OVERLAPPING DOMAIN DECOMPOSITION METHODS: NEW ALGEBRAIC COARSE SPACES

Pierre Jolivet, Hussam Al Daas

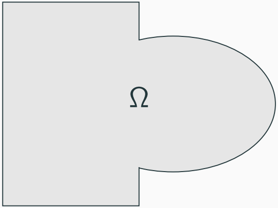
June 20, 2022

ICMS@Strathclyde: Solvers for frequency-domain wave problems and applications

Introduction

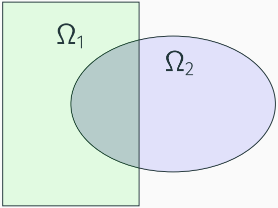
Overlapping Schwarz methods

Consider the linear system: $Ax = b \in \mathbb{K}^n$



Consider the linear system: $Ax = b \in \mathbb{K}^n$

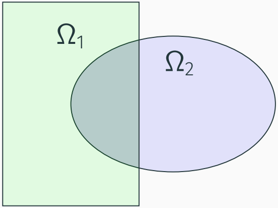
Two-way decomposition of $[\![1; n]\!]$ and restriction operators



Consider the linear system: $Ax = b \in \mathbb{K}^n$

Two-way decomposition of $[\![1; n]\!]$ and restriction operators

$$M_{\text{ASM}}^{-1} = \sum_{i=1}^{N=2} R_i^T (R_i A R_i^T)^{-1} R_i$$



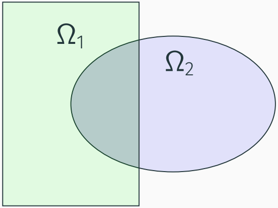
[Schwarz 1870]

Consider the linear system: $Ax = b \in \mathbb{K}^n$

Two-way decomposition of $[\![1; n]\!]$ and restriction operators

$$M_{\text{ASM}}^{-1} = \sum_{i=1}^{N=2} R_i^T (R_i A R_i^T)^{-1} R_i$$

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N=2} \tilde{R}_i^T (R_i A R_i^T)^{-1} R_i$$



[Schwarz 1870] [Cai and Sarkis 1999]

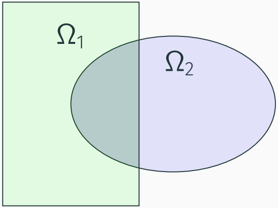
Overlapping Schwarz methods

Consider the linear system: $Ax = b \in \mathbb{K}^n$

Two-way decomposition of $[\![1; n]\!]$ and restriction operators

$$M_{\text{ASM}}^{-1} = \sum_{i=1}^{N=2} R_i^T (R_i A R_i^T)^{-1} R_i$$

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N=2} \tilde{R}_i^T (R_i A R_i^T)^{-1} R_i$$



[Schwarz 1870] [Cai and Sarkis 1999]

$\rightarrow$ no guaranteed scalability for large N

Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$

Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$
- define locally $W_i = \begin{bmatrix} \Lambda_1 & \cdots & \Lambda_{\gamma_i} \end{bmatrix}$

Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$
- define locally $W_i = \begin{bmatrix} \Lambda_1 & \cdots & \Lambda_{\gamma_i} \end{bmatrix}$
- use $Z = \begin{bmatrix} \tilde{R}_1^T W_1 & \cdots & \tilde{R}_N^T W_N \end{bmatrix}$ to compute $A_C = Z^T A Z$

Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$
- define locally $W_i = \begin{bmatrix} \Lambda_1 & \cdots & \Lambda_{\gamma_i} \end{bmatrix}$
- use $Z = \begin{bmatrix} \tilde{R}_1^T W_1 & \cdots & \tilde{R}_N^T W_N \end{bmatrix}$ to compute $A_C = Z^T A Z$
- coarse correction $M^{-1}_{\text{additive}} = Z A_C^{-1} Z^T + M^{-1}_{\text{RAS}}$

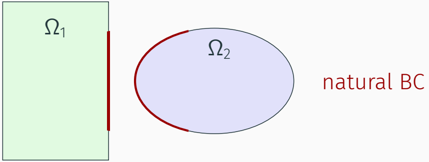
Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$
- define locally $W_i = \begin{bmatrix} \Lambda_1 & \cdots & \Lambda_{\gamma_i} \end{bmatrix}$
- use $Z = \begin{bmatrix} \tilde{R}_1^T W_1 & \cdots & \tilde{R}_N^T W_N \end{bmatrix}$ to compute $A_C = Z^T A Z$
- coarse correction $M_{\text{additive}}^{-1} = Z A_C^{-1} Z^T + M_{\text{RAS}}^{-1}$

How to choose N_i?

- unassembled matrices



Coarse correction

GenEO in a nutshell [Spillane et al. 2013]

- solve concurrently $N_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$
- define locally $W_i = \begin{bmatrix} \Lambda_1 & \cdots & \Lambda_{\gamma_i} \end{bmatrix}$
- use $Z = \begin{bmatrix} \tilde{R}_1^T W_1 & \cdots & \tilde{R}_N^T W_N \end{bmatrix}$ to compute $A_C = Z^T A Z$
- coarse correction $M_{\text{additive}}^{-1} = Z A_C^{-1} Z^T + M_{\text{RAS}}^{-1}$

How to choose N_i?

