Sparse factorization
using low rank submatrices

Cleve Ashcraft
LSTC
cleve@lstc.com

2010 MUMPS User Group Meeting
April 15-16, 2010
Toulouse, FRANCE

ftp.lstc.com:outgoing/cleve/MUMPS10_Ashcraft.pdf
LSTC
Livermore Software Technology Corporation
Livermore, California 94550

- Founded by John Hallquist of LLNL, 1980’s
- Public domain versions of DYNA and NIKE codes
- LS-DYNA: implicit/explicit, nonlinear finite element analysis code
- Multiphysics capabilities
  - Fluid/structure interaction
  - Thermal analysis
  - Acoustics
  - Electromagnetics
Multifrontal Algorithm

• very large, sparse $LDL^T$ and $LDU$ factorizations
• tree structure organizes factor storage, solve and factor operations
• medium-to-large sparse linear systems located at each leaf node of the tree
• medium-sized dense linear systems located at each interior node of the tree
• dense matrix-matrix operations at each interior node
• sparse matrix-matrix adds between nodes
Multifrontal tree
Multifrontal tree – polar representation
Multifrontal Algorithm

● 40M equations, present frontier at LSTC
● serial, SMP, MPP, hybrid, GPU
● large problems, > 1M - 10M dof, require out-of-core storage of factor entries, even on distributed memory systems
● IO cost largely hidden during the factorization
● IO cost dominant during the solves
● eigensolver $\Rightarrow$ several right hand sides
● many applications, e.g., Newton’s method, have a single right hand side
Low Rank Approximations

- hierarchical matrices, Hackbusch, Bebendorf, Leborne, others
- semi-separable matrices, Gu, others
- submatrices are numerically rank deficient
- method of choice for Boundary Elements (BEM)
- now applied to Finite Elements (FEM)

\[ A \approx XY^T \]

- storage = \( r(m + n) \) vs \( mn \)
- reduction in ops = \( \frac{r}{\min(m, n)} \)
Multifrontal Algorithm + Low Rank Approximations

- At each leaf node in the multifrontal tree — use standard multifrontal
- At each interior node in the multifrontal tree — low rank matrix-matrix multiplies and sums
- Between nodes —
  low rank matrix sums
- Dramatic reduction in storage
- Dramatic reduction in operations
- Excellent approximation properties for finite element operators
- Our experience is with potential equations, elasticity with solids and shells
Outline

• graph, tree, matrix perspectives
• experiments – 2-D potential equation
• low rank computations
• blocking strategies
• summary
One subgraph

One subtree

One submatrix

\[
\begin{bmatrix}
A_{\Omega_I,\Omega_I} & A_{\Omega_I,S} & A_{\Omega_I,\partial S} \\
A_{\Omega_J,\Omega_J} & A_{\Omega_J,S} & A_{\Omega_J,\partial S} \\
A_{S,\Omega_I} & A_{S,\Omega_J} & A_S,S & A_S,\partial S \\
A_{\partial S,\Omega_I} & A_{\partial S,\Omega_J} & A_{\partial S,S} & A_{\partial S,\partial S}
\end{bmatrix}
\]
Portion of original matrix

A ordered by domains, separator and external boundary

nz = 9052
Portion of factor matrix

L ordered by domains, separator and external boundary

nz = 48899
Schur complement matrix

\[
\begin{bmatrix}
\hat{A}_{S,S} & \hat{A}_{S,\partial S} \\
\hat{A}_{\partial S,S} & \hat{A}_{\partial S,\partial S}
\end{bmatrix}
\]

= 0

Schur complement, separator and external boundary

nz = 20659
Outline

• graph, tree, matrix perspectives
• experiments – 2-D potential equation
• low rank computations
• blocking strategies
• summary
Computational experiments

• Compute $L_{S,S}$, $L_{\partial S,S}$ and $\tilde{A}_{\partial S,\partial S}$
• Find domain decompositions of $S$ and $\partial S$
• Form block matrices, e.g., $L_{S,S} = \sum_{K \geq J} L_{K,J}$
• Find singular value decompositions of each $L_{K,J}$
• Collect all singular values

\[
\{\sigma\} = \sum_{K \geq J} \sum_{i=1}^{\min(|K|,|J|)} \sigma_{i}^{(K,J)}
\]

