Parallel Multilevel Low-Rank approximation preconditioners

Yousef Saad

Department of Computer Science and Engineering

University of Minnesota

Modelling 2019

Olomouc, Sept. 18, 2019

First:

- Joint work with Yuanzhe Xi [to Join Emory University] ...

... and:

- Ruipeng Li, Geoffrey Dillon, Vasilis Kalantzis, Tianshi Xu

- Work supported by NSF-DMS

Preconditioners in 'algebraic' DD context

Common framework: Partition mesh, 'distribute' matrix, then exploit a form of Schwarz technique ...

... or a form of 'approximate' Schur complement technique

➤ In recent years: many researchers have discovered the importance of some form of 'low-rank correction'

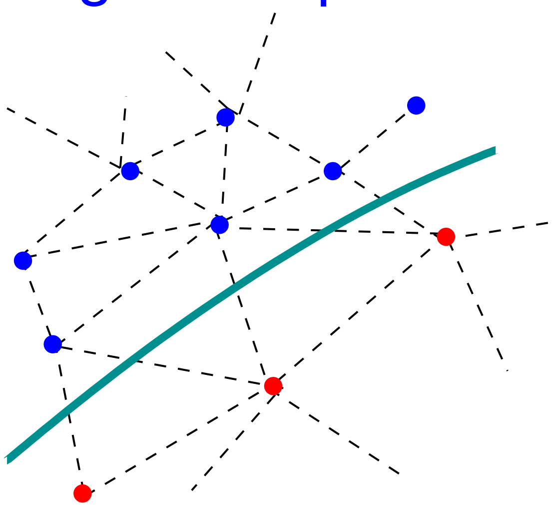
➤ aka 'deflation', aka 'SA', ...

➤ This talk: Our work in LR correction techniques

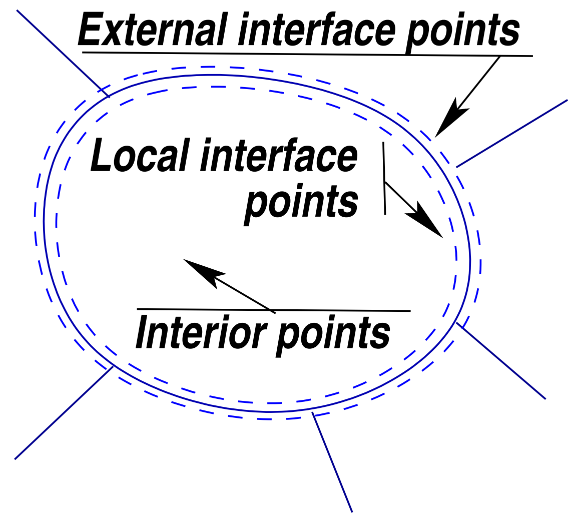
Schur complement + low-rank correction techniques

- Algebraic DD: Partition graph using ‘edge separation’:

Edge Separation:



Local view:



