UNIVERSITY OF MINNESOTA TWIN CITIES

Multilevel Low-Rank Preconditioners

Yousef Saad

Department of Computer Science and Engineering

University of Minnesota

Modelling 2014 – June 2,2014

Dedicated to Owe Axelsson at the occasion of his 80th birthday

Acknowledments

- Joint work with Ruipeng Li
- Work supported by NSF-DMS

Intro: ILU-type preconditioners

Problem:

To solve linear systems $\boxed{Ax = b}$

Common approach: ['grey-box' solvers]

Krylov subspace accelerator (e.g., GMRES, BiCGSTAB)
+ Preconditioner

Common preconditioners:

Incomplete LU factorizations;
Relaxation-type;
AMG; ...

Common difficulties of ILUs:

Often fail for indefinite problems
Not too good for highly parallel environments [GPUs]

Alternatives to ILU preconditioners

- Time to think about (radical) alternatives?
  - Preconditioners requiring few 'irregular' computations ...
  - .. that trade volume of computations for speed,
  - .. and, if possible, more robust for indefinite case
- Possible candidates: Methods based on Multilevel Low-Rank (MLR) approximations
- Low-rank approximation techniques can be seen everywhere in computational sciences
- Common approach: truncated SVD ..
- .. and more often now : random sampling

Related work:

- Work on H-matrices [Hackbusch and co-workers, B. Khoromskij, L. Grasedyck, S. Leborne, + many others..]
- Work on HSS matrices [e.g., J. Xia, S. Chandrasekaran, M. Gu, and X-S. Li 2010.]
- Work on 'balanced incomplete factorizations' (R. Bru et al.)
- Work on "sweeping preconditioners" by Engquist and Ying.
- Work on computing the diagonal of a matrix inverse [Jok Tang and YS (2010) ..]

MULTI-LEVEL LOW-RANK PRECONDITIONERS

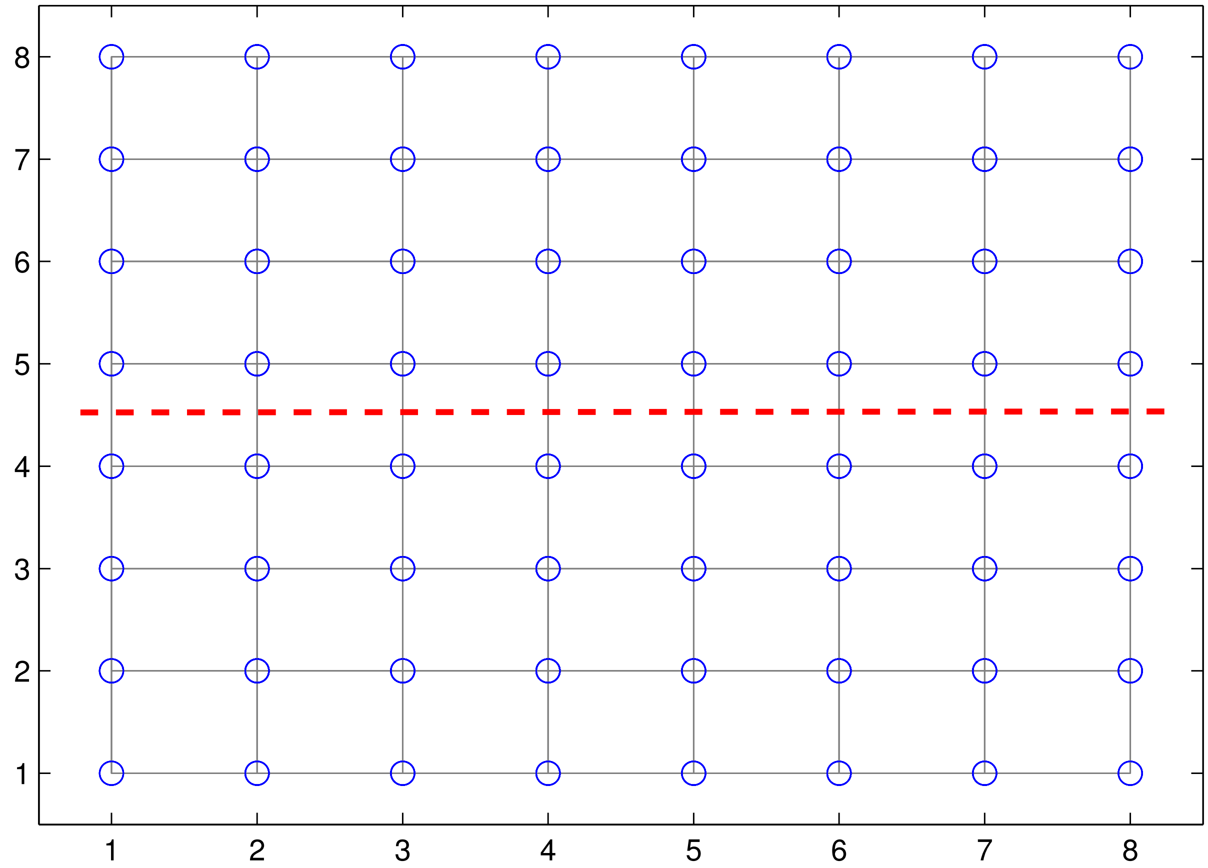
Low-rank Multilevel Approximations

➤ Starting point: symmetric matrix derived from a 5-point discretization of a 2-D Pb on $n_x \times n_y$ grid

$$A = \left(\begin{array}{cccc|ccc} A_1 & D_2 & & & & & \\ D_2 & A_2 & D_3 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & D_\alpha & A_\alpha & D_{\alpha+1} & & \\ \hline & & & D_{\alpha+1} & A_{\alpha+1} & \ddots & \\ & & & & \ddots & \ddots & \ddots \\ & & & & & D_{n_y} & A_{n_y} \end{array}\right)$$

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \equiv \begin{pmatrix} A_{11} & \\ & A_{22} \end{pmatrix} + \begin{pmatrix} & A_{12} \\ A_{21} & \end{pmatrix}$$

Corresponding splitting on FD mesh:



- $A_{11} \in \mathbb{R}^{m \times m}$, $A_{22} \in \mathbb{R}^{(n-m)\times(n-m)}$

- In the simplest case second matrix is:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & \\ & A_{22} \end{pmatrix} + \begin{pmatrix} & -I \\ -I & \end{pmatrix}$$

- Write 2nd matrix as:

$$\begin{pmatrix} & -I \\ -I & \end{pmatrix} = \begin{pmatrix} +I & \\ & +I \end{pmatrix} - \begin{pmatrix} I & I \\ I & I \end{pmatrix}$$

$$\mathbf{E}\,\mathbf{E}^T$$

$$\mathbf{E}^T = \begin{pmatrix} I & I \end{pmatrix}$$

- Thus: $\boldsymbol{A} = \underbrace{(\boldsymbol{A} + \boldsymbol{E}\boldsymbol{E}^T)}_{\boldsymbol{B}} - \boldsymbol{E}\boldsymbol{E}^T$
- Note: $\boldsymbol{E} \in \mathbb{R}^{n \times n_x}$, $\boldsymbol{X} \in \mathbb{R}^{n_x \times n_x}$
- n_x = | separator | = [$O(n^{1/2})$ in 2-D, $O(n^{2/3})$ in 3-D]

$$\boldsymbol{A} = \boldsymbol{B} - \boldsymbol{E}\boldsymbol{E}^T,$$
$$\boldsymbol{B} := \begin{pmatrix} \boldsymbol{B}_1 & \\ & \boldsymbol{B}_2 \end{pmatrix} \in \mathbb{R}^{n \times n}, \quad \boldsymbol{E} := \begin{pmatrix} \boldsymbol{E}_1 \\ \boldsymbol{E}_2 \end{pmatrix} \in \mathbb{R}^{n \times n_x},$$

- Next step: use Sherman-Morrison formula:

$$\boldsymbol{A}^{-1} = \boldsymbol{B}^{-1} + (\boldsymbol{B}^{-1}\boldsymbol{E})\boldsymbol{X}^{-1}(\boldsymbol{B}^{-1}\boldsymbol{E})^T$$
$$\boldsymbol{X} = \boldsymbol{I} - \boldsymbol{E}^T\boldsymbol{B}^{-1}\boldsymbol{E}$$

Multilevel Low-Rank (MLR) algorithm

- Use in a recursive framework [apply recursively to B_1, B_2]
- Next step: low-rank approx. for $B^{-1}E$

$$B^{-1}E \approx U_k V_k^T, \quad U_k \in \mathbb{R}^{n\times k}, \quad V_k \in \mathbb{R}^{n_x\times k},$$

- Replace $B^{-1}E$ by $U_k V_k^T$ in $X = I - (E^T B^{-1})E$:

$$X \approx G_k = I - V_k U_k^T E, \quad (\in \mathbb{R}^{n_x\times n_x})$$ Leads to ...