• Split matrix $L_{S,S} = M_{S,S} + N_{S,S}$ using singular values $\{\sigma\}$.
• We want $\|N_{S,S}\|_F \leq 10^{-14} \|L_{S,S}\|_F$
$255 \times 255$ diagonal block factor $L_{S,S}$

43% dense, relative accuracy $10^{-14}$

$L_{S,S} =$
$754 \times 255$ lower block factor $L_{\partial \mathcal{S}, \mathcal{S}}$
16\% dense, relative accuracy $10^{-14}$

$L_{\partial \mathcal{S}, \mathcal{S}} =$
$754 \times 754$ update matrix $\tilde{A}_{\partial S, \partial S}$

21% dense, relative accuracy $10^{-14}$

$\tilde{A}_{\partial S, \partial S} =$
approximating 255 x 255 separator factor matrix $L_{3,3}$
754 × 255 factor matrix $L_{\partial S, S}$

storage vs accuracy

approximating 754 × 255 interface–separator factor matrix $L_{4,3}$

$\log_{10}(\|M\|_F / \|A\|_F)$ vs fraction of dense storage

- 3 x 1 blocks
- 6 x 2 blocks
- 9 x 3 blocks
- 12 x 4 blocks
754 × 754 update matrix $\hat{A}_{\partial S, \partial S}$

storage vs accuracy

approximating 754 x 754 schur complement matrix Ahat44

$\log_{10}(1 - ||M||_F / ||A||_F)$

fraction of dense storage
Outline

• graph, tree, matrix perspectives
• experiments – 2-D potential equation
• low rank computations
• blocking strategies
• summary
How to compute low rank submatrices?

- **SVD** – singular value decomposition
  \[ A = U \Sigma U^T \]
  where \( U \) and \( V \) are orthogonal and \( \Sigma \) is diagonal
- **Gold standard, expensive, \( O(n^3) \) ops**

- **QR factorization**
  \[ AP = QR \]
  where \( Q \) is orthogonal and \( R \) is upper triangular, and \( P \) is a permutation matrix
- **Silver standard, moderate cost, \( O(rn^2) \) ops**
row norms of $R$ vs singular values of $A$

$754 \times 754$ update matrix $\hat{A}_{\partial S, \partial S}$

$94 \times 94$ submatrix size

near, mid and far interactions
SVD vs $QR$ with column pivoting — Conclusions:

- Column pivoting $QR$ factorization does well.
- Row norms of $R$ track singular values $\sigma$
- The numerical rank of $R$ is greater than needed, but not that much greater
- For more accuracy/less storage, two sided orthogonal factorizations
  - $AP = ULV$, $U$ and $V$ orthogonal, $L$ triangular
  - $PAQ = UBV$, $U$ and $V$ orthogonal, $B$ bidiagonal
  track the singular values very closely.
Type of approximation of submatrices

- **Submatrix** $L_{I,J}$ of $L_{\partial S,S}$, $\|L_{I,J}\|_F = 2.92 \times 10^{-2}$

<table>
<thead>
<tr>
<th>factorization</th>
<th>numerical rank</th>
<th>total entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>—</td>
<td>4900</td>
</tr>
<tr>
<td>QR</td>
<td>20</td>
<td>2681</td>
</tr>
<tr>
<td>ULV</td>
<td>18</td>
<td>2680</td>
</tr>
<tr>
<td>SVD</td>
<td>18</td>
<td>2538</td>
</tr>
</tbody>
</table>

- **Submatrix** $L_{I,J}$ of $L_{\partial S,S}$, $\|L_{I,J}\|_F = 2.04 \times 10^{-3}$

