



Multilevel Low-Rank preconditioners

Yousef Saad

*Department of Computer Science
and Engineering*

University of Minnesota

Sparse Days 2013 – Toulouse, June 18, 2013

First:

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

Intro: ILU preconditioners

Problem:

to solve linear systems $Ax = b$

Common approach:

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

Alternatives to ILU preconditioners

➤ What would be a good alternative?

Wish-list:

- A preconditioner requiring few 'irregular' computations
- Something that trades **volume** of computations for **speed**
- If possible something that is robust for indefinite case

➤ Good candidate:

- Multilevel Low-Rank (MLR) approximate inverse preconditioners

Related work:

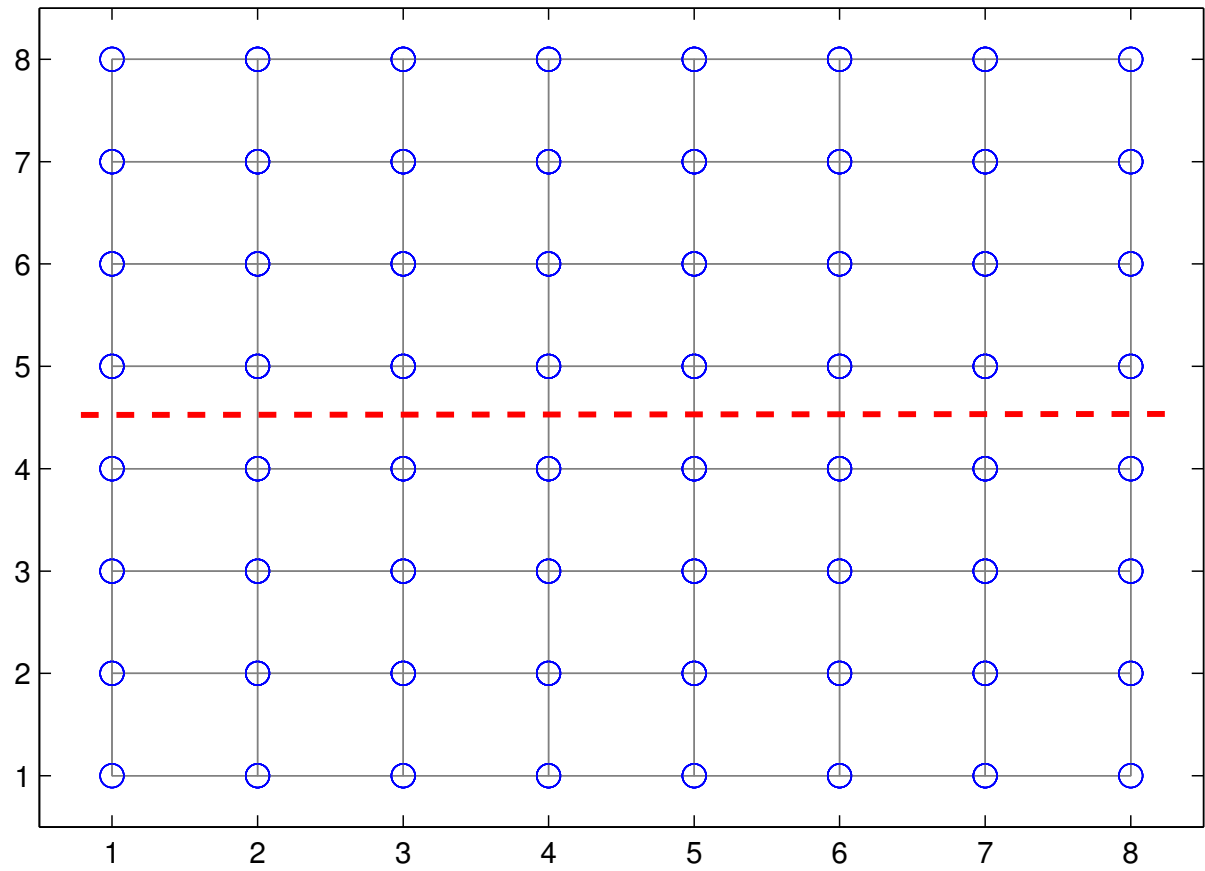
- Work on HSS matrices [e.g., JIANLIN XIA, SHIVKUMAR CHANDRASEKARAN, MING GU, AND XIAOYE S. LI, *Fast algorithms for hierarchically semiseparable matrices*, Numerical Linear Algebra with Applications, 17 (2010), pp. 953–976.]
- Work on H-matrices [Hackbusch, ...]
- Work on ‘balanced incomplete factorizations’ (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) ..]

