Domain Decomposition Methods: Theory and Applications

Frédéric Nataf

Laboratory J.L. Lions (LJLL), CNRS, Alpines Inria and Sorbonne University

---

IWR Colloquium – December 2018

Outline

Outline

Framework: Scientific computing

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

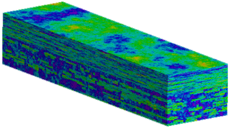
E.g. Darcy pressure equation,
P^1-finite elements:

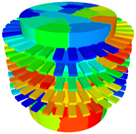
$$\mathbf{A}\mathbf{u} = \mathbf{f}$$

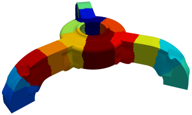
$$\mathrm{cond}(\mathbf{A}) \sim \frac{\kappa_{\max}}{\kappa_{\min}} h^{-2}$$

Goal:
Parallel iterative solvers
robust in size and heterogeneities

Applications:
flow in heterogeneous / stochastic / layered media
structural mechanics
electromagnetics
graph Laplacian, ...





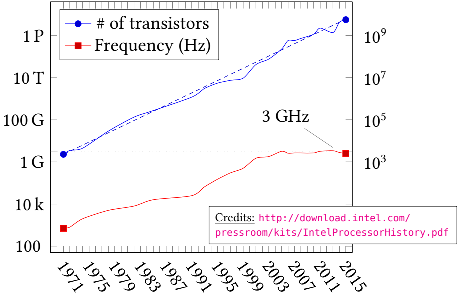


80% of the elapsed time for typical engineering applications

Changes in hardware ⇒ Go parallel

Since year 2004:

- CPU frequency stalls at 3 GHz due to the heat wall.



- 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.

Physical limitations of communication vs. computation

Energy

- a 32-bit floating-point operation requires 3.1 pJ
- whereas the same DRAM read requires 640 pJ.

Speed

- Infiniband latency 1μ sec., 3,000 operations at 3GHz
- Minimum latency for an internode distance of 3 meters: 0.01μ sec. 10 operations at 3GHz

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)
- Minimize sequential task
- Redundant computations are welcome if they decrease communication

$Au = f$? Panorama of linear solvers

Direct Solvers

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

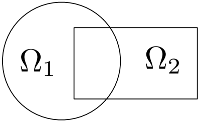
Iterative Methods

- 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, ...) Frequency decomposition methods
- Domain decomposition methods (O. Widlund, C. Farhat, J. Mandel, P.L. Lions,) are a naturally parallel compromise

The original Schwarz Method (H.A. Schwarz, 1870)



$$-\Delta(u) = f \quad \text{in } \Omega$$
$$u = 0 \quad \text{on } \partial\Omega.$$

Schwarz Method : $(u_1^n, u_2^n) \to (u_1^{n+1}, u_2^{n+1})$ with

$$\begin{aligned} -\Delta(u_1^{n+1}) &= f \quad \text{in } \Omega_1 \\ u_1^{n+1} &= 0 \text{ on } \partial\Omega_1 \cap \partial\Omega \\ u_1^{n+1} &= u_2^n \quad \text{on } \partial\Omega_1 \cap \overline{\Omega_2}. \end{aligned} \qquad \begin{aligned} -\Delta(u_2^{n+1}) &= f \quad \text{in } \Omega_2 \\ u_2^{n+1} &= 0 \text{ on } \partial\Omega_2 \cap \partial\Omega \\ u_2^{n+1} &= u_1^{n+1} \quad \text{on } \partial\Omega_2 \cap \overline{\Omega_1}. \end{aligned}$$

Parallel algorithm.

Consider the discretized Poisson problem: $Au = f \in \mathbb{R}^n$.

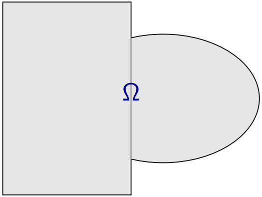
Given a decomposition of $[\![1;n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- 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 $\mathbb{R}^{[\![1;n]\!]}$.

$u^m \longrightarrow u^{m+1}$ by solving 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 $A_i := R_i A R_i^T$.



An introduction to Additive Schwarz

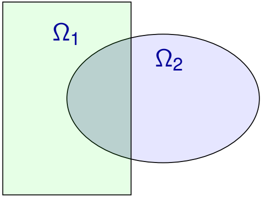
Consider the discretized Poisson problem: $Au = f \in \mathbb{R}^n$.
Given a decomposition of $[\![1; n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- 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 $\mathbb{R}^{[\![1;n]\!]}$.

$u^m \longrightarrow u^{m+1}$ by solving 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 $A_i := R_i A R_i^T$.



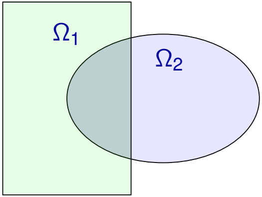
Consider the discretized Poisson problem: $Au = f \in \mathbb{R}^n$.
Given a decomposition of $[\![1;n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- 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 $\mathbb{R}^{[\![1;n]\!]}$.

$u^m \longrightarrow u^{m+1}$ by solving 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 $A_i := R_i A R_i^T$.

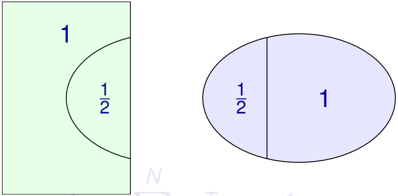


An introduction to Additive Schwarz II

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, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$. $M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i$.

