Scalable Domain Decomposition Preconditioners For Heterogeneous Elliptic Problems

Pierre Jolivet, F. Hecht,
F. Nataf, C. Prud'homme

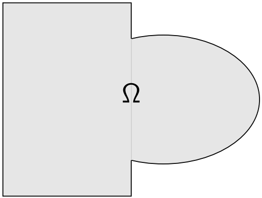
Laboratoire Jacques-Louis Lions
Laboratoire Jean Kuntzmann
INRIA Rocquencourt

4th workshop on `FreeFem++`

December 12nd, 2013

A short introduction to DDM

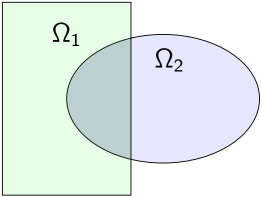
Consider the linear system: $Au = f \in \mathbb{R}^n$.



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



A short introduction to DDM

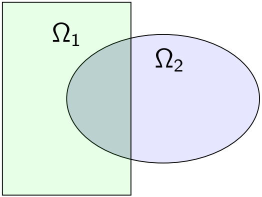
Consider the linear system: $Au = f \in \mathbb{R}^n$.
Given a decomposition of $[\![1; n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

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

Then solve concurrently:

$$u_1^{m+1} = u_1^m + A_{11}^{-1} R_1 (f - Au^m) \qquad u_2^{m+1} = u_2^m + A_{22}^{-1} R_2 (f - Au^m)$$

where $u_i = R_i u$ and $A_{ij} := R_i A R_j^T$.

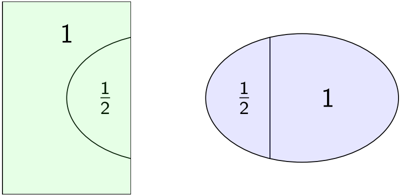


A short introduction II

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

Duplicated unknowns coupled via a *partition of unity*:

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

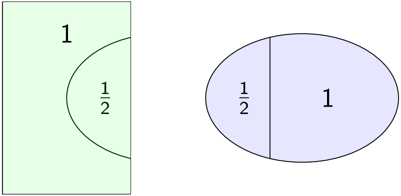


A short introduction II

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

Duplicated unknowns coupled via a *partition of unity*:

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



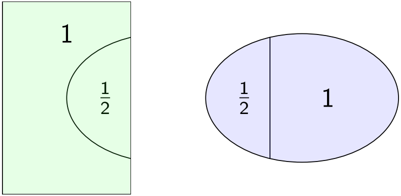
Then, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

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

$$M^{-1} = \sum_{i=1}^{N} R_i^T D_i A_{ii}^{-1} R_i.$$

Contributions and goals

Based on algebraic results with the p. of u., we propose:

1. a reformulation of the global matrix-vector product eliminating the need of a global ordering,
2. a construction of a so-called "coarse operator" to enhance a simple preconditioner.

We are interested in the solution of various SPD systems, independently of:

- the discretization order,
- the contrast in the coefficients,
- the number of subdomains.

Using the overlap to its fullest extent

DDM methods are seldom used as standalone solvers.

Krylov methods and overlapping Schwarz methods

$$Au \implies \text{efficient global matrix-vector product}$$

Using the overlap to its fullest extent

DDM methods are seldom used as standalone solvers.

Krylov methods and overlapping Schwarz methods

$$Au = \sum_{j=1}^{N} A R_j^T D_j R_j u$$

Using the overlap to its fullest extent

DDM methods are seldom used as standalone solvers.

Krylov methods and overlapping Schwarz methods

$$R_i A u = \sum_{j=1}^{N} R_i A R_j^T D_j R_j u$$

Using the overlap to its fullest extent

DDM methods are seldom used as standalone solvers.

Krylov methods and overlapping Schwarz methods

$$R_i A u = \sum_{j=1}^{N} R_i A R_j^T D_j R_j u = \sum_{j \in \overline{\mathcal{O}_i}} A_{ij} D_j R_j u$$

Using the overlap to its fullest extent

DDM methods are seldom used as standalone solvers.

