LU factorization and its communication avoiding version

Laura Grigori

EPFL and PSI

October 15, 2024

Plan

LU factorization

Block LU factorization

Communication avoiding LU factorization

Norms and other notations

$$
\begin{aligned}
\|\mathbf{A}\|_F &= \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}|a_{ij}|^2} \\
\|\mathbf{A}\|_2 &= \sigma_{max}(A) \\
\|\mathbf{A}\|_\infty &= \max_{1\le i\le n}\sum_{j=1}^{n}|a_{ij}| \\
\|\mathbf{A}\|_1 &= \max_{1\le j\le n}\sum_{i=1}^{n}|a_{ij}|
\end{aligned}
$$

Inequalities $|x| \le |y|$ and $|\mathbf{A}| \le |\mathbf{B}|$ hold componentwise.

Plan

LU factorization

Block LU factorization

Communication avoiding LU factorization

Algebra of the LU factorization

LU factorization

Compute the factorization $\mathbf{\Pi A} = \mathbf{LU}$

Example

Given the matrix

$$\mathbf{A} = \begin{pmatrix} 3 & 1 & 3 \\ 6 & 7 & 3 \\ 9 & 12 & 3 \end{pmatrix}$$

Let

$$\mathbf{M}_1 = \begin{pmatrix} 1 & & \\ -2 & 1 & \\ -3 & & 1 \end{pmatrix}, \quad \mathbf{M}_1\mathbf{A} = \begin{pmatrix} 3 & 1 & 3 \\ 0 & 5 & -3 \\ 0 & 9 & -6 \end{pmatrix}$$

Algebra of the LU factorization

- In general

$$\mathbf{A}^{(k+1)} = \mathbf{M}_k \mathbf{A}^{(k)} := \begin{pmatrix} \mathbf{I}_{k-1} & & & & \\ & 1 & & & \\ & -m_{k+1,k} & 1 & & \\ & \dots & & \ddots & \\ & -m_{n,k} & & & 1 \end{pmatrix} \mathbf{A}^{(k)}, \text{ where}$$

$$\mathbf{M}_k = \mathbf{I} - m_k e_k^T, \quad \mathbf{M}_k^{-1} = \mathbf{I} + m_k e_k^T$$

where e_k is the k-th unit vector, $m_k = (0, \dots, 0, 1, m_{k+1,k}, \dots, m_{n,k})^T$, $e_i^T m_k = 0, \forall i \leq k$

- The factorization can be written as

$$\mathbf{M}_{n-1} \dots \mathbf{M}_1 \mathbf{A} = \mathbf{A}^{(n)} = \mathbf{U}$$

Algebra of the LU factorization

- We obtain

$$
\begin{aligned}
\mathbf{A} &= \mathbf{M}_1^{-1} \dots \mathbf{M}_{n-1}^{-1} \mathbf{U} \\
&= (\mathbf{I} + m_1 e_1^T) \dots (I + m_{n-1} e_{n-1}^T) \mathbf{U} \\
&= \left(\mathbf{I} + \sum_{i=1}^{n-1} m_i e_i^T \right) \mathbf{U} \\
&= \begin{pmatrix} 1 & & & \\ m_{21} & 1 & & \\ \vdots & \vdots & \ddots & \\ m_{n1} & m_{n2} & \dots & 1 \end{pmatrix} \mathbf{U} = \mathbf{LU}
\end{aligned}
$$

The need for pivoting

- For stability, avoid division by small diagonal elements
- For example

$$\mathbf{A} = \begin{pmatrix} 0 & 3 & 3 \\ 3 & 1 & 3 \\ 6 & 2 & 3 \end{pmatrix} \tag{1}$$

  has an LU factorization if we permute the rows of matrix A

$$\mathbf{\Pi A} = \begin{pmatrix} 6 & 2 & 3 \\ 0 & 3 & 3 \\ 3 & 1 & 3 \end{pmatrix} = \begin{pmatrix} 1 & & \\ & 1 & \\ 0.5 & & 1 \end{pmatrix} \cdot \begin{pmatrix} 6 & 2 & 3 \\ & 3 & 3 \\ & & 1.5 \end{pmatrix} \tag{2}$$

- Partial pivoting allows to bound the multipliers $m_{ik} \leq 1$ and hence $|L| \leq 1$
- Factorization referred to as LU with partial pivoting (LUPP) or also Gaussian elimination with partial pivoting GEPP

Solving $Ax = b$ by using Gaussian elimination

Composed of 4 steps

1. Factor $\mathbf{\Pi A} = \mathbf{LU}$, $(2/3)n^3)$ flops
2. Compute $\mathbf{\Pi}b$ to solve $\mathbf{LU}x = \mathbf{\Pi}b$
3. Forward substitution: solve $\mathbf{L}y = \mathbf{\Pi} * b$, n^2 flops
4. Backward substitution: solve $\mathbf{U}x = y$, n^2 flops

Wilkinson's backward error stability result

Growth factor g_W defined as

$$g_W = \frac{\max_{i,j,k} |a_{ij}^k|}{\max_{i,j} |a_{ij}|}$$

Note that

$$|u_{ij}| = |a_{ij}^i| \leq g_W \max_{i,j} |a_{ij}|$$

Theorem (Wilkinson's backward error stability result, see also [N.J.Higham, 2002] for more details)

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ and let $\hat{x}$ be the computed solution of $Ax = b$ obtained by using GEPP. Then

$$(\mathbf{A} + \Delta\mathbf{A})\hat{x} = b, \qquad \|\Delta\mathbf{A}\|_\infty \leq n^2 \gamma_{3n} g_W(n) \|\mathbf{A}\|_\infty,$$

where $\gamma_n = n\epsilon/(1 - n\epsilon)$, ϵ is machine precision and assuming $n\epsilon < 1$.