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

$$A = \left(\begin{array}{ccc|ccc} A_1 & D_2 & & & & \\ D_2 & A_2 & D_3 & & & \\ & \cdots & \cdots & \cdots & & \\ & & D_\alpha & A_\alpha & D_{\alpha+1} & \\ \hline & & & D_{\alpha+1} & A_{\alpha+1} & \cdots \\ & & & & \cdots & \cdots \\ & & & & & D_{n_y} & A_{n_y} \end{array} \right)$$
$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \equiv \begin{pmatrix} A_{11} & \\ & A_{22} \end{pmatrix} + \begin{pmatrix} & A_{12} \\ A_{21} & \end{pmatrix}$$

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:

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

➤ Write 2nd matrix as:

$$\begin{array}{|c|c|} \hline & \\ \hline & -I \\ \hline -I & \\ \hline & \\ \hline \end{array} = \begin{array}{|c|c|} \hline & \\ \hline +I & \\ \hline & +I \\ \hline & \\ \hline \end{array} - \begin{array}{|c|c|} \hline & \\ \hline I & I \\ \hline I & I \\ \hline & \\ \hline \end{array}$$

$$\mathbf{E}^T = \begin{array}{|c|c|} \hline & \\ \hline I & I \\ \hline \end{array} \qquad \mathbf{E} \mathbf{E}^T$$

➤ Above splitting can be rewritten as

$$A = \underbrace{(A + EE^T)}_B - EE^T$$

$$A = B - EE^T,$$
$$B := \begin{pmatrix} B_1 & \\ & B_2 \end{pmatrix} \in \mathbb{R}^{n \times n}, \quad E := \begin{pmatrix} E_1 \\ E_2 \end{pmatrix} \in \mathbb{R}^{n \times n_x},$$

Note: $B_1 := A_{11} + E_1 E_1^T$, $B_2 := A_{22} + E_2 E_2^T$.

➤ Shermann-Morrison formula:

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

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

➤ Note: $E \in \mathbb{R}^{n \times n_x}$, $X \in \mathbb{R}^{n_x \times n_x}$

➤ n_x = number of points in separator [$O(n^{1/2})$ for 2-D mesh, $O(n^{2/3})$ for 3-D mesh]

● Use in a recursive framework

● Similar idea was used for computing the diagonal of the inverse [J. Tang YS '10]

- First thought : approximate X and exploit recursivity

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

- However wont work : cost explodes with # levels

- Alternative: low-rank approx. for $B^{-1}E$

$$B^{-1}E \approx U_k V_k^T,$$

$$U_k \in \mathbb{R}^{n \times k},$$

$$V_k \in \mathbb{R}^{n_x \times k},$$

Multilevel Low-Rank (MLR) algorithm

➤ Method: Use low-rank approx. for $B^{-1}E$

$$B^{-1}E \approx U_k V_k^T,$$

$$U_k \in \mathbb{R}^{n \times k}, \\ V_k \in \mathbb{R}^{n_x \times k},$$

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

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

Preconditioner

$$M^{-1} = B^{-1} + U_k H_k U_k^T, \quad H_k = V_k^T G_k^{-1} V_k$$

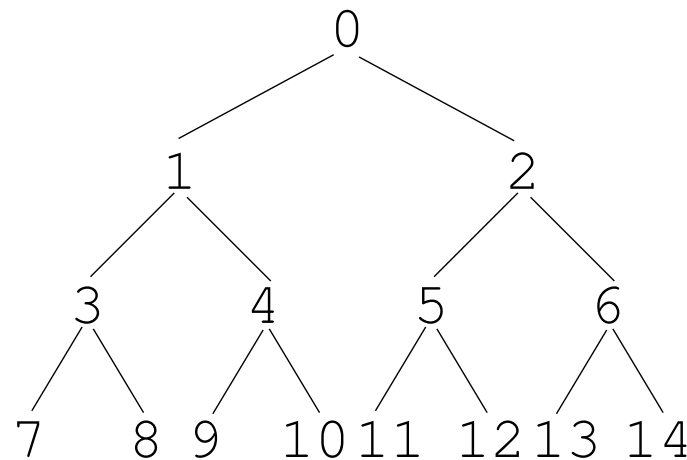
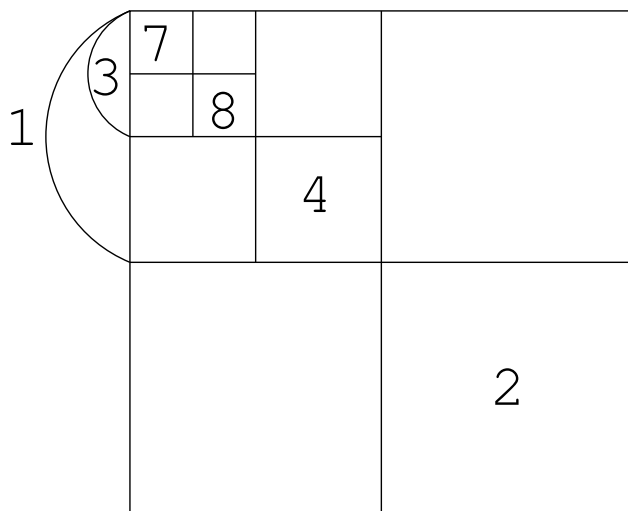
↖ Use recursivity