Krylov methods and overlapping Schwarz methods

$$R_i A u = \sum_{j=1}^{N} R_i A R_j^T D_j R_j u = \sum_{j \in \overline{\mathcal{O}_i}} A_{ij} D_j R_j u$$

$$= \sum_{j \in \overline{\mathcal{O}_i}} R_i R_j^T A_{jj} D_j R_j u.$$

local unknowns on Ω_j

- no need for the global matrix, only local to neighbors mappings.
  $\hookrightarrow$ explicit point-to-point communications via $R_i R_j^T$.
- reuse of the operators from the preconditioner, A_{ii}.

$\mathcal{O}_i$ are the neighbors of Ω_i, $\overline{\mathcal{O}_i} = \mathcal{O}_i \cup \{i\}$.

Limitations of one-level methods

One-level methods don't require exchange of global information.

This hampers numerical scalability of such preconditioners.

Two-level preconditioners I

A common technique in the field of DDM, MG, deflation:

introduce an auxiliary "coarse" problem.

Let Z be a rectangular matrix. Define

$$E := Z^T A Z\,.$$

Z has $\mathcal{O}(N)$ columns, hence E is much smaller than A.

Two-level preconditioners I

A common technique in the field of DDM, MG, deflation:

introduce an auxiliary “coarse” problem.

Let Z be a rectangular matrix. Define

$$E := Z^T A Z\,.$$

Z has $\mathcal{O}(N)$ columns, hence E is much smaller than A.
Enrich the original preconditioner, e.g. additively

$$P^{-1} = M^{-1} + Z E^{-1} Z^T,$$

c.f. (Tang et al. 2009).

Two-level preconditioners II

The construction of Z and the assembly of E are challenging.

Let each domain compute concurrently ν_i vectors $\left\{\Lambda_{i_j}\right\}_{j=1}^{\nu_i}$.
Define local dense rectangular matrices:

$$W_i = \begin{bmatrix} D_i\Lambda_{i_1} & D_i\Lambda_{i_2} & \cdots & D_i\Lambda_{i_{\nu_i}} \end{bmatrix}.$$

Then, define the global deflation matrix as:

$$Z = \begin{bmatrix} R_1^T W_1 & R_2^T W_2 & \cdots & R_N^T W_N \end{bmatrix}$$

Generalized eigenvalue problems

For theoretical justification of Z, see (Spillane et al. 2011).
Solved by ARPACK concurently:

$$A_i^N \Lambda_j = \lambda_j D_i R_{i,0}^T R_{i,0} A_i^N D_i \Lambda_j$$

where

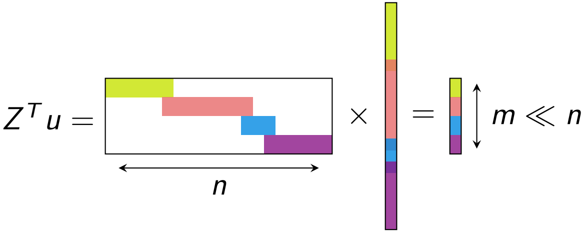
- A_i^N is the local unassembled matrix,
- $R_{i,0}$ is the restriction from Ω_i to $\Omega_i \bigcap \left(\bigcup_{j \in \mathcal{O}_i} \Omega_j \right)$.

Workflow during one coarse operator correction

How to compute $ZE^{-1}Z^Tu \in \mathbb{R}^n$?

Workflow during one coarse operator correction

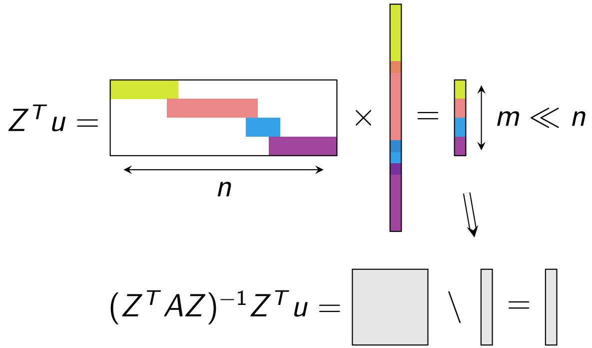
How to compute $Z^T u \in \mathbb{R}^n$?