- unassembled matrices
- local SPSD splitting [Al Daas and Grigori 2019]

Coarse correction formulae

Given a fine-level preconditioner $M_\star^{-1}$ and $Q = ZA_C^{-1}Z^T$

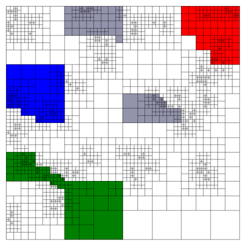
$$
\begin{aligned}
M_{\text{additive}}^{-1} &= Q + M_\star^{-1} \\
M_{\text{balanced}}^{-1} &= Q + (I - AQ)^T M_\star^{-1} (I - AQ) \\
M_{\text{deflated}}^{-1} &= Q + M_\star^{-1} (I - AQ)
\end{aligned}
$$

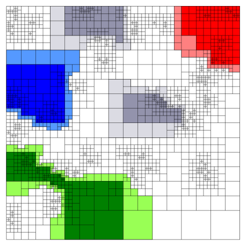
HPDDM and PETSc

- specific implementation [Jolivet, Nataf, et al. 2013]

HPDDM and PETSc

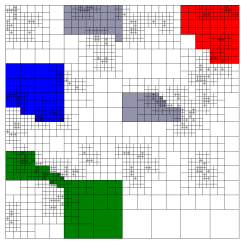
- specific implementation [Jolivet, Nataf, et al. 2013]
- interfaced in PETSc [Jolivet, Roman, and Zampini 2021]
  - user-provided N_i (`PCHPDDMSetAuxiliaryMat`)
  - automatically assembled on `DMPlex`

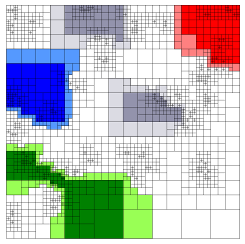




HPDDM and PETSc

- specific implementation [Jolivet, Nataf, et al. 2013]
- interfaced in PETSc [Jolivet, Roman, and Zampini 2021]
  - user-provided N_i (`PCHPDDMSetAuxiliaryMat`)
  - automatically assembled on `DMPlex`

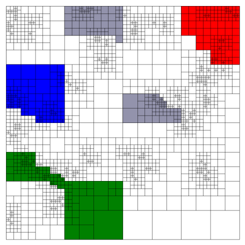


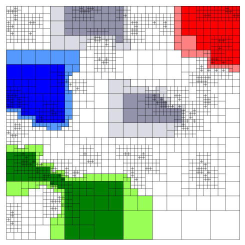


  - different RHS (`PCHPDDMSetRHSMat`) [Bootland et al. 2021]

HPDDM and PETSc

- specific implementation [Jolivet, Nataf, et al. 2013]
- interfaced in PETSc [Jolivet, Roman, and Zampini 2021]
  - user-provided N_i (`PCHPDDMSetAuxiliaryMat`)
  - automatically assembled on `DMPlex`





  - different RHS (`PCHPDDMSetRHSMat`) [Bootland et al. 2021]

→ still low-level and *not* algebraic

Outline

Introduction

Sparse least squares

General sparse problems

When coarse spaces are not needed

Conclusion

Sparse least squares

Abstract setting

Rectangular matrices

- overdetermined linear systems $Ax = b$
- generalized singular value decompositions (A, B)

$$U_A^T A G = C \qquad U_B^T B G = S$$

Abstract setting

Rectangular matrices

- overdetermined linear systems $Ax = b$
- generalized singular value decompositions (A, B)

$$U_A^T A G = C \qquad U_B^T B G = S$$

Solvers

- direct: QR factorization
- iterative:
  - LSQR [Paige and Saunders 1982]
  - normal equations

Abstract setting

Rectangular matrices

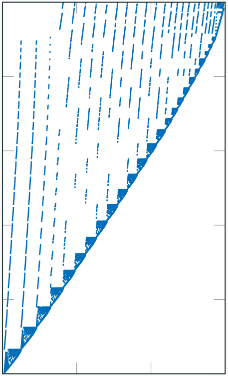
- overdetermined linear systems $Ax = b$
- generalized singular value decompositions (A, B)

$$U_A^T A G = C \qquad U_B^T B G = S$$

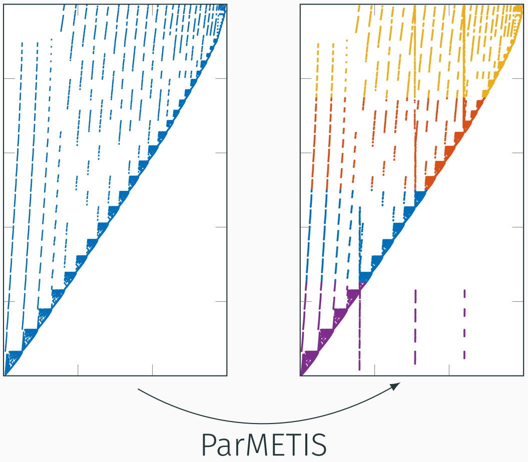
Solvers

- direct: QR factorization
- iterative:
  - LSQR [Paige and Saunders 1982]
  - normal equations $\quad \rightarrow$ preconditioner for $A^T A$