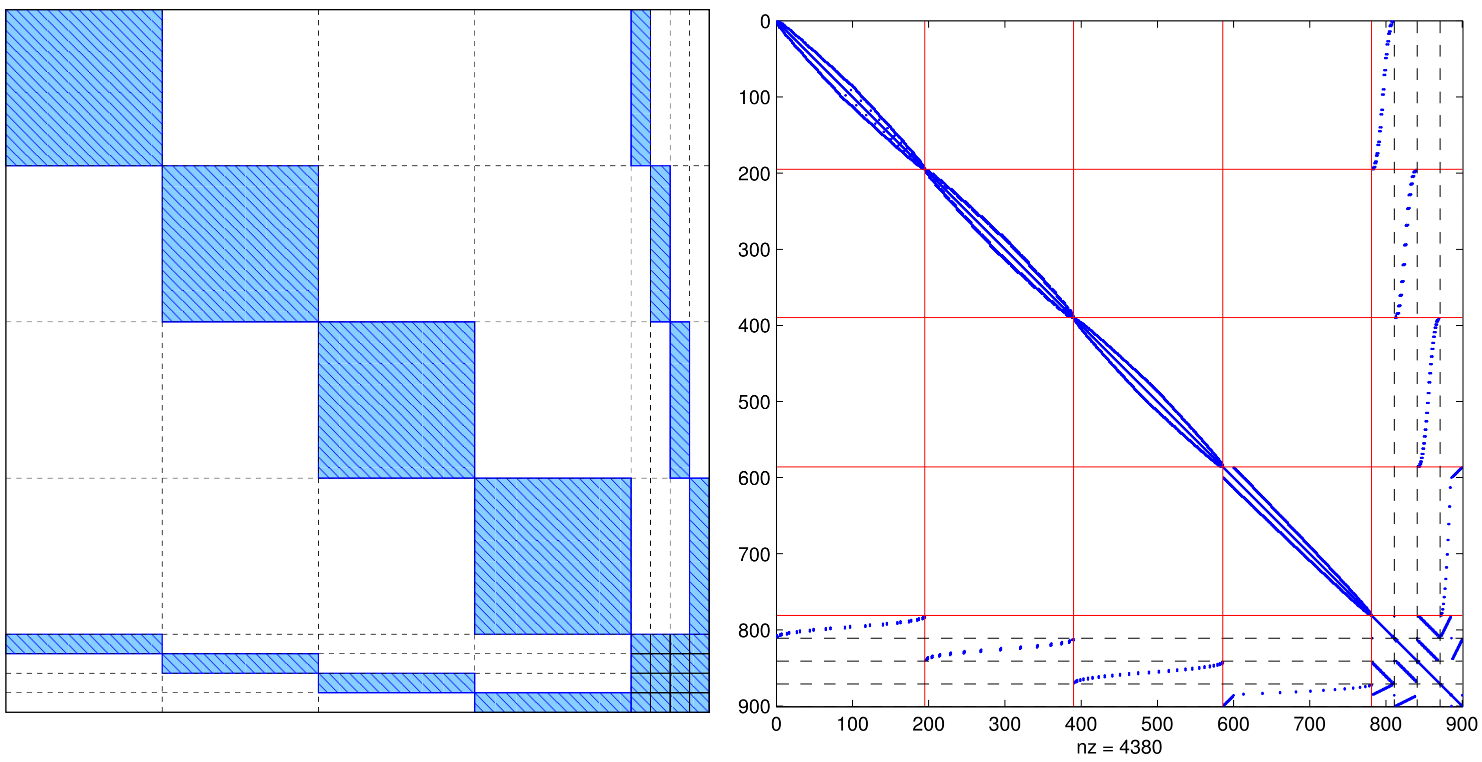
Local Equations

$$\begin{pmatrix} B_i & E_i \\ E_i^T & C_i \end{pmatrix} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \begin{pmatrix} 0 \\ \sum_{j \in N_i} E_{ij} y_j \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}$$

- Assume (for now) A is Symmetric Positive Definite (SPD)

Recall: The global system

- Global matrix has the form $\begin{pmatrix} B & E \\ E^T & C \end{pmatrix}$



Schur Complement System

Background:

$$\begin{pmatrix} B & E \\ E^T & C \end{pmatrix} = \begin{pmatrix} I & \\ E^T B^{-1} & I \end{pmatrix} \begin{pmatrix} B & E \\ & S \end{pmatrix} \qquad S = C - E^T B^{-1} E$$

- $S \in \mathbb{R}^{s \times s}$ == 'Schur complement' matrix
- Solution obtained from two solves with B, one with S

Next: Find approximate inverse of S.

- Assume C is SPD and let $C = LL^T$. Then:

$$S = L \left(I - L^{-1} E^T B^{-1} E L^{-T} \right) L^T \equiv L(I - H)L^T.$$

- Define: $H = L^{-1} E^T B^{-1} E L^{-T}$
- Can show: $\lambda_j(H) \in \ [0, 1)$

Decay properties of $\mathrm{S}^{-1} - \mathrm{C}^{-1}$

- We have: $\boldsymbol{S^{-1} = L^{-T}(I - H)^{-1}L^{-1}}$
- Can we write: $\boxed{\boldsymbol{S^{-1} = C^{-1}} + \text{Low rank correction ?}}$

$$\boldsymbol{S^{-1} - C^{-1} = L^{-T}(I - (I - H)^{-1})L^{-1} \equiv L^{-T} X L^{-1}}$$