operations & `MPI_Gather`

Workflow during one coarse operator correction

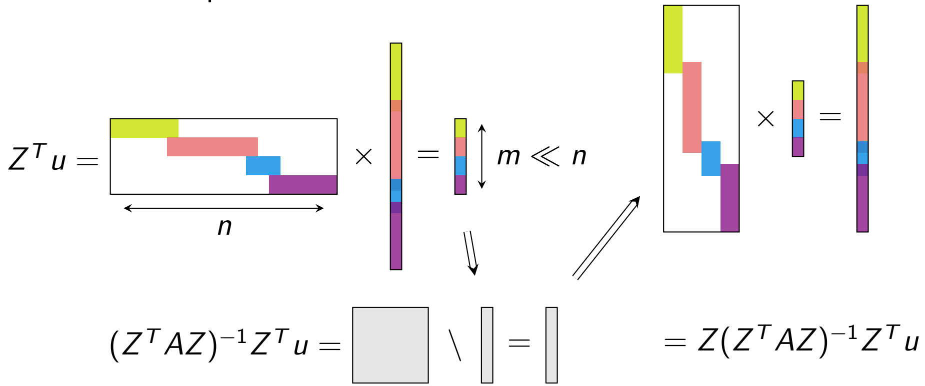
How to compute $E^{-1}Z^T u \in \mathbb{R}^n$?



operations & MPI_Gather + linear solve

Workflow during one coarse operator correction

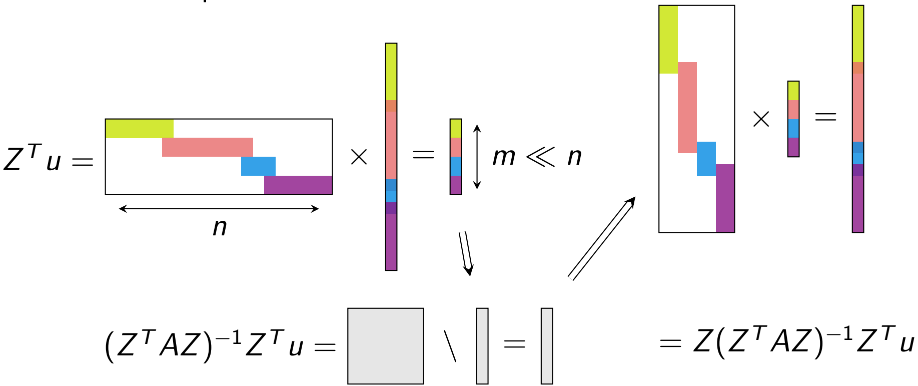
How to compute $ZE^{-1}Z^Tu \in \mathbb{R}^n$?



operations & `MPI_Gather` + linear solve + `MPI_Scatter` & operations

Workflow during one coarse operator correction

How to compute $ZE^{-1}Z^Tu \in \mathbb{R}^n$?

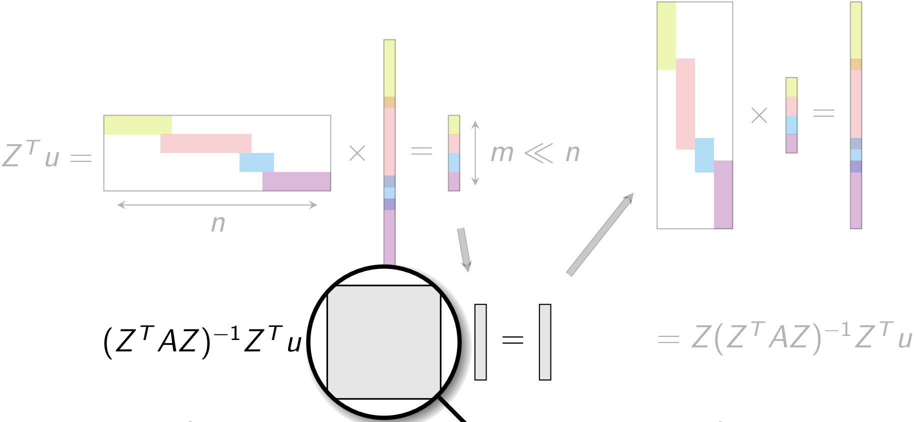


operations & `MPI_Gather` + linear solve + `MPI_Scatter` & operations

Communication pattern $\implies$ global reduction at the coarse level.

Workflow during one coarse operator correction

How to compute $ZE^{-1}Z^Tu \in \mathbb{R}^n$?



operations & MPI_Gather + linear solve + MPI_Scatter & operations

Communication pattern $\Longrightarrow$ global reduction at the coarse level.

Distribution of the coarse operator

How can one solve $E^{-1}z = c \in \mathbb{R}^m$?