mk13-b5 from the SSMC

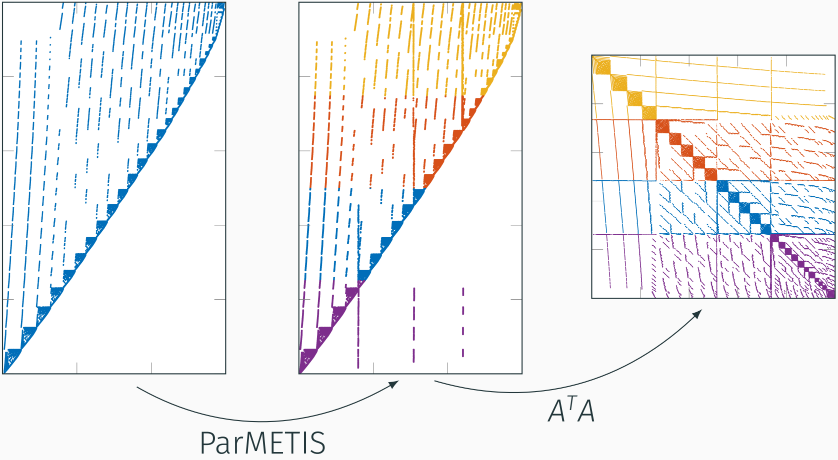


mk13-b5 from the SSMC on 4 processes



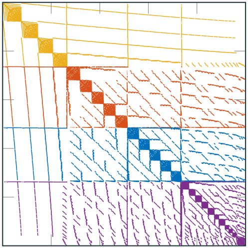
Academic example

mk13-b5 from the SSMC on 4 processes



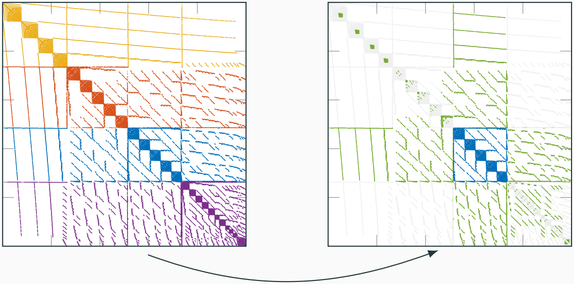
Preconditioning $A^T A$

- AMG
- (incomplete) Cholesky



Preconditioning $A^T A$

- AMG
- (incomplete) Cholesky
- domain decomposition



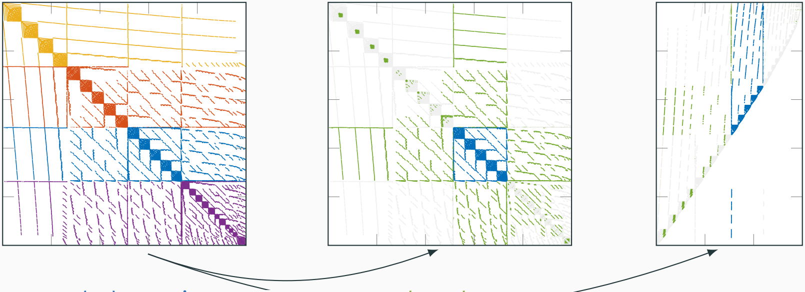
Subdomain #3 + one overlap layer

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N=4} \tilde{R}_i^T (R_i A^T A R_i^T)^{-1} R_i$$

Cholesky decompositions

Preconditioning $A^T A$

- AMG
- (incomplete) Cholesky
- domain decomposition

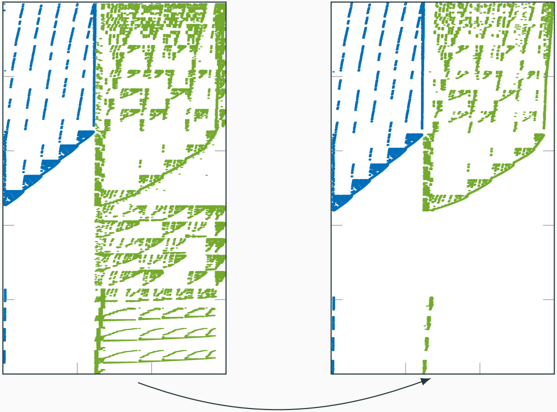


Subdomain #3 + one overlap layer

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N=4} \tilde{R}_i^T (R_i A^T A R_i^T)^{-1} R_i$$

QR decompositions

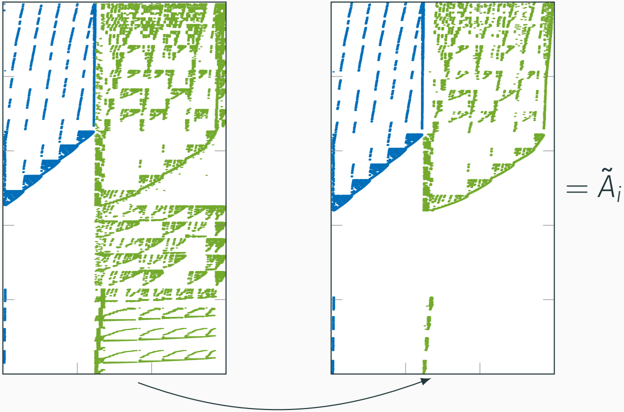
SPSD SPLITTING



$= \tilde{A}_i$

Zero rows on the overlap that are zero in the subdomain

SPSD SPLITTING

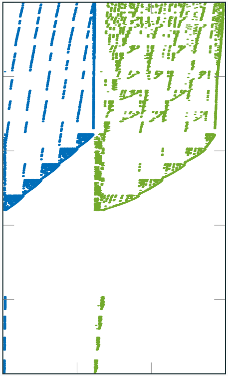


Zero rows on the overlap that are zero in the subdomain

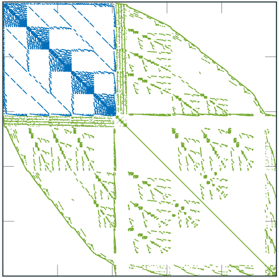
- solve concurrently $\tilde{A}_i^T \tilde{A}_i \Lambda_k = \lambda_k \tilde{R}_i A^T A \tilde{R}_i^T \Lambda_k$
- theory in [Al Daas, Jolivet, and Scott 2022]