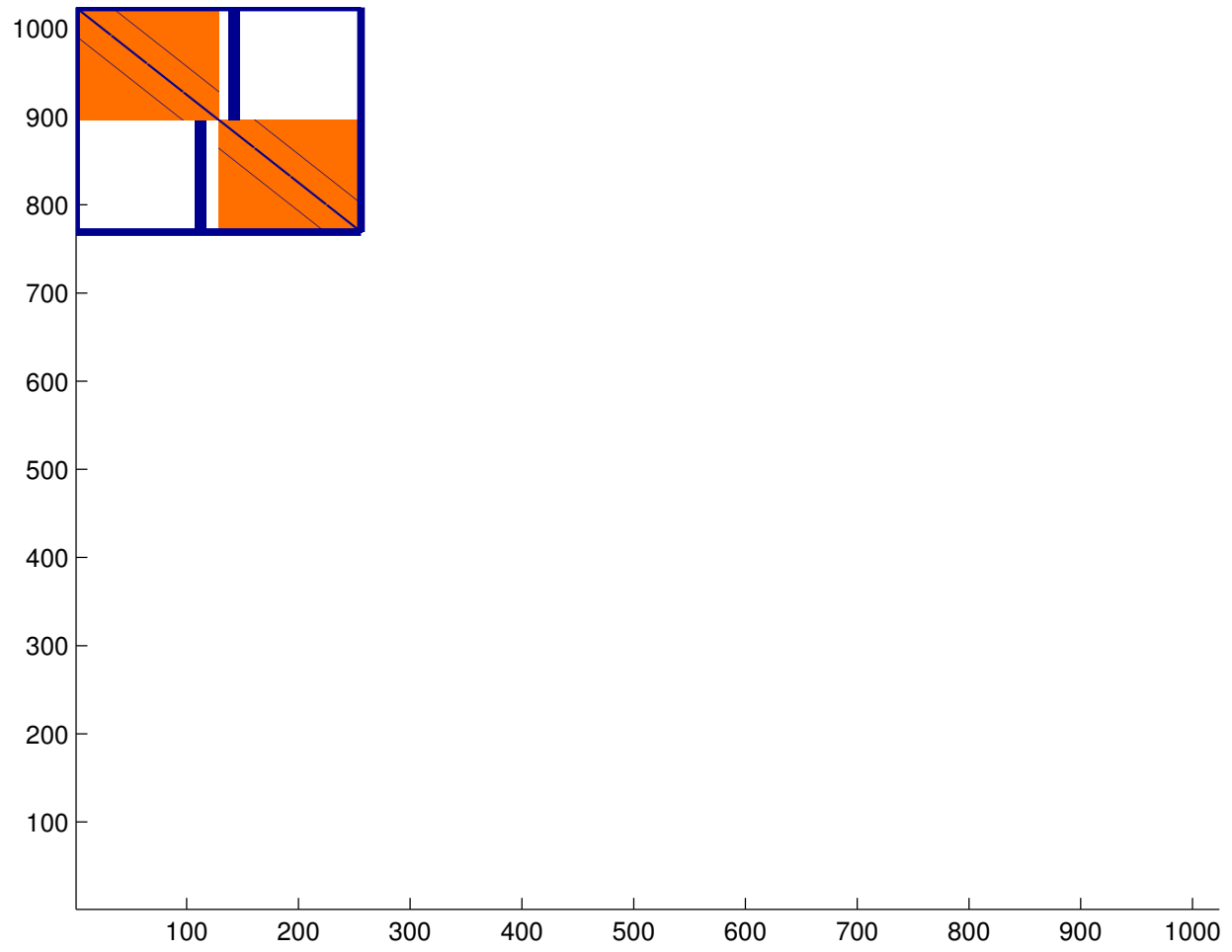
➤ We can show :