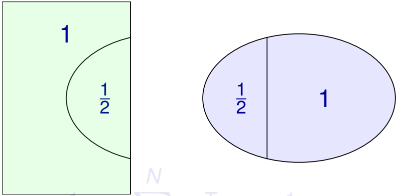
RAS algorithm (Cai & Sarkis, 1999) Equivalence with a weighted Block Jacobi method with overlap

An introduction to Additive Schwarz 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, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

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

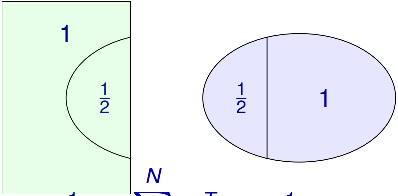
RAS algorithm (Cai & Sarkis, 1999) Equivalence with a weighted Block Jacobi method with overlap

An introduction to Additive Schwarz 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, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

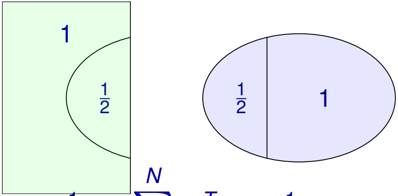
$$M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i .$$

RAS algorithm (Cai & Sarkis, 1999) Equivalence with a weighted Block Jacobi method with overlap

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, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$. $M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i$.

RAS algorithm (Cai & Sarkis, 1999) Equivalence with a weighted Block Jacobi method with overlap

Algebraic formulation - RAS and ASM

Schwarz algorithm iterates on a pair of local functions (u_m^1, u_m^2)
RAS algorithm iterates on the global function u^m

Schwarz and RAS

Discretization of the classical Schwarz algorithm and the iterative RAS algorithm:

$$U^{n+1} = U^n + M_{RAS}^{-1} r^n \,, r^n := F - A\,U^n .$$

are equivalent

$$U^n = R_1^T D_1 U_1^n + R_2^T D_2 U_2^n \,.$$

(Efstathiou and Gander, 2002).

Operator M_{RAS}^{-1} is used as a preconditioner in Krylov methods for non symmetric problems.

Algebraic formulation - RAS and ASM

Schwarz algorithm iterates on a pair of local functions (u_m^1, u_m^2)
RAS algorithm iterates on the global function u^m

Schwarz and RAS

Discretization of the classical Schwarz algorithm and the iterative RAS algorithm:

$$U^{n+1} = U^n + M_{RAS}^{-1} r^n \, , r^n := F - A\,U^n .$$

are equivalent

$$U^n = R_1^T D_1 U_1^n + R_2^T D_2 U_2^n \, .$$

(Efstathiou and Gander, 2002).

Operator M_{RAS}^{-1} is used as a preconditioner in Krylov methods for non symmetric problems.

ASM: a symmetrized version of RAS

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

A symmetrized version: Additive Schwarz Method (ASM),

$$M_{ASM}^{-1} := \sum_{i=1}^{N} R_i^T A_i^{-1} R_i \qquad (1)$$

is used as a preconditioner for the conjugate gradient (CG) method.

Although RAS is more efficient, ASM is amenable to condition number estimates.

Chronological curiosity: First paper on Additive Schwarz dates back to 1989 whereas RAS paper was published in 1998

Adding a coarse space

One level methods are not scalable. We add a coarse space correction (*aka* second level or deflation)
Let V_H be the coarse space and Z be a basis, $V_H = \text{span}\, Z$, writing $R_0 = Z^T$ we define the two level preconditioner as:

$$M^{-1}_{ASM,2} := R_0^T (R_0 A R_0^T)^{-1} R_0 + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i .$$

The Nicolaides approach (1987) is to use the kernel of the operator as a coarse space, this is the constant vectors, in local form this writes:

$$Z := (R_i^T D_i R_i \mathbf{1})_{1 \le i \le N}$$

where D_i are chosen so that we have a partition of unity:

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

Key notion: Stable splitting (J. Xu, 1989)

Theoretical convergence result

Theorem (Widlund, Dryija)

Let $M_{ASM,2}^{-1}$ be the two-level additive Schwarz method:

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

where δ is the size of the overlap between the subdomains and H the subdomain size.

This does indeed work very well

Number of subdomains	8	16	32	64
ASM	18	35	66	128
ASM + Nicolaides	20	27	28	27

Fails for highly heterogeneous problems
You need a larger and adaptive coarse space

Adaptive Coarse space for highly heterogeneous Darcy and (compressible) elasticity problems:
Geneo .EVP per subdomain:

Find $V_{j,k} \in \mathbb{R}^{\mathcal{N}_j}$ and $\lambda_{j,k} \geq 0$:

$$D_j R_j A R_j^T D_j V_{j,k} = \lambda_{j,k} A_j^{Neu} V_{j,k}$$

In the two-level ASM, let τ be a user chosen parameter:
Choose eigenvectors $\lambda_{j,k} \geq \tau$ per subdomain:

$$Z := \left(R_j^T D_j V_{j,k} \right)_{\lambda_{j,k} \geq \tau}^{j=1,\ldots,N}$$

This automatically includes Nicolaides CS made of Zero Energy Modes.

Adaptive Coarse space for highly heterogeneous Darcy and (compressible) elasticity problems:
Geneo .EVP per subdomain:

Find $V_{j,k} \in \mathbb{R}^{\mathcal{N}_j}$ and $\lambda_{j,k} \geq 0$:

$$D_j R_j A R_j^T D_j V_{j,k} = \lambda_{j,k} A_j^{Neu} V_{j,k}$$

In the two-level ASM, let τ be a user chosen parameter:
Choose eigenvectors $\lambda_{j,k} \geq \tau$ per subdomain:

$$Z := \left(R_j^T D_j V_{j,k}\right)_{\lambda_{j,k} \geq \tau}^{j=1,\ldots,N}$$

This automatically includes Nicolaides CS made of Zero

Energy Modes.