The growth factor

- The LU factorization is backward stable if the growth factor is small (grows linearly with n).
- For partial pivoting, the growth factor $g(n) \leq 2^{n-1}$, and this bound is attainable.
- In practice it is on the order of $n^{2/3} - n^{1/2}$

Exponential growth factor for Wilkinson matrix

$$\mathbf{A} = diag(\pm 1) \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ -1 & 1 & 0 & \ldots & 0 & 1 \\ -1 & -1 & 1 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 1 \\ -1 & -1 & \cdots & -1 & 1 & 1 \\ -1 & -1 & \cdots & -1 & -1 & 1 \end{bmatrix}$$

Experimental results for special matrices

Several errror bounds for GEPP

matrix	cond(A,2)	g_W	$\|\|L\|\|_1$	cond(U,2)	$\frac{\|\|\Pi A - LU\|\|_F}{\|\|A\|\|_F}$
hadamard	1.0E+0	4.1E+3	4.1E+3	5.3E+5	0.0E+0
randsvd	6.7E+7	4.7E+0	9.9E+2	1.4E+10	5.6E-15
chebvand	3.8E+19	2.0E+2	2.2E+3	4.8E+22	5.1E-14
frank	1.7E+20	1.0E+0	2.0E+0	1.9E+30	2.2E-18
hilb	8.0E+21	1.0E+0	3.1E+3	2.2E+22	2.2E-16

- Two reasons considered to be important for the average case stability [Trefethen and Schreiber, 90]:
  - the multipliers in **L** are small
  - the correction introduced at each elimination step is of rank 1

Plan

LU factorization

Block LU factorization

Communication avoiding LU factorization

Block formulation of the LU factorization

Partitioning of matrix A of size $n \times n$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$

where $\mathbf{A}_{11}$ is of size $b \times b$, $\mathbf{A}_{21}$ is of size $(m-b) \times b$, $\mathbf{A}_{12}$ is of size $b \times (n-b)$ and $\mathbf{A}_{22}$ is of size $(m-b) \times (n-b)$.

Block LU algebra

The first iteration computes the factorization:

$$\mathbf{\Pi}_1^T \mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}}_{11} & \bar{\mathbf{A}}_{12} \\ \bar{\mathbf{A}}_{21} & \bar{\mathbf{A}}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{21} & \mathbf{I}_{n-b} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ & \mathbf{A}^1 \end{bmatrix}$$

The algorithm continues recursively on the trailing matrix $\mathbf{A}^1$.

Block LU factorization - the algorithm

1. Compute the LU factorization with partial pivoting of the first block column

$$\mathbf{\Pi}_1 \begin{pmatrix} \mathbf{A}_{11} \\ \mathbf{A}_{21} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11} \\ \mathbf{L}_{21} \end{pmatrix} \mathbf{U}_{11}$$

2. Pivot by applying the permutation matrix $\mathbf{\Pi}_1^T$ on the entire matrix,

$$\bar{\mathbf{A}} = \mathbf{\Pi}_1^T \mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}}_{11} & \bar{\mathbf{A}}_{12} \\ \bar{\mathbf{A}}_{21} & \bar{\mathbf{A}}_{22} \end{bmatrix}$$

3. Solve the triangular system

$$\mathbf{L}_{11}\mathbf{U}_{12} = \bar{\mathbf{A}}_{12}$$

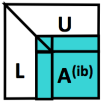
4. Update the trailing matrix,

$$\mathbf{A}^1 = \bar{\mathbf{A}}_{22} - \mathbf{L}_{21}\mathbf{U}_{12}$$

5. Compute recursively the block LU factorization of $\mathbf{A}^1$.

LU Factorization as in ScaLAPACK

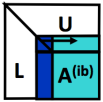
LU factorization on a $P = \sqrt{P}x\sqrt{P}$ grid of processors



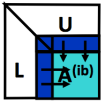
For ib = 1 to n-1 step b

$A(ib) = \mathbf{A}(ib : n, ib : n)$

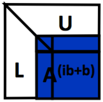
1. Compute panel factorization
   - find pivot in each column, swap rows



2. Apply all row permutations
   - broadcast pivot information along the rows
   - swap rows at left and right
3. Compute block row of **U**
   - broadcast right diagonal block of **L** of current panel



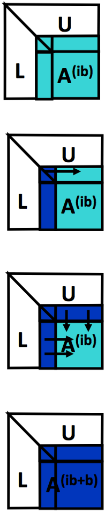
4. Update trailing matrix
   - broadcast right block column of **L**
   - broadcast down block row of **U**



Cost of LU Factorization in ScaLAPACK

LU factorization on a $P = \sqrt{P} x \sqrt{P}$ grid of processors
For ib = 1 to n-1 step b
$A(ib) = \mathbf{A}(ib:n, ib:n)$

