Matrix Computations
Chapter 2 Linear systems and LU decomposition
Section 2.2 Pivoting for LU Decomposition

Jie Lu
ShanghaiTech University

Pivoting

- Previously, we assume all the pivots are nonzero. What if some $a_{kk}^{(k-1)}$ happens to be zero?

- Gaussian elimination is known to be numerically unstable when a pivot is close to zero
  - Relatively small pivots can cause large entries in **L** and **U** and thus non-negligible error in solution due to round-off errors

- Pivoting: Find permutations of A with a proper LU decomposition
  - Partial pivoting, complete pivoting, rook pivoting, etc.

Permutation Matrix

A square matrix with exactly one entry of 1 in each row and each column and 0 elsewhere is a permutation matrix

Example: Let $\mathbf{\Pi}$ be a 4×4 permutation matrix and $A \in \mathbb{R}^4$

$$\mathbf{\Pi} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_3 & \mathbf{a}_4 \end{bmatrix} = \begin{bmatrix} — \tilde{\mathbf{a}}_1^T — \\ — \tilde{\mathbf{a}}_2^T — \\ — \tilde{\mathbf{a}}_3^T — \\ — \tilde{\mathbf{a}}_4^T — \end{bmatrix}$$

- $\mathbf{\Pi A}$ is obtained by swapping row 1 and row 4 of $\mathbf{A}$
- $\mathbf{A\Pi}$ is obtained by swapping column 1 and column 4 of $\mathbf{A}$

$$\mathbf{\Pi A} = \begin{bmatrix} — \tilde{\mathbf{a}}_4^T — \\ — \tilde{\mathbf{a}}_2^T — \\ — \tilde{\mathbf{a}}_3^T — \\ — \tilde{\mathbf{a}}_1^T — \end{bmatrix} \qquad \mathbf{A\Pi} = \begin{bmatrix} \mathbf{a}_4 & \mathbf{a}_2 & \mathbf{a}_3 & \mathbf{a}_1 \end{bmatrix}$$

Permutation Matrix

Example:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{PA} = \begin{bmatrix} \text{—}\tilde{\mathbf{a}}_2^T\text{—} \\ \text{—}\tilde{\mathbf{a}}_4^T\text{—} \\ \text{—}\tilde{\mathbf{a}}_1^T\text{—} \\ \text{—}\tilde{\mathbf{a}}_3^T\text{—} \end{bmatrix}, \qquad \mathbf{AP} = \begin{bmatrix} \mathbf{a}_3 & \mathbf{a}_1 & \mathbf{a}_4 & \mathbf{a}_2 \end{bmatrix}$$

Note that $\mathbf{P}$ can be decomposed as

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Interchange Permutations

Let $\mathbf{\Pi}_k \in \mathbb{R}^{n\times n}$, $k = 1, \ldots, m \leq n$ be the $n \times n$ identity matrix $\mathbf{I}$ with row k and row $piv(k)$ swapped, which are called interchange permutations

Let $\mathbf{P} = \mathbf{\Pi}_m \cdots \mathbf{\Pi}_1$

- $\mathbf{\Pi}_k$ is symmetric (but $\mathbf{P}$ may not be symmetric)
- $\mathbf{P}^T = \mathbf{\Pi}_1 \cdots \mathbf{\Pi}_m$
- If $piv = [1, \ldots, m]^T$, then $\mathbf{P} = \mathbf{I}$

Computation of $\mathbf{Px}$, $\mathbf{x} \in \mathbb{R}^n$

```
for k=1:m  % overwrite x with Px
     x(k)↔ x(piv(k)) % swap entry k and entry piv(k)
end
```

Computation of $\mathbf{P}^T\mathbf{x}$, $\mathbf{x} \in \mathbb{R}^n$

```
for k=m:-1:1
     x(k)↔ x(piv(k))
end
```

No flops needed for permutation (but affect performance nontrivially)

Partial Pivoting

Recall Upper Triangularization in Section 2.1
Given $\mathbf{A}^{(k-1)}$, $k = 1, \ldots, n-1$,

1. Find $piv(k) = arg\ \max_{j\in[k,n]} |\mathbf{A}^{(k-1)}(j,k)|$
2. Let $\mathbf{\Pi}_k \in \mathbb{R}^{n\times n}$ be the interchange permutation that swaps row k and row $piv(k)$ of $\mathbf{I}$
3. Determine the Gauss Transformation $\mathbf{M}_k = \mathbf{I}_n - \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T$, where

$$\boldsymbol{\tau}^{(k)} = \begin{bmatrix} \mathbf{0}_k \\ (\mathbf{\Pi}_k\mathbf{A}^{(k-1)})(k+1:n,k)/(\mathbf{\Pi}_k\mathbf{A}^{(k-1)})(k,k) \end{bmatrix}$$