Theory of GenEO

Under two technical assumptions.

Theorem (Spillane, Dolean, Hauret, N., Pechstein, Scheichl (Num. Math. 2013))

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

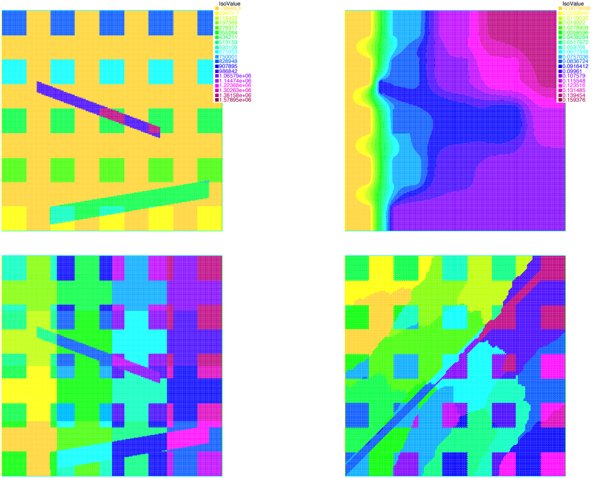
$$\kappa(M_{ASM,2}^{-1}A) \le (1+k_0)\Big[2 + k_0\,(2k_0+1)\Big(1+\tau\Big)\Big]$$

Possible criterion for picking τ: (used in our Numerics)

$$\tau := \min_{j=1,\dots,N} \frac{H_j}{\delta_j}$$

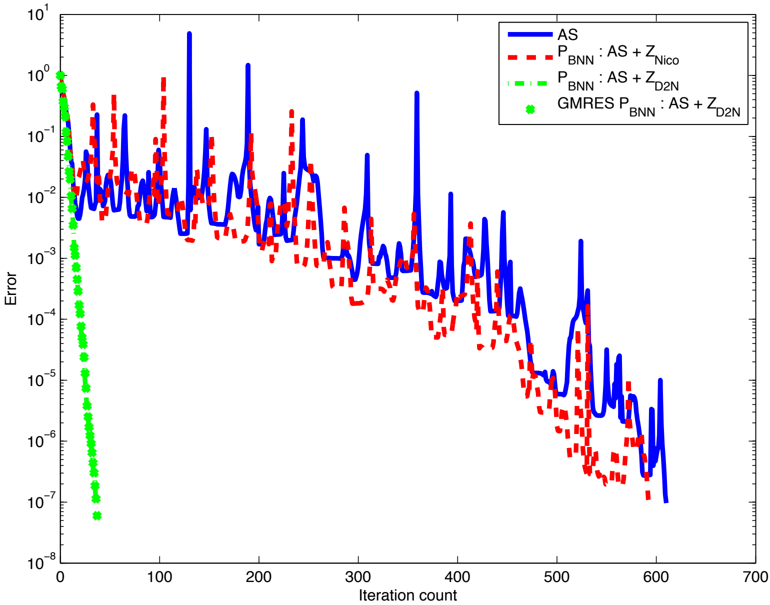
$H_j \dots$ subdomain diameter, $\delta_j \dots$ overlap

Numerical results (Darcy)



Channels and inclusions: $1 \leq \alpha \leq 1.5 \times 10^6$, the solution and partitionings (Metis or not)

Convergence



Outline

P.L. Lions' Algorithm (1988)

$$
\begin{array}{l}
-\Delta(u_1^{n+1}) = f \quad \text{in } \Omega_1, \\
u_1^{n+1} = 0 \quad \text{on } \partial\Omega_1 \cap \partial\Omega, \\
(\dfrac{\partial}{\partial n_1} + \alpha)(u_1^{n+1}) = (-\dfrac{\partial}{\partial n_2} + \alpha)(u_2^n) \quad \text{on } \partial\Omega_1 \cap \overline{\Omega_2},
\end{array}
$$

(n_1 and n_2 are the outward normal on the boundary of the subdomains)

$$
\begin{array}{l}
-\Delta(u_2^{n+1}) = f \quad \text{in } \Omega_2, \\
u_2^{n+1} = 0 \quad \text{on } \partial\Omega_2 \cap \partial\Omega \\
(\dfrac{\partial}{\partial n_2} + \alpha)(u_2^{n+1}) = (-\dfrac{\partial}{\partial n_1} + \alpha)(u_1^n) \quad \text{on } \partial\Omega_2 \cap \overline{\Omega_1}.
\end{array}
$$

with $\alpha > 0$. Overlap is not necessary for convergence.
Parameter α can be optimized for.
Extended to the Helmholtz equation (B. Després, 1991)
a.k.a FETI 2 LM (Two-Lagrange Multiplier) Method, 1998.

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \leq i \leq N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \le i \le N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \leq i \leq N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate

   $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \le i \le N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate

   $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \le i \le N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate
   $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \leq i \leq N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate
   $\Rightarrow$???

ORAS: Optimized RAS

1. P.L. Lions algorithm at the continuous level (partial differential equation)
2. Algebraic formulation for overlapping subdomains $\Rightarrow$ Let B_i be the matrix of the Robin subproblem in each subdomain $1 \leq i \leq N$, define $M_{ORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} R_i$, *Optimized multiplicative, additive, and restricted additive Schwarz preconditioning, St Cyr et al, 2007*
3. Symmetric variant $\Rightarrow$
   1. $M_{OAS}^{-1} := \sum_{i=1}^{N} R_i^T B_i^{-1} R_i$ (Natural but K.O.)
   2. $M_{SORAS}^{-1} := \sum_{i=1}^{N} R_i^T D_i B_i^{-1} D_i R_i$ (O.K.)
4. Adaptive Coarse space with prescribed targeted convergence rate
   $\Rightarrow$???