1. Compute panel factorization
   - $\# messages = O(n \log_2 \sqrt{P})$
2. Apply all row permutations
   - $\# messages = O(n/b(\log_2 \sqrt{P} + \log_2 \sqrt{P}))$
3. Compute block row of **U**
   - $\# messages = O(n/b \log_2 \sqrt{P})$
4. Update trailing matrix
   - $\# messages = O(n/b(\log_2 \sqrt{P} + \log_2 \sqrt{P})$



Cost of parallel block LU

Consider that we have a $\sqrt{P} \times \sqrt{P}$ grid, block size b

$$\gamma \cdot \left(\frac{2/3n^3}{P} + \frac{n^2 b}{\sqrt{P}} \right) + \beta \cdot \frac{n^2 \log P}{\sqrt{P}} + \alpha \cdot \left(1.5n \log P + \frac{3.5n}{b} \log P \right).$$

Plan

LU factorization

Block LU factorization

Communication avoiding LU factorization

The LU factorization of a tall skinny matrix

First try the obvious generalization of TSQR. Consider first column block, call it $\tilde{\mathbf{A}}$

$$
\begin{aligned}
\tilde{\mathbf{A}} &= \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{pmatrix} = \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T \mathbf{L}_1^{(2)} \mathbf{U}_1^{(2)} \\ (\mathbf{\Pi}_2^{(2)})^T \mathbf{L}_2^{(2)} \mathbf{U}_2^{(2)} \\ (\mathbf{\Pi}_3^{(2)})^T \mathbf{L}_3^{(2)} \mathbf{U}_3^{(2)} \\ (\mathbf{\Pi}_4^{(2)})^T \mathbf{L}_4^{(2)} \mathbf{U}_4^{(2)} \end{pmatrix} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} \begin{pmatrix} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{pmatrix} \\
&= \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T & & & \\ & (\mathbf{\Pi}_2^{(2)})^T & & \\ & & (\mathbf{\Pi}_3^{(2)})^T & \\ & & & (\mathbf{\Pi}_4^{(2)})^T \end{pmatrix} \begin{pmatrix} \mathbf{L}_1^{(2)} & & & \\ & \mathbf{L}_2^{(2)} & & \\ & & \mathbf{L}_3^{(2)} & \\ & & & \mathbf{L}_4^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{pmatrix}
\end{aligned}
$$

$$
\begin{pmatrix} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{pmatrix} = \begin{pmatrix} (\mathbf{\Pi}_1^{(1)})^T \mathbf{L}_1^{(1)} \mathbf{U}_1^{(3)} \\ (\mathbf{\Pi}_3^{(1)})^T \mathbf{L}_3^{(1)} \mathbf{U}_1^{(3)} \end{pmatrix} = \left(\begin{array}{c|c} (\mathbf{\Pi}_1^{(1)})^T & \\ \hline & (\mathbf{\Pi}_3^{(1)})^T \end{array} \right) \left(\begin{array}{c|c} \mathbf{L}_1^{(1)} & \\ \hline & \mathbf{L}_3^{(1)} \end{array} \right) \begin{pmatrix} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{pmatrix} = (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} \begin{pmatrix} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{pmatrix}
$$

$$
\begin{pmatrix} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{pmatrix} = (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}
$$

The final factorization is:

$$
\tilde{\mathbf{A}} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}
$$

The LU factorization of a tall skinny matrix

First try the obvious generalization of TSQR. Consider first column block, call it $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} = \begin{pmatrix}\mathbf{A}_1\\\mathbf{A}_2\\\mathbf{A}_3\\\mathbf{A}_4\end{pmatrix} = \begin{pmatrix}(\mathbf{\Pi}_1^{(2)})^T\mathbf{L}_1^{(2)}\mathbf{U}_1^{(2)}\\(\mathbf{\Pi}_2^{(2)})^T\mathbf{L}_2^{(2)}\mathbf{U}_2^{(2)}\\(\mathbf{\Pi}_3^{(2)})^T\mathbf{L}_3^{(2)}\mathbf{U}_3^{(2)}\\(\mathbf{\Pi}_4^{(2)})^T\mathbf{L}_4^{(2)}\mathbf{U}_4^{(2)}\end{pmatrix} = (\mathbf{\Pi}^{(2)})^T\mathbf{L}^{(2)}\left(\begin{array}{c}\mathbf{U}_1^{(2)}\\\mathbf{U}_2^{(2)}\\\hline\mathbf{U}_3^{(2)}\\\mathbf{U}_4^{(2)}\end{array}\right)$$

$$= \begin{pmatrix}(\mathbf{\Pi}_1^{(2)})^T & & & \\ & (\mathbf{\Pi}_2^{(2)})^T & & \\ & & (\mathbf{\Pi}_3^{(2)})^T & \\ & & & (\mathbf{\Pi}_4^{(2)})^T\end{pmatrix}\begin{pmatrix}\mathbf{L}_1^{(2)} & & & \\ & \mathbf{L}_2^{(2)} & & \\ & & \mathbf{L}_3^{(2)} & \\ & & & \mathbf{L}_4^{(2)}\end{pmatrix}\left(\begin{array}{c}\mathbf{U}_1^{(2)}\\\mathbf{U}_2^{(2)}\\\hline\mathbf{U}_3^{(2)}\\\mathbf{U}_4^{(2)}\end{array}\right)$$