Preconditioner

$$M^{-1} = B^{-1} + U_k H_k U_k^T$$

(on B^{-1}:) Use recursivity

- Can show : $H_k = (I - U_k^T E V_k)^{-1}$ and $H_k^T = H_k$

Other options explored

➤ Another thought : approximate $\boldsymbol{X}$ (only) and exploit recursivity

$$\boldsymbol{B}^{-1}[\boldsymbol{v} + \boldsymbol{E}\tilde{\boldsymbol{X}}^{-1}\boldsymbol{E}^T\boldsymbol{B}^{-1}\boldsymbol{v}]$$

➤ However won't work: cost explodes with # levels [recursivity]

➤ We will see later how we can use this in DD framework

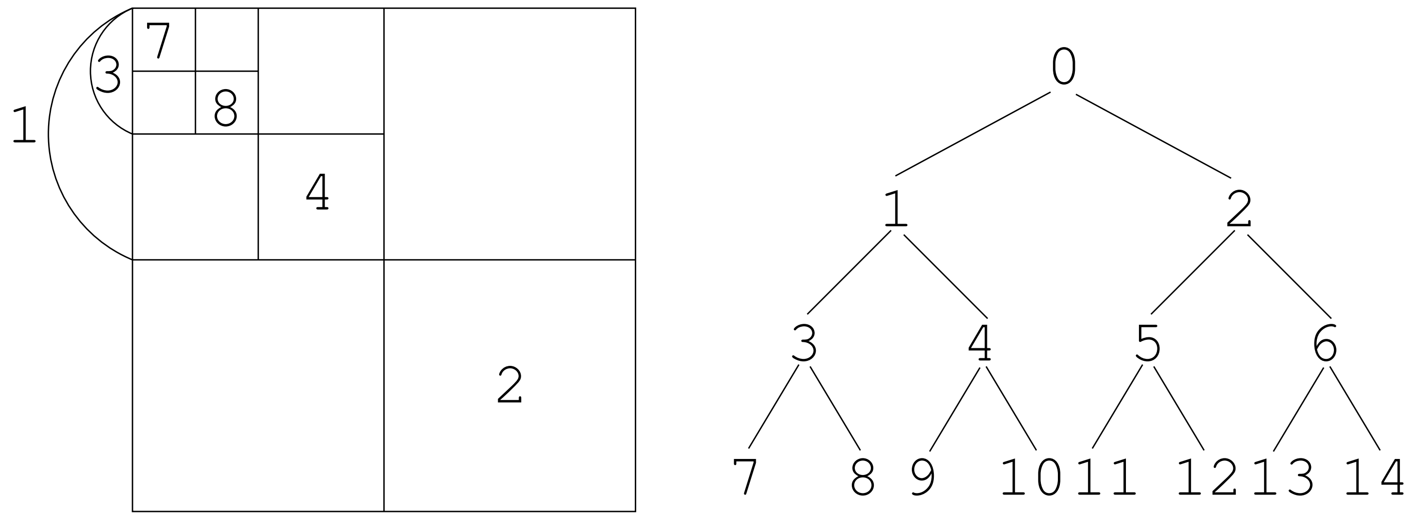
➤ Another possibility: approximate $\boldsymbol{B}^{-1}\boldsymbol{E}$ on one side only:

$$\boldsymbol{M}^{-1} = \boldsymbol{B}^{-1} + \boldsymbol{B}^{-1}\boldsymbol{E}\boldsymbol{G}_k^{-1}\boldsymbol{V}_k\boldsymbol{U}_k^T = \boldsymbol{B}^{-1}[\boldsymbol{I} + \boldsymbol{E}\boldsymbol{G}_k^{-1}\boldsymbol{V}_k\boldsymbol{U}_k^T]$$

➤ However, can show that this is equivalent to the previous method

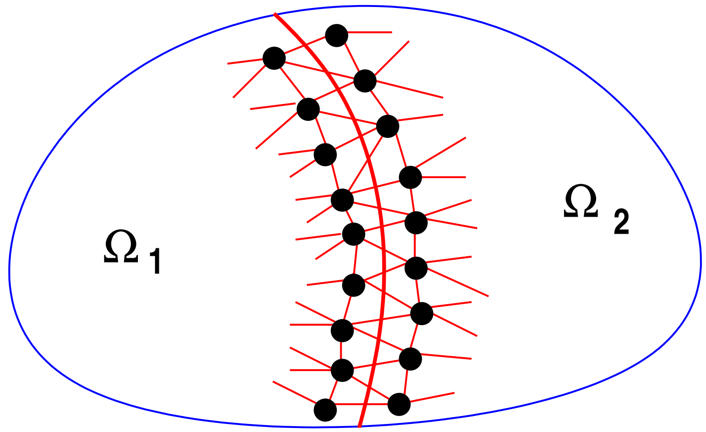
Recursive multilevel framework

- $A_i = B_i + E_i E_i^T$, $B_i \equiv \begin{pmatrix} B_{i_1} & \\ & B_{i_2} \end{pmatrix}$.
- Next level, set $A_{i_1} \equiv B_{i_1}$ and $A_{i_2} \equiv B_{i_2}$
- Repeat on A_{i_1}, A_{i_2}
- Last level, factor A_i (IC, ILU)
- Binary tree structure:



Generalization: Domain Decomposition framework

Domain partitioned into 2 domains with an edge separator



- Matrix can be permuted to:

$$PAP^T = \left(\begin{array}{cc|cc} \hat{B}_1 & \hat{F}_1 & & \\ \hat{F}_1^T & C_1 & & -X \\ \hline & & \hat{B}_2 & \hat{F}_2 \\ & -X^T & \hat{F}_2^T & C_2 \end{array}\right)$$

- Interface nodes in each domain are listed last.

- Each matrix $\hat{B}_i$ is of size $n_i \times n_i$ (interior var.) and the matrix C_i is of size $m_i \times m_i$ (interface var.)

$$\text{Let:} \qquad E_\alpha = \begin{pmatrix} 0 \\ \alpha I \\ 0 \\ \frac{X^T}{\alpha} \end{pmatrix} \qquad \text{then we have:}$$

$$PAP^T = \begin{pmatrix} B_1 & \\ & B_2 \end{pmatrix} - EE^T \quad \text{with} \quad B_i = \begin{pmatrix} \hat{B}_i & \hat{F}_1 \\ \hat{F}_i^T & C_i + D_i \end{pmatrix}$$

$$\text{and} \quad \begin{cases} D_1 = \alpha^2 I \\ D_2 = \frac{1}{\alpha^2} X^T X \end{cases} .$$

- α used for balancing
- Better results when using diagonals instead of αI

Theory: 2-level analysis for model problem

- Interested in eigenvalues γ_j of

$$A^{-1} - B^{-1} = B^{-1} E X^{-1} E^T B^{-1}$$

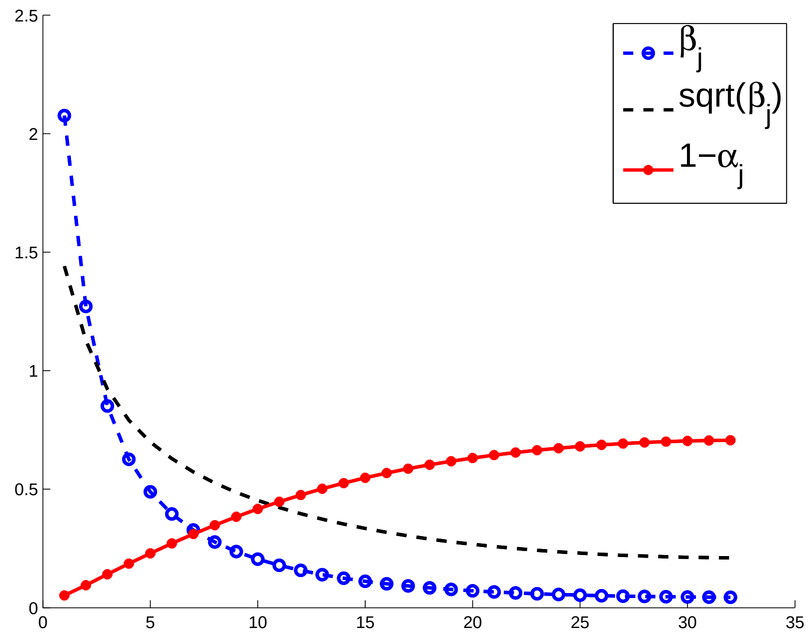
when A = Pure Laplacean .. They are:

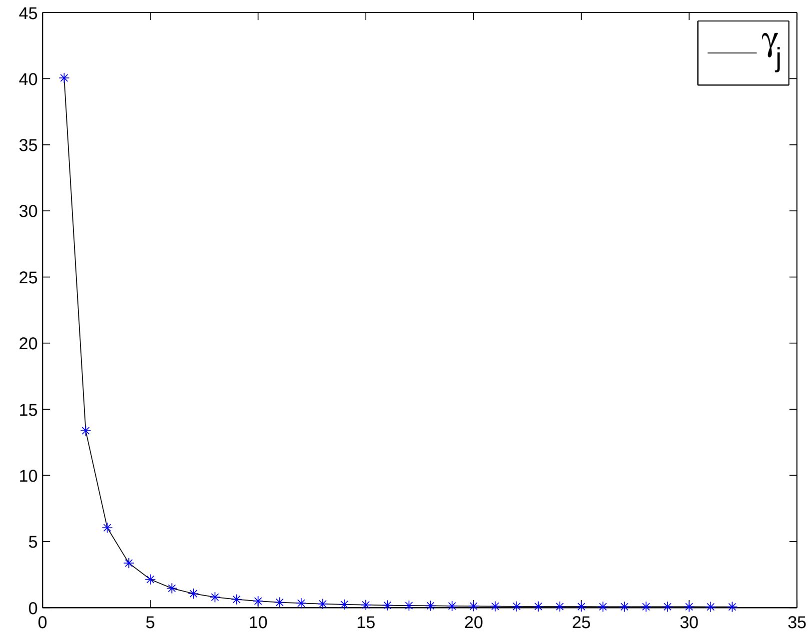
$$\gamma_j = \frac{\beta_j}{1-\alpha_j}, \quad j = 1, \cdots, n_x \quad \text{with:}$$

$$\beta_j = \sum_{k=1}^{n_y/2} \frac{\sin^2 \frac{n_y k \pi}{n_y+1}}{4\left(\sin^2 \frac{k\pi}{n_y+1} + \sin^2 \frac{j\pi}{2(n_x+1)}\right)^2},$$

$$\alpha_j = \sum_{k=1}^{n_y/2} \frac{\sin^2 \frac{n_y k \pi}{n_y+1}}{\sin^2 \frac{k\pi}{n_y+1} + \sin^2 \frac{j\pi}{2(n_x+1)}} .$$

➤ Decay of the γ_j's when $nx = ny = 32$.