4. $\mathbf{A}^{(k)} = \mathbf{M}_k(\mathbf{\Pi}_k\mathbf{A}^{(k-1)})$ (which satisfies $\mathbf{A}^{(k)}(k+1:n,k) = \mathbf{0}$)

Upon completing the above process, we have

$$\mathbf{M}_{n-1}\mathbf{\Pi}_{n-1}\cdots\mathbf{M}_1\mathbf{\Pi}_1\mathbf{A} = \mathbf{U}$$

Note that all the elements in $\boldsymbol{\tau}^{(k)}(k+1:n)$ are ≤ 1 in absolute value

Partial Pivoting (cont'd)

Example: $\mathbf{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}$

Partial Pivoting (cont'd)

Partial Pivoting (cont'd)

Computation of $\mathbf{L}$ with Partial Pivoting

Define $\mathbf{P} = \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_1$ and for each $k = 1, \ldots, n-1$,

$$\tilde{\mathbf{M}}_k = (\mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1})\mathbf{M}_k(\mathbf{\Pi}_{k+1} \cdots \mathbf{\Pi}_{n-1})$$

Note: $\tilde{\mathbf{M}}_k$ **is a Gauss transformation**

$$\tilde{\mathbf{M}}_k = (\mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1}) \cdot (\mathbf{I} - \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T) \cdot (\mathbf{\Pi}_{k+1} \cdots \mathbf{\Pi}_{n-1}) = \mathbf{I} - \tilde{\boldsymbol{\tau}}^{(k)}\mathbf{e}_k^T$$

with $\tilde{\boldsymbol{\tau}}^{(k)} = \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1}\boldsymbol{\tau}^{(k)}$ (Why?)

Computation of $\mathbf{L}$ with Partial Pivoting (cont'd)

Example: Let $n = 4$

$$\begin{aligned}\tilde{\mathbf{M}}_3\tilde{\mathbf{M}}_2\tilde{\mathbf{M}}_1\mathbf{P}\mathbf{A} &= \mathbf{M}_3 \cdot (\mathbf{\Pi}_3\mathbf{M}_2\mathbf{\Pi}_3) \cdot (\mathbf{\Pi}_3\mathbf{\Pi}_2\mathbf{M}_1\mathbf{\Pi}_2\mathbf{\Pi}_3) \cdot (\mathbf{\Pi}_3\mathbf{\Pi}_2\mathbf{\Pi}_1)\mathbf{A} \\ &= \mathbf{M}_3\mathbf{\Pi}_3\mathbf{M}_2\Pi_2\mathbf{M}_1\Pi_1\mathbf{A} = \mathbf{U}\end{aligned}$$

We can easily extend this to general n and obtain

$$\tilde{\mathbf{M}}_{n-1}\cdots\tilde{\mathbf{M}}_1\mathbf{P}\mathbf{A} = \mathbf{U}$$

In addition, let

$$\mathbf{L} = \tilde{\mathbf{M}}_1^{-1}\cdots\tilde{\mathbf{M}}_{n-1}^{-1} = (\mathbf{I} + \tilde{\boldsymbol{\tau}}^{(1)}\mathbf{e}_1^T)\cdots(\mathbf{I} + \tilde{\boldsymbol{\tau}}^{(n-1)}\mathbf{e}_{n-1}^T) = \mathbf{I} + \sum_{k=1}^{n-1}\tilde{\boldsymbol{\tau}}^{(k)}\mathbf{e}_k^T$$

where the absolute value of each entry of $\mathbf{L}$ is ≤ 1 (Why?)