Leveraging the flexibility of SLEPc/PETSc

- ST for solving $\tilde{A}_i^T\tilde{A}_i\Lambda_k = \lambda_k\tilde{R}_iA^TA\tilde{R}_i^T\Lambda_k$



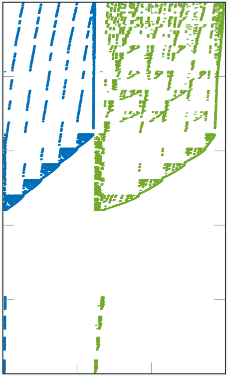
QR factorization of $\tilde{A}_i$



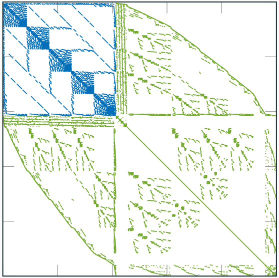
LL^T factorization of $\tilde{A}_i^T\tilde{A}_i$

Leveraging the flexibility of SLEPc/PETSc

- ST for solving $\tilde{A}_i^T\tilde{A}_i\Lambda_k = \lambda_k\tilde{R}_iA^TA\tilde{R}_i^T\Lambda_k$



QR factorization of $\tilde{A}_i$



LL^T factorization of $\tilde{A}_i^T\tilde{A}_i$

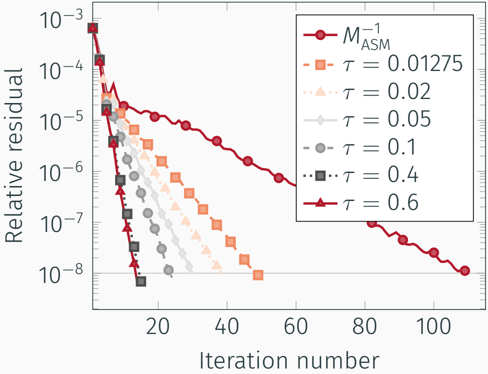
- symbolic factorization reused for the fine-level correction

Numerical results on 256 processes

Identifier	M^{-1}_{balanced}	BoomerAMG	GAMG
mesh_deform	**13**		35
EternityII_E	**43**		63
lp_stocfor3	**34**		513
deltaX	**23**		784
sc205-2r	**54**		195
stormg2-125	**42**		
Rucci1	**21**	118	364
image_interp	**11**	40	203
mk13-b5	19	**11**	
pds-100	18	**16**	35
fome21	20	**16**	20
sgpf5y6	224		**163**
Hardesty2	**30**	88	404
Delor338K	**10**		
watson_2	**15**		64
LargeRegFile	41	**19**	
cont11_l	**30**	53	723

Adaptive preconditioning

watson_2 from the SSMC on 256 processes



τ	n_C	Iterations
0.01275	2,400	49
0.02	2,683	39
0.05	3,049	30
0.1	3,337	24
0.4	8,979	15
0.6	30,246	14

Strong scaling

Hardesty3 ($m = 8{,}217{,}820$ and $n = 7{,}591{,}564$)

N	Iterations	Eigensolve	Setup	Solve	n_C	Total	Speedup
16	113	2,417.4	24.5	301.3	4,800	2,743.2	—
32	117	1,032.7	14.1	154.2	9,600	1,201.0	2.3
64	129	887.2	11.4	112.3	19,200	1,010.9	2.7
128	144	224.1	6.9	55.4	38,400	286.3	9.6
256	97	128.0	6.7	32.2	76,800	166.9	16.4
512	87	45.5	13.0	26.9	153,391	85.3	32.2
1,024	85	23.8	20.2	35.3	303,929	79.3	34.6
2,048	55	14.6	31.4	43.2	497,704	89.1	30.8
4,096	59	11.7	30.8	44.9	695,774	87.3	31.4

Extension to SPD matrices

- SPSD splitting $\tilde{A}_i$ of A from SPSD splitting of A^2 $(= A^T A)$

$$\tilde{X}_i = \begin{bmatrix} A_{Ii} & A_{I\Gamma i} & \\ A_{\Gamma Ii} & A_{\Gamma i} & A_{\Gamma\Delta i} \end{bmatrix}$$

Extension to SPD matrices

- SPSD splitting $\tilde{A}_i$ of A from SPSD splitting of A^2 $(= A^T A)$

$$\tilde{X}_i = \begin{bmatrix} A_{Ii} & A_{I\Gamma i} & \\ A_{\Gamma Ii} & A_{\Gamma i} & A_{\Gamma\Delta i} \end{bmatrix}$$