Some constraints:

1. E cannot be centralized on a single MPI process,
2. E cannot be distributed on all MPI processes,
3. the solution must be computed <u>fast</u> and <u>reliably</u>.

Distribution of the coarse operator

How can one solve $E^{-1}z = c \in \mathbb{R}^m$?

Some constraints:

1. E cannot be centralized on a single MPI process,
2. E cannot be distributed on all MPI processes,
3. the solution must be computed <u>fast</u> and <u>reliably</u>.

$\Longrightarrow$ use a direct solver with a distributed matrix on few *master* processes (number chosen at runtime).

Assembly for Schwarz methods

Recalling $E = ZAZ^T$, it can be proven that the block (i, j)

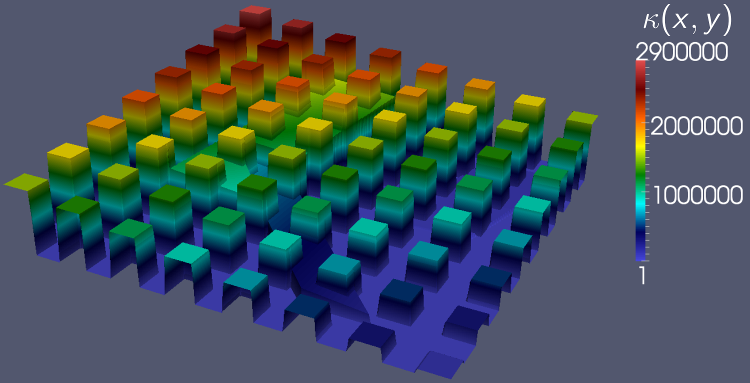
$$\begin{aligned} E_{ij} &= W_i^T A_{ij} W_j \\ &= W_i^T R_i R_j^T A_{jj} W_j \end{aligned}$$

1. compute locally $T_i = A_{ii} W_i$ ($\texttt{csrmm}$),
2. send to each neighbor, $S_j = R_j R_i^T T_i$,
3. receive from each neighbor $U_j = R_i R_j^T T_j$,
4. compute locally $E_{i,i} = W_i^T T_i$ ($\texttt{gemm}$),
5. compute locally $E_{i,j} = W_i^T U_j$ ($\texttt{gemm}$).

Note:
- steps 2 and 3 overlap with step 4,
- if $j \notin \mathcal{O}_i$, $R_i R_j^T = 0$.