<table>
<thead>
<tr>
<th>factorization</th>
<th>numerical rank</th>
<th>total entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>—</td>
<td>4900</td>
</tr>
<tr>
<td>QR</td>
<td>14</td>
<td>1896</td>
</tr>
<tr>
<td>ULV</td>
<td>10</td>
<td>1447</td>
</tr>
<tr>
<td>SVD</td>
<td>10</td>
<td>1410</td>
</tr>
</tbody>
</table>
Operations with low rank submatrices

\[ A = U_A V_A^T, \quad B = U_B V_B^T, \quad C = U_C V_C^T \]

See Bebendorf, “Hierarchical Matrices”, Chapter 1

- **Multiplication** \( A = B C \), \( \text{rank}(A) \leq \min(\text{rank}(B), \text{rank}(C)) \)

\[
A = U_A V_A^T = \left( U_B V_B^T \right) \left( U_C V_C^T \right) = BC \\
= U_B \left( V_B^T U_C \right) V_C^T \\
= U_B \left( \left( V_B^T U_C \right) V_C^T \right) \\
= \left( U_B \left( V_B^T U_C \right) \right) V_C^T
\]

- **Addition** \( A = B + C \), \( \text{rank}(A) \leq \text{rank}(B) + \text{rank}(C) \)

\[
A = U_A V_A^T = \left( U_B V_B^T \right) + \left( U_C V_C^T \right) = B + C \\
= \left[ U_B \ U_C \right] \left[ V_B \ V_C \right]^T
\]
Multiplication $A = B \ C$

$A = U_A V_A^T$, \quad $B = U_B V_B^T$, \quad $C = U_C V_C^T$

$$A = B \ C = \begin{pmatrix}
\vdots \\
\end{pmatrix} \begin{pmatrix}
\vdots \\
\end{pmatrix}
$$

$$= \left( (m \times h) (h \times l) \right) \left( (l \times k) (k \times n) \right)
$$

$$= (m \times \min(h, k)) (\min(h, k) \times n)$$
Near-near matrix product

near–near interaction, $L_{2,1} L_{2,1}^T$

- $\sigma(L_{21})$
- $\sigma(L_{21}^* L_{21}^T)$
mid-near matrix product

mid–near interaction, $L_{2,1} L_{1,1}^T$

\[
\sigma(L_{21}) \quad \sigma(L_{11}) \quad \sigma(L_{21}^* L_{11}^T)
\]
far-near matrix product

far–near interaction, $L_{5,1}^T L_{6,1}$
mid-mid matrix product

mid–mid interaction, $L_{1,1} L_{1,1}^T$

\[
\sigma(L_{11})
\]

\[
\sigma(L_{11}^* L_{11}^T)
\]
far-mid matrix product

mid–far interaction, $L_{6,1} L_{1,1}^T$

$\sigma(L_{61})$  
$\sigma(L_{61}L_{61}^T)$
far-far matrix product

far–far interaction, $L_{6,1} L_{6,1}^T$

\[ \sigma(L_{61}) \quad \sigma(L_{61}^* L_{61}^T) \]
Addition $A = B + C$

$\text{rank}(A) \leq \text{rank}(B) + \text{rank}(C)$

$$A = U_A V_A^T = (U_B V_B^T) + (U_C V_C^T) = B + C$$

$$= [U_B \ U_C] \ [V_B \ V_C]^T$$

$$= (Q_1 R_1) (Q_2 R_2)^T$$

$$= Q_1 \left( R_1 R_2^T \right) Q_2^T$$

$$= Q_1 (Q_3 R_3) Q_2^T$$

$$= (Q_1 Q_3) \left( R_3 Q_2^T \right)$$

$$= (Q_1 Q_3) \left( Q_2 R_3^T \right)^T$$

- $R_1 R_2^T$ usually has low numerical rank
- examples follow
update matrix, diagonal block
\[ \hat{A}_{3,3} = L_{3,1} L_{3,1}^T + L_{3,2} L_{3,2}^T + L_{3,3} L_{3,3}^T \]
update matrix, mid-distance off-diagonal block

\[ \hat{A}_{5,3} = L_{5,1}L_{3,1}^T + L_{5,2}L_{3,2}^T + L_{5,3}L_{3,3}^T \]
update matrix, far-distance off-diagonal block

\[ \hat{A}_{7,3} = L_{7,1}L^T_{3,1} + L_{7,2}L^T_{3,2} + L_{7,3}L^T_{3,3} \]
Outline

