AMS526: Numerical Analysis I (Numerical Linear Algebra)

Lecture 4: Gaussian Elimination and LU Factorization with Pivoting

Xiangmin Jiao

Stony Brook University

Outline

1. Gaussian Elimination

2. Gaussian Elimination with Pivoting

Review of Gaussian Elimination

- Given linear system $Ax = b$, where $A \in \mathbb{R}^{n \times n}$ is nonsingular, Gaussian elimination transforms the linear system to
$$Ux = y$$
where U is upper triangular, and then solves $Ux = y$ by back substitution
- Gaussian elimination performs three types of operations
  1. Add a multiple one equation to another
  2. Interchange two equations
  3. Multiply an equation by a nonzero constant
- Gaussian elimination can be represented as transformation of augmented matrix
$$[A \mid b] \to [U \mid y]$$
through operations are on rows of matrix
- A is nonsingular if and only if U is nonsingular

Gaussian Elimination and LU Factorization

- Gaussian elimination without row interchanges can be viewed as "triangular triangularization" of *nonsingular* $A \in \mathbb{R}^{n\times n}$

$$\underbrace{L_{n-1}\cdots L_2 L_1}_{L^{-1}} A = U$$

- Then $A = LU$. It is also called LU factorization (without pivoting)
- For augmented matrix, $L^{-1}[A \mid b] = [A \mid y]$, so $y = L^{-1}b$
- Example of LU factorization of 4×4 matrix A

$$\xrightarrow{L_1} \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix}}_{L_1 A} \xrightarrow{L_2} \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & 0 & \mathbf{x} & \mathbf{x} \\ & 0 & \mathbf{x} & \mathbf{x} \end{bmatrix}}_{L_2 L_1 A} \xrightarrow{L_3} \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & 0 & \mathbf{x} \end{bmatrix}}_{L_3 L_2 L_1 A}$$

What is Matrix L_1?

- In Gaussian elimination, at first step A is transformed to $A^{(1)}$ by

$$\begin{aligned} \ell_{i1} &= a_{i1}/a_{11}, \qquad i = 2, \ldots, n \\ a_{ij}^{(1)} &= a_{ij} - \ell_{i1} a_{1j}, \qquad i = 2, \ldots, n, \quad j = 2, \ldots, n \end{aligned}$$

- In matrix form, we have $A^{(1)} = L_1 A$, where

$$L_1 = \begin{bmatrix} 1 & & & & \\ -\ell_{2,1} & 1 & & & \\ -\ell_{3,1} & & 1 & & \\ \vdots & & & \ddots & \\ -\ell_{n,1} & & & & 1 \end{bmatrix}$$

What is Matrices L_k?

- At step k, eliminate entries below $a_{kk}^{(k-1)}$:
- Let $a_{:,k}^{(k-1)}$ be kth column of $L_{k-1} \cdots L_1 A$, then

$$a_{:,k}^{(k-1)} = [a_{1k}^{(k-1)}, a_{2k}^{(k-1)}, \cdots, a_{kk}^{(k-1)}, a_{k,k+1}^{(k-1)}, \cdots a_{n,k}^{(k-1)}]^T$$

$$L_k a_{:,k}^{(k-1)} = [a_{1k}^{(k-1)}, a_{2k}^{(k-1)}, \cdots, a_{kk}^{(k-1)}, 0, \cdots 0]^T$$

- The *multipliers* $\ell_{jk} = a_{jk}^{(k-1)} / a_{kk}^{(k-1)}$ appear in L_k

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -\ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & -\ell_{n,k} & & & 1 \end{bmatrix}$$

Forming The L Matrix

- Let $\ell_k = [\underbrace{0, \cdots, 0}_{k}, \ell_{k+1,k}, \cdots, \ell_{n,k}]^T$ and $e_k = [\underbrace{0, \cdots, 0}_{k-1}, 1, \cdots, 0]^T$, then $L_k = I - \ell_k e_k^T$
- Luckily, L matrix contains the multipliers $\ell_{jk} = x_{jk}/x_{kk}$

$$L = L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \dots & \ell_{n,n-1} & 1 \end{bmatrix}$$

and is said to be a *unit lower triangular matrix*

Proof of Structure of L Matrix

- First, $L_k^{-1} = I + \ell_k e_k^T$, because $e_k^T \ell_k = 0$ and

$$(I - \ell_k e_k^T)(I + \ell_k e_k^T) = I - \ell_k e_k^T \ell_k e_k^T = I$$

- Second, $L_1^{-1} L_2^{-1} \cdots L_{k+1}^{-1} = I + \sum_{j=1}^{k+1} \ell_j e_j^T$, since (prove by induction)

$$(I + \sum_{j=1}^{k} \ell_j e_j^T)(I + \ell_{k+1} e_{k+1}^T) = I + \sum_{j=1}^{k+1} \ell_j e_j^T + \sum_{j=1}^{k} \ell_j (e_j^T \ell_{k+1}) e_{k+1}^T$$

  where $e_j^T \ell_{k+1} = 0$ for $j < k+1$

- In other words, L is "union" of nonzero entries in $L_1^{-1}, L_2^{-1}, \cdots, L_{n-1}^{-1}$