Note $\sqrt{\beta_j}$ are the singular values of $B^{-1}E$.

In this particular case 3 eigenvectors will capture 92 % of the inverse whereas 5 eigenvectors will capture 97% of the inverse.

EXPERIMENTS

A few MATLAB experiments

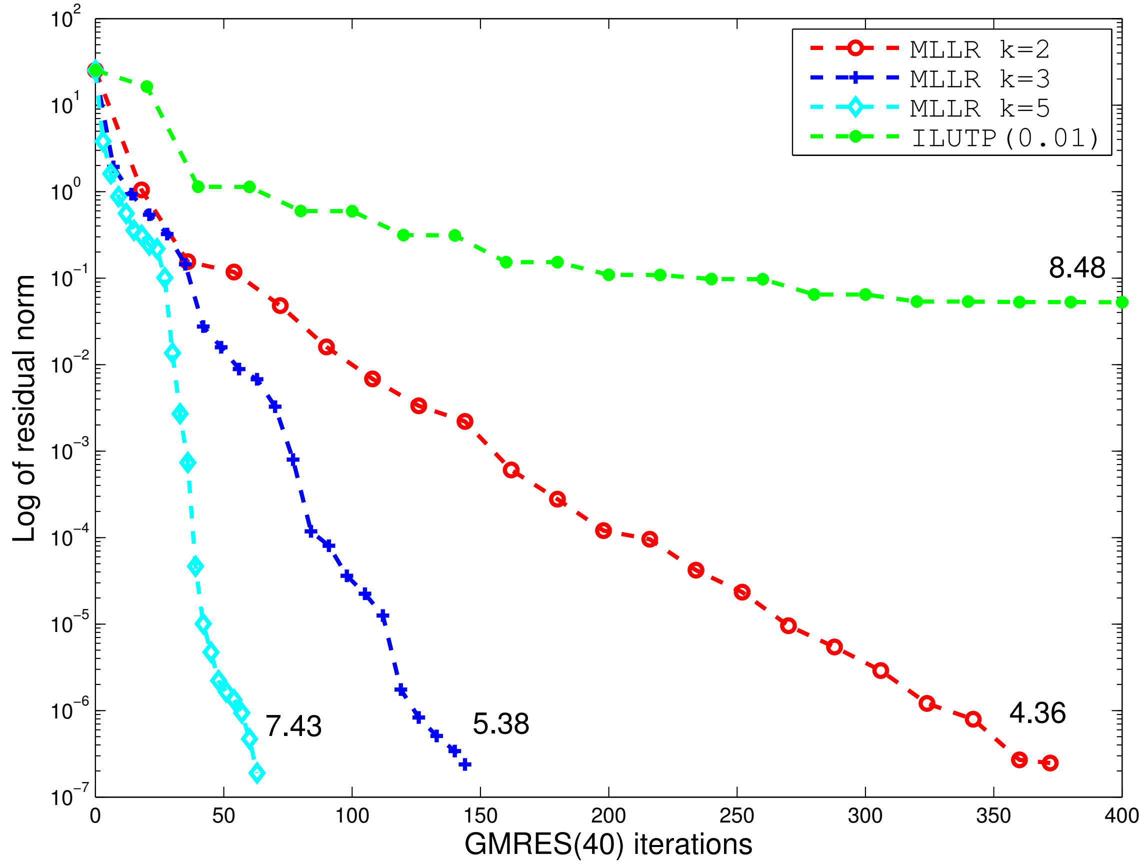
- Matlab first – Small problems ; ‘real’ tests later
- Helmholtz-like problem:

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} - \rho u = -6 - \rho\left(2x^2 + y^2\right) \text{ in } \Omega,$$

+ Boundary conditions so solution is known

- ρ = constant selected to make problem more or less difficult
- Finite differences on a 66×66 mesh (matrix size 4,096).
- MLR starts converging for $k = 2$.
- $\rho = 845$ selected so original Laplacean is shifted by 0.2
- 60 negative eigenvalues, smallest = -0.1953...

Comparison with ILUTP



ILUTP vs. MLR (E) – # levels = 7 for MLR

k	nlev=7		nlev=6		nlev=5		nlev=4		nlev=3	
2	318	3.56	372	4.36	261	4.77	183	4.80	47	5.53
3	192	4.78	144	5.38	144	5.59	102	5.41	38	5.94
4	181	6.03	132	6.41	74	6.41	45	6.02	35	6.35
5	75	7.20	63	7.43	39	7.22	33	6.63	31	6.76
6	45	8.52	41	8.46	35	8.04	29	7.24	28	7.16

MLR: GMRES(40) iteration counts and fill ratios

Helmoltz-like equation - a 3D case

- Similar set-up to 2D case. $26 \times 26 \times 26$ grid $\rightarrow$ size $n = 24^3 = 13,824$
- $\rho = 312.5$ selected so the shift is 0.5 - making the problem very indefinite [60 negative eigenvalues, $\lambda_{min} = -0.4527..$]

GMRES(40)-MLR iteration counts and fill ratios

k	nlev=6		nlev=5		nlev=4	
2	377	5.49	177	6.66	114	8.46
4	293	6.97	138	7.84	88	9.35
6	187	8.46	101	9.03	73	10.23
8	116	9.95	78	10.22	51	11.12

- ILUTP fails even for very small values of droptol (large fill)

General matrices

- 17 matrices from the Univ. Florida sparse matrix collection + one from a shell problem.
- 7 matrices are SPD
- Size varies from $n = 1,224$ (HB/bcsstm27) to $n = 9,000$ (AG-Monien/3elt1 dual)
- nnz varies from $nnz = 5,300$ (HB/bcspwr06) to $nnz = 355,460$ (Boeing/bcsstk38).
- Only indefinite cases shown

MATRICES (Non SPD)	MLR				ICT/ILUTP	
	nlev	k	fill-ratio	#its	fill-ratio	#its
HB/bcsstm27	4	50	1.8	26	2.3	73
HB/bcspwr06	4	5	3.1	6	5.2	F
HB/bcspwr07	5	5	3.2	6	4.8	F
HB/bcspwr08	4	5	2.1	17	5.8	F
HB/blckhole	5	50	12.8	32	21.8	F
HB/jagmesh3	4	5	5.9	30	9.7	111
Boeing/nasa1824	4	60	3.6	116	4.9	150
AG-Monien/3elt_dual	6	5	9.3	12	13.9	F
AG-Monien/airfoil1_dual	6	5	9.5	5	12.7	F
AG-Monien/ukerbe1_dual	4	5	9.1	25	10.5	F
SHELL/COQUE8E3	3	70	5.0	83	5.06	F

MLR vs. ICT/ILUTP

'Real tests' – Experimental setting

- Hardware: Intel Xeon X5675 processor (12 MB Cache, 3.06 GHz, 6-core)
- C/C++; Intel Math Kernel Library (MKL,version 10.2)
- Stop when: $\|r_i\| \leq 10^{-8}\|r_0\|$ or `its` exceeds 500
- Model Problems in 2-D/3-D:

$$-\Delta u - cu = g \text{ in } \Omega \quad + \text{B.C.}$$

- 2-D: $g(x,y) = -\left(x^2 + y^2 + c\right) e^{xy}; \quad \Omega = (0,1)^3.$
- 3-D: $g(x,y,z) = -6 - c\left(x^2 + y^2 + z^2\right); \quad \Omega = (0,1)^3.$
- F.D. Differences discret.

Symmetric indefinite cases

- $c > 0$ in $-\Delta u - cu$; i.e., $-\Delta$ shifted by $-sI$.
- 2D case: $s = 0.01$, 3D case: $s = 0.05$
- MLR + GMRES(40) compared to ILDLT + GMRES(40)
- 2-D problems: #lev$= 4$, rank$= 5, 7, 7$
- 3-D problems: #lev$= 5$, rank$= 5, 7, 7$
- ILDLT failed for most cases
- Difficulties in MLR: #lev cannot be large, [no convergence]
- inefficient factorization at the last level (memory, CPU time)

Grid	ILDLT-GMRES				MLR-GMRES			
	fill	p-t	its	i-t	fill	p-t	its	i-t
256^2	6.5	0.16	F		6.0	0.39	84	0.30
512^2	8.4	1.25	F		8.2	2.24	246	6.03
1024^2	10.3	10.09	F		9.0	15.05	F	
$32^2 \times 64$	5.6	0.25	61	0.38	5.4	0.98	62	0.22
64^3	7.0	1.33	F		6.6	6.43	224	5.43
128^3	8.8	15.35	F		6.5	28.08	F	