- Thus, $\boldsymbol{S^{-1} = C^{-1} + L^{-T} X L^{-1}}$. Note:

$$\boldsymbol{\lambda_k(X) = \frac{\lambda_k(H)}{1 - \lambda_k(H)}}$$

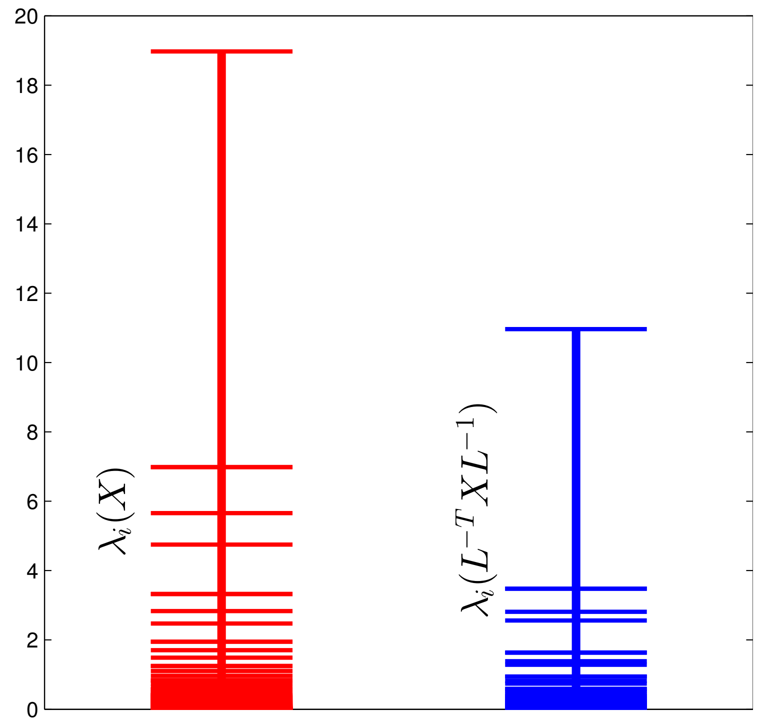
- Well separated when $\boldsymbol{\lambda_k \to 1}$

Decay properties of $\mathrm{S}^{-1} - \mathrm{C}^{-1}$

- Example: 2-D Laplacian, $n_x = n_y = 32$, 4 subdomains
- $\Lambda(X)$ and $\Lambda(S^{-1} - C^{-1}) = \Lambda(L^{-T} X L^{-1})$

5 eigenvectors:
82.5% of X, 85.1% of $L^{-T} X L^{-1}$

10 eigenvectors:
89.7% of X, 91.4% of $L^{-T} X L^{-1}$



- Closed form analysis available for 2D Laplaceans

Low-rank approximation

- Preconditioner for A:

$$M = \begin{pmatrix} I & \\ E^T B^{-1} & I \end{pmatrix} \begin{pmatrix} B & E \\ & \tilde{S} \end{pmatrix}$$

- $(n - s)$ of $\lambda_i(AM^{-1}) = 1$, the other $s \to \lambda_i(S\tilde{S}^{-1})$
- Eigendecomposition $H = U\Lambda U^T$. Replace Λ with $\tilde{\Lambda}$
- Recall $S^{-1} = L^{-T}(I - H)^{-1}L^{-1}$, and rewrite

$$S^{-1} = L^{-T}U(I - \Lambda)^{-1}U^T L^{-1}$$
$$\tilde{S}^{-1} = L^{-T}U(I - \tilde{\Lambda})^{-1}U^T L^{-1}$$

- Can show: $\lambda(S\tilde{S}^{-1}) = \dfrac{1 - \lambda_i}{1 - \tilde{\lambda}_i}, \quad i = 1, \ldots, s$

Numerical Experiments

- Intel Xeon X5675 (12 MB Cache, 3.06 GHz, 6-core), Xeon X5560 (8 MB Cache, 2.8 GHz, 4-core) at MSI
- Written in C/C++, MKL; OpenMP parallelism
- Accelerators: CG, GMRES(40)
- Partitioning with METIS