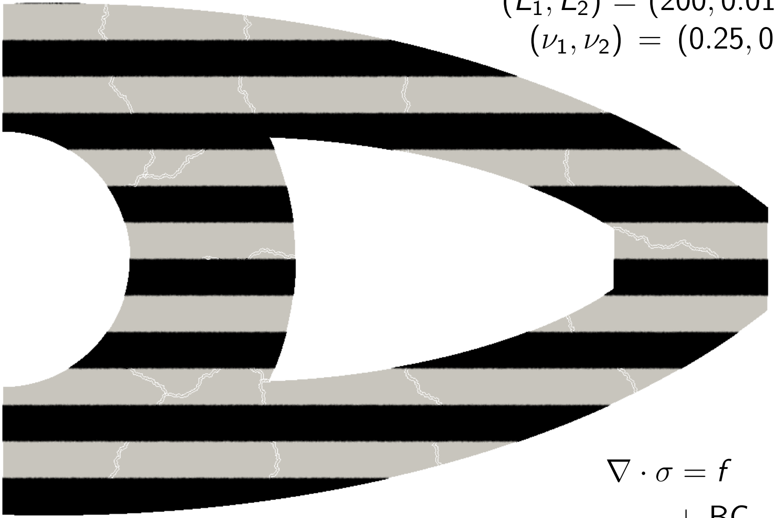
Example of heterogeneous coefficients

$$\nabla \cdot (\kappa \nabla u) = f$$
$$+ \text{BC}$$



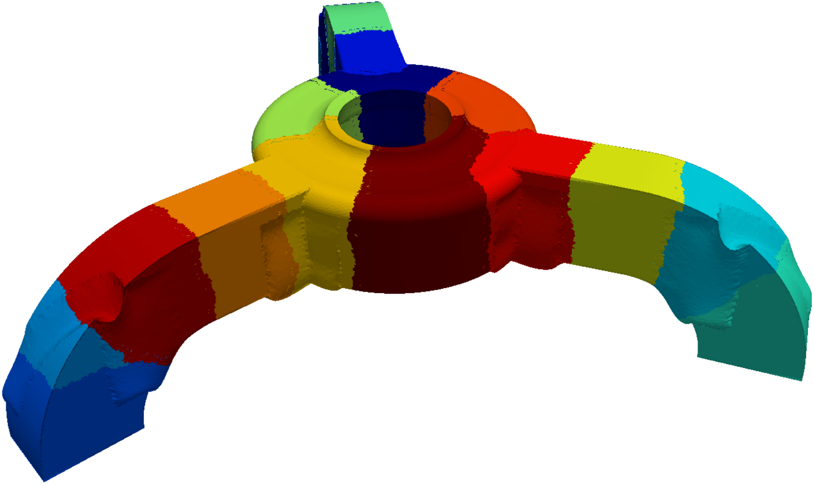
2D geometry

$$(E_1, E_2) = (200, 0.01) \text{ GPa}$$
$$(\nu_1, \nu_2) = (0.25, 0.45)$$



$$\nabla \cdot \sigma = f$$
$$+ \text{ BC}$$

3D geometry



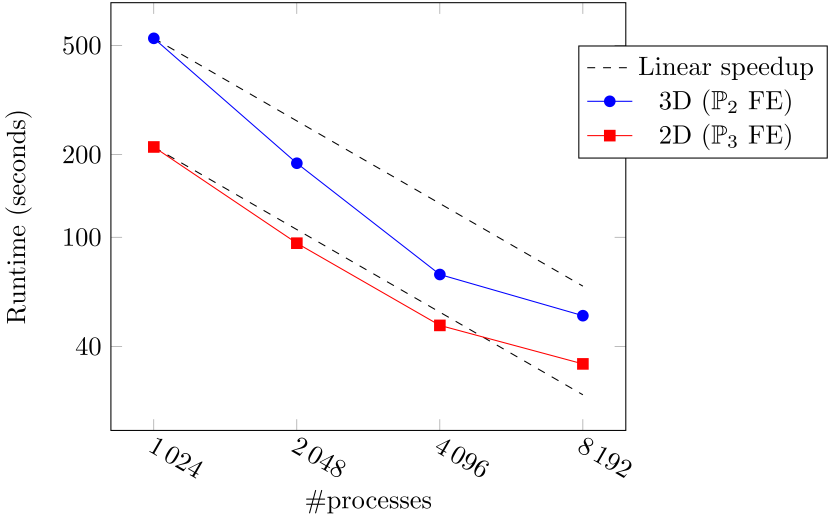
Machine used for scaling runs

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.

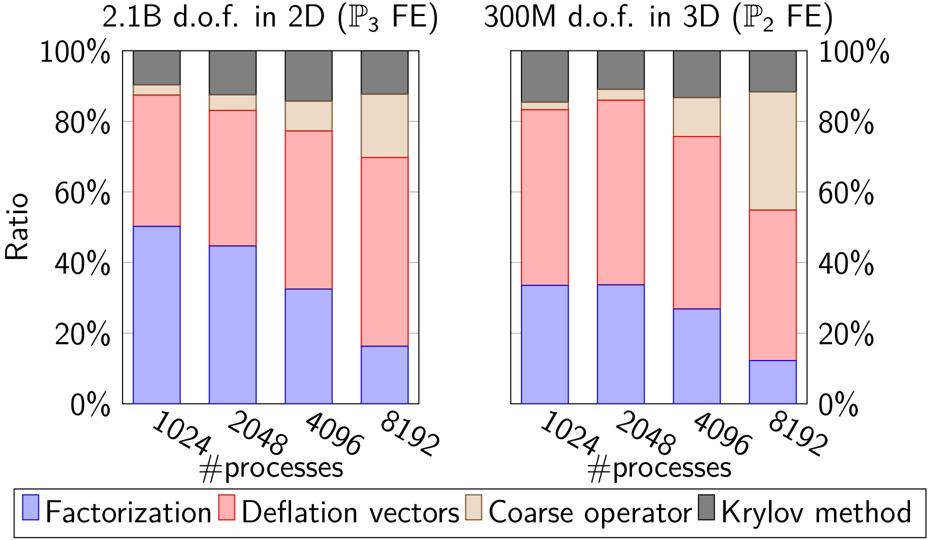
Strong scaling (linear elasticity)