Fictitious Space Lemma

Lemma (Fictitious Space Lemma, Nepomnyaschikh 1991)

Let H and H_D be two Hilbert spaces. Let a be a symmetric positive bilinear form on H and b on H_D. Suppose that there exists a linear operator $\mathcal{R} : H_D \to H$, such that

- *$\mathcal{R}$ is surjective.*
- *there exists a positive constant c_R such that*

$$a(\mathcal{R}u_D, \mathcal{R}u_D) \leq c_R \cdot b(u_D, u_D) \;\; \forall u_D \in H_D \, . \tag{2}$$

- *Stable decomposition: there exists a positive constant c_T such that for all $u \in H$ there exists $u_D \in H_D$ with $\mathcal{R}u_D = u$ and*

$$c_T \cdot b(u_D, u_D) \leq a(\mathcal{R}u_D, \mathcal{R}u_D) = a(u, u) \, . \tag{3}$$

Fictitious Space Lemma (continued)

Lemma (FSL continued)

We introduce the adjoint operator $\mathcal{R}^* : H \to H_D$ *by* $(\mathcal{R}u_D, u) = (u_D, \mathcal{R}^* u)_D$ *for all* $u_D \in H_D$ *and* $u \in H$*. Then we have the following spectral estimate*

$$c_T \cdot a(u, u) \leq a\left(\mathcal{R}B^{-1}\mathcal{R}^* Au,\, u\right) \leq c_R \cdot a(u, u)\,, \quad \forall u \in H \qquad (4)$$

which proves that the eigenvalues of operator $\mathcal{R}B^{-1}\mathcal{R}^* A$ *are bounded from below by* c_T *and from above by* c_R*.*

- $H := \mathbb{R}^{\#\mathcal{N}}$ and the a-bilinear form:

$$a(\mathbf{U}, \mathbf{V}) := \mathbf{V}^T A \mathbf{U}. \tag{5}$$

where A is the matrix of the problem we want to solve.

- H_D is a product space and b a bilinear form defined by

$$H_D := \prod_{i=1}^{N} \mathbb{R}^{\#\mathcal{N}_i} \text{ and } b(\mathcal{U}, \mathcal{V}) := \sum_{i=1}^{N} \mathbf{V}_i^T B_i \mathbf{U}_i, \, . \tag{6}$$

- The linear operator $\mathcal{R}_{SORAS}$ is defined as

$$\mathcal{R}_{SORAS} : H_D \longrightarrow H, \, \mathcal{R}_{SORAS}(\mathcal{U}) := \sum_{i=1}^{N} R_i^T D_i \mathbf{U}_i. \tag{7}$$

We have: $M_{SORAS}^{-1} = \mathcal{R}_{SORAS} \, B^{-1} \, \mathcal{R}_{SORAS}^{*}$.

Estimate for the **one level** SORAS

Let k_0 be the maximum number of neighbors of a subdomain and γ_1 be defined as:

$$\gamma_1 := \max_{1\le i\le N} \max_{\mathbf{U}_i \in \mathbb{R}^{\#\mathcal{N}_i}\setminus\{0\}} \frac{(D_i\mathbf{U}_i)^T A_i (D_i\mathbf{U}_i)}{\mathbf{U}_i^T B_i \mathbf{U}_i}$$

We can take $c_R := k_0\,\gamma_1$.
Let k_1 be the maximum multiplicity of the intersection between subdomains and τ_1 be defined as:

$$\tau_1 := \min_{1\le i\le N} \min_{U_i \in \mathbb{R}^{\#\mathcal{N}_i}\setminus\{0\}} \frac{\mathbf{U}_i{}^T A_i^{Neu} \mathbf{U}_i}{\mathbf{U}_i{}^T B_i \mathbf{U}_i} .$$

We can take $c_T := \frac{\tau_1}{k_1}$.
We have:

$$\frac{\tau_1}{k_1} \le \lambda(M_{SORAS}^{-1} A) \le k_0\,\gamma_1 .$$

Definition (Generalized Eigenvalue Problem for the upper bound)

$$\text{Find } (\mathbf{U}_{ik}, \mu_{ik}) \in \mathbb{R}^{\#\mathcal{N}_i} \setminus \{0\} \times \mathbb{R} \text{ such that}$$
$$D_i A_i D_i \mathbf{U}_{ik} = \mu_{ik} B_i \, \mathbf{U}_{ik} \quad . \tag{8}$$

Let $\gamma > 0$ be a user-defined threshold, we define $Z_{geneo}^{\gamma} \subset \mathbb{R}^{\#\mathcal{N}}$ as the vector space spanned by the family of vectors $(R_i^T D_i \mathbf{U}_{ik})_{\mu_{ik} > \gamma, 1 \le i \le N}$ corresponding to eigenvalues larger than γ.

Control of the lower bound

Definition (Generalized Eigenvalue Problem for the lower bound)

For each subdomain $1 \leq j \leq N$, we introduce the generalized eigenvalue problem

$$\begin{array}{c} \text{Find } (\mathbf{V}_{jk}, \lambda_{jk}) \in \mathbb{R}^{\#\mathcal{N}_j} \setminus \{0\} \times \mathbb{R} \text{ such that} \\ A_j^{Neu} \mathbf{V}_{jk} = \lambda_{jk} B_j \mathbf{V}_{jk} \ . \end{array} \quad (9)$$

Let $\tau > 0$ be a user-defined threshold, we define $Z_{geneo}^{\tau} \subset \mathbb{R}^{\#\mathcal{N}}$ as the vector space spanned by the family of vectors $(R_j^T D_j \mathbf{V}_{jk})_{\lambda_{jk} < \tau, 1 \leq j \leq N}$ corresponding to eigenvalues smaller than τ.