General symmetric matrices - Test matrices

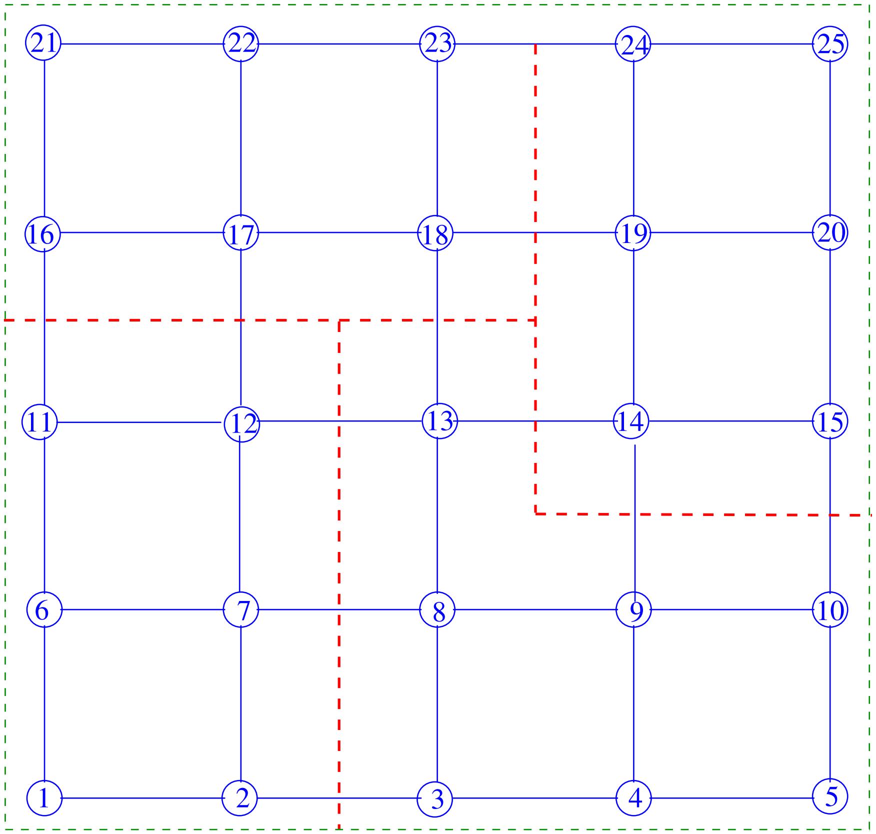
MATRIX	N	NNZ	SPD	DESCRIPTION
Andrews/Andrews	60,000	760,154	yes	computer graphics pb.
Williams/cant	62,451	4,007,383	yes	FEM cantilever
UTEP/Dubcova2	65,025	1,030,225	yes	2-D/3-D PDE pb.
Rothberg/cfd1	70,656	1,825,580	yes	CFD pb.
Schmid/thermal1	82,654	574,458	yes	thermal pb.
Rothberg/cfd2	123,440	3,085,406	yes	CFD pb.
Schmid/thermal2	1,228,045	8,580,313	yes	thermal pb.
Cote/vibrobox	12,328	301,700	no	vibroacoustic pb.
Cunningham/qa8fk	66,127	1,660,579	no	3-D acoustics pb.
Koutsovasilis/F2	71,505	5,294,285	no	structural pb.

MATRIX	ICT/ILDLT				MLR-CG/GMRES					
	fill	p-t	its	i-t	k	lev	fill	p-t	its	i-t
Andrews	2.6	0.44	32	0.16	2	6	2.3	1.38	27	0.08
cant	4.3	2.47	F	19.01	10	5	4.3	7.89	253	5.30
Dubcova2	1.4	0.14	42	0.21	4	4	1.5	0.60	47	0.09
cfd1	2.8	0.56	314	3.42	5	5	2.3	3.61	244	1.45
thermal1	3.1	0.15	108	0.51	2	5	3.2	0.69	109	0.33
cfd2	3.6	1.14	F	12.27	5	4	3.1	4.70	312	4.70
thermal2	5.3	4.11	148	20.45	5	5	5.4	15.15	178	14.96

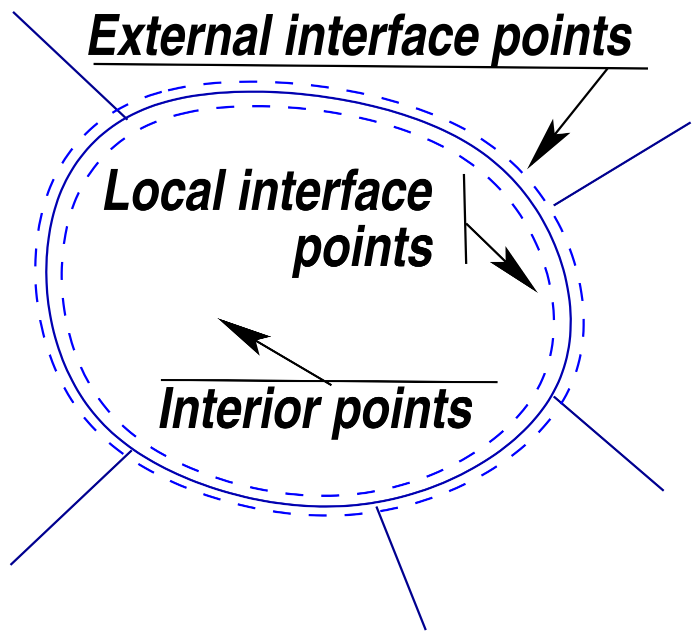
MATRIX	ICT/ILDLT				MLR-CG/GMRES					
	fill	p-t	its	i-t	k	lev	fill	p-t	its	i-t
vibrobox	3.3	0.19	F	1.06	10	4	3.0	0.45	183	0.22
qa8fk	1.8	0.58	56	0.60	2	8	1.6	2.33	75	0.36
F2	2.3	1.37	F	13.94	5	5	2.5	4.17	371	7.29

Avoiding recursivity: 'standard' DD framework

- Work in progress
- Goal: avoid recursivity
- Consider a domain partition in p domains using vertex- based partitioniong (edge-separation)
- Interface nodes in each domain are listed last.



Local view:

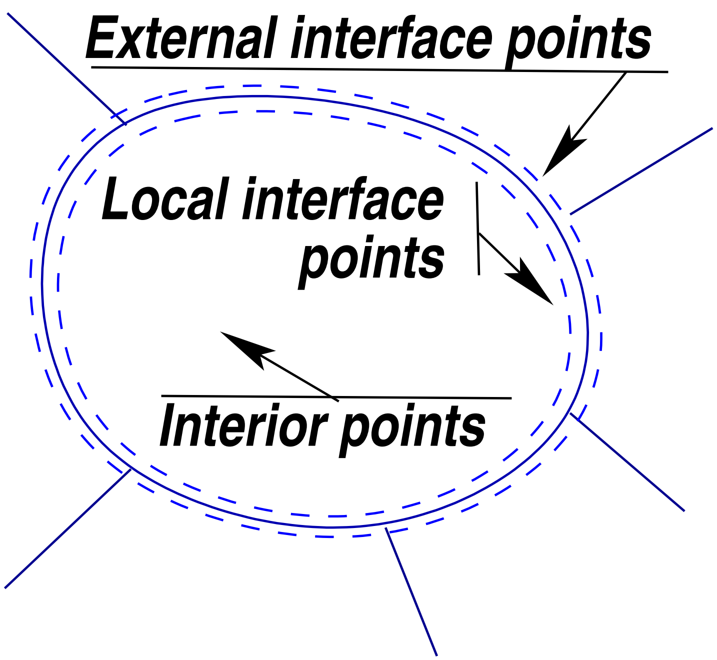


$$\begin{pmatrix} B_i & F_i \\ E_i^T & C_i \end{pmatrix} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \begin{pmatrix} 0 \\ \sum_{j \in N_i} E_{ij} y_j \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}$$

The global system: Global view

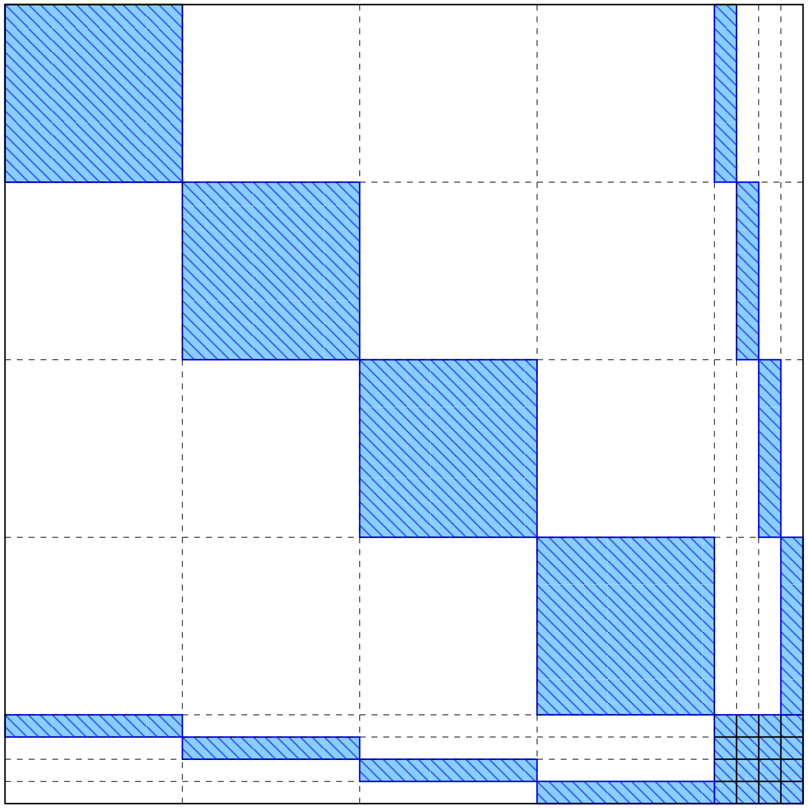
- Global system can be permuted to the form →
- u_i's internal variables
- y interface variables