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



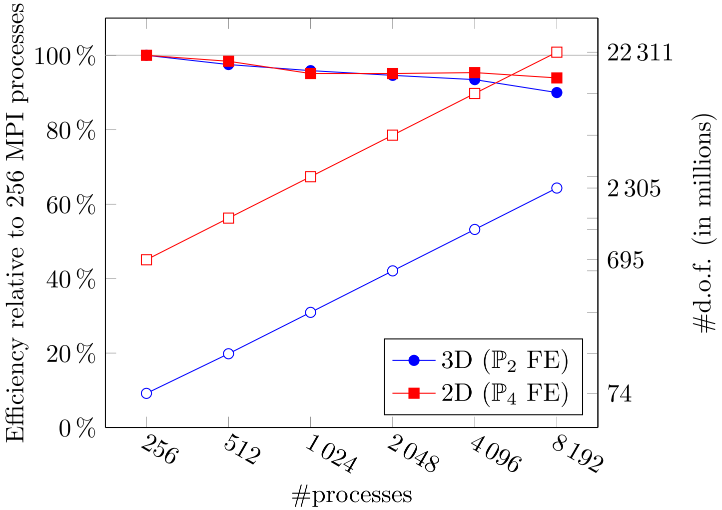
Strong scaling (linear elasticity)

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



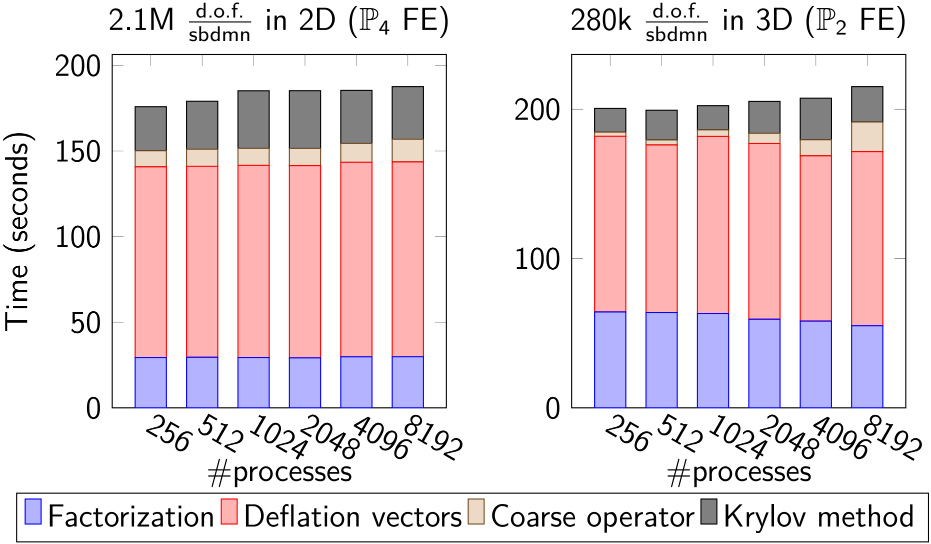
Weak scaling (scalar diffusion equation)

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



Weak scaling (scalar diffusion equation)

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



Distributed global matrix

Because there is overlap, the A_{ii} can be used to assemble A.

Local to global mapping $\implies$ distribution of the global matrix à la PETSc (split row-wise).

The mapping is computed via a double-sweep algorithm.

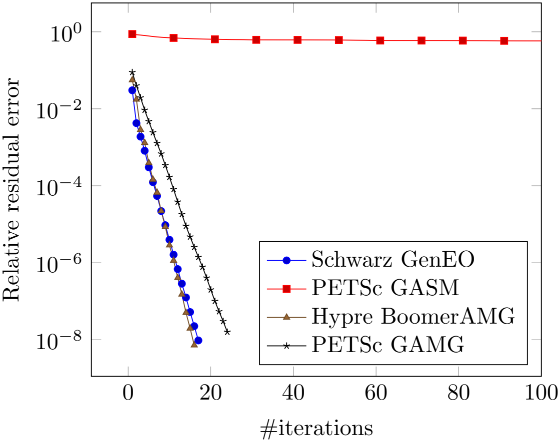
Comparison

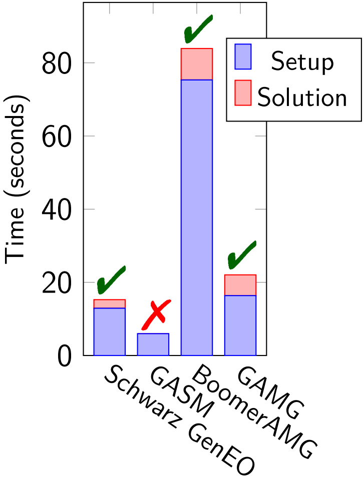
Comparing performance of setup and solution phases between our solver against purely algebraic (+ near null space) solvers:

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

Solution of a linear system I

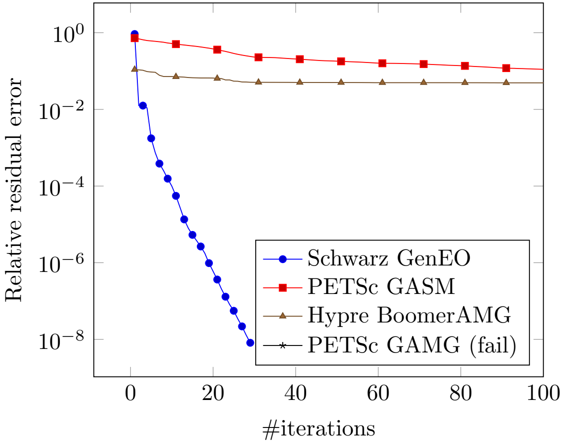
Homogeneous 3D Poisson equation discretized by $\mathbb{P}_1$ FE solved on 2,048 MPI processes, 111M d.o.f.

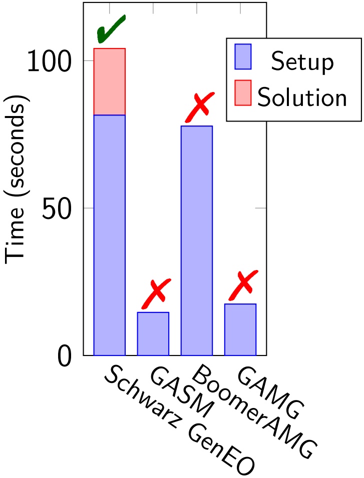




Solution of a linear system II

Heterogeneous 3D linear elasticity equation discretized by $\mathbb{P}_2$ FE solved on 2,048 MPI processes, 127M d.o.f.





Final words

Limitations:

- scaling of the coarse operator in 3D beyond 10k subdomains,
- deflation vectors need elementary matrices to be computed.

Summary:

- scalable framework for building two-level preconditioners for both Schwarz or substructuring methods (FETI-1),
- easily interfacable (FEM, FVM) without a global ordering.

Outlooks:

- adaptive (re)construction/recycling of the coarse operator,
- nonlinear and saddle point problems.

Final words

Limitations:

- scaling of the coarse operator in 3D beyond 10k subdomains,
- deflation vectors need elementary matrices to be computed.

Summary:

- scalable framework for building two-level preconditioners for both Schwarz or substructuring methods (FETI-1),
- easily interfacable (FEM, FVM) without a global ordering.

Outlooks:

- adaptive (re)construction/recycling of the coarse operator,
- nonlinear and saddle point problems.

Thank you !

Spillane, N., V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl (2011). "A robust two-level domain decomposition preconditioner for systems of PDEs". In: *Comptes Rendus Mathematique* 349.23, pp. 1255–1259.

Tang, J., R. Nabben, C. Vuik, and Y. Erlangga (2009). "Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods". In: *Journal of Scientific Computing* 39.3, pp. 340–370.

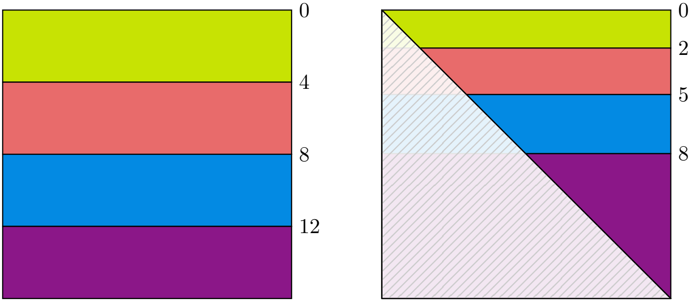
Solvers parameters

- Schwarz GenEO: $\nu_i = 20$, overlap $= 1$ (geometric).
- PETSc GASM: overlap $= 10$ (algebraic).
- Hypre BoomerAMG: HMIS coarsening, extended "classical" interpolation, no CF-relaxation, 2 levels of aggressive coarsening.
- PETSc GAMG: 1 smoothing step, `-mg_levels_ksp_type richardson -mg_levels_pc_type sor`.