Therefore, LU decomposition with pivoting is equivalent to

$$\mathbf{P}\mathbf{A} = \mathbf{L}\mathbf{U}$$

LU with Partial Pivoting

Find $\mathbf{L}, \mathbf{U}$ s.t. $\mathbf{PA} = \mathbf{LU}$ in MATLAB

```
for k=1:n-1
     [~ ,piv]=max(abs(A(k:n,k))); piv=piv-1+k;
     A([k piv],:)=A([piv k],:)  % swap row k and row piv
% If A(k,k)=0, then nothing to do
     if A(k,k)~ =0
         rows=k+1:n
         A(rows,k)=A(rows,k)/A(k,k) % compute τ^(k)(k+1:n)
         A(rows,rows)=A(rows,rows)-A(rows,k)*A(k,rows)
    end
end
```

In the above code, $A(k, k:n)$ represents $\mathbf{U}(k, k:n)$ and $A(k+1:n, k)$ represents $\mathbf{L}(k+1:n, k)$ (We already know the diagonal entries of $\mathbf{L}$ are 1)

$\mathcal{O}(n^2)$ comparisons for searching for the pivots

$\mathcal{O}(2n^3/3)$ flops

LU with Complete Pivoting

Complete Pivoting: Permute the largest entry of $A^{(k-1)}(k:n,k:n)$ in absolute value into the (k,k)-entry

- Require both row and column swaps

$$(rowpiv(k), colpiv(k)) = \arg\max_{(i,j)\in[k,n]\times[k,n]} |\mathbf{A}^{(k-1)}(i,j)|$$
$$\mathbf{A}^{(k-1)}(k, 1:n) \leftrightarrow \mathbf{A}^{(k-1)}(rowpiv(k), 1:n)$$
$$\mathbf{A}^{(k-1)}(1:n, k) \leftrightarrow \mathbf{A}^{(k-1)}(1:n, colpiv(k))$$

Then apply Gauss Transform to obtain $\mathbf{A}^{(k)}$ s.t. $\mathbf{A}^{(k)}(k+1:n,k) = \mathbf{0}$

The above Upper Triangularization gives

$$\mathbf{A}^{(k)} = \mathbf{M}_k\mathbf{\Pi}_k\mathbf{A}^{(k-1)}\Gamma_k = \mathbf{M}_k\mathbf{\Pi}_k\cdots\mathbf{M}_1\mathbf{\Pi}_1\mathbf{A}\mathbf{\Gamma}_1\cdots\mathbf{\Gamma}_k, \quad k = 1,\ldots,n-1$$
$$\mathbf{A}^{(n-1)} = \mathbf{U}$$

LU with Complete Pivoting

- $\mathcal{O}(n^3)$ comparisons and $\mathcal{O}(\frac{2}{3}n^3)$ flops
  - Much more costly than partial pivoting
  - But lead to much smaller bound on growth factor, which reflects the safety of applying Gaussian elimination (cf. Section 3.4.5 in textbook)

- $\mathbf{PAQ}^T = \mathbf{LU}$
  - $\mathbf{P} = \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_1$, where $\mathbf{\Pi}_k$ interchanges row k and row $rowpiv(k)$ of $\mathbf{I}$
  - $\mathbf{Q} = \mathbf{\Gamma}_{n-1} \cdots \mathbf{\Gamma}_1$, where $\mathbf{\Gamma}_k$ interchanges row k and row $colpiv(k)$ of $\mathbf{I}$
  - $\mathbf{U}$ is upper triangular, $\mathbf{L}$ is unit lower triangular with $|\ell_{ij}| \leq 1$

Solving Linear System via LU with Pivoting

Solve $\mathbf{Ax} = \mathbf{b}$ using $\mathbf{PAQ}^T = \mathbf{LU}$

1. Solve $\mathbf{Lz} = \mathbf{Pb}$ for $\mathbf{z}$ (Forward Substitution $\mathcal{O}(n^2)$)
2. Solve $\mathbf{Uy} = \mathbf{z}$ for $\mathbf{y}$ (Back Substitution $\mathcal{O}(n^2)$)
3. Set $\mathbf{x} = \mathbf{Q}^T y$

$\mathbf{Q} = \mathbf{I}$ for partial pivoting

Discussion

- When you call `lu(A)` or `A\b` in MATLAB, it always performs pivoting
- Apart from solving linear systems, LU decomposition is also used to
  - Compute $\mathbf{A}^{-1}$ (solve n linear systems): let $\mathbf{B} = \mathbf{A}^{-1}$

$$\mathbf{AB} = \mathbf{I} \quad \Longleftrightarrow \quad \mathbf{Ab}_i = \mathbf{e}_i, \; i = 1, \ldots, n$$

  - Compute $\det(\mathbf{A})$:

$$\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{U}) = \prod_{i=1}^{n} u_{ii}$$

- Another way of pivoting: Let the pivot be the element in $A^{(k-1)}(k:n, k:n)$ that has the maximal absolute value in both its row and its column (Rook Pivoting)