SLR, indefinite model problems

- $-\boldsymbol{\Delta}$ shifted by $-s\boldsymbol{I}$. 2D: $s = 0.01$, 3D: $s = 0.05$

Grid	ILDLT-GMRES				RAS-GMRES				SLR-GMRES					
	fill	p-t	its	i-t	fill	p-t	its	i-t	nd	rk	fill	p-t	its	i-t
256^2	8.2	.17	F	-	6.3	.13	F	-	8	32	6.4	.21	33	.125
512^2	8.4	.70	F	-	8.4	.72	F	-	16	64	7.6	2.1	93	1.50
1024^2	13	5.1	F	-	19	22	F	-	8	128	11	25	50	4.81
40^3	6.9	.25	54	.54	6.7	.25	99	.30	64	32	6.7	.49	23	.123
64^3	9.0	1.4	F	-	11.8	2.2	F	-	128	64	9.1	3.9	45	1.16
100^3	15	11	F	-	12	15	F	-	128	180	15	63	88	13.9

‘Non-standard’ DD framework: HID ordering

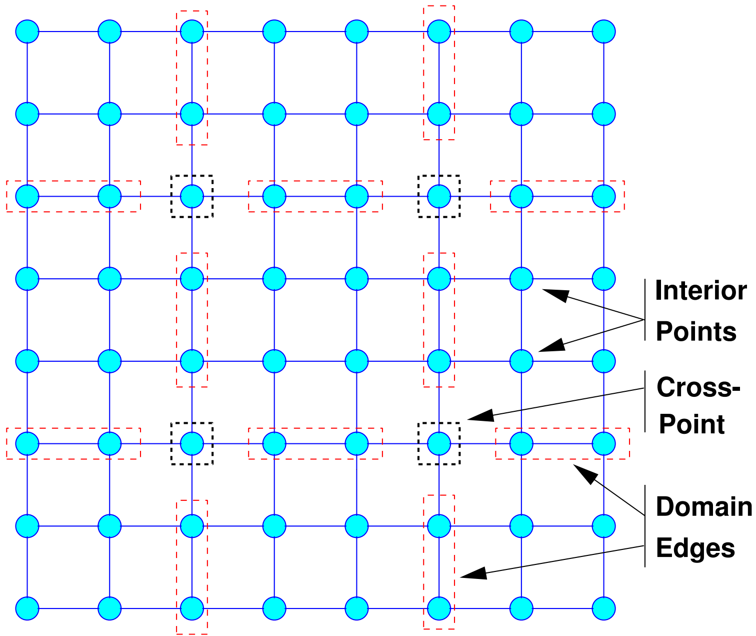
- Issue: Schur complement can become large (3D Pbs)

- Remedy: Use Hierarchical Interface Decomposition (HID) - Henon and YS’05

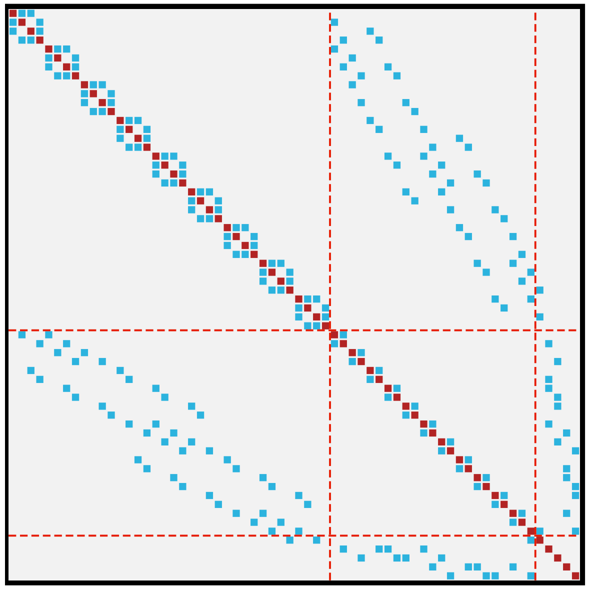
Goal: Define a method that descends into interface variables in a hierarchical way $\rightarrow$ need a hierarchy of ’interfaces’.