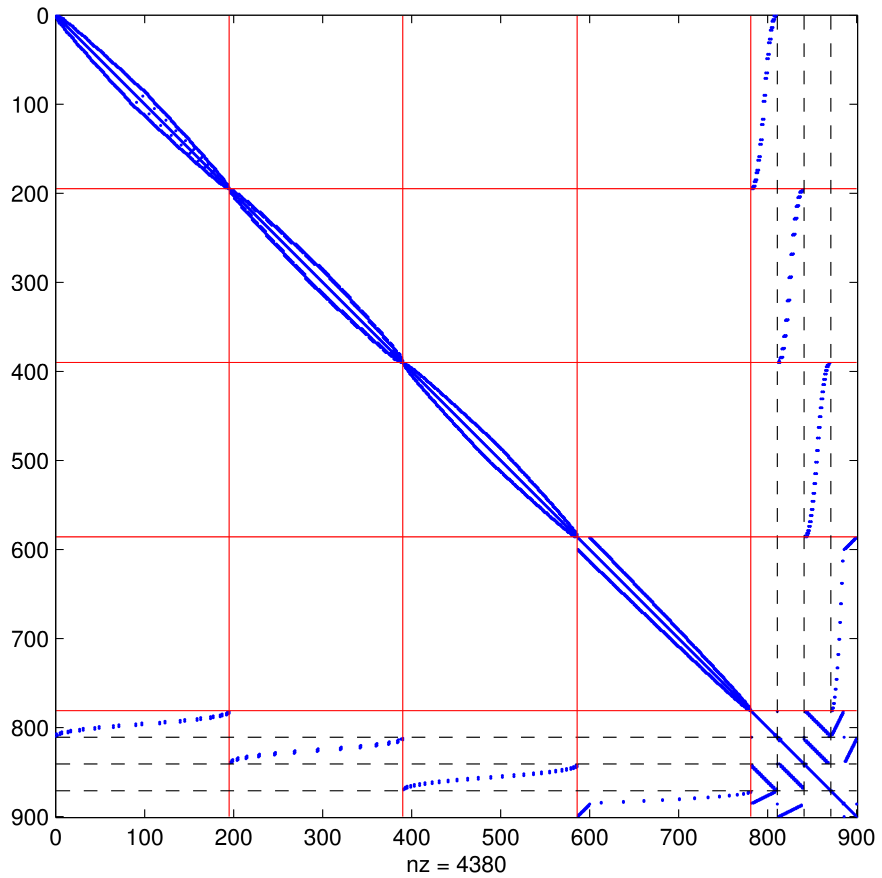
$$\begin{pmatrix} B_1 & & & \dots & \hat{F}_1 \\ & B_2 & & \dots & \hat{F}_2 \\ \vdots & & \ddots & & \vdots \\ & & & B_p & \hat{F}_p \\ \hat{E}_1^T & \hat{E}_2^T & \dots & \hat{E}_p^T & C \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_p \\ y \end{pmatrix} = b$$



- $\hat{F}_i$ maps local interface points to interior points in domain Ω_i
- $\hat{E}_i^T$ does the reverse operation

Example:





Splitting

- Split as: $$A \equiv \begin{pmatrix} B & \hat{F} \\ \hat{E}^T & C \end{pmatrix} = \begin{pmatrix} B & \\ & C \end{pmatrix} + \begin{pmatrix} & \hat{F} \\ \hat{E}^T & \end{pmatrix}$$

- Define: $F \equiv \begin{pmatrix} \alpha^{-1}\hat{F} \\ -\alpha I \end{pmatrix}$; $E \equiv \begin{pmatrix} \alpha^{-1}\hat{E} \\ -\alpha I \end{pmatrix}$ Then:

$$\left[\begin{array}{c|c} B & \hat{F} \\ \hline \hat{E}^T & C \end{array}\right] = \left[\begin{array}{c|c} B + \alpha^{-2}\hat{F}\hat{E}^T & 0 \\ \hline 0 & C + \alpha^2 I \end{array}\right] - FE^T.$$

- Property: $\hat{F}\hat{E}^T$ is 'local', i.e., no inter-domain couplings $\rightarrow$

$$A_0 \equiv \left[\begin{array}{c|c} B + \alpha^{-2}\hat{F}\hat{E}^T & 0 \\ \hline 0 & C + \alpha^2 I \end{array}\right]$$

$=$ block-diagonal

Low-Rank Approximation DD preconditioners

Sherman-Morrison $\rightarrow$

$$A^{-1} = A_0^{-1} + A_0^{-1} F G^{-1} E^T A_0^{-1}$$
$$G \equiv I - E^T A_0^{-1} F$$

Options:

(a) Approximate $A_0^{-1}F, E^T A_0^{-1}, G^{-1}$ [as before]
(b) Approximate only G^{-1} [new]

- (b) requires 2 solves with A_0.

Let $G \approx G_k$
Preconditioner $\rightarrow$

$$M^{-1} = A_0^{-1} + A_0^{-1} F G_k^{-1} E^T A_0^{-1}$$

Symmetric Positive Definite case

- Recap: Let $\boxed{G \equiv I - E^T A_0^{-1} E \equiv I - H}$. Then

$$A^{-1} = A_0^{-1} + A_0^{-1} E G^{-1} E^T A_0^{-1}$$

- Approximate G^{-1} by G_k^{-1} $\rightarrow$ preconditioner:

$$M^{-1} = A_0^{-1} + (A_0^{-1} E) G_k^{-1} (E^T A_0^{-1})$$

- Matrix A_0 is SPD
- Can show: $0 \le \lambda_j(H) < 1$.

- Next, approximate H as $H \approx U\tilde{D}U^T$ – Then:

$$(I - U\tilde{D}U^T)^{-1} = I + U[(I - \tilde{D})^{-1} - I]U^T.$$

- Now take rank-k approximation to H:

$H \approx U_k D_k U_k^T \qquad G_k = I - U_k D_k U_k^T \quad \rightarrow$

$$G_k^{-1} \equiv (I - U_k D_k U_k^T)^{-1} = I + U_k[(I - D_k)^{-1} - I]U_k^T$$

- Observation: $A^{-1} = M^{-1} + A_0^{-1}E[G^{-1} - G_k^{-1}]E^T A_0^{-1}$
- G_k: k largest eigenvalues of G matched – others set == 0
- Result: AM^{-1} has
  - $n - s + k$ eigenvalues == 1
  - All others between 0 and 1

Alternative: reset lowest eigenvalues to constant

- Let $H = U\Lambda U^T$ = exact (full) diagonalization of H
- We replaced Λ by:

$$\begin{pmatrix} \lambda_1 & & & & & & \\ & \lambda_2 & & & & & \\ & & \ddots & & & & \\ & & & \lambda_k & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{pmatrix}$$

- Alternative: replace Λ by

$$\begin{pmatrix} \lambda_1 & & & & & & \\ & \lambda_2 & & & & & \\ & & \ddots & & & & \\ & & & \lambda_k & & & \\ & & & & \theta & & \\ & & & & & \ddots & \\ & & & & & & \theta \end{pmatrix}$$

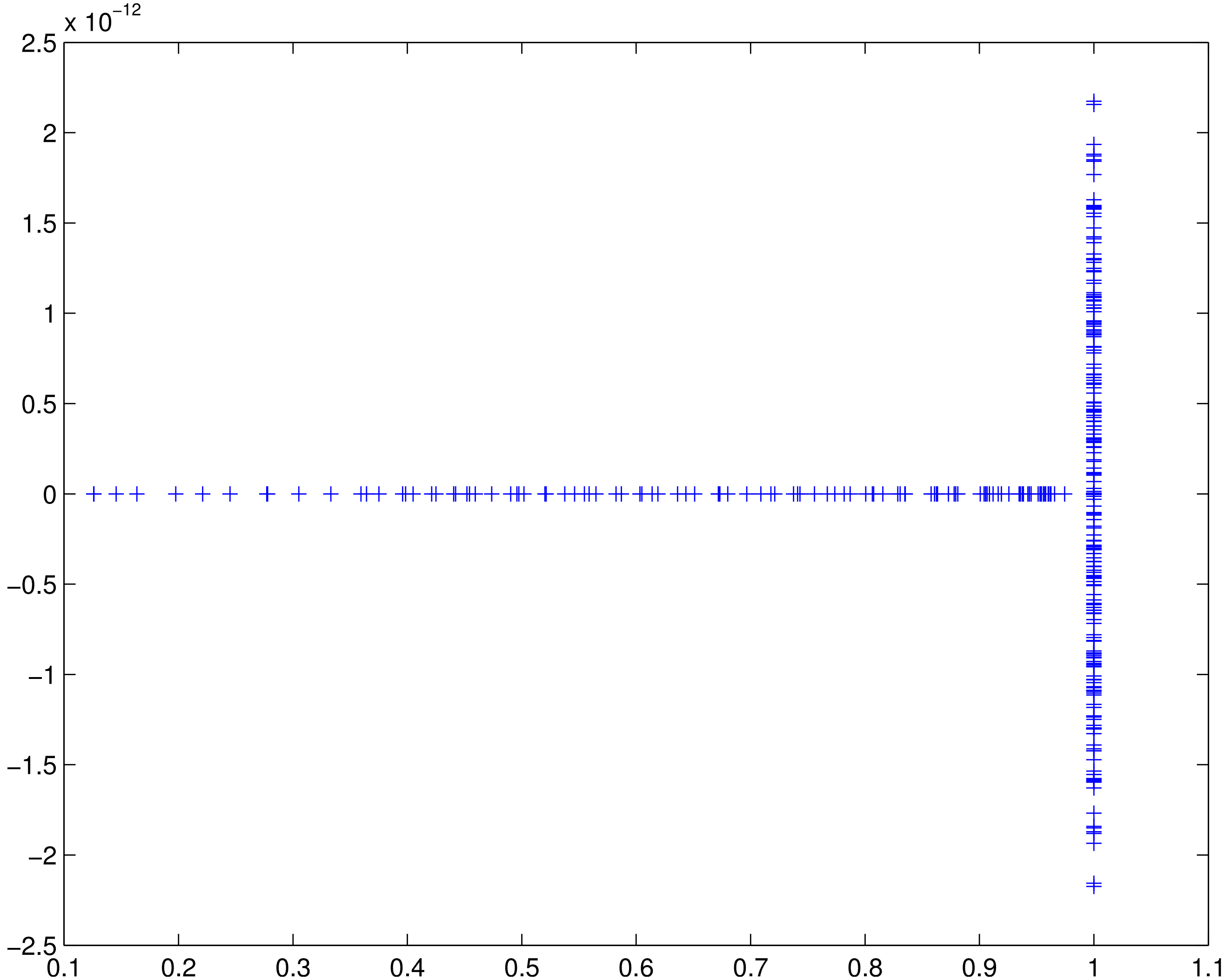
- Interesting case: $\theta = \lambda_{k+1}$
- Question: related approximation to G^{-1}?