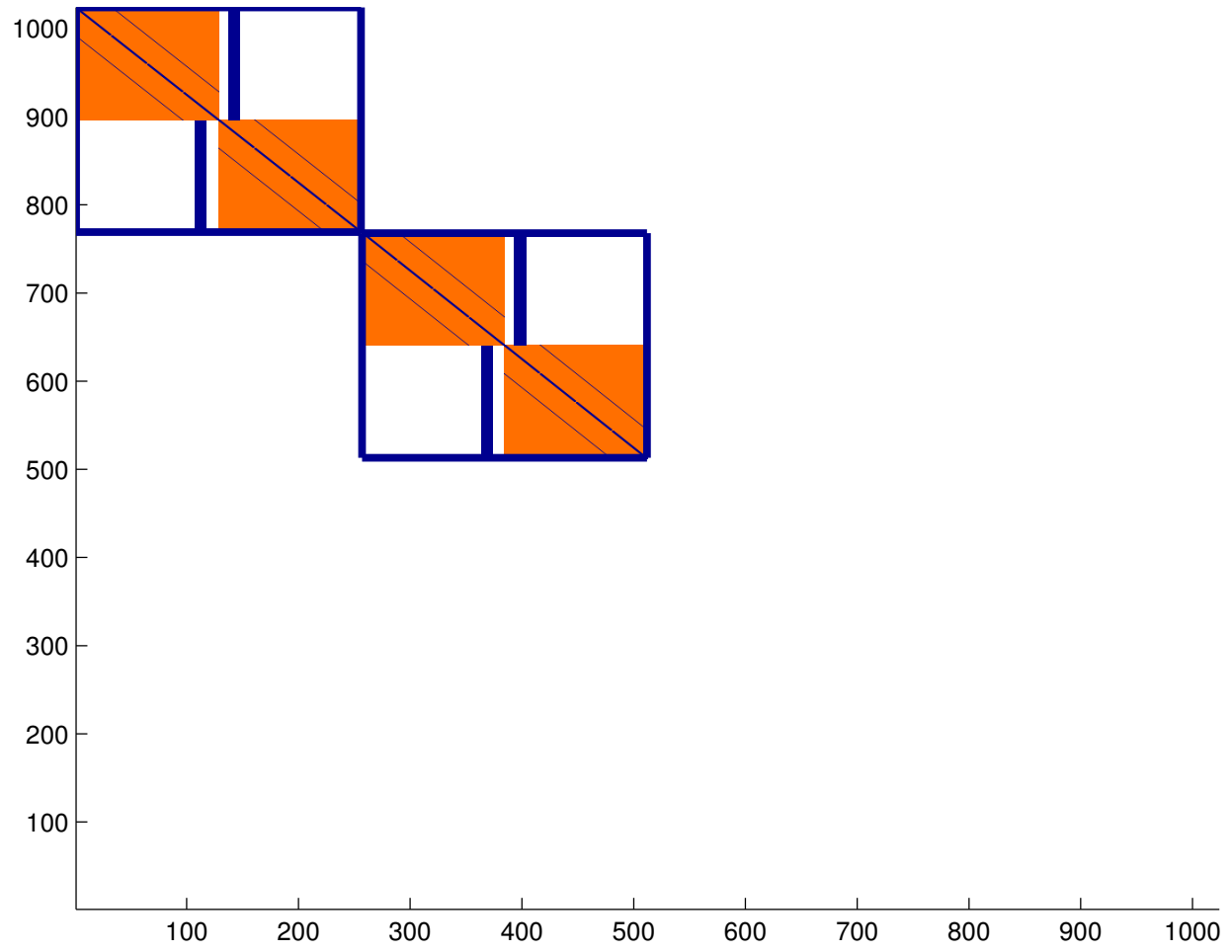
$$H_k = (I - U_k^T E V_k)^{-1} \quad \text{and} \\ H_k^T = H_k$$