$$\left(\begin{array}{c}\mathbf{U}_1^{(2)}\\\mathbf{U}_2^{(2)}\\\hline\mathbf{U}_3^{(2)}\\\mathbf{U}_4^{(2)}\end{array}\right) = \begin{pmatrix}(\mathbf{\Pi}_1^{(1)})^T\mathbf{L}_1^{(1)}\mathbf{U}_1^{(3)}\\(\mathbf{\Pi}_3^{(1)})^T\mathbf{L}_3^{(1)}\mathbf{U}_1^{(3)}\end{pmatrix} = \left(\begin{array}{c|c}(\mathbf{\Pi}_1^{(1)})^T & \\\hline & (\mathbf{\Pi}_3^{(1)})^T\end{array}\right)\left(\begin{array}{c|c}\mathbf{L}_1^{(1)} & \\\hline & \mathbf{L}_3^{(1)}\end{array}\right)\left(\begin{array}{c}\mathbf{U}_1^{(1)}\\\hline\mathbf{U}_3^{(1)}\end{array}\right) = (\mathbf{\Pi}^{(1)})^T\mathbf{L}^{(1)}\left(\begin{array}{c}\mathbf{U}_1^{(1)}\\\hline\mathbf{U}_3^{(1)}\end{array}\right)$$

$$\left(\begin{array}{c}\mathbf{U}_1^{(1)}\\\hline\mathbf{U}_3^{(1)}\end{array}\right) = (\mathbf{\Pi}^{(0)})^T\mathbf{L}^{(0)}\mathbf{U}^{(0)}$$

The final factorization is:

$$\tilde{\mathbf{A}} = (\mathbf{\Pi}^{(2)})^T\mathbf{L}^{(2)}(\mathbf{\Pi}^{(1)})^T\mathbf{L}^{(1)}(\mathbf{\Pi}^{(0)})^T\mathbf{L}^{(0)}\mathbf{U}^{(0)}$$

The LU factorization of a tall skinny matrix

First try the obvious generalization of TSQR. Consider first column block, call it $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{pmatrix} = \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T \mathbf{L}_1^{(2)} \mathbf{U}_1^{(2)} \\ (\mathbf{\Pi}_2^{(2)})^T \mathbf{L}_2^{(2)} \mathbf{U}_2^{(2)} \\ (\mathbf{\Pi}_3^{(2)})^T \mathbf{L}_3^{(2)} \mathbf{U}_3^{(2)} \\ (\mathbf{\Pi}_4^{(2)})^T \mathbf{L}_4^{(2)} \mathbf{U}_4^{(2)} \end{pmatrix} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} \left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right)$$

$$= \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T & & & \\ & (\mathbf{\Pi}_2^{(2)})^T & & \\ & & (\mathbf{\Pi}_3^{(2)})^T & \\ & & & (\mathbf{\Pi}_4^{(2)})^T \end{pmatrix} \begin{pmatrix} \mathbf{L}_1^{(2)} & & & \\ & \mathbf{L}_2^{(2)} & & \\ & & \mathbf{L}_3^{(2)} & \\ & & & \mathbf{L}_4^{(2)} \end{pmatrix} \left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right)$$

$$\left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right) = \begin{pmatrix} (\mathbf{\Pi}_1^{(1)})^T \mathbf{L}_1^{(1)} \mathbf{U}_1^{(3)} \\ (\mathbf{\Pi}_3^{(1)})^T \mathbf{L}_3^{(1)} \mathbf{U}_1^{(3)} \end{pmatrix} = \left(\begin{array}{c|c} (\mathbf{\Pi}_1^{(1)})^T & \\ \hline & (\mathbf{\Pi}_3^{(1)})^T \end{array}\right) \left(\begin{array}{c|c} \mathbf{L}_1^{(1)} & \\ \hline & \mathbf{L}_3^{(1)} \end{array}\right) \left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right) = (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} \left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right)$$

$$\left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right) = (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}$$

The final factorization is:

$$\tilde{\mathbf{A}} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}$$

The LU factorization of a tall skinny matrix

First try the obvious generalization of TSQR. Consider first column block, call it $\tilde{\mathbf{A}}$

$$\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{pmatrix} = \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T \mathbf{L}_1^{(2)} \mathbf{U}_1^{(2)} \\ (\mathbf{\Pi}_2^{(2)})^T \mathbf{L}_2^{(2)} \mathbf{U}_2^{(2)} \\ (\mathbf{\Pi}_3^{(2)})^T \mathbf{L}_3^{(2)} \mathbf{U}_3^{(2)} \\ (\mathbf{\Pi}_4^{(2)})^T \mathbf{L}_4^{(2)} \mathbf{U}_4^{(2)} \end{pmatrix} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} \left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right)$$

$$= \begin{pmatrix} (\mathbf{\Pi}_1^{(2)})^T & & & \\ & (\mathbf{\Pi}_2^{(2)})^T & & \\ & & (\mathbf{\Pi}_3^{(2)})^T & \\ & & & (\mathbf{\Pi}_4^{(2)})^T \end{pmatrix} \begin{pmatrix} \mathbf{L}_1^{(2)} & & & \\ & \mathbf{L}_2^{(2)} & & \\ & & \mathbf{L}_3^{(2)} & \\ & & & \mathbf{L}_4^{(2)} \end{pmatrix} \left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right)$$