- concurrent economy-size SVD

Extension to SPD matrices

- SPSD splitting $\tilde{A}_i$ of A from SPSD splitting of A^2 $(= A^T A)$

$$\tilde{X}_i = \begin{bmatrix} A_{Ii} & A_{I\Gamma i} & \\ A_{\Gamma Ii} & A_{\Gamma i} & A_{\Gamma\Delta i} \end{bmatrix}$$

- concurrent economy-size SVD
- inverse of $\tilde{A}_i$ as a function of $\tilde{X}_i = U_i \Sigma_i V_i^T$

Extension to SPD matrices

- SPSD splitting $\tilde{A}_i$ of A from SPSD splitting of A^2 $(= A^T A)$

$$\tilde{X}_i = \begin{bmatrix} A_{Ii} & A_{I\Gamma i} & \\ A_{\Gamma Ii} & A_{\Gamma i} & A_{\Gamma\Delta i} \end{bmatrix}$$

- concurrent economy-size SVD
- inverse of $\tilde{A}_i$ as a function of $\tilde{X}_i = U_i \Sigma_i V_i^T$

$$\tilde{A}_i u = v \impliedby \underline{\tilde{A}_i} \begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} v \\ 0 \end{bmatrix}$$

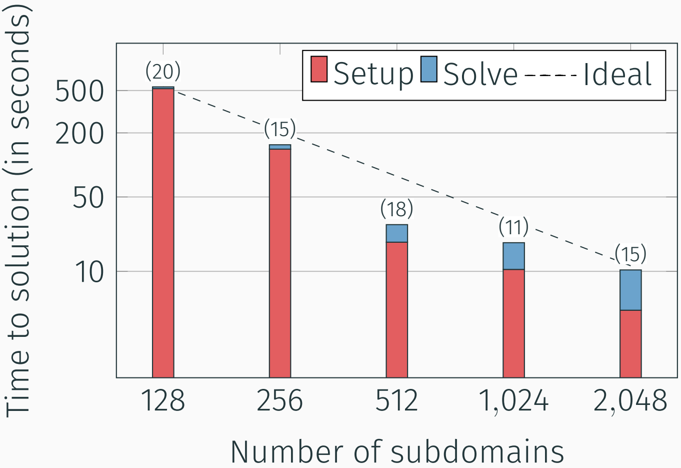
$$\widetilde{\underline{A}_i}^{-1} = V_i(\Sigma_i + \sigma_{1i}\varepsilon I)^{-1} V_i^\top + \sigma_{1i}^{-1}\varepsilon^{-1}(I - V_i V_i^\top).$$

- [Al Daas and Jolivet 2022]

Numerical results on 256 processes

Identifier	M^{-1}_{deflated}	BoomerAMG	GAMG
s3rmt3m3	**4**		
vanbody	**18**		
nasasrb	**10**		
Dubcova2	**5**	76	56
s3dkt3m2	**49**		
shipsec8	**7**		
ship_003	**9**		
boneS01	**16**		
bmwcra_1	**20**		
G2_circuit	19	**11**	26
pwtk	**47**		
af_4_k101	**18**		
parabolic_fem	17	**8**	16
tmt_sym	14	**10**	17
ecology2	45	**12**	18

Strong scaling



- tmt_sym with an increasing number of MPI processes
- slope of −3 represents ideal linear decrease

Multilevel results

Identifier	Outer iterations	Inner iterations	n	$n_{C,2}$	$n_{C,3}$	GC
s3rmt3m3	4	10	5,357	5,321	2,240	2.41
parabolic_fem	17	3	525,825	21,736	3,838	1.05

Multilevel results

Identifier	Outer iterations	Inner iterations	n	$n_{C,2}$	$n_{C,3}$	GC
s3rmt3m3	4	10	5,357	5,321	2,240	2.41
parabolic_fem	17	3	525,825	21,736	3,838	1.05

Identifier	Outer iterations	Inner iterations	n	$n_{C,2}$	$n_{C,3}$
diffusion	11	5	4,173,281	60,144	12,040
elasticity	8	11	30,633,603	14,880	5,120

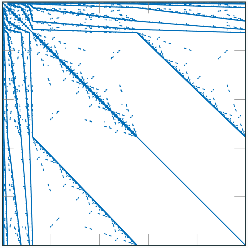
General sparse problems

Abstract setting

- limited theory for nonsymmetric preconditioning
- SPSD splitting for SPD diagonally dominant matrices
- theory in [Al Daas, Jolivet, and Rees 2022]