OpenMPI bindings for hybrid runs:

`--bind-to-socket --bycore`.

Distribution of the coarse operator



Uniform distribution Non-uniform distribution

Distribution of E when built with $N = 16$ using 4 masters. On the right, the number of values per master is roughly the same if the values below the diagonal are dropped (symmetric coarse operator).

Timings for assembling the coarse operator

3D

N	P		$\dim(E)$		$\lvert\mathcal{O}_i\rvert$ (average)		Memory cost of "E^{-1}"		Time	
256	4		5120		11.5		38 MB		2.78 s	
512	6		10240		12.4		78 MB		3.35 s	
1024	8	8	20480	22528	13.0	12.0	156 MB	93 MB	4.42 s	11.25 s
2048	12	12	40960	40960	13.8	12.9	332 MB	138 MB	6.91 s	5.68 s
4096	18	22	73728	73728	14.2	13.7	434 MB	172 MB	10.75 s	8.04 s
8192	64	48	131072	131072	14.7	14.6	420 MB	241 MB	19.92 s	17.30 s

2D

N	P		$\dim(E)$		$\lvert\mathcal{O}_i\rvert$ (average)		Memory cost of "E^{-1}"		Time	
256	2		5376		5.5		21 MB		9.39 s	
512	4		10240		5.6		32 MB		9.96 s	
1024	10	8	20480	24576	5.7	5.5	65 MB	57 MB	9.92 s	10.14 s
2048	14	12	38912	40960	5.8	5.7	94 MB	83 MB	10.05 s	6.20 s
4096	22	18	81920	73728	5.9	5.8	99 MB	73 MB	10.87 s	5.10 s
8192	36	36	163840	122880	5.9	5.8	152 MB	118 MB	13.27 s	6.96 s

Strong scaling (linear elasticity)

	N	Factorization	Deflation	Solution	#it.	Total	#d.o.f.
3D	1 024	177.86 s	264.03 s	77.41 s	28	530.56 s	$293.98 \cdot 10^6$
	2 048	62.69 s	97.29 s	20.39 s	23	186.04 s	
	4 096	19.64 s	35.70 s	9.73 s	20	73.12 s	
	8 192	6.33 s	22.08 s	6.05 s	27	51.76 s	
2D	1 024	37.01 s	131.76 s	34.29 s	28	213.20 s	$2.14 \cdot 10^9$
	2 048	17.55 s	53.83 s	17.52 s	28	95.10 s	
	4 096	6.90 s	27.07 s	8.64 s	23	47.71 s	
	8 192	2.01 s	20.78 s	4.79 s	23	34.54 s	

Weak scaling (scalar diffusion equation)

	N	Factorization	Deflation	Solution	#it.	Total	#d.o.f.
3D	256	64.24 s	117.74 s	15.81 s	13	200.57 s	$74.62 \cdot 10^6$
	512	63.97 s	112.17 s	19.93 s	18	199.41 s	$144.70 \cdot 10^6$
	1 024	63.22 s	118.58 s	16.18 s	14	202.40 s	$288.80 \cdot 10^6$
	2 048	59.43 s	117.59 s	21.34 s	17	205.26 s	$578.01 \cdot 10^6$
	4 096	58.14 s	110.68 s	27.89 s	20	207.47 s	$1.15 \cdot 10^9$
	8 192	54.96 s	116.64 s	23.64 s	17	215.15 s	$2.31 \cdot 10^9$
2D	256	29.40 s	111.35 s	25.71 s	29	175.85 s	$695.96 \cdot 10^6$
	512	29.60 s	111.52 s	27.99 s	28	179.07 s	$1.39 \cdot 10^9$
	1 024	29.43 s	112.18 s	33.63 s	28	185.16 s	$2.79 \cdot 10^9$
	2 048	29.18 s	112.23 s	33.74 s	28	185.20 s	$5.58 \cdot 10^9$
	4 096	29.80 s	113.69 s	31.02 s	26	185.38 s	$11.19 \cdot 10^9$
	8 192	29.83 s	113.81 s	30.67 s	25	187.57 s	$22.31 \cdot 10^9$