$$\left(\begin{array}{c} \mathbf{U}_1^{(2)} \\ \mathbf{U}_2^{(2)} \\ \hline \mathbf{U}_3^{(2)} \\ \mathbf{U}_4^{(2)} \end{array}\right) = \begin{pmatrix} (\mathbf{\Pi}_1^{(1)})^T \mathbf{L}_1^{(1)} \mathbf{U}_1^{(3)} \\ (\mathbf{\Pi}_3^{(1)})^T \mathbf{L}_3^{(1)} \mathbf{U}_1^{(3)} \end{pmatrix} = \left(\begin{array}{c|c} (\mathbf{\Pi}_1^{(1)})^T & \\ \hline & (\mathbf{\Pi}_3^{(1)})^T \end{array}\right) \left(\begin{array}{c|c} \mathbf{L}_1^{(1)} & \\ \hline & \mathbf{L}_3^{(1)} \end{array}\right) \left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right) = (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} \left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right)$$

$$\left(\begin{array}{c} \mathbf{U}_1^{(1)} \\ \hline \mathbf{U}_3^{(1)} \end{array}\right) = (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}$$

The final factorization is:

$$\tilde{\mathbf{A}} = (\mathbf{\Pi}^{(2)})^T \mathbf{L}^{(2)} (\mathbf{\Pi}^{(1)})^T \mathbf{L}^{(1)} (\mathbf{\Pi}^{(0)})^T \mathbf{L}^{(0)} \mathbf{U}^{(0)}$$

Obvious generalization of TSQR to LU

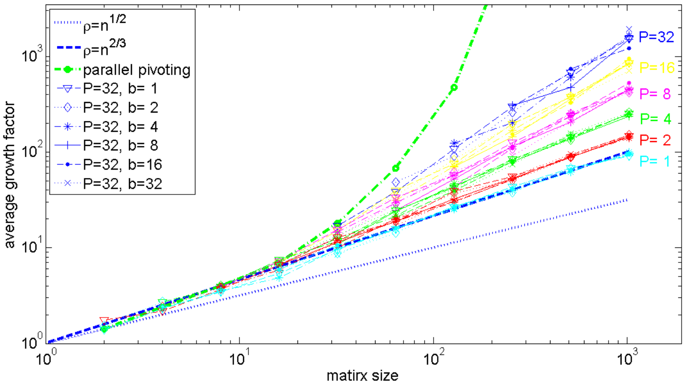
- Block parallel pivoting:

$$
\begin{tikzcd}[row sep=small]
\mathbf{A}_1 \arrow[r] & \mathbf{U}_1^{(2)} \arrow[dr] & & \\
\mathbf{A}_2 \arrow[r] & \mathbf{U}_2^{(2)} \arrow[r] & \mathbf{U}_1^{(1)} \arrow[dr] & \\
\mathbf{A}_3 \arrow[r] & \mathbf{U}_3^{(2)} \arrow[dr] & & \mathbf{U}_1^{(0)} \\
\mathbf{A}_4 \arrow[r] & \mathbf{U}_4^{(2)} \arrow[r] & \mathbf{U}_3^{(1)} \arrow[ur] &
\end{tikzcd}
$$

- Block pairwise pivoting:
  - uses a flat tree and is optimal in the sequential case
  - introduced by Barron and Swinnerton-Dyer, 1960: block LU factorization used to solve a system with 100 equations on EDSAC 2 computer using an auxiliary magnetic-tape
  - used in PLASMA for multicore architectures and FLAME for out-of-core algorithms and for multicore architectures

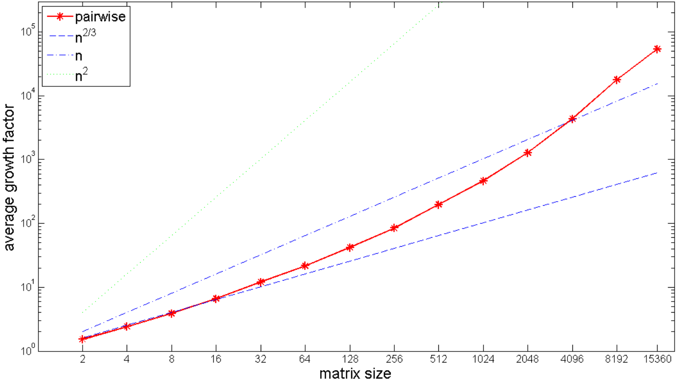
$$
\begin{tikzcd}[row sep=small]
\mathbf{A}_1 \arrow[r] & \mathbf{U}_1^{(3)} \arrow[r] & \mathbf{U}_1^{(2)} \arrow[r] & \mathbf{U}_1^{(1)} \arrow[r] & \mathbf{U}_1^{(0)} \\
\mathbf{A}_2 \arrow[urr] & & & & \\
\mathbf{A}_3 \arrow[uurrr] & & & & \\
\mathbf{A}_4 \arrow[uuurrrr] & & & &
\end{tikzcd}
$$

Block parallel pivoting



- Unstable for large number of processors P
- When P=number rows, it corresponds to parallel pivoting, known to be unstable (Trefethen and Schreiber, 90)

Block pairwise pivoting



- Results shown for random matrices
- Will become unstable for large matrices

Tournament pivoting - the overall idea

- At each iteration of a block algorithm

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} \quad \text{where } \tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A}_{11} \\ \mathbf{A}_{21} \end{pmatrix}$$

where $\mathbf{A}_{11}$ is of size $b \times b$, $\mathbf{A}_{21}$ is of size $(m-b) \times b$, $\mathbf{A}_{12}$ is of size $b \times (n-b)$ and $\mathbf{A}_{22}$ is of size $(m-b) \times (n-b)$.