- Result: Let $\gamma = 1/(1-\theta)$. Then approx. to G^{-1} is:

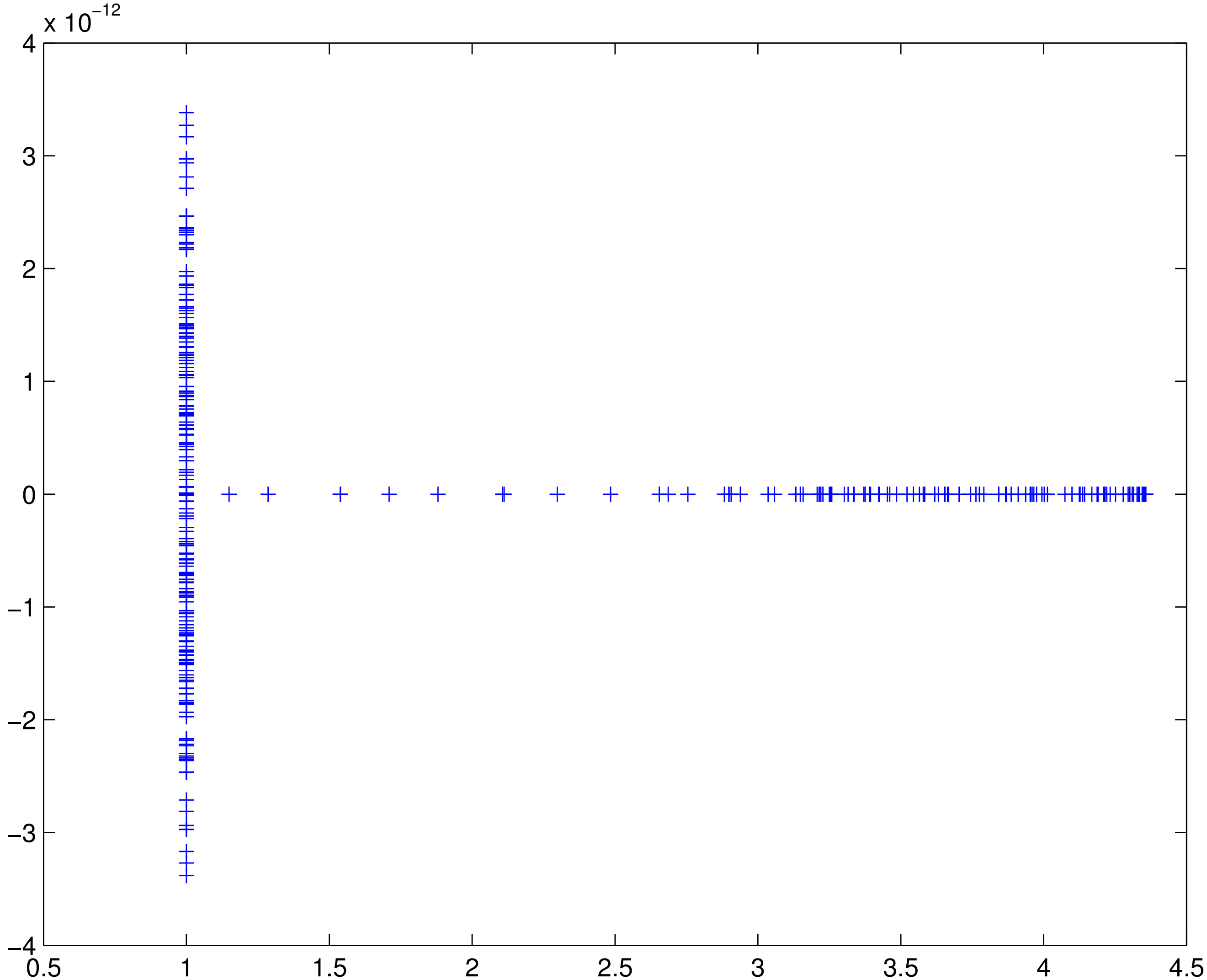
$$G_{k,\theta}^{-1} \equiv \gamma I + U_k[(I - D_k)^{-1} - \gamma I]U_k^T$$

- G_k: k largest eigenvalues of G matched – others set == θ
- $\theta = 0$ yields previous case
- When $\lambda_{k+1} \leq \theta < 1$ we get
- Result: AM^{-1} has
  - $n - s + k$ eigenvalues == 1
  - All others ≥ 1
- Next: An example for a 900×900 Laplacean, 4 domains, $s = 119$

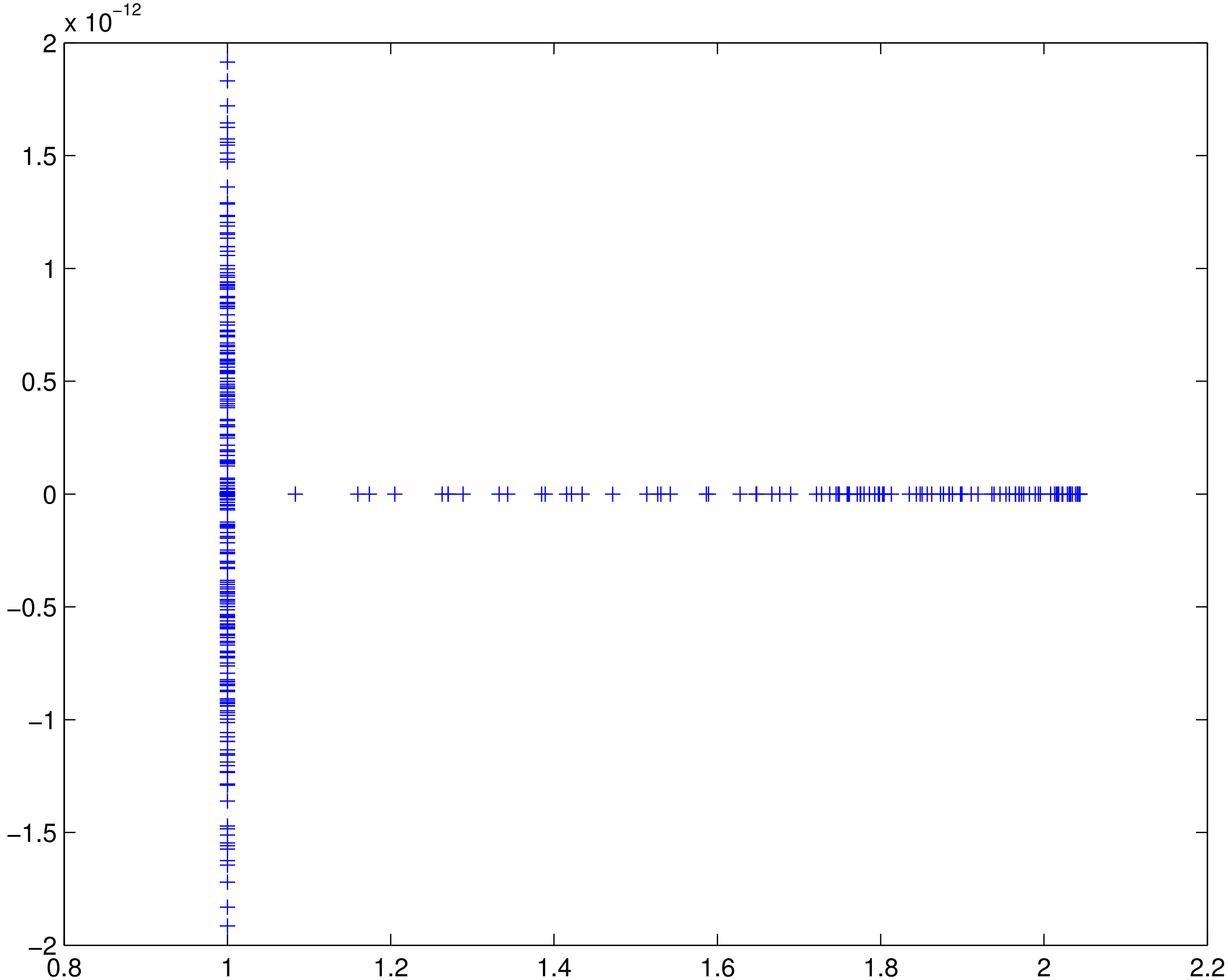
$k = 5$ Eigenvalues of AM^{-1} for the case $\theta = 0$



$k = 5$ Eigenvalues of AM^{-1} for the case $\theta = \lambda_{k+1}$



$k = 15$ Eigenvalues of AM^{-1} for the case $\theta = \lambda_{k+1}$



Proposition Assume θ is so that $\boxed{\lambda_{k+1} \le \theta < 1}$. Then the eigenvalues η_i of AM^{-1} satisfy:

$$1 \le \eta_i \le 1 + \frac{1}{1-\theta} \|A^{1/2} A_0^{-1} E\|_2^2.$$

- Experiments: For the Laplacean (FD) and when $\alpha = 1$,

$$\|A^{1/2} A_0^{-1} E\|_2^2 = \|E^T A_0^{-1} A A_0^{-1} E\|_2 \approx \frac{1}{4}$$

regardless of the mesh-size. Being investigated.