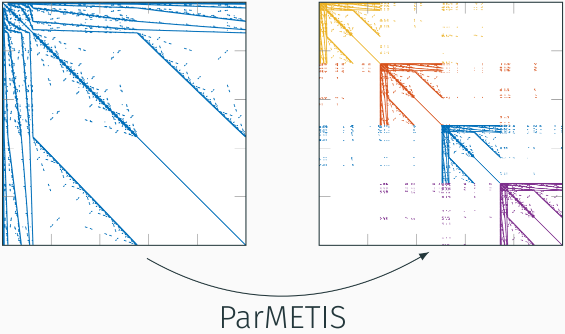
Academic example

Dubcova3 from the SSMC



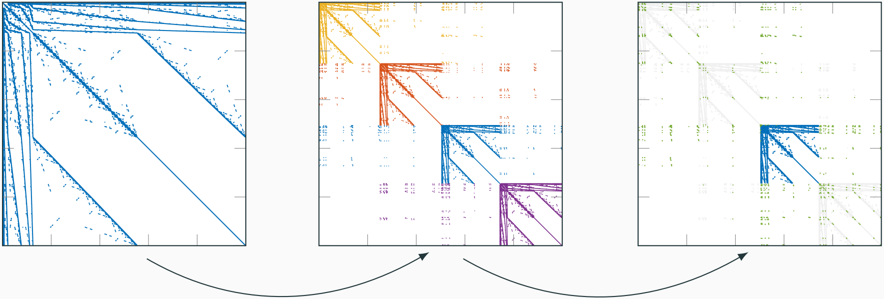
Academic example

Dubcova3 from the SSMC on 4 processes



Academic example

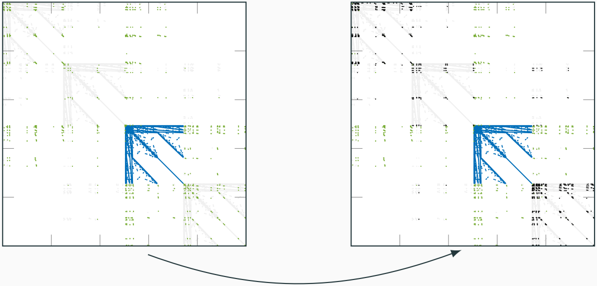
Dubcova3 from the SSMC on 4 processes



ParMETIS

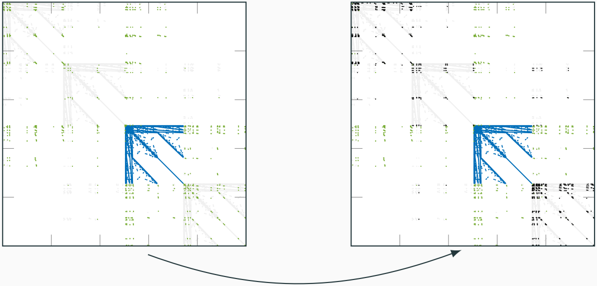
Subdomain #3 +
one overlap layer

SPSD SPLITTING



Diagonal lumping on the overlap with off-process coefficients

SPSD splitting



Diagonal lumping on the overlap with off-process coefficients

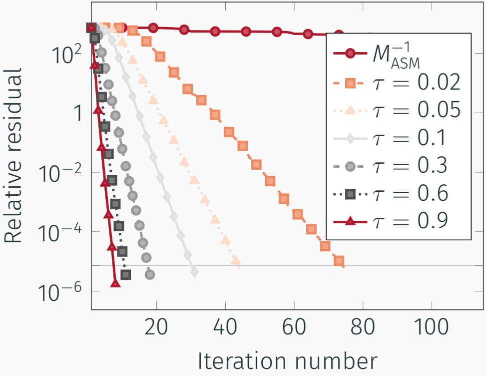
- define $\tilde{A}_i$ as $R_i A R_i^T$ lumped for overlapping unknowns
- solve concurrently $\tilde{A}_i \Lambda_k = \lambda_k \tilde{R}_i A \tilde{R}_i^T \Lambda_k$

Numerical results on 256 processes

Identifier	M^{-1}_{deflated}	BoomerAMG	GAMG
light_in_tissue	**6**		53
finan512	**6**	7	8
consph	**93**		
Dubcova3	**7**	72	71
CO	26	**25**	
nxp1	**20**		
CoupCons3D	**20**		26
parabolic_fem	**5**	8	16
Chevron4	**5**		
apache2	**8**	11	35
tmt_sym	**5**	10	17
tmt_unsym	**6**	13	18
ecology2	**6**	12	18
thermal2	26	**14**	20
atmosmodj	**7**	8	17
G3_circuit	**8**	12	35
Transport	**9**	10	98
memchip	36	**15**	
circuit5M_dc	7	**5**	

Adaptive preconditioning

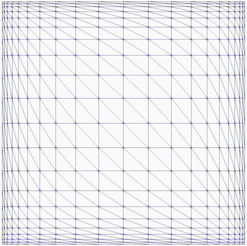
G3_circuit from the SSMC on 256 processes



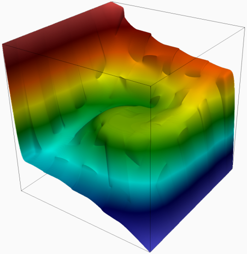
τ	n_0	Iterations
0.02	739	75
0.05	1,831	44
0.1	3,608	31
0.3	11,784	18
0.6	34,402	11
0.9	71,385	8

Convection–diffusion equation

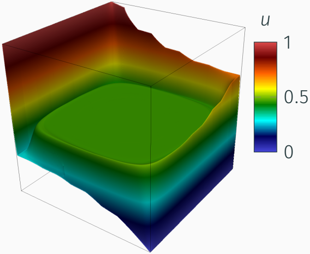
$$\nabla\cdot(Vu) - \nu\nabla\cdot(\kappa\nabla u) = 0 \text{ in } \Omega \qquad u = 0 \text{ in } \Gamma_0 \qquad u = 1 \text{ in } \Gamma_1$$