LU Factorization without Pivoting

- Factorize $A \in \mathbb{R}^{n\times n}$ into $A = LU$

Gaussian elimination without pivoting

$$
\begin{aligned}
&U = A, L = I \\
&\textbf{for } k = 1 \textbf{ to } n-1 \\
&\qquad \textbf{for } j = k+1 \textbf{ to } n \\
&\qquad\qquad \ell_{jk} = u_{jk}/u_{kk} \\
&\qquad\qquad u_{j,k:n} = u_{j,k:n} - \ell_{jk} u_{k,k:n}
\end{aligned}
$$

- Flop count $\sim \sum_{k=1}^{n} 2(n-k)(n-k) \sim 2\sum_{k=1}^{n} k^2 \sim 2n^3/3$
- In practice, L overwrites lower-triangular part of A and U overwrites upper-triangular part of A
- Question: What happens if u_{kk} is 0?
- Answer: The algorithm will break due to division by zero!

Theorem of LU Factorization

Theorem

Let A be an $n \times n$ matrix whose leading principal submatrices are all nonsingular. Then A can be factorized in exactly one way into a product $A = LU$, such that L is unit lower triangular and U is upper triangular.

- However, the requirement of nonsingular leading principal submatrices is too strong!
- Analogous to LDL^T with respect to Cholesky factorization, a variant is LU is LDV, where D is diagonal and V is unit upper triangular
- More importantly, what happens if u_{kk} is ***nearly*** 0 (or a submatrix is ***nearly*** singular)?

Outline

Gaussian Elimination with Partial Pivoting

- At step k, we divide by u_{kk} (i.e., $a_{kk}^{(k-1)}$), which would break if u_{kk} is 0 (or close to 0), which can happen even if A is nonsingular
- Other nonzero entry in kth column below diagonal can be *pivot*

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & a_{ik}^{(k-1)} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ & \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ & 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ & a_{ik}^{(k-1)} & \times & \times & \times \\ & 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix}$$

and we permute (interchange) row i with row k

Partial Pivoting

- We choose nonzero entry with largest magnitude as pivot
- kth step of Gaussian elimination of partial pivoting

$$\underset{\text{Pivot selection}}{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & x_{ik} & \mathbf{x} & \mathbf{x} \\ & \times & \times & \times \end{bmatrix}} \xrightarrow{P_k} \underset{\text{Row interchange}}{\begin{bmatrix} \times & \times & \times & \times \\ & x_{kk} & \mathbf{x} & \mathbf{x} \\ & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ & \times & \times & \times \end{bmatrix}} \xrightarrow{L_k} \underset{\text{Elimination}}{\begin{bmatrix} \times & \times & \times & \times \\ & x_{kk} & \times & \times \\ & 0 & \mathbf{x} & \mathbf{x} \\ & 0 & \mathbf{x} & \mathbf{x} \end{bmatrix}}$$

  and we interchange row i with row k
- P_k is a permutation matrix, obtained by interchanging two rows of I
- Product of permutation matrices is also a permutation matrix
- For any permutation matrix P, all its entries are zeros and ones, and $PP^T = P^T P = I$

Matrix Notation of Partial Pivoting

- In terms of matrices, it becomes $\underbrace{L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1}_{L^{-1}P}A = U$
- $PA = LU$ or $A = P^TLU$. This is LU factorization with partial pivoting
- $P = P_{n-1}\cdots P_2P_1$ and $L = \left(L'_{n-1}\cdots L'_2L'_1\right)^{-1}$, where $L'_k = P_{n-1}\cdots P_{k+1}L_kP_{k+1}^{-1}\cdots P_{n-1}^{-1}$
- It is easy to verify that $L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1 = \left(L'_{n-1}\cdots L'_2L'_1\right)\left(P_{n-1}\cdots P_2P_1\right)$
- $L'_k = I - P_{n-1}\cdots P_{k+1}\ell_k e_k^T$, and $\left(L'_k\right)^{-1} \equiv I + P_{n-1}\cdots P_{k+1}\ell_k e_k^T$.
- In other words, L'_k is obtained by continued permutation of $\ell_k e_k^T$ along with $A^{(k+)}$, and lower-triangular part of L is "union" of permuted $\ell_k e_k^T$

Algorithm of Gaussian Elimination with Partial Pivoting

- Factorize $A \in \mathbb{R}^{n\times n}$ into $A = P^T LU$

Gaussian elimination with partial pivoting

$$
\begin{aligned}
&U = A, L = I,\ P = I \\
&\textbf{for } k = 1 \textbf{ to } n-1 \\
&\qquad i \leftarrow \arg\max_{i\geq k} |u_{ik}| \\
&\qquad u_{k,k:n} \leftrightarrow u_{i,k:n} \\
&\qquad \ell_{k,1:k-1} \leftrightarrow \ell_{i,1:k-1} \\
&\qquad p_{k,:} \leftrightarrow p_{i,:} \\
&\qquad \textbf{for } j = k+1 \textbf{ to } m \\
&\qquad\qquad \ell_{jk} = u_{jk}/u_{kk} \\
&\qquad\qquad u_{j,k:n} = u_{j,k:n} - \ell_{jk} u_{k,k:n}
\end{aligned}
$$

- Flop count $\sim \sum_{k=1}^{n} 2(n-k)(n-k) \sim 2\sum_{k=1}^{n} k^2 \sim 2n^3/3$, same as without pivoting
- Question: Can u_{kk} be 0?

