LUDecomposition

2020年3月9日 星期一 09:22

LUDecomposition

Solving System of Linear Equations

System of linear equations

n unknowns, m linear equations

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases}$$

In matrix notations, it can be written as

$$Ax = b,$$

where

$$A = \underbrace{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}}_{\text{coefficient matrix}} \quad x = \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}}_{\text{unknown vector}} \quad b = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}}_{\text{right-hand side (RHS)}}$$

1. Solvablity of system of linear equations

- The solution of $Ax=b$ must fall into one of the 3 cases.
  - Case I: $Ax=b$ has no solution. Example: $\begin{pmatrix}1&0\\1&0\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}=\begin{pmatrix}-1\\1\end{pmatrix}$ has no solution.
  - Case II: $Ax=b$ has a unique solution. Example: $\begin{pmatrix}1&0\\0&1\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}=\begin{pmatrix}2\\3\end{pmatrix}$ has a unique solution $x=\begin{pmatrix}2\\3\end{pmatrix}$. $\begin{pmatrix}x_1\\x_2\end{pmatrix}=\begin{pmatrix}2\\3\end{pmatrix}$
  - Case III: $Ax=b$ has infinitely many solutions.

$\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}=\begin{pmatrix}a\\1\\1-a\end{pmatrix}$

Example: $\begin{pmatrix}1&1&1\\0&1&0\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}=\begin{pmatrix}2\\1\end{pmatrix}$ has solutions in the form of $x=\begin{pmatrix}0\\1\\1\end{pmatrix}+\begin{pmatrix}a\\0\\-a\end{pmatrix}$ $\forall a \in \mathbb{R}$.

- Thm: Let $A\in\mathbb{R}^{m\times n}$ and $b\in\mathbb{R}^m$. Then the linear equation $Ax=b$ is solvable (i.e., $\exists$ at least one $x\in\mathbb{R}^n$ s.t. $Ax=b$) if and only if $b\in Ran(A)$.

Proof. By the definition of $Ran(A)$.

Some results on the solutions of $Ax=b$

- $Ax=b$ has at least one solution, (i.e. either (Case II) or (Case III)) $\Leftrightarrow b\in Ran(A)$ (By definition of A).
- $Null(A)\neq\{0\} \Rightarrow Ax=b$ cannot has a unique solution (i.e. either (Case I) or (Case III)) (proved by the definition of $Null(A)$)
- $m<n$ (i.e., less equations than unknowns) $\Rightarrow Ax=b$ cannot has a unique solution (i.e. either (Case I) or (Case III)) (proved by using the last statement)
- $m>n$ (i.e., more equations than unknowns) $\Rightarrow \exists b$ s.t. $Ax=b$ has no solution. (i.e. $\exists b$ in (Case I))

We focus only on the case when A is nonsingular.

- Definition: An $n\times n$ square matrix A is non-singular if $Rank(A)=n$.
- A is nonsingular $\Leftrightarrow Ran(A)=\mathbb{R}^n$
- Definition: An $n\times n$ square matrix A is invertible if there exists an $n\times n$ matrix, denoted by A^{-1}, such that $AA^{-1}=A^{-1}A=I$.
- Thm: Let $A\in\mathbb{R}^{n\times n}$ be an $n\times n$ square matrix. Then, A is nonsingular $\Leftrightarrow$ A is invertible $\Leftrightarrow$ $Ax=b$ has a unique solution for any $b\in\mathbb{R}^n$. $\Leftrightarrow \det(A)\neq 0$

Proof. From a standard linear algebra textbook.

(Proof 略)

Theorem 1: Let $A\in\mathbb{R}^{m\times n}$ and $b\in\mathbb{R}^m$. The linear equation $Ax=b$ is solvable (i.e. at least one sol.) if and only if $b\in Ran(A)$

Proof: From the definition of $Ran(A)$ $\quad$ def. $:=\{Ay \mid y\in\mathbb{R}^n\}$

- $Ax=b$ has a solution $\Rightarrow b=Ax\in Ran(A)$
- $b\in Ran(A) \Rightarrow \exists x$ s.t. $b=Ax$ ▨

Theorem 2: $Null(A)\neq\{0\} \Rightarrow Ax=b$ cannot has a unique solution.

Proof: Since $Null(A)\neq\{0\}$, $\exists x_0\neq 0$ s.t. $x_0\in Null(A)$ $\Rightarrow Ax_0=0$

如果 $Null(A)\neq\{0\}$，那就表示有一個 x_0 在 $Null(A)$ 中而 x_0 不是 0，而等於 $Ax_0=0$（Null space 定義）

So, if $\exists$ a sol. x_1 of $Ax_1=b$, then

$A(x_1+x_0)=Ax_1+Ax_0=b+0=b$

$\Rightarrow x_1+x_0$ is also a solution. but $x_1+x_0\neq x_1$ ($\because x_0\neq 0$)

$=$ we have more than one solutions. ▨

Corollary: ① If $m<n$, A = wide matrix

$\Rightarrow$ 因為 $Null(A)$ def. $:=\{x\in\mathbb{R}^n \mid Ax=0\}$, $Null(A)\in\mathbb{R}^n$，但 $Rank(A)\leq m$

所以 $Null(A)$ 至少都是 $\mathbb{R}^{n-m}>0$

$\Rightarrow Null(A)\neq\{0\}$

(Thm 2) $\Rightarrow Ax=b$ cannot has a unique solution

$\Rightarrow$ 不考慮 case 1.

② If $m>n$, then $\exists b$ s.t. $Ax=b$ has no solution.

Proof. A = tall matrix

Then, $\dim(Ran(A))\leq n$

And $Ran(A)\subset\mathbb{R}^m$ (def.) $\Rightarrow Ran(A)\subsetneq\mathbb{R}^m$ (Range of A smaller than $\mathbb{R}^m$ but not $\mathbb{R}^m$)

$\exists b_0\notin Ran(A)$ but $b_0\in\mathbb{R}^m \overset{\text{thm 1}}{\Rightarrow}$ there is no solution of $Ax=b_0$.

也就是說，$Ran(A)$ much smaller than $\mathbb{R}^m$, By Thm 1，你總可以找到一個 b_0 s.t. $Ax=b_0$ 無 solution.

$\Rightarrow$ 不考慮 case 2.

Proof 略:

The solution $Ax=b$

$A^{-1}Ax=A^{-1}b$

$x=A^{-1}b$

(invertible $\Rightarrow$ unique sol.)

因為找 inverse 太麻煩 所以不直接 solve x.

所以我們用 guassian 去 solve x.

2. Gaussian Elimination

An 3×3 Example:

$$\begin{cases} 2x_1+x_2-x_3=1 & ① \\ 4x_1+5x_2-3x_3=-3 & ② \\ -2x_1+5x_2-2x_3=-8 & ③ \end{cases}$$

Step 1: Eliminate x_1 in ②&③

Row operations:

$$\begin{matrix} ①\to① \\ ②