Two level SORAS-GENEO-2 preconditioner

Definition (Two level SORAS-GENEO-2 preconditioner)

Let P_0 denote the a-orthogonal projection on the SORAS-GENEO-2 coarse space

$$Z_{\text{GenEO-2}} := Z^{\tau}_{geneo} \bigoplus Z^{\gamma}_{geneo} ,$$

the two-level SORAS-GENEO-2 preconditioner is defined:

$$\boxed{M^{-1}_{SORAS,2} := P_0 A^{-1} + (I_d - P_0)\, M^{-1}_{SORAS}\, (I_d - P_0^T)}$$

where $P_0 A^{-1} = R_0^T (R_0\, A\, R_0^T)^{-1} R_0$, see J. Mandel, (1992) and BFGS algorithm (1970).

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

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

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

$$\text{Find } (\mathbf{V}_{jk}, \lambda_{jk}) \in \mathbb{R}^{\#\mathcal{N}_j} \setminus \{0\} \times \mathbb{R} \text{ such that}$$
$$A_j^{Neu}\mathbf{V}_{jk} = \lambda_{jk} D_j B_j D_j \mathbf{V}_{jk}\ .$$

FSL and other DDM

GenEO technique yields adaptive coarse spaces with a targeted condition number for:

- Additive Schwarz method
- Hybrid Schwarz method
- Balancing Neumann Neumann and FETI
- Optimized Schwarz method (Haferssas, Jolivet, N. 2017)

See also Schwarz algorithm for the Schur complement (Poirel, Agullo & Giraud)

For a comprehensive presentation:

”An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation”, V. Dolean, P. Jolivet and F Nataf, `https://www.ljll.math.upmc.fr/nataf/OT144DoleanJolivetNataf_full.pdf`, Lecture Notes, SIAM, 2015.

Outline

Neumann sub matrix and Domain Specific Language

In addition to the original matrix, we need for each subdomain j a Neumann matrix A_j^{Neu}.

DD methods, access to the local Neumann matrix:

- Ask the developer of the simulation code to provide it (HPDDM, Jolivet & N.)
- Infer it from some assumption on the problem at hand (e.g. Graph Laplacian)
- Use Domain Specific Language (DSL) for finite element (or volume) method

Note that a related question in multigrid is the access to the near kernel of the matrix:

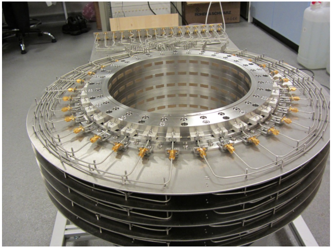
- user provided near kernel of the matrix (GAMG with PETSc)
- Infer it from some assumption on the problem at hand (e.g. Graph Laplacian) (Notay-Napov 2016).

Why use a DS(E)L (FreeFem++, Feel++, Dune, Fenics or Firedrake) instead of `C/C++/Fortran/..` ?

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

```
varf a(u, v) = int3d(mesh)([dx(u), dy(u), dz(u)]' * [dx(v), dy(v), dz(v)])
               - int3d(mesh)(f * v) + on(boundary_mesh)(u = 0) ,
```

- access to the variational formulation is then natural and that's what we need.



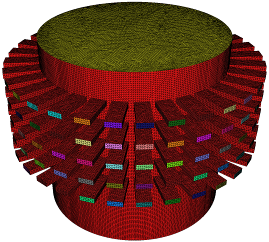


Figure: Antennas and mesh – interior diameter 28,5 cm

In-house open source libraries (LGPL) linked to many third-party libraries:

- HPDDM (High Performance Domain Decomposition Methods) for massively parallel computing
- FreeFem++(-mpi) for the parallel simulation of equations from physics by the finite element method (FEM).

Parallel Software : HPDDM, ffddm and FreeFem++

HPDDM (P. Jolivet, F.N.) and ffddm (P.-H. Tournier, F.N.)

- Implements parallel algorithms: Domain Decomposition methods and Block solvers
- 2 billions unknowns in three dimension solved in 210 seconds on 8100 cores
- Interfaced with FreeFem++ and Feel++
- HPDDM can be interfaced with a C++, C, Fortran or Python code
- ffddm is a pure FreeFem++ implementation of DDM

FreeFem++

- versatile parallel simulation tools: fluid and solid mechanics, electromagnetism, quantum physics, . . .
- documentation in English (Franglish say), Japanese, Spanish and Chinese 如何使用FreeFem++
- teaching, research, prototyping in some big companies and in some small/medium companies as a production code

Numerical results via a Domain Specific Language

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

- Metis Karypis and Kumar 1998
- SCOTCH Chevalier and Pellegrini 2008
- UMFPACK Davis 2004
- ARPACK Lehoucq et al. 1998
- MPI Snir et al.
- Intel MKL
- PARDISO Schenk et al. 2004
- MUMPS Amestoy et al. 1998
- PETSc solvers Balay et al.
- Slepc via PETSc

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

FreeFem++ and DDM $\Rightarrow$ HPC for ALL!