An Alternative Implementation

- In practice, L and U overwrite A, and P is represented by a vector

> Gaussian elimination with partial pivoting (alternative)
>
> $p = [1, 2, \cdots, n]$;
> **for** $k = 1$ **to** $n - 1$
> $\quad i \leftarrow \arg\max_{i \geq k} |a_{ik}|$
> $\quad a_{k,1:n} \leftrightarrow a_{i,1:n}$
> $\quad p_k \leftrightarrow p_i$
> $\quad a_{k+1:n,k} \leftarrow a_{k+1:n,k} / a_{k,k}$
> $\quad A_{k+1:n,k+1:n} \leftarrow A_{k+1:n,k+1:n} - a_{k+1:n,k} * a_{k,k+1:n}$

- Using LU factorization with partial pivoting to solve $Ax = b$:
  1. $A = P^T LU$; (LU factorization with partial pivoting)
  2. $Ly = Pb$; (Forward substitution, where $(Pb)_i = b(p_i)$)
  3. $Ux = y$; (Back substitution)
- If the augmented matrix is used, then first two steps are merged.

Complete Pivoting

- More generally, pivot can be chosen from entries (i, j), $i \geq k$, $j \geq k$

$$\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & \mathbf{x} & x_{ij} & \mathbf{x} \\ & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ & \mathbf{x} & 0 & \mathbf{x} \\ & \times & x_{ij} & \times \\ & \mathbf{x} & 0 & \mathbf{x} \end{bmatrix}$$

  and we then permute row i with row k, column j with column k
- In matrix operations, complete can be expressed as

$$\underbrace{L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1}_{L^{-1}P}\,A\,\underbrace{Q_1Q_2\cdots Q_{n-1}}_{Q} = U$$

- Therefore, $PAQ = LU$ where $P = P_{n-1}\cdots P_2P_1$ and $L = \left(L'_{n-1}\cdots L'_2L'_1\right)^{-1}$

Complete Pivoting

- By choosing largest absolute value among these entries, complete pivoting gives better stability theoretically (to be discussed later)
- Similar to LU with complete pivoting, LDL^T with pivoting (for symmetric indefinite systems) permutes both rows and columns

$$PAP^T = LDL^T$$

  - Pivot is found among diagonal entries
  - Permuting both rows and columns is necessary for preserving symmetry

- Complete pivoting is typically not used in practice, because it increases cost in searching pivot and complexity of implementation
  - complete pivoting incurs $O(n^3)$ comparisons, whereas partial pivoting incurs $O(n^2)$

Banded Linear Systems

- A matrix A is *banded* if there is a narrow band around the main diagonal such that all of the entries of A outside of the band are zero
- If A is $n \times n$, and there exist p and q such that $a_{ij} = 0$ whenever $i > j + p$ or $j > i + q$, then we say A is banded with bandwidth $p + q + 1$
- p and q are the *lower bandwidth* and *upper bandwidth*, respectively
- In $A = LU$ (LU without pivoting), L has lower bandwidth p and U has upper bandwidth q. Total flop count is about $2npq$
- In $PA = LU$ (LU with column pivoting), row-interchanges enlarge bandwidth.
    - U has upper bandwidth $p + q$
    - L has at most $p + 1$ nonzeros per column, so it would have lower bandwidth p if ℓ_k is not permuted after eliminating kth column
    - Total flop count is about $2np(p + q)$

Reference: Gene H. Golub and Charles F. Van Loan, *Matrix Computations*, 3rd edition, John Hopkins University Press, 1996. Section 4.3.

Alternative Linear Solvers

Gauss-Jordan elimination: At the kth step, use pivot to eliminate nonzeros in column k both above and below diagonal. It is more expensive than Gaussian elimination. How much more?

Cramer's rule $x_i = \det(A^{(i)})/\det(A)$, where $A^{(i)}$ is obtained from A by replacing ith column by b. This is overly expensive and unstable, so it is never used except for $n = 2$ or 3

Strassen's method (and similar recursive algorithms) for matrix-matrix multiplication can be modified to solve linear systems in $O(n^s)$ flops for $s < 3$, but are not advantageous in practice

These are direct methods. Other important classes of methods include

- *iterative methods* (later) and
- *multigrid methods*