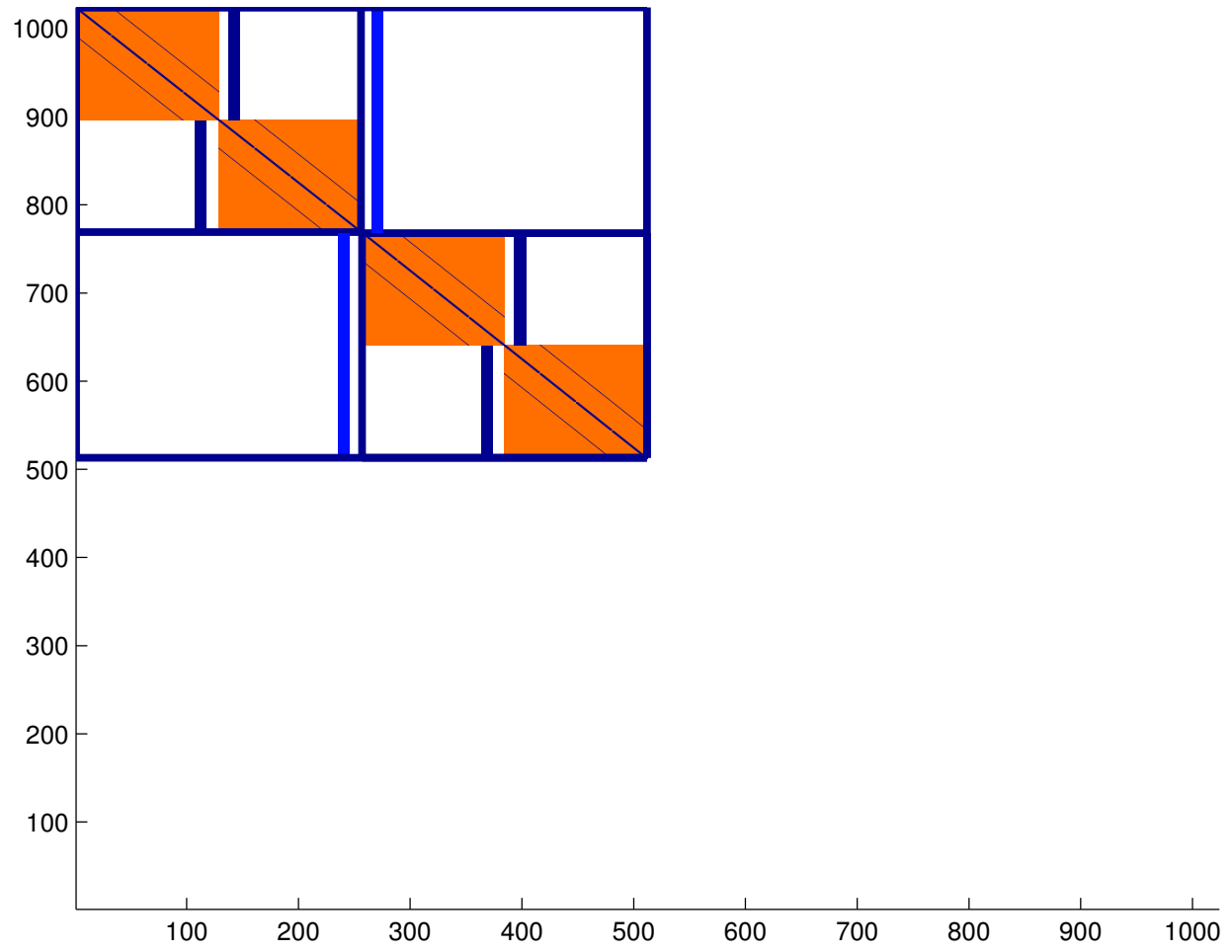
Recursive multilevel framework

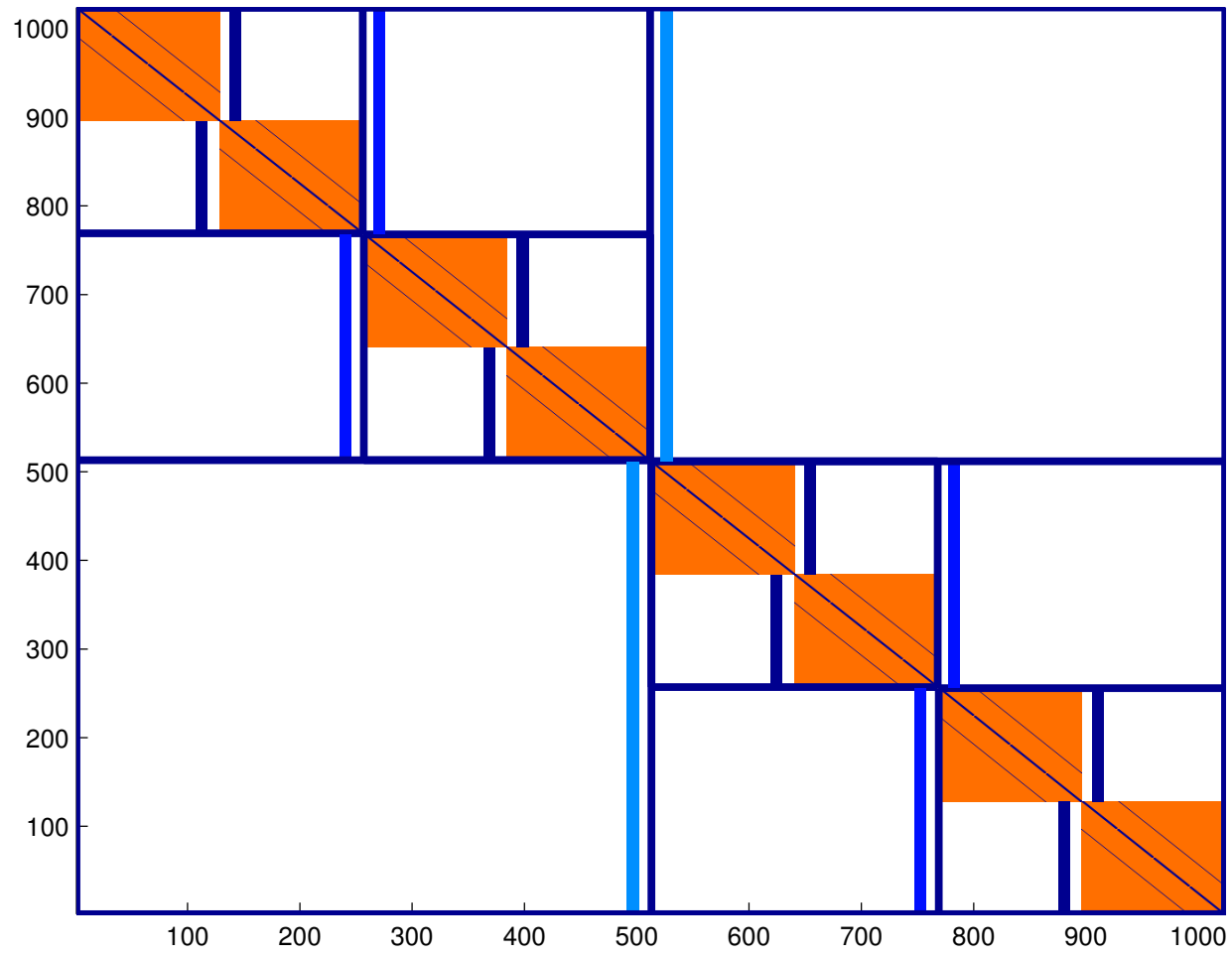
- $A_i = B_i + E_i E_i^T$, $B_i \equiv \begin{pmatrix} B_{i_1} & \\ & B_{i_2} \end{pmatrix}$.
- Next level, set $A_{i_1} \equiv B_{i_1}$ and $A_{i_2} \equiv B_{i_2}$
- Repeat on A_{i_1}, A_{i_2}
- Last level, factor A_i (IC, ILU)
- Binary tree structure:





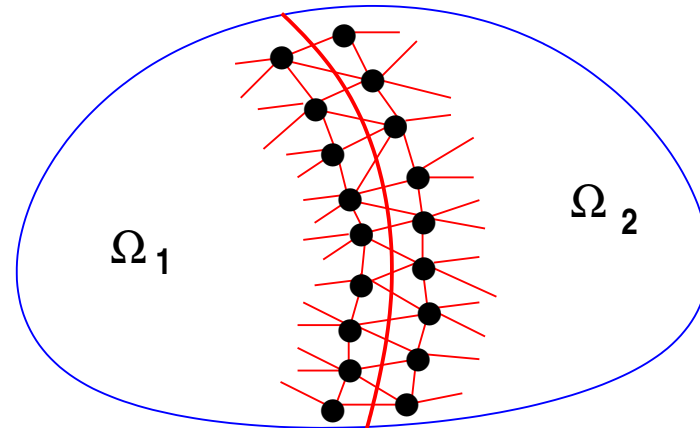






Generalization: Domain Decomposition framework

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.)

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

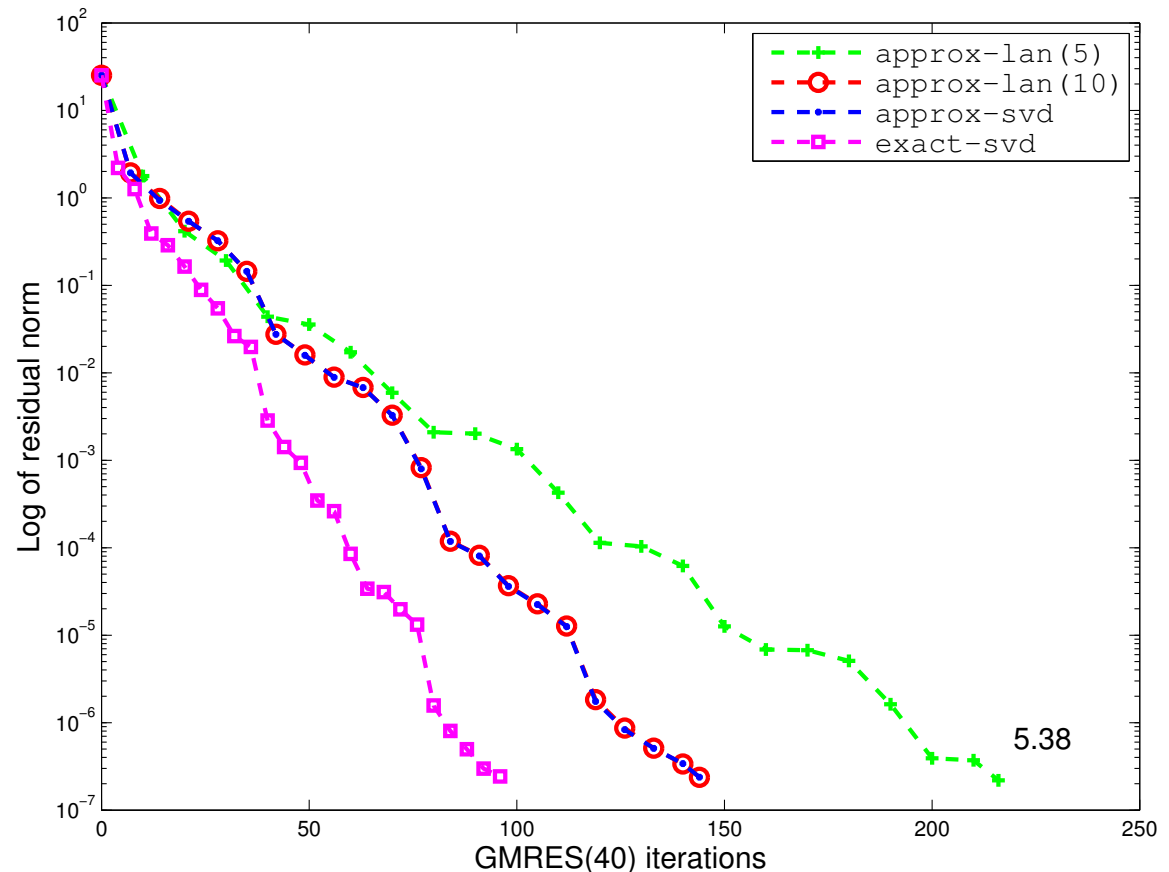
$$PAP^T = \begin{pmatrix} B_1 & \\ & B_2 \end{pmatrix} - EE^T \quad \text{with} \quad B_i = \begin{pmatrix} \hat{B}_i & \hat{F}_1 \\ \hat{F}_i^T & C_i + D_i \end{pmatrix}$$

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

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

First: How to compute the singular vectors?