A new interface with P.-H. Tournier: diffusion-3d-minimal-ddm.edp

```
meshN ThGlobal = cube(getARGV("−global", 10), getARGV("−global",
      10), getARGV("−global", 10),
    [x, y, z], label = LL);        // global mesh

macro Varf(varfName, meshName, PhName)
    varf varfName(u,v) = intN(meshName)(grad(u)' * grad(v)) +
        intN(meshName)(v) + on(1, u = 1.0); // EOM

// Domain decomposition
ffddmbuildDDmesh( Lap , ThGlobal , mpiCommWorld )
ffddmbuildDDfespace( Lap , Lap , real , def , init , P1 )
ffddmsetupOperator(Lap ,Lap , Varf)
```

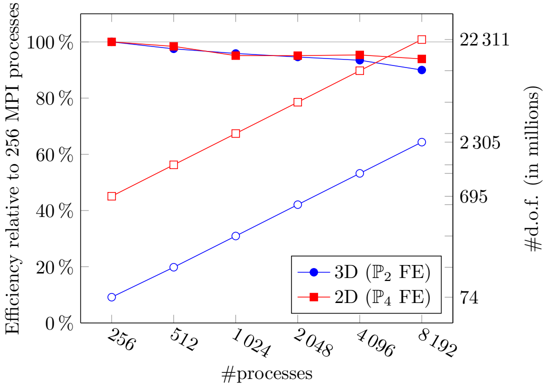
A new interface with P.-H. Tournier: diffusion-3d-minimal-ddm.edp

```
// Domain decomposition solve
real[int] rhs(LapVhi.ndof);
ffddmbuildrhs(Lap , Varf , rhs )
LapVhi def(u) ;

// Two-level Schwarz solve
ffddmsetupPrecond(Lap,Varf)
ffddmgeneosetup(Lap,Varf)
ffddmset(Lap,corr,"BNN");
real[int] x0(LapVhi.ndof);
x0 = 0;
u[] = LapfGMRES(x0, rhs, 1.e-6, 200, "right");
Lapwritesummary//process 0 prints convergence history
ffddmplot(Lap,u, "Lap Global solution with fGMRES");
```

Weak scalability in three dimensions

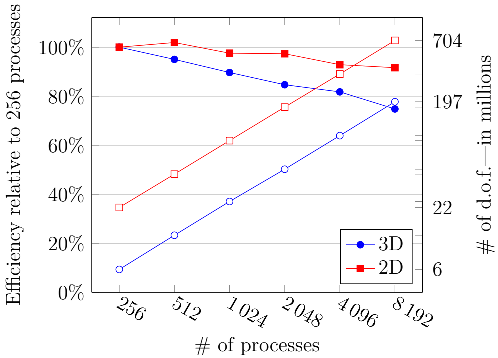
Darcy problems with heterogeneous coefficients with automatic mesh partition



Efficiency for a 3D problem. Direct solvers in the subdomains. Final size: 2 billion unknowns. Wall-clock time: $\simeq$ 200s.

Weak scalability for heterogeneous elasticity (with FreeFem++ and HPDDM)

Rubber Steel sandwich with automatic mesh partition



(a) Timings of various simulations

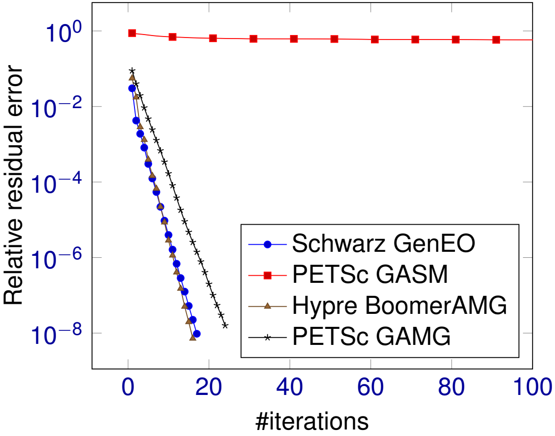
Figure: 200 millions unknowns in 3D wall-clock time: 200. sec. , Haferssas, Jolivet & N., 2016

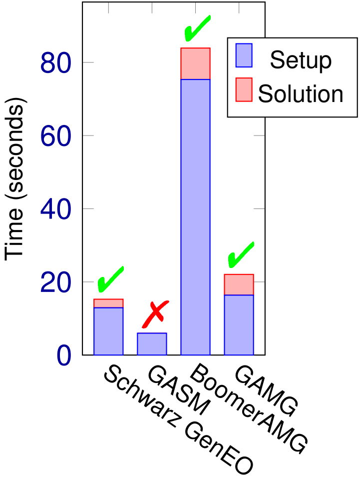
Comparison to multigrid

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

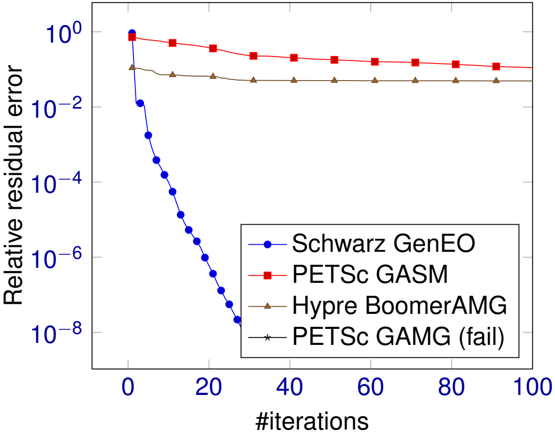
- GASM – one-level domain decomposition method (ANL),
- Hypre BoomerAMG – algebraic multigrid (LLNL),
- GAMG – algebraic multrigrid (ANL/LBL).