- Ideas of this type in the Domain Decomposition context (PDEs) by Smith and Widlund (89) – [“Wirebasket” techniques]

The hierarchical decomposition of a graph - example



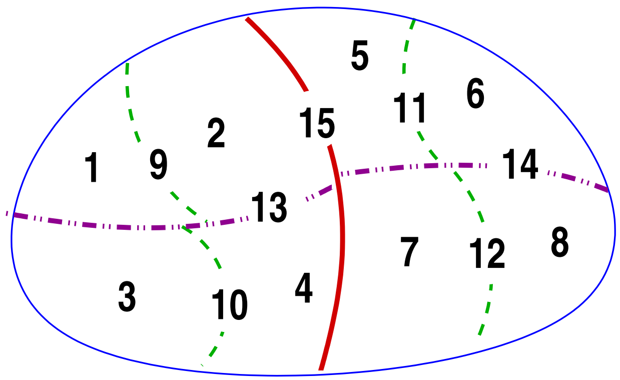
Graph

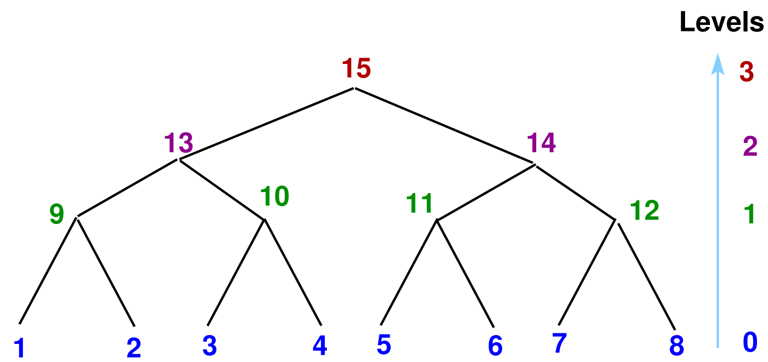


Matrix pattern

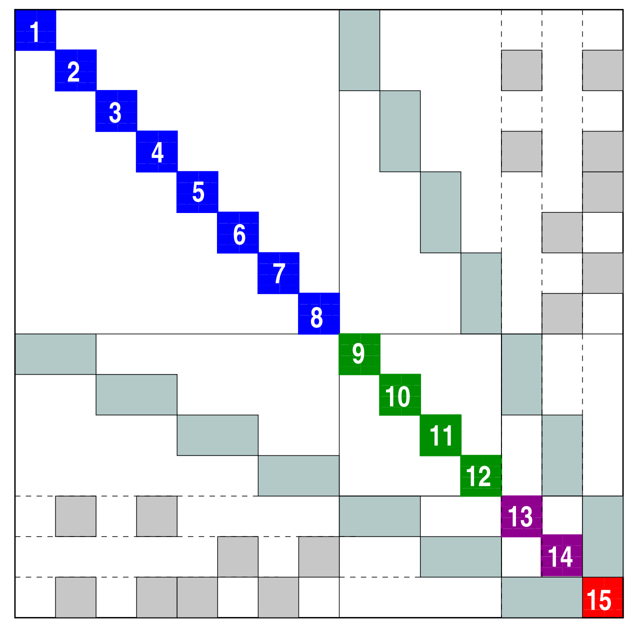
- C^1 = subdomain interiors; C^2 = sets of edges; C^3 = cross-points
- Label by levels $\to$ block-diagonal structure at each level

- Easy way to get an HID: Nested Dissection ordering





Up: $\mathbf{3}$-level partition of a $\mathbf{2}$-D domain. An HID tree with connector level information.
Right: Non-zero pattern of the re-ordered matrix.



Recursive preconditioner

$$A_l = \begin{pmatrix} B_l & E_l \\ E_l^T & C_l \end{pmatrix} \quad \text{and} \quad C_l = A_{l+1} \quad \text{for} \quad l = 0 : L-1,$$

A_0 == HID-reordered matrix A

A_l == matrix C_{l-1} for $l = 1, 2, \cdots, L$

A_L == submatrix associated with the top-level connector.