- Best upper bound for $\theta = \lambda_{k+1}$

- Assume above is true and set $\theta = \lambda_{k+1}$. Then $\kappa(AM^{-1}) \le$ constant, if k large enough so that $\lambda_{k+1} \le$ constant.

- i.e., need to capture sufficient part of spectrum

The symmetric indefinite case

- Appeal of this approach over ILU: approximate inverse $\rightarrow$ Not as sensitive to indefiniteness
- Part of the results shown still hold
- But $\lambda_i(H)$ can be > 1 now.
- Parameter α now plays a more important role
- Work still in progress

Example: Take Laplacean on a 30×30 FD grid.

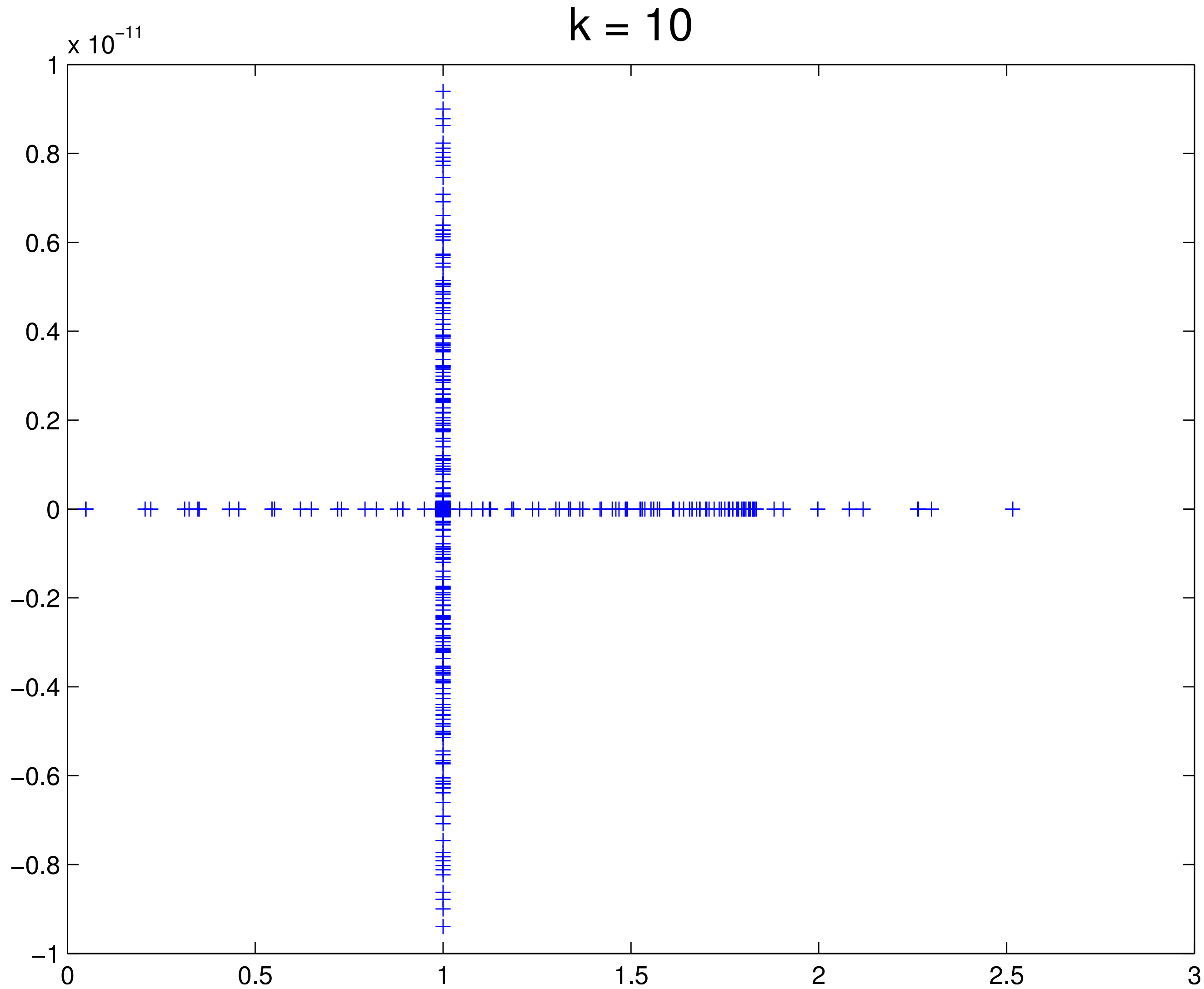
- Subtract $0.4I$ – result: 26 negative eigenvalues

$$\lambda_{min} = -0.379477..., \qquad \lambda_{max} = 7.579477...$$

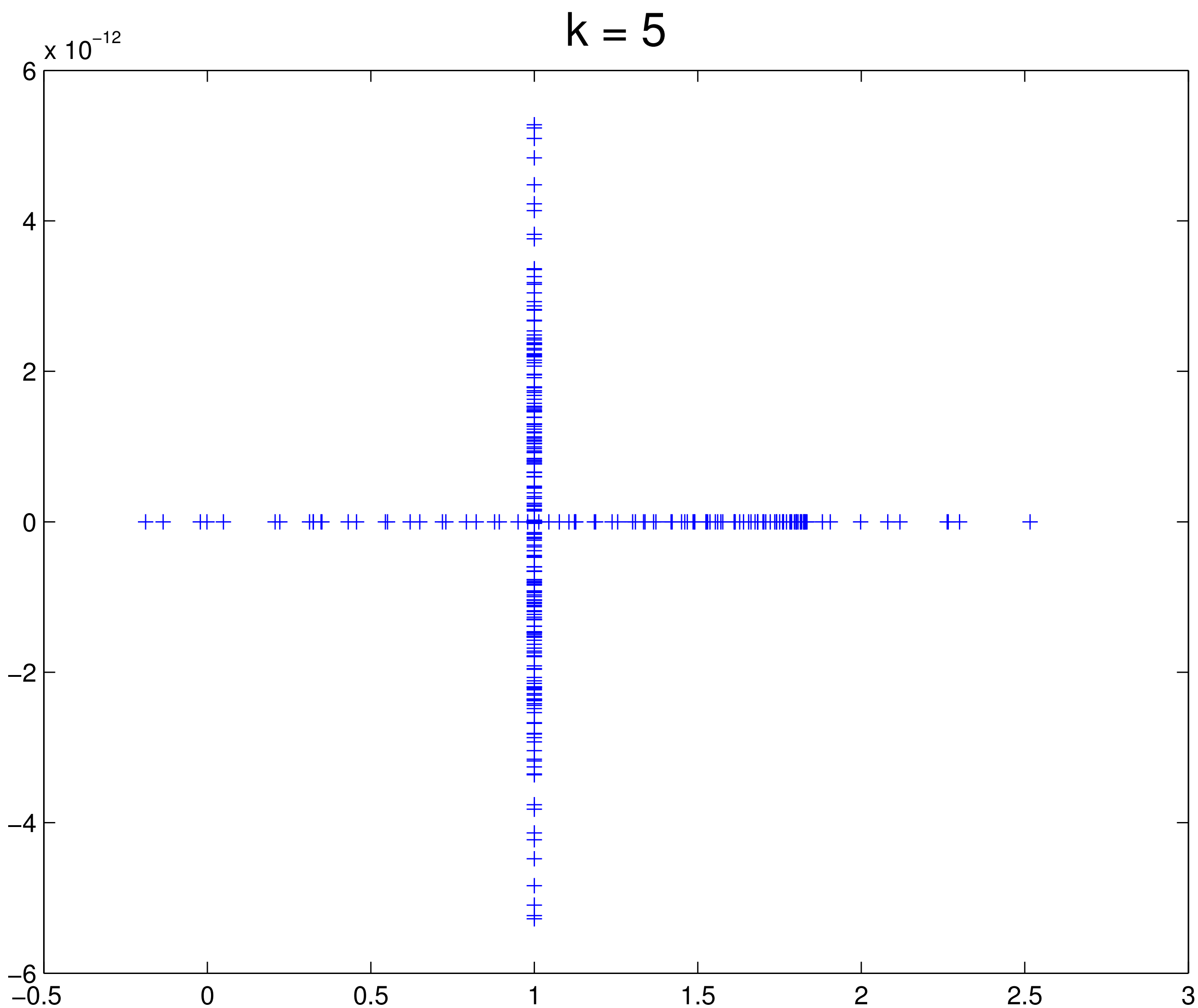
- Use $\alpha = 4.0$, $\theta = 0.9$;
- We do test for $k = 10$ and then $k = 5$

$k = 10$ Eigenvalues of AM^{-1} $[\theta = 0.90, \alpha = 4]$

- All eigenvalues are > 0



$k = 5$ Eigenvalues of AM^{-1} $[\theta = 0.90, \alpha = 4]$



➤ 5 eigenvalues are < 0

Parallel implementations

- Recall :

$$M^{-1} = A_0^{-1}\left[I + EG_{k,\theta}^{-1}E^TA_0^{-1}\right]$$
$$G_{k,\theta}^{-1} = \gamma I + U_k[(I - D_k)^{-1} - \gamma I]U_k^T$$

- Steps involved in applying M^{-1} to a vector x :

ALGORITHM : 1 *Preconditioning operation*

1. $z = A_0^{-1}x$ // $\hat{B}_i$-solves and $C_\alpha-$ solve
2. $y = E^Tz$ // Interior points to interface (Loc.)
3. $y_k = G_{k,\theta}^{-1}y$ // Use Low-Rank approx.
4. $z_k = Ey_k$ // Interface to interior points (Loc.)
5. $u = A_0^{-1}(x + z_k)$ // $\hat{B}_i$-solves and $C_\alpha-$ solve