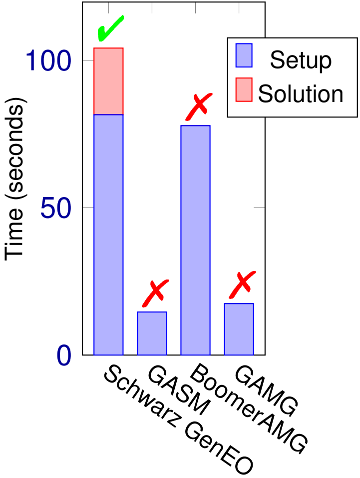
Homogeneous 3D Poisson equation discretized by $\mathbb{P}_1$ FE solved on 2,048 MPI processes, 111M d.o.f.





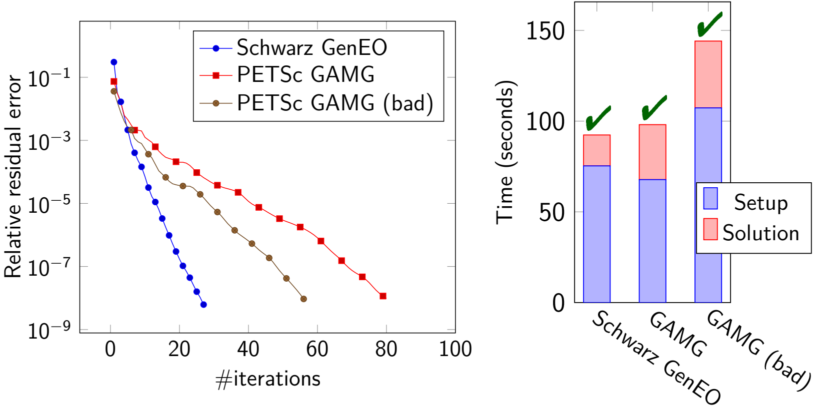
Heterogeneous 3D linear elasticity equation discretized by $\mathbb{P}_2$ FE solved on 4,096 MPI processes, 127M d.o.f.





Solution of a linear system II

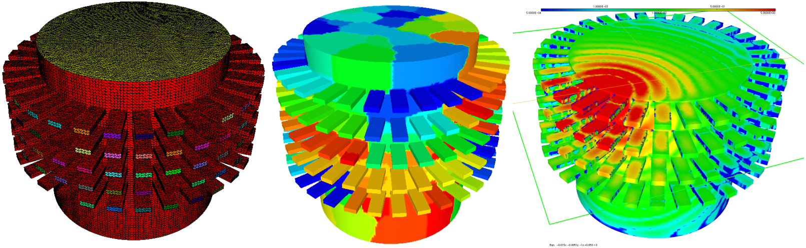
Heterogeneous 3D linear elasticity equation discretized by $\mathbb{P}_2$ FE solved on 4,096 MPI processes, 262M d.o.f.



Wave propagation phenomena

The coarse space, if necessary, is built from a coarse grid. If a coarse problem is solved, it is approximately solved by a one level DD method. See also I.G. Graham, E.A. Spence and J. Zou, DD with local impedance conditions for the Helmholtz equation,

- Mesh with 2.3M degrees of freedom;
- Domain decomposition methods with impedance interface conditions, twice as fast as Dirichlet interface conditions;
- Parallel computing on 64 cores on SGI UV2000 at UPMC : 3s per emitter, 5 mn as a whole.



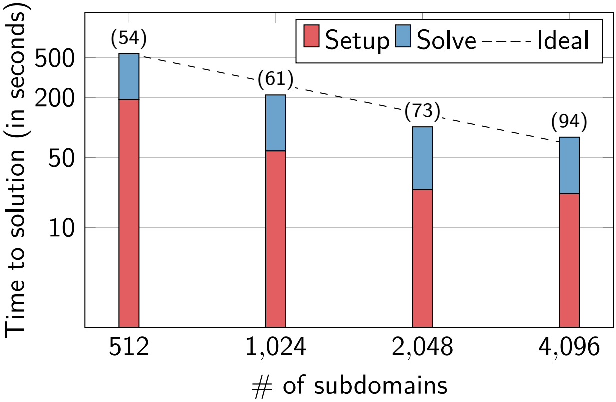
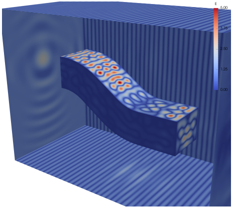
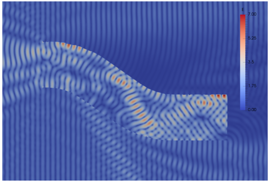


Figure: Maxwell 3D with edge elements of degree 2 - 119M d.o.f.

Bonazzoli, Dolean, Graham, Spence, Tournier, 2018.
order 2 edge elements (Nedelec), 10 pts per wavelength
$f = 10\,GHz$: $n \approx 1.07 \times 10^8$ $\quad$ $f = 16\,GHz$: $n \approx 1.98 \times 10^8$

f	N_{sub}	# it	inner it	Total	Setup	GMRES	inner
10GHz	1536	32	1527	515.8	383.2	132.6	61.8
10GHz	3072	33	2083	285.0	201.6	83.4	40.6
16GHz	3072	43	3610	549.2	336.8	212.4	118.6
16GHz	6144	46	4744	363.0	210.5	152.5	96.8





5 points per wave length, P2 FE Simulations réalisées sur Occigen (CINES) noeuds Haswell

		cartesian mesh			adaptive mesh		
f	# cores	# dofs	# it	sec.	# dofs	# it	sec.
5	384	22 M	167	58	11 M	125	25
10	3072	176 M	340	121	85 M	253	59
20	12288	—	—	—	678 M	438	218