Ω



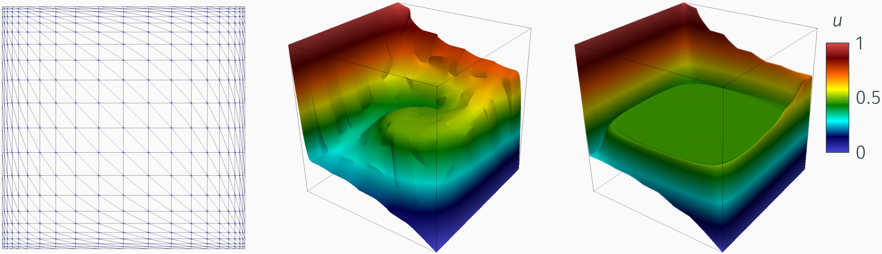
$\nu = 10^{-2}$



$\nu = 10^{-4}$

Convection–diffusion equation

$\nabla\cdot(Vu) - \nu\nabla\cdot(\kappa\nabla u) = 0 \text{ in } \Omega \qquad u = 0 \text{ in } \Gamma_0 \qquad u = 1 \text{ in } \Gamma_1$



Ω $\qquad$ $\nu = 10^{-2}$ $\qquad$ $\nu = 10^{-4}$

Dimension	k	N	n	ν = 1	ν = 10^{-2}	ν = 10^{-4}
2	1	1,024	$6.3 \cdot 10^6$	23 (52,875)	19 (52,759)	21 (28,235)
3	2	4,096	$8.1 \cdot 10^6$	18 ($1.8 \cdot 10^5$)	11 ($1.6 \cdot 10^5$)	29 (76,853)

Structure-aware lumping

- `MatGetRowSum` used in `PCHPDDM` for diagonal lumping

Structure-aware lumping

- `MatGetRowSum` used in `PCHPDDM` for diagonal lumping
- vector finite elements group multiple unknowns
- no `MatGetBlockRowSum`, e.g., $bs = 3$ for 2D elasticity

Structure-aware lumping

- `MatGetRowSum` used in `PCHPDDM` for diagonal lumping
- vector finite elements group multiple unknowns
- no `MatGetBlockRowSum`, e.g., $bs = 3$ for 2D elasticity
- MFEM ex2p with a call to `MatSetBlockSize`

Dimension	k	N	n	Iterations
2	4	1,024	$1.8 \cdot 10^6$	15 (77,378)
3	3	1,024	$2 \cdot 10^6$	34 ($1.5 \cdot 10^5$)

When coarse spaces are not needed

Block Krylov methods

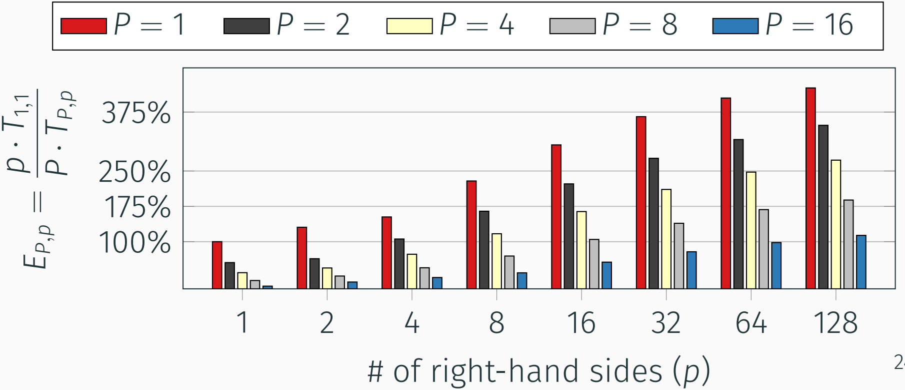
Performance

- higher arithmetic intensity
- fewer synchronizations with more data
- (sometimes) faster convergence

Block Krylov methods

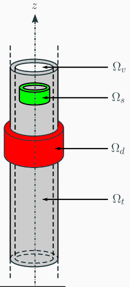
Performance

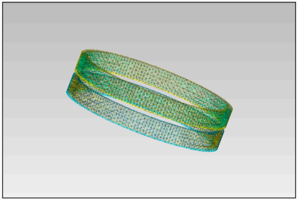
- higher arithmetic intensity
- fewer synchronizations with more data
- (sometimes) faster convergence



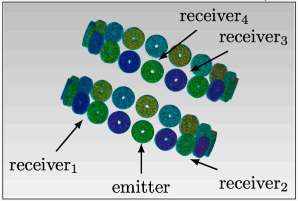
Eddy-current testing I

- stewardship of SG inside PWR
- inverse problem for reconstructing deposits
- Helmholtz equation with many RHSs





SAX probe



SMX probe

Eddy-current testing II

- block Krylov methods are memory-demanding
- split a single large block into multiple smaller blocks
- use recycling w/ blocking [Audibert et al. 2020]

$$AX = B \implies A \begin{bmatrix} X_1 & \cdots & X_p \end{bmatrix} = \begin{bmatrix} B_1 & \cdots & B_p \end{bmatrix}$$

Krylov method	p	# of RHS/p	Time	/RHS	Speedup
GMRES(40)	779	1	1.8 h	8.3 sec	—

Eddy-current testing II

- block Krylov methods are memory-demanding
- split a single large block into multiple smaller blocks
- use recycling w/ blocking [Audibert et al. 2020]

$$AX = B \implies A \begin{bmatrix} X_1 & \cdots & X_p \end{bmatrix} = \begin{bmatrix} B_1 & \cdots & B_p \end{bmatrix}$$