- Each leading block B_l in A_l has a block-diagonal structure

Goal: Explore multilevel strategies to approximate the factorization of A_l

➤ Recall factorization:

$$A_l = \begin{pmatrix} I & \\ E_l^T B_l^{-1} & I \end{pmatrix} \begin{pmatrix} B_l & \\ & S_l \end{pmatrix} \begin{pmatrix} I & B_l^{-1} E_l \\ & I \end{pmatrix}$$

$$S_l = C_l - E_l^T B_l^{-1} E_l$$

Main Observation: $S_l^{-1} - C_l^{-1}$ nearly small rank

➤ Rank bounded by number of cross-points (connectors at level l that intersect with connectors of higher levels)..

Idea: Write

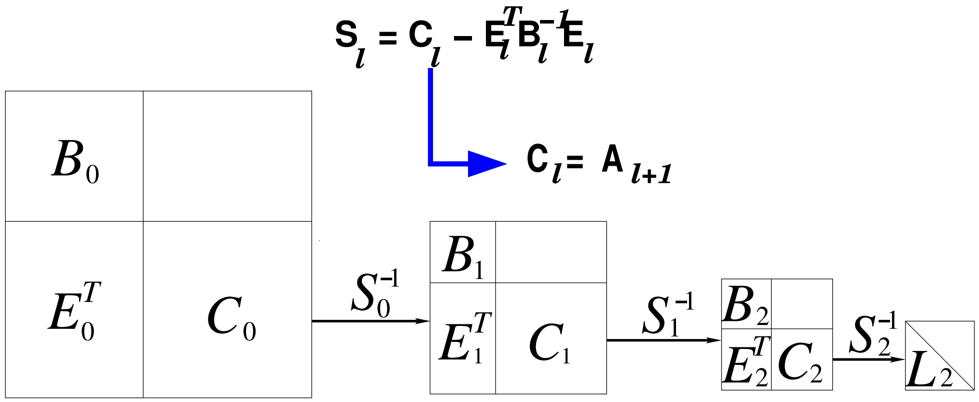
$$A_l^{-1} = \begin{pmatrix} I & -B_l^{-1}E_l \\ & I \end{pmatrix} \begin{pmatrix} B_l^{-1} & \\ & S_l^{-1} \end{pmatrix} \begin{pmatrix} I & \\ -E_l^T B_l^{-1} & I \end{pmatrix}.$$

- Approximate S_l^{-1} as $S_l^{-1} \approx C_l^{-1} - W_l H_l W_l^T$
- Next: set $C_l = A_{l+1}$ $\rightarrow$ exploit recursivity
- Last level: use (incomplete) Cholesky.
- Next: illustration for 3 levels.

- At levels $l = 0, 1, 2$ express A_l^{-1} as :

$$A_l^{-1} = \begin{pmatrix} I & -B_l^{-1}E_l \\ & I \end{pmatrix} \begin{pmatrix} B_l^{-1} & \\ & S_l^{-1} \end{pmatrix} \begin{pmatrix} I & \\ -E_l^T B_l^{-1} & I \end{pmatrix}.$$

- S_l^{-1} needed $\rightarrow$ Approximate as $S_l^{-1} \approx C_l^{-1} + W_l H_l W_l^T$
- C_l^{-1} needed $\rightarrow$ if $l == 2$ get $C_2 \approx L_2 L_2^T$,
  else set $A_{l+1} = C_l$ & go to next level



Computing the low-rank correction

- Let $C = LL^T$ and define

$$\boxed{G = L^{-1}(C - S)L^{-T} = L^{-1}\left(E^T B^{-1} E\right) L}$$

We have $S = L(I - G)L^T \rightarrow$

$$\begin{aligned} S^{-1} - C^{-1} &= L^{-T}\left[(I - G)^{-1} - I\right] L^{-1} \\ &= L^{-T}\left[G(I - G)^{-1}\right] L^{-1}. \end{aligned}$$

- Use Lanczos algorithm to get a few of the largest eigenvalues of G with associated eigenvectors:

$$[W_l, \Sigma_l] = \texttt{eigs}(C_l^{-1} E_l^T B_l^{-1} E_l, k) \rightarrow$$

$$\boxed{S_l^{-1} - C_l^{-1} \approx W_l H_l W_l^T, \quad \text{with} \quad H_l = \Sigma_l (I - \Sigma_l)^{-1}.}$$

- Need to solve with $C_l \rightarrow$ exploit recursivity

Current work: extension to nonsymmetric case – GeMSLR

- Use Arnoldi instead of Lanczos.
- Parallel code called GeMSLR developed in C++
- Complex version available
- Details skipped.