  - Preprocess $\tilde{\mathbf{A}}$ to find at low communication cost good pivots for the LU factorization of $\tilde{\mathbf{A}}$, return a permutation matrix $\mathbf{\Pi}_1$
  - Permute the pivots to top, ie compute $\mathbf{\Pi}_1\mathbf{A}$
  - Compute LU with no pivoting of first block column $\mathbf{\Pi}_1\tilde{\mathbf{A}}$, update trailing matrix, obtain

$$\mathbf{\Pi}_1\mathbf{A} = \begin{pmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{21} & \mathbf{I}_{n-b} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ & \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{U}_{12} \end{pmatrix}$$

Tournament pivoting

$$
\begin{tikzcd}
\begin{bmatrix} 2 & 4 \\ 0 & 1 \\ 2 & 0 \\ 1 & 2 \end{bmatrix} \arrow[r] & \begin{bmatrix} 2 & 4 \\ 2 & 0 \end{bmatrix} \arrow[r] & \begin{bmatrix} 4 & 1 \\ 2 & 4 \end{bmatrix} \arrow[r] & \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} \\
\begin{bmatrix} 2 & 0 \\ 0 & 0 \\ 4 & 1 \\ 1 & 0 \end{bmatrix} \arrow[r] & \begin{bmatrix} 4 & 1 \\ 2 & 0 \end{bmatrix} \arrow[ur] & & \\
\begin{bmatrix} 0 & 1 \\ 1 & 4 \\ 0 & 0 \\ 0 & 2 \end{bmatrix} \arrow[r] & \begin{bmatrix} 1 & 4 \\ 0 & 2 \end{bmatrix} \arrow[r] & \begin{bmatrix} 4 & 2 \\ 1 & 4 \end{bmatrix} \arrow[uur] & \\
\begin{bmatrix} 2 & 1 \\ 0 & 2 \\ 1 & 0 \\ 4 & 2 \end{bmatrix} \arrow[r] & \begin{bmatrix} 4 & 2 \\ 0 & 2 \end{bmatrix} \arrow[ur] & &
\end{tikzcd}
$$

Tournament pivoting for a tall skinny matrix

Consider the first block of columns distributed across 4 processors,

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{bmatrix}$$

- At the leaves of the binomial tree
  Compute GEPP factorization of each $\mathbf{A}_I \in \mathbb{R}^{m/4 \times b}$

$$\begin{aligned}
\mathbf{\Pi}_1^{(2)}\mathbf{A}_1 &= \mathbf{L}_1^{(2)}\mathbf{U}_1^{(2)}, \\
\mathbf{\Pi}_2^{(2)}\mathbf{A}_2 &= \mathbf{L}_2^{(2)}\mathbf{U}_2^{(2)}, \\
\mathbf{\Pi}_3^{(2)}\mathbf{A}_3 &= \mathbf{L}_3^{(2)}\mathbf{U}_3^{(2)}, \\
\mathbf{\Pi}_4^{(2)}\mathbf{A}_4 &= \mathbf{L}_4^{(2)}\mathbf{U}_4^{(2)}.
\end{aligned}$$

Tournament pivoting for a tall skinny matrix

- Perform $\log_2 P$ times GEPP factorization of selected $2b \times b$ rows
  - Two LU factorizations with partial pivoting are computed in parallel to obtain two new sets of pivot rows,

$$\mathbf{A}_1^{(1)} := \begin{pmatrix} \mathbf{\Pi}_1^{(2)} \mathbf{A}_1(1:b,:) \\ \mathbf{\Pi}_2^{(2)} \mathbf{A}_2(1:b,:) \end{pmatrix}, \quad \mathbf{\Pi}_1^{(1)} \mathbf{A}_1^{(1)} = \mathbf{L}_1^{(1)} \mathbf{U}_1^{(1)},$$

$$\mathbf{A}_3^{(1)} := \begin{pmatrix} \mathbf{\Pi}_3^{(2)} \mathbf{A}_3(1:b,:) \\ \mathbf{\Pi}_4^{(2)} \mathbf{A}_4(1:b,:) \end{pmatrix}, \quad \mathbf{\Pi}_3^{(1)} \mathbf{A}_3^{(1)} = \mathbf{L}_3^{(1)} \mathbf{U}_3^{(1)}$$