- Done in Matlab:
- Compute sing. vectors via Matlab's `svds` on (exact) $B^{-1}E$; or approximate $B^{-1}E$; or via bi-Lanczos procedure ($B^{-1}E$ not computed)
- Difference →



MLR($k = 3$) by `svds` vs. `lan(m)`

Theory: 2-level analysis for model problem

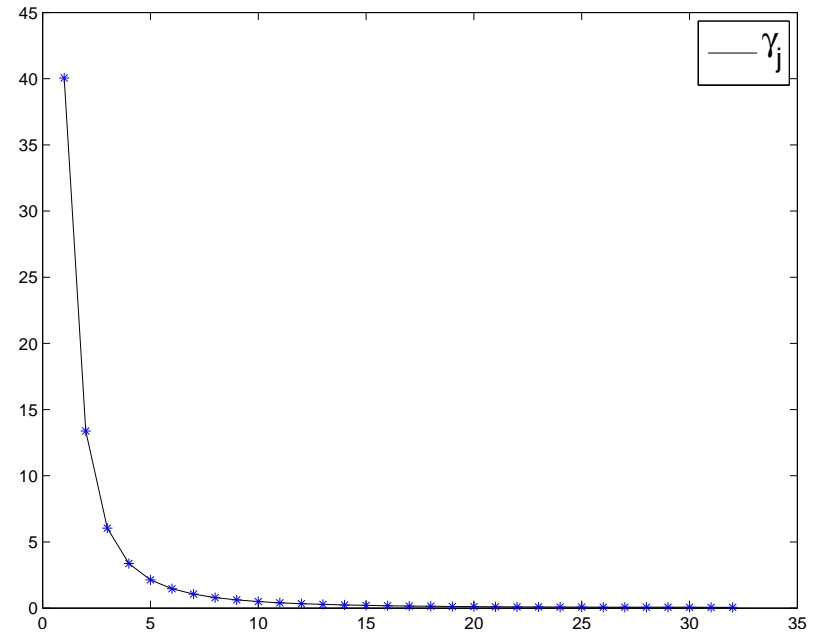
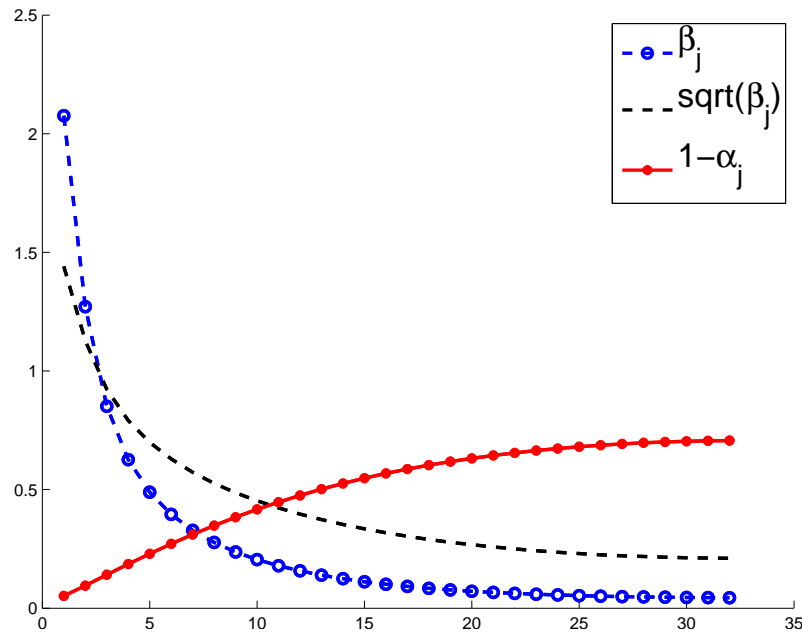
➤ Interested in eigenvalues γ_j of

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

when A = Pure Laplacean .. They are:

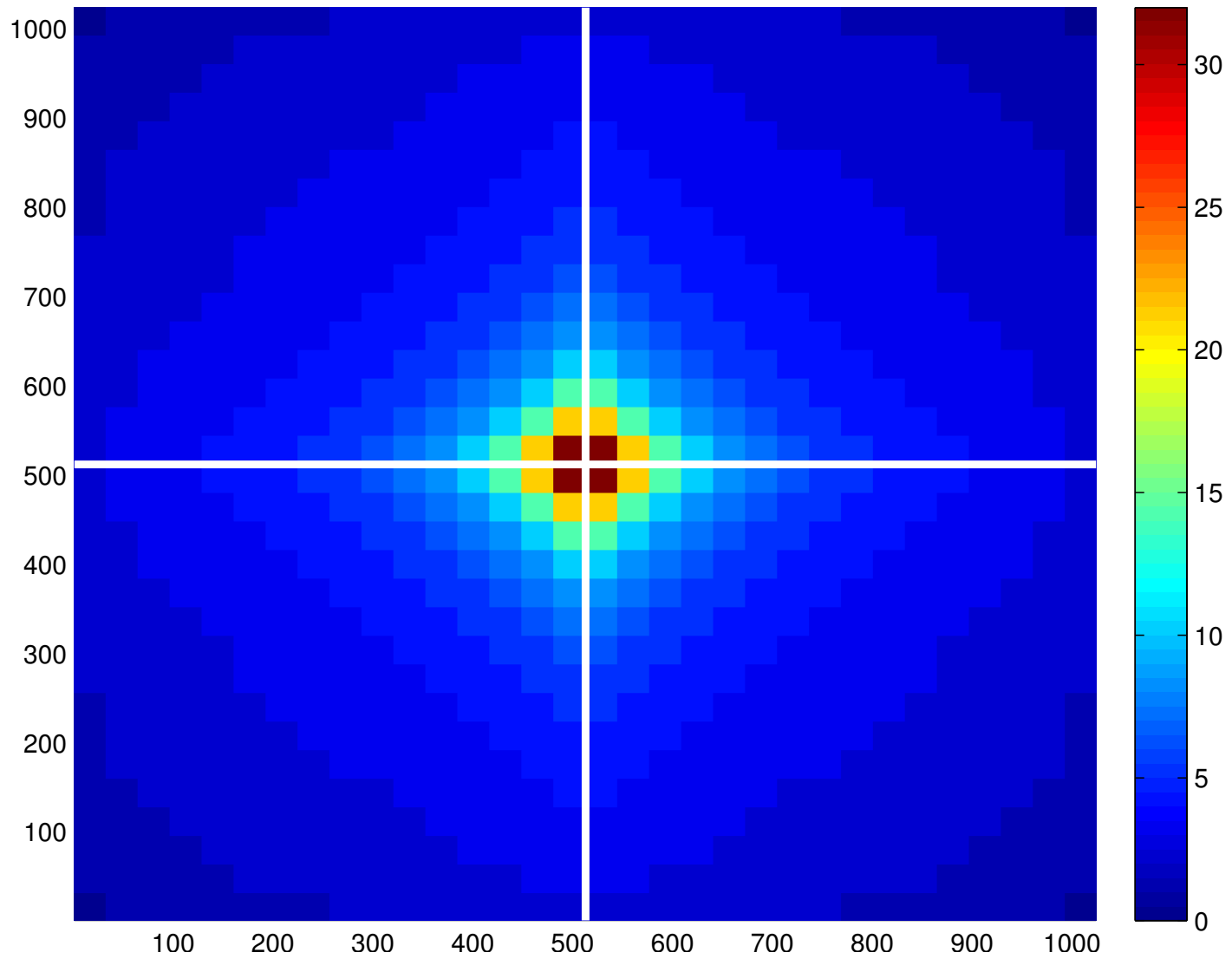
$$\gamma_j = \frac{\beta_j}{1 - \alpha_j}, \quad j = 1, \dots, n_x \quad \text{with:}$$
$$\beta_j = \sum_{k=1}^{n_y/2} \frac{\sin^2 \frac{n_y k \pi}{n_y + 1}}{4 \left(\sin^2 \frac{k \pi}{n_y + 1} + \sin^2 \frac{j \pi}{2(n_x + 1)} \right)^2},$$
$$\alpha_j = \sum_{k=1}^{n_y/2} \frac{\sin^2 \frac{n_y k \pi}{n_y + 1}}{\sin^2 \frac{k \pi}{n_y + 1} + \sin^2 \frac{j \pi}{2(n_x + 1)}}.$$

► Decay of the γ_j 's when $n_x = n_y = 32$.



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

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

Tests: SPD cases

- SPD cases, pure Laplacean ($c = 0$ in previous equations)
- MLR + PCG compared to IC + PCG
- 2-D problems: #lev= 5, rank= 2
- 3-D problems: #lev= 5, 7, 10, rank= 2

Grid	N	ICT-CG				MLR-CG			
		fill	p-t	its	i-t	fill	p-t	its	i-t
256^2	65K	3.1	0.08	69	0.19	3.2	0.45	84	0.12
512^2	262K	3.2	0.32	133	1.61	3.5	1.57	132	1.06
1024^2	1,048K	3.4	1.40	238	15.11	3.5	4.66	215	9.77
$32^{2.64}$	65K	2.9	0.14	33	0.10	3.0	0.46	43	0.08
64^3	262K	3.0	0.66	47	0.71	3.1	3.03	69	0.63
128^3	2,097K	3.0	6.59	89	13.47	3.2	24.61	108	10.27

- Set-up times for MLR preconditioners are higher
- Bear in mind the ultimate target architecture [SIMD...]

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.

Generalization of MLR via DD

- **DD:** `PartGraphRecursive` from METIS
- balancing with diagonals
- higher ranks used in two problems (`cant` and `vibrobox`)
- Show SPD cases first then non-SPD

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
cf1	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
cf2	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

Conclusion

- Promising alternatives to ILUs can be found in new forms of approximate inverse techniques
- Seek “data-sparsity” instead of regular sparsity
- More needs to be done to exploit additional structure in MLR [2-level blocks, compression,...]