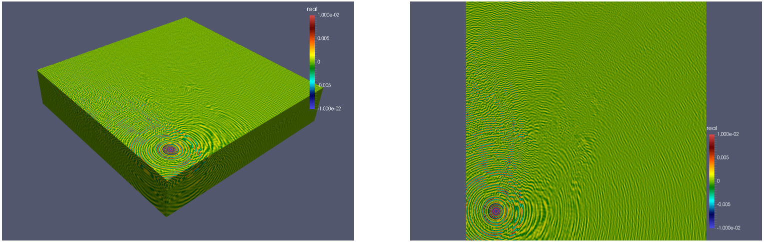


Figure: Simulations with FreeFem++ (P.H. Tournier)

Radiative transfer problem

$$(\boldsymbol{s} \cdot \nabla + \beta(\boldsymbol{x}))I(\boldsymbol{x}, \boldsymbol{s}) - \sigma_s(\boldsymbol{x}) \oint_{\mathcal{S}} I(\boldsymbol{x}, \boldsymbol{s}')\Phi(\boldsymbol{s}, \boldsymbol{s}')\ \mathrm{d}\boldsymbol{s}'$$
$$- \kappa(\boldsymbol{x}) I_b(\boldsymbol{x}) = 0 \qquad \forall \boldsymbol{x} \in \Omega, \boldsymbol{s} \in \mathcal{S}$$

One billion unknowns in 60 seconds with 8192 MPI processes

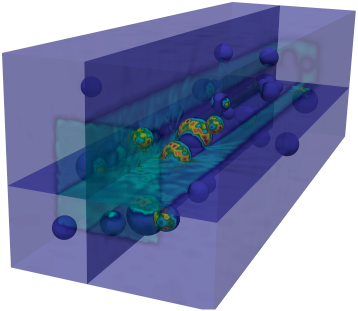


Figure: Badria, Jolivet, Rousseau, Le Corre, Digonnet and Favennec, 2018 – FreeFem++ script

Outline

Conclusion

Summary

- DDM are naturally parallel and communication avoiding
- Targeted convergence rate for SPD problems
- FreeFem++ integration of GenEO via HPDDM yields a versatile and powerful tool

Work in progress

- Multigrid like three (or more) level methods (Stability w.r.t to approximate coarse solves N. 2018)
- Firedrake (yafem DSL) integration of GenEO via geneo4PETSc

Open questions

- Theoretical framework for saddle point, non symmetric or undefinite problems (Graham, Spence 2017)

Bibliography

Available on HAL and Software on freefem.org and github:

- P. Jolivet, V. Dolean, F. Hecht, F. Nataf, C. Prud'homme, N. Spillane, "High Performance domain decomposition methods on massively parallel architectures with FreeFem++", J. of Numerical Mathematics, 2012 vol. 20.
- N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, R. Scheichl, "Abstract Robust Coarse Spaces for Systems of PDEs via Generalized Eigenproblems in the Overlaps", *Numerische Mathematik*, 2013.
- V. Dolean, P. Jolivet and F Nataf, "An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation", SIAM, 2015.
- P. Jolivet and F Nataf, "HPDDM: high-performance unified framework for domain decomposition methods", https://github.com/hpddm/hpddm , MPI-C++ library with C++, C and Python interfaces, 2014.
- F. Houssen and F. Nataf, "geneo4PETSc - Implementation of the GenEO preconditioner with PETSc and SLEPc." https://github.com/geneo4PETSc/geneo4PETSc

Thanks to GENCI for the HPC resources of OCCIGEN at CINES under the allocations 2017-067730

THANK YOU FOR YOUR ATTENTION!

Available on HAL and Software on freefem.org and github:

- P. Jolivet, V. Dolean, F. Hecht, F. Nataf, C. Prud'homme, N. Spillane, "High Performance domain decomposition methods on massively parallel architectures with FreeFem++", J. of Numerical Mathematics, 2012 vol. 20.
- N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, R. Scheichl, "Abstract Robust Coarse Spaces for Systems of PDEs via Generalized Eigenproblems in the Overlaps", *Numerische Mathematik*, 2013.
- V. Dolean, P. Jolivet and F Nataf, "An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation", SIAM, 2015.
- P. Jolivet and F Nataf, "HPDDM: high-performance unified framework for domain decomposition methods", https://github.com/hpddm/hpddm , MPI-C++ library with C++, C and Python interfaces, 2014.
- F. Houssen and F. Nataf, "geneo4PETSc - Implementation of the GenEO preconditioner with PETSc and SLEPc." https://github.com/geneo4PETSc/geneo4PETSc

Thanks to GENCI for the HPC resources of OCCIGEN at CINES under the allocations 2017-067730

THANK YOU FOR YOUR ATTENTION!

Bibliography

Available on HAL and Software on freefem.org and github:

- P. Jolivet, V. Dolean, F. Hecht, F. Nataf, C. Prud'homme, N. Spillane, "High Performance domain decomposition methods on massively parallel architectures with FreeFem++", J. of Numerical Mathematics, 2012 vol. 20.
- N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, R. Scheichl, "Abstract Robust Coarse Spaces for Systems of PDEs via Generalized Eigenproblems in the Overlaps", *Numerische Mathematik*, 2013.
- V. Dolean, P. Jolivet and F Nataf, "An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation", SIAM, 2015.
- P. Jolivet and F Nataf, "HPDDM: high-performance unified framework for domain decomposition methods", https://github.com/hpddm/hpddm , MPI-C++ library with C++, C and Python interfaces, 2014.
- F. Houssen and F. Nataf, "geneo4PETSc - Implementation of the GenEO preconditioner with PETSc and SLEPc." https://github.com/geneo4PETSc/geneo4PETSc

Thanks to GENCI for the HPC resources of OCCIGEN at CINES under the allocations 2017-067730

THANK YOU FOR YOUR ATTENTION!