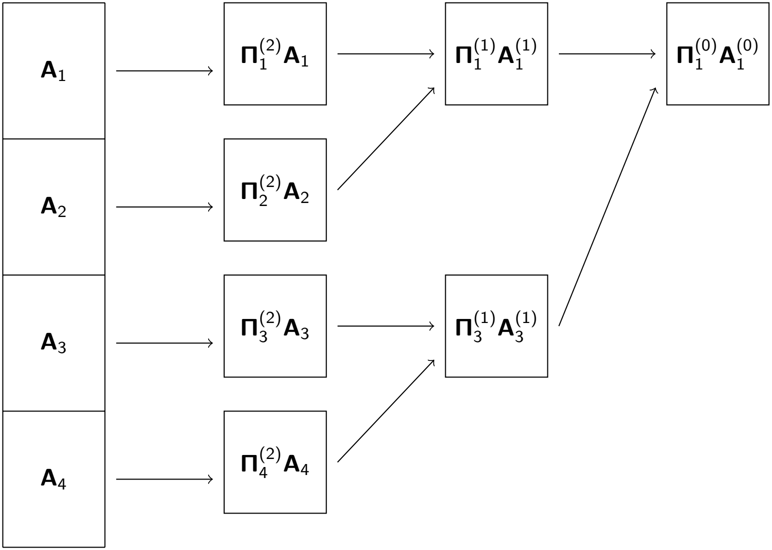
  - At the root of the tree:

$$\mathbf{A}_1^{(0)} := \begin{bmatrix} \mathbf{\Pi}_1^{(1)} \mathbf{A}_1^{(1)}(1:b,:) \\ \mathbf{\Pi}_3^{(1)} \mathbf{A}_3^{(1)}(1:b,:) \end{bmatrix}, \quad \mathbf{\Pi}_1^{(0)} \mathbf{A}_1^{(0)} = \mathbf{L}_1^{(0)} \mathbf{U}_1^{(0)}$$

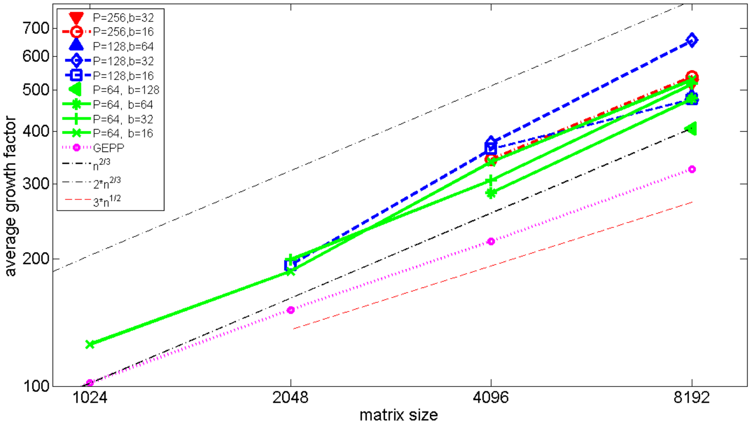
- Permute to leading positions the global pivot rows $\mathbf{\Pi}_1^{(0)} \mathbf{A}_1^{(0)}(1:b,:)$
- Let $\mathbf{\Pi}_1 \in \mathbb{R}^{n \times n}$ be the matrix that reflects this permutation
- Perform LU factorization with no pivoting of the permuted matrix

$$\mathbf{\Pi}_1 \begin{bmatrix} \mathbf{A}_{11} \\ \mathbf{A}_{21} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} \\ \mathbf{L}_{21} \end{bmatrix} \mathbf{U}_{11}. \tag{3}$$

Tournament pivoting



Growth factor for binary tree based CALU



- Random matrices from a normal distribution
- Same behaviour for all matrices in our test, and $|\mathbf{L}| \leq 4.2$

Our "proof of stability" for CALU

- CALU as stable as GEPP in following sense:
  In exact arithmetic, CALU process on a matrix A is equivalent to GEPP process on a larger matrix G whose entries are blocks of A and zeros.
- Example of one step of tournament pivoting:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \\ \mathbf{A}_{31} & \mathbf{A}_{32} \end{pmatrix},$$

$$\begin{tikzcd} \mathbf{A}_{11} \arrow[rr] & & \bar{\mathbf{A}}_{11} \\ \mathbf{A}_{21} \arrow[dr] & & \\ & \mathbf{A}_{21} \arrow[uur] & \\ \mathbf{A}_{31} \arrow[ur] & & \end{tikzcd}$$

$$\mathbf{G} = \begin{pmatrix} \bar{\mathbf{A}}_{11} & & \bar{\mathbf{A}}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{21} & \\ & -\mathbf{A}_{31} & \mathbf{A}_{32} \end{pmatrix}$$

- Proof possible by using original rows of **A** during tournament pivoting (not the computed rows of **U**).

Outline of “proof of stability”

- After the factorization of first panel by CALU, $\mathbf{A}^s_{32}$ (the Schur complement of $\mathbf{A}_{32}$) is not bounded as in GEPP,

$$\begin{pmatrix} \mathbf{\Pi}_{11} & \mathbf{\Pi}_{12} & \\ \mathbf{\Pi}_{21} & \mathbf{\Pi}_{22} & \\ & & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \\ \mathbf{A}_{31} & \mathbf{A}_{32} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{A}}_{11} & \bar{\mathbf{A}}_{12} \\ \bar{\mathbf{A}}_{21} & \bar{\mathbf{A}}_{22} \\ \mathbf{A}_{31} & \mathbf{A}_{32} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{L}}_{11} & & \\ \bar{\mathbf{L}}_{21} & \mathbf{I}_b & \\ \bar{\mathbf{L}}_{31} & & \mathbf{I}_{m-2b} \end{pmatrix} \cdot \begin{pmatrix} \bar{\mathbf{U}}_{11} & \bar{\mathbf{U}}_{12} \\ & \bar{\mathbf{A}}^s_{22} \\ & \mathbf{A}^s_{32} \end{pmatrix}$$

