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

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

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

\[
E^T = \begin{pmatrix}
I & I
\end{pmatrix}
\]
Above splitting can be rewritten as

\[
A = \left( A + EE^T \right) - EE^T
\]

\[
B := \begin{pmatrix} B_1 \\ B_2 \end{pmatrix} \in \mathbb{R}^{n \times n}, \quad E := \begin{pmatrix} E_1 \\ E_2 \end{pmatrix} \in \mathbb{R}^{n \times n_x},
\]

Note: \( B_1 := A_{11} + E_1 E_1^T \), \( B_2 := A_{22} + E_2 E_2^T \).
Shermann-Morrison formula:

\[ A^{-1} = B^{-1} + B^{-1}E(I - ETB^{-1}E)^{-1}ETB^{-1} \]

\[ A^{-1} = B^{-1} + (B^{-1}E)X^{-1}(B^{-1}E)^T \]
\[ X = I - ETB^{-1}E \]

- **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

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

However won't work: cost explodes with # levels

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

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

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

$V_k \in \mathbb{R}^{n_x \times k}$. 
**Multilevel Low-Rank (MLR) algorithm**

- **Method:** Use low-rank approx. for $B^{-1}E$
  
  \[ B^{-1}E \approx U_k V_k^T, \quad U_k \in \mathbb{R}^{n \times k}, \quad V_k \in \mathbb{R}^{n_x \times k}, \]

- Replace $B^{-1}E$ by $U_k V_k^T$ in $X = I - (E^T B^{-1})E$:
  \[ X \approx G_k = I - V_k U_k^T E, \quad (\in \mathbb{R}^{n_x \times n_x}) \]

  **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:**
  \[
  H_k = (I - U_k^T E V_k)^{-1}
  \]

  and

  \[
  H_k^T = H_k
  \]
Recursive multilevel framework

- $A_i = B_i + E_i E_i^T$, $B_i \equiv \begin{pmatrix} B_{i1} \\ B_{i2} \end{pmatrix}$.

- Next level, set $A_{i1} \equiv B_{i1}$ and $A_{i2} \equiv B_{i2}$

- Repeat on $A_{i1}, A_{i2}$

- 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:

\[
PAP^T = \begin{pmatrix} \hat{B}_1 & \hat{F}_1 \\ \hat{F}_1^T & C_1 \end{pmatrix} \begin{pmatrix} \hat{B}_2 & \hat{F}_2 \\ \hat{F}_2^T & C_2 \end{pmatrix} \begin{pmatrix} 0 & -X \\ -X^T & 0 \end{pmatrix}
\]

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

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

and 
\[
\begin{cases}
D_1 = \alpha^2 I \\
D_2 = \frac{1}{\alpha^2}X^T X
\end{cases}
\]

$\alpha$ used for balancing

Better results when using diagonals instead of $\alpha I$
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 $\rightarrow$

MLR($k = 3$) by svds vs. lan(m)
Theory: 2-level analysis for model problem

Interested in eigenvalues $\gamma_j$ of

$$A^{-1} - B^{-1} = B^{-1}EX^{-1}E^TB^{-1}$$

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 $\gamma_j$’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
**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:

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

- 2-D: $g(x, y) = - (x^2 + y^2 + c) e^{xy}; \quad \Omega = (0, 1)^3$.
- 3-D: $g(x, y, z) = -6 - c (x^2 + y^2 + z^2); \quad \Omega = (0, 1)^3$.
- 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
<table>
<thead>
<tr>
<th>Grid</th>
<th>$N$</th>
<th>ICT-CG</th>
<th>MLR-CG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>fill</td>
<td>p-t</td>
</tr>
<tr>
<td>$256^2$</td>
<td>65K</td>
<td>3.1</td>
<td>0.08</td>
</tr>
<tr>
<td>$512^2$</td>
<td>262K</td>
<td>3.2</td>
<td>0.32</td>
</tr>
<tr>
<td>$1024^2$</td>
<td>1,048K</td>
<td>3.4</td>
<td>1.40</td>
</tr>
<tr>
<td>$32^2.64$</td>
<td>65K</td>
<td>2.9</td>
<td>0.14</td>
</tr>
<tr>
<td>$64^3$</td>
<td>262K</td>
<td>3.0</td>
<td>0.66</td>
</tr>
<tr>
<td>$128^3$</td>
<td>2,097K</td>
<td>3.0</td>
<td>6.59</td>
</tr>
</tbody>
</table>

- 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)
<table>
<thead>
<tr>
<th>Grid</th>
<th>IDLT-GMRES</th>
<th>MLR-GMRES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fill</td>
<td>p-t</td>
</tr>
<tr>
<td>256²</td>
<td>6.5</td>
<td>0.16</td>
</tr>
<tr>
<td>512²</td>
<td>8.4</td>
<td>1.25</td>
</tr>
<tr>
<td>1024²</td>
<td>10.3</td>
<td>10.09</td>
</tr>
<tr>
<td>32² × 64</td>
<td>5.6</td>
<td>0.25</td>
</tr>
<tr>
<td>64³</td>
<td>7.0</td>
<td>1.33</td>
</tr>
<tr>
<td>128³</td>
<td>8.8</td>
<td>15.35</td>
</tr>
</tbody>
</table>
## General symmetric matrices - Test matrices

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

- DD: \texttt{PartGraphRecursive} from \texttt{METIS}
- balancing with diagonals
- higher ranks used in two problems (\texttt{cant} and \texttt{vibrobox})
- Show SPD cases first then non-SPD
<table>
<thead>
<tr>
<th>MATRIX</th>
<th>ICT/ILDLT</th>
<th>MLR-CG/GMRES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fill</td>
<td>p-t</td>
</tr>
<tr>
<td>Andrews</td>
<td>2.6</td>
<td>0.44</td>
</tr>
<tr>
<td>cant</td>
<td>4.3</td>
<td>2.47</td>
</tr>
<tr>
<td>Dubcova2</td>
<td>1.4</td>
<td>0.14</td>
</tr>
<tr>
<td>cfd1</td>
<td>2.8</td>
<td>0.56</td>
</tr>
<tr>
<td>thermal1</td>
<td>3.1</td>
<td>0.15</td>
</tr>
<tr>
<td>cfd2</td>
<td>3.6</td>
<td>1.14</td>
</tr>
<tr>
<td>thermal2</td>
<td>5.3</td>
<td>4.11</td>
</tr>
<tr>
<td>MATRIX</td>
<td>ICT/ILDLT</td>
<td>MLR-CG/GMRES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
<td>--------------</td>
</tr>
<tr>
<td></td>
<td>fill</td>
<td>p-t</td>
</tr>
<tr>
<td>vibrobox</td>
<td>3.3</td>
<td>0.19</td>
</tr>
<tr>
<td>qa8fk</td>
<td>1.8</td>
<td>0.58</td>
</tr>
<tr>
<td>F2</td>
<td>2.3</td>
<td>1.37</td>
</tr>
</tbody>
</table>
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,...]