Strong Scaling Result

- Recursive K-way + GeMSLR + FGMRES.
- 128*128*128 7pt Laplacian using rank = 20.

	Time (sec.)		Speed-up	
np	Precond	Solve	Precond	Solve
1	30.52	50.00	1.00	1.00
2	12.02	23.19	2.03	2.16
4	7.67	11.67	3.98	4.28
8	4.30	6.39	7.10	7.82
16	2.65	4.27	11.52	11.71
32	1.50	2.50	20.35	20.00
64	0.88	1.42	34.68	35.21

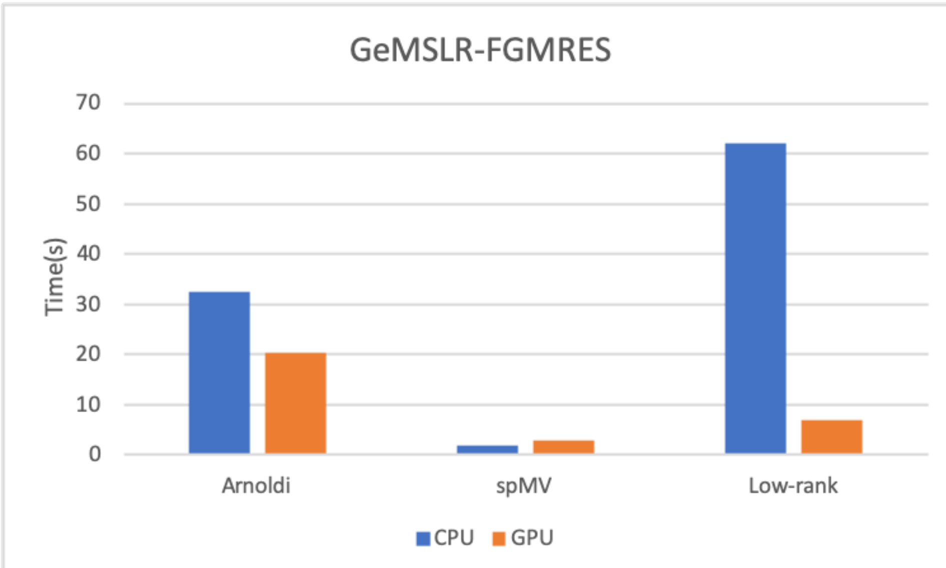
Strong Scaling Result

- Recursive K-way + GeMSLR + FGMRES.
- 256*256*256 7pt Laplacian with rank = 20.

	Time (sec.)		Speed-up	
np	Precond	Solve	Precond	Solve
16	40.33	62.26	1.00	1.00
32	22.48	33.92	1.79	1.84
64	13.07	18.71	3.08	3.33
128	7.73	10.50	5.22	5.93
256	4.86	6.22	8.30	10.01

GPU-acceleration

- Solve $\boldsymbol{C}^{-1}(\boldsymbol{I} - \boldsymbol{W}\boldsymbol{H}\boldsymbol{W}^{\boldsymbol{H}})\boldsymbol{x}$.



*Test on 72*72*72 3D Laplacean with rank = 100. Haswell Xeon E5-2680 v3 nodes + NVidia Tesla K40m GPUs*

Conclusion

➤ New Mantra: Seek "rank-sparsity" or "spectral sparsity" instead of regular sparsity

➤ Current work: (1) Good HID partitioniers; (2) General purpose code (in prog.) (3) *Very* highly indefinite problems

Advantages of Multilevel Low-Rank preconditioners:

(1) Approximate inverses $\longrightarrow$ less sensitive to indefiniteness;
(2) Exploit dense computations; (3) Easy to update.

➤ Visit `http://cs.umn.edu/~saad`
for this talk, papers, codes, ...