• graph, tree, matrix perspectives
• experiments – 2-D potential equation
• low rank computations
• blocking strategies
• summary
Blocking Strategies

- Active data structures
  
  \[
  \begin{bmatrix}
  L_{\bar{J},\bar{J}} \\
  L_{\partial \bar{J},\bar{J}} \quad \tilde{A}_{\partial \bar{J},\partial J}
  \end{bmatrix}
  \]

- partition \( \bar{J} \) and \( \partial \bar{J} \) independently

- for 2-d problems, \( \bar{J} \) and \( \partial \bar{J} \) are 1-d manifolds

- for 3-d problems, \( \bar{J} \) and \( \partial \bar{J} \) are 2-d manifolds

- we need mesh partitioning of the separator and the boundary of a region \( \bar{J} \)

- each index set of a partition is a segment
Segment partition

Segment wirebasket domain decomposition
\[ L_{J,J} = \sum_{\sigma, \tau} L_{\sigma,\tau} \quad \sigma \times \tau \subseteq J \times J \]

\[ L_{\partial J,J} = \sum_{\sigma, \tau} L_{\sigma,\tau} \quad \sigma \times \tau \subseteq \partial J \times J \]

\[ \hat{A}_{\partial J,\partial J} = \sum_{\sigma, \tau} \hat{A}_{\sigma,\tau} \quad \sigma \times \tau \subseteq \partial J \times \partial J \]
Strategy I

- The partition of $\partial J$ (local to node $J$) conforms to the partitions of ancestors $K$, $K \cap \partial J \neq \emptyset$

- Advantages:
  - Update assembly is simplified
  
  \[
  \hat{A}_{\sigma_2,\tau_2}^{(p(J))} = \hat{A}_{\sigma_2,\tau_2}^{(p(J))} + \hat{A}_{\sigma_1,\tau_1}^{(J)}
  \]

  where $\sigma_1 \subseteq \sigma_2$, $\tau_1 \subseteq \tau_2$

  - One destination for each $\hat{A}_{\sigma_1,\tau_1}^{(J)}$

- Disadvantages:
  - Partition of $\partial J$ can be fragmented, more segments, smaller size, less efficient storage
Strategy II

- The partition of $\partial J$ (local to node $J$) need not conform to the partitions of ancestors

- Advantages:
  - Partition of $\partial J$ can be optimized better since $\partial J$ is small and localized

- Disadvantages:
  - Update assembly is more complex

\[
\widehat{A}_{\sigma_1 \cap \sigma_2, \tau_1 \cap \tau_2}^{(p(J))} = \widehat{A}_{\sigma_1 \cap \sigma_2, \tau_1 \cap \tau_2}^{(p(J))} + \widehat{A}_{\sigma_1 \cap \sigma_2, \tau_1 \cap \tau_2}^{(J)}
\]

where $\sigma_1 \cap \sigma_2 \neq \emptyset$, $\tau_1 \cap \tau_2 \neq \emptyset$

- Several destinations for a submatrix $\widehat{A}_{\sigma_1, \tau_1}$
Outline

• graph, tree, matrix perspectives
• experiments – 2-D potential equation
• low rank computations
• blocking strategies
• summary
Multifrontal tree

Multifrontal Factorization

- each leaf node is a $d$-dimensional sparse FEM matrix
- each interior node is a $(d - 1)$-dimensional dense BEM matrix
- use low rank storage and computation at each interior node
Call to action

• progress to date in low rank factorizations driven by iterative methods
• work needed from direct methods community
• start from industrial strength multifrontal code
  – serial, SMP, MPP, hybrid, GPU
  – pivoting for stability, out-of-core, singular systems, null spaces
Call to action

• Many challenges
  – partition of space, partition of interface
  – added programming complexity of low rank matrices
  – challenges to pivot for stability
  – challenges/opportunities to implement in parallel

• Payoff will be huge
  – reduction in storage footprint
  – reduction in computational work
  – take direct methods to next level of problem size