A_0 Solves Note:

$$A_0 = \begin{pmatrix} \hat{B}_1 & & & & \\ & \hat{B}_2 & & & \\ & & \ddots & & \\ & & & \hat{B}_p & \\ & & & & C_\alpha \end{pmatrix}$$

- Recall $\hat{B}_i = B_i + \alpha^{-2} E_i E_i^T$
- A solve with A_0 amounts to all p $\hat{B}_i$-solves and a C_α-solve
- Can replace C_α^{-1} by a low degree polynomial [Chebyshev]
- Can use any solver for the $\hat{B}_i$'s

Parallel tests: Itasca (MSI)

- HP linux cluster- with Xeons 5560 (“Nehalem”) processors

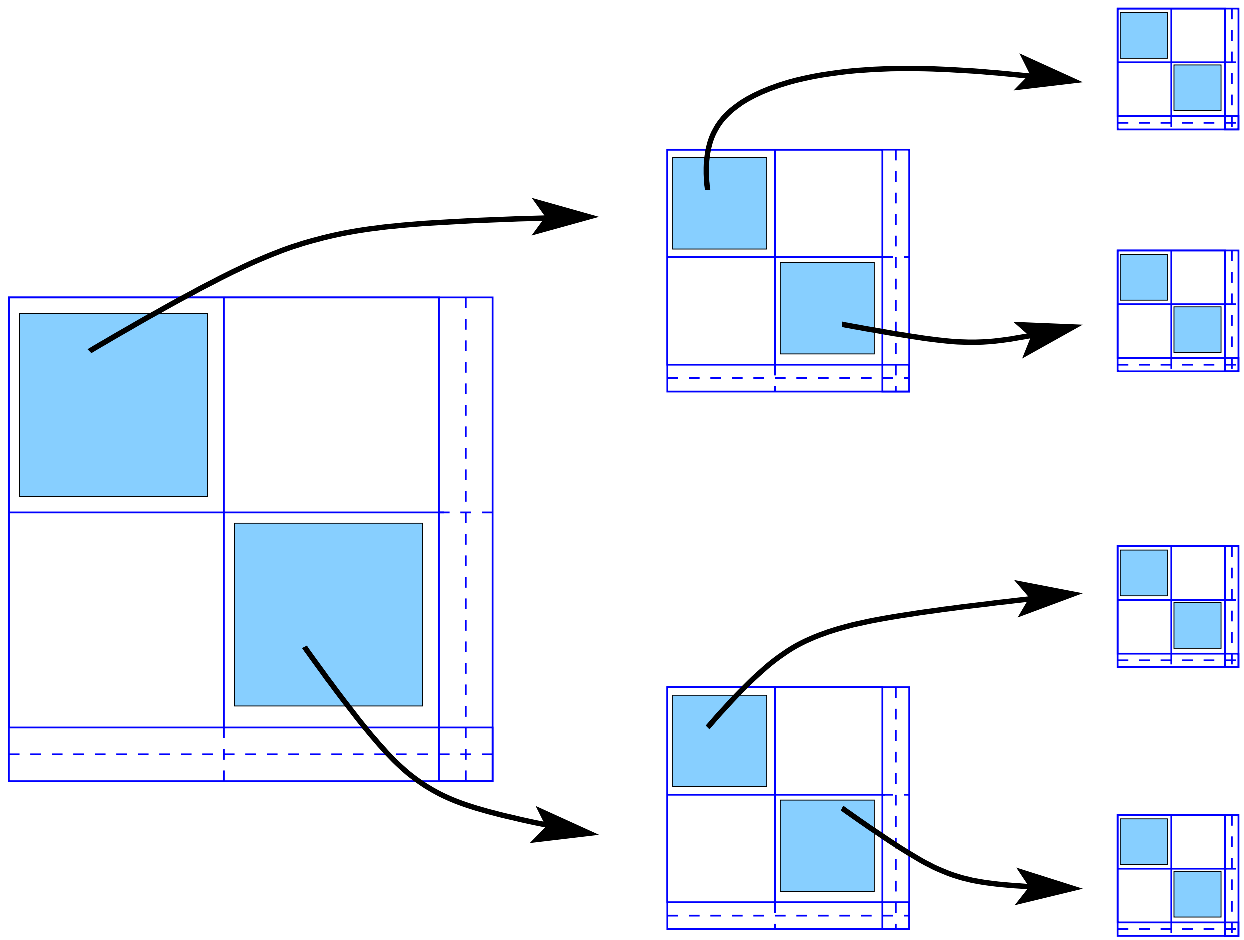
2-D

Mesh	Nproc	Rank	#its	Prec-t	Iter-t
256×256	2	8	29	2.30	.343
512×512	8	16	57	2.62	.747
1024×1024	32	32	96	3.30	1.32
2048×2048	128	64	154	4.84	2.38

3-D

Mesh	Nproc	Rank	#its	Prec-t	Iter-t
$32 \times 32 \times 32$	2	8	12	1.09	.0972
$64 \times 64 \times 64$	16	16	31	1.18	.381
$128 \times 128 \times 128$	128	32	62	2.42	.878

Mixing Divide & Conquer and standard DD



Mixing Divide & Conquer and standard DD

- Must use a two-sided approximation

- Back to recursive version
- Recall $\longrightarrow$

$$A^{-1} = A_0^{-1} + (A_0^{-1}E)G^{-1}(A_0^{-1}E)^T$$
$$G \equiv I - E^T(A_0^{-1}E)$$

- Use a 2-domain partitioning + recursion

- Approximate $A_0^{-1}E$ by a low-rank matrix - get related approximation to G $\longrightarrow$:

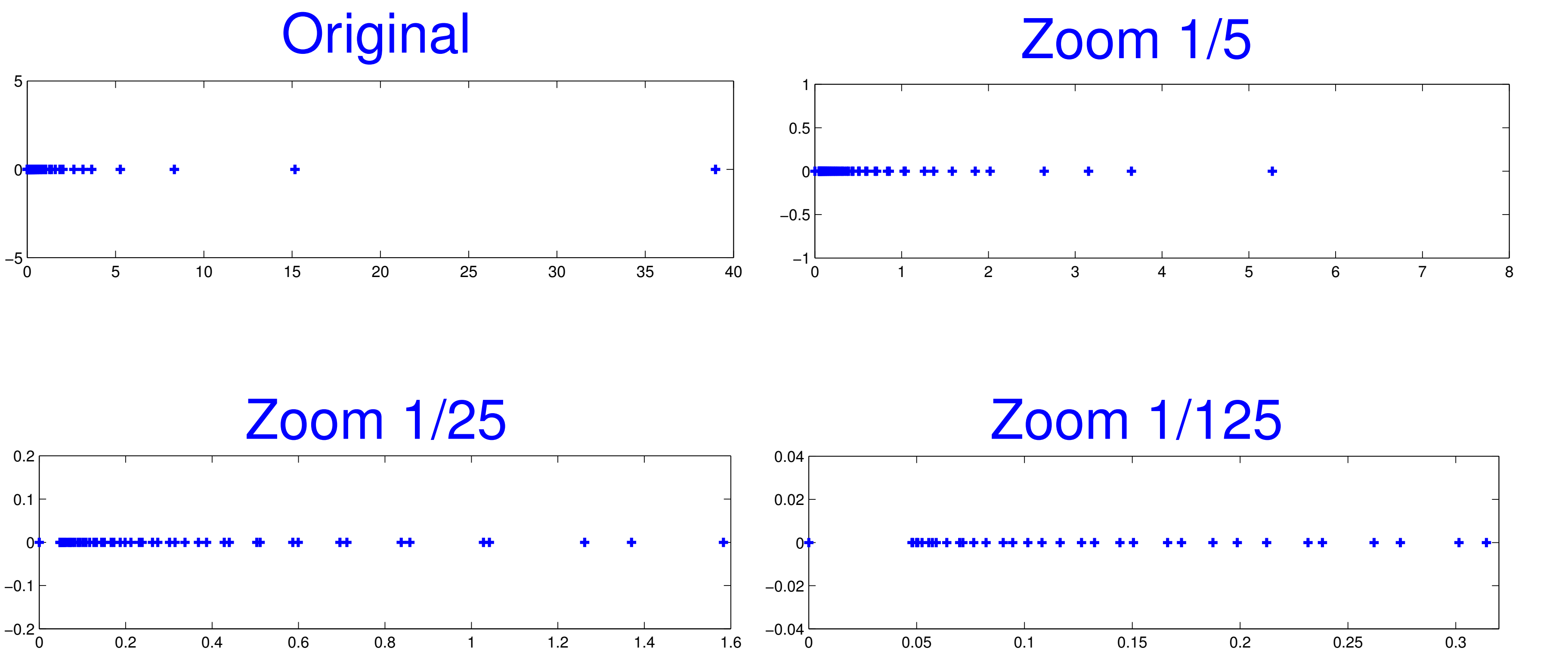
$$A_0^{-1}E \approx U_k V_k^T \longrightarrow$$

$$M^{-1} = A_0^{-1} + U_k X_k^{-1} U_k^T$$
$$X_k = I - U_k^T E V_k$$

- Advantage: Natural way of splitting - no need for balancing

$A^{-1} - A_0^{-1}$ *is nearly low-rank*

- Similar to experiment shown earlier earlier
- Eigenvalues of $A^{-1} - A_0^{-1}$ & 3 zooms closing in on 0



Conclusion

- Promising alternatives to ILUs can be found in new forms of approximate inverse techniques
- Seek “data-sparsity” instead of regular sparsity
- DD approch easier to implement, easier to understand than recursive approach

Advantages of Multilevel Low-Rank preconditioners:

- Approximate inverses $\longrightarrow$ less sensitive to indefiniteness
- Exploit dense computations
- Easy to update