- but $\mathbf{A}^s_{32}$ can be obtained by GEPP on larger matrix $\mathbf{G}$ formed from blocks of $\mathbf{A}$

$$\mathbf{G} = \begin{pmatrix} \bar{\mathbf{A}}_{11} & & \bar{\mathbf{A}}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{21} & \\ & -\mathbf{A}_{31} & \mathbf{A}_{32} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{L}}_{11} & & \\ \mathbf{A}_{21}\bar{\mathbf{U}}_{11}^{-1} & \mathbf{L}_{21} & \\ & -\mathbf{L}_{31} & \mathbf{I}_{m-2b} \end{pmatrix} \begin{pmatrix} \bar{\mathbf{U}}_{11} & & \bar{\mathbf{U}}_{12} \\ & \mathbf{U}_{21} & -\mathbf{L}_{21}^{-1}\mathbf{A}_{21}\bar{\mathbf{U}}_{11}^{-1}\bar{\mathbf{U}}_{12} \\ & & \mathbf{A}^s_{32} \end{pmatrix}$$

- GEPP on G does not permute and

$$\begin{aligned} \mathbf{L}_{31}\mathbf{L}_{21}^{-1}\mathbf{A}_{21}\bar{\mathbf{U}}_{11}^{-1}\bar{\mathbf{U}}_{12} + \mathbf{A}^s_{32} &= \mathbf{L}_{31}\mathbf{U}_{21}\bar{\mathbf{U}}_{11}^{-1}\bar{\mathbf{U}}_{12} + \mathbf{A}^s_{32} = \mathbf{A}_{31}\bar{\mathbf{U}}_{11}^{-1}\bar{\mathbf{U}}_{12} + \mathbf{A}^s_{32} \\ &= \bar{\mathbf{L}}_{31}\bar{\mathbf{U}}_{12} + \mathbf{A}^s_{32} = \mathbf{A}_{32} \end{aligned}$$

Growth factor in exact arithmetic

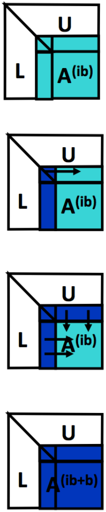
- Matrix of size m-by-n, reduction tree of height $H = \log_2(P)$
- In practice growth factor for GEPP and CALU is on the order of $n^{2/3} - -n^{1/2}$

	matrix of size $m \times (b+1)$		
	TSLU(b,H)		GEPP
	upper bound	attained	upper bound
$\lvert L \rvert$	2^{bH}	$2^{(b-2)H-(b-1)}$	1
g_W	$2^{b(H+1)}$	2^b	2^b
	matrix of size $m \times n$		
	CALU(b,H)		GEPP
	upper bound	attained	upper bound
$\lvert L \rvert$	2^{bH}	$2^{(b-2)H-(b-1)}$	1
g_W	$2^{n(H+1)-1}$	2^{n-1}	2^{n-1}

Cost of LU with tournament pivoting

LU factorization on a $P = \sqrt{P} x \sqrt{P}$ grid of processors
For ib = 1 to n-1 step b
$A(ib) = A(ib : n, ib : n)$

1. Compute panel factorization
   - $\#messages = O(n/b \log_2 \sqrt{P})$
2. Apply all row permutations
   - $\#messages = O(n/b(\log_2 \sqrt{P} + \log_2 \sqrt{P}))$
3. Compute block row of U
   - $\#messages = O(n/b \log_2 \sqrt{P})$
4. Update trailing matrix
   - $\#messages = O(n/b(\log_2 \sqrt{P} + \log_2 \sqrt{P})$



CALU based on TSLU

Cost of CALU vs ScaLAPACK's PDGETRF

- $n \times n$ matrix on $\sqrt{P} \times \sqrt{P}$ processor grid, block size b
- Flops: $(2/3)n^3/P + 3/2n^2 b \log_2 P/\sqrt{P}$ vs $(2/3)n^3/P + n^2 b/P^{1/2}$
- Bandwidth: $n^2 \log_2 P/\sqrt{P}$ vs same
- Latency: $3n \log_2 P/b$ vs $1.5n \log_2 P$

Close to optimal (modulo log P factors)

- Assume $O(n^2/P)$ memory/processor, $O(n^3)$ algorithm
- Choose b near $n/\sqrt{P}$ (its upper bound)
- Bandwidth lower bound: $\Omega(n^2/\sqrt{P})$ – just $\log_2 P$ smaller
- Latency lower bound: $\Omega(\sqrt{P})$ – just polylog(P) smaller

Performance vs ScaLAPACK

- Parallel TSLU (LU on tall-skinny matrix)
  - IBM Power 5
    Up to 4.37x faster (16 procs, $1M \times 150$)
  - Cray XT4
    Up to 5.52x faster (8 procs, $1M \times 150$)
- Parallel CALU (LU on general matrices)
  - Intel Xeon (two socket, quad core)
    Up to 2.3x faster (8 cores, $10^6 \times 500$)
  - IBM Power 5
    Up to 2.29x faster (64 procs, 1000×1000)
  - Cray XT4
    Up to 1.81x faster (64 procs, 1000×1000)

Acknowledgement

- Figures from upcoming book on communication avoiding algorithms with G. Ballard, E. Carson, and J. Demmel

References (1)

N.J.Higham (2002).
Accuracy and Stability of Numerical Algorithms.
SIAM, second edition.