Krylov method	p	# of RHS/p	Time	/RHS	Speedup
GMRES(40)	779	1	1.8 h	8.3 sec	—
GCRODR(40, 5)	779	1	1.4 h	6.4 sec	1.3

Eddy-current testing II

- block Krylov methods are memory-demanding
- split a single large block into multiple smaller blocks
- use recycling w/ blocking [Audibert et al. 2020]

$$AX = B \implies A\begin{bmatrix}X_1 & \cdots & X_p\end{bmatrix} = \begin{bmatrix}B_1 & \cdots & B_p\end{bmatrix}$$

Krylov method	p	# of RHS/p	Time	/RHS	Speedup
GMRES(40)	779	1	1.8 h	8.3 sec	—
GCRODR(40, 5)	779	1	1.4 h	6.4 sec	1.3
BGMRES(30)	8	98	3.8 min	0.29 sec	28.6
	4	195	3.9 min	0.30 sec	27.6

Eddy-current testing II

- block Krylov methods are memory-demanding
- split a single large block into multiple smaller blocks
- use recycling w/ blocking [Audibert et al. 2020]

$$AX = B \implies A \begin{bmatrix} X_1 & \cdots & X_p \end{bmatrix} = \begin{bmatrix} B_1 & \cdots & B_p \end{bmatrix}$$

Krylov method	p	# of RHS/p	Time	/RHS	Speedup
GMRES(40)	779	1	1.8 h	8.3 sec	—
GCRODR(40, 5)	779	1	1.4 h	6.4 sec	1.3
BGMRES(30)	8	98	3.8 min	0.29 sec	28.6
	4	195	3.9 min	0.30 sec	27.6
BGCRODR(30, 1)	8	98	2.6 min	0.20 sec	41.5
	4	195	3.1 min	0.24 sec	34.6

Conclusion

Final words

- new algebraic coarse spaces for Schwarz methods
- available in PETSc with `--download-hpddm`
- investigate other problems: BEM, saddle point...

Final words

- new algebraic coarse spaces for Schwarz methods
- available in PETSc with `--download-hpddm`
- investigate other problems: BEM, saddle point…

Thank you!

References I

Al Daas, Hussam and Laura Grigori (2019). "A class of efficient locally constructed preconditioners based on coarse spaces". In: *SIAM Journal on Matrix Analysis and Applications* 40.1, pp. 66–91.

Al Daas, Hussam and Pierre Jolivet (2022). "A robust algebraic multilevel domain decomposition preconditioner for general sparse symmetric positive definite matrices". In: *SIAM Journal on Scientific Computing*, submitted for publication. URL: `https://arxiv.org/pdf/2109.05908.pdf`.

Al Daas, Hussam, Pierre Jolivet, and Tyrone Rees (2022). "Efficient algebraic two-level Schwarz preconditioner for sparse matrices". In: *SIAM Journal on Scientific Computing*, submitted for publication. URL: `https://arxiv.org/pdf/2201.02250.pdf`.

Al Daas, Hussam, Pierre Jolivet, and Jennifer A. Scott (2022). "A robust algebraic domain decomposition preconditioner for sparse normal equations". In: *SIAM Journal on Scientific Computing* 44 (3), A1047–A1068. URL: `https://github.com/prj-/aldaas2021robust`.

References II

Audibert, Lorenzo, Hugo Girardon, Houssem Haddar, and Pierre Jolivet (2020). "Inversion of eddy-current signals using a level-set method and block Krylov solvers". In: *SIAM Journal on Scientific Computing*, submitted for publication. URL: `https://hal.archives-ouvertes.fr/hal-03043491`.

Bootland, Niall, Victorita Dolean, Pierre Jolivet, and Pierre-Henri Tournier (2021). "A comparison of coarse spaces for Helmholtz problems in the high-frequency regime". In: *Computers & Mathematics with Applications* 98, pp. 239–253.

Cai, Xiao-Chuan and Marcus Sarkis (1999). "A restricted additive Schwarz preconditioner for general sparse linear systems". In: *SIAM Journal on Scientific Computing* 21.2, pp. 792–797.

Jolivet, Pierre, Frédéric Hecht, Frédéric Nataf, and Christophe Prud'homme (2013). "Scalable domain decomposition preconditioners for heterogeneous elliptic problems". In: *Proceedings of the 2013 International Conference on High Performance Computing, Networking, Storage and Analysis*. SC13. ACM, 80:1–80:11.

References III

- Jolivet, Pierre, Jose E. Roman, and Stefano Zampini (2021). "KSPHPDDM and PCHPDDM: extending PETSc with robust overlapping Schwarz preconditioners and advanced Krylov methods". In: *Computers & Mathematics with Applications* 84, pp. 277–295. URL: `https://github.com/prj-/jolivet2020petsc`.
- Paige, Christopher C. and Michael A. Saunders (1982). "LSQR: an algorithm for sparse linear equations and sparse least squares". In: *ACM Transactions on Mathematical Software* 8.1, pp. 43–71.
- Schwarz, Hermann (1870). "Über einen Grenzübergang durch alternierendes Verfahren". In: *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich* 15, pp. 272–286.
- Spillane, Nicole, Victorita Dolean, Patrice Hauret, Frédéric Nataf, Clemens Pechstein, and Robert Scheichl (2013). "Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps". In: *Numerische Mathematik* 126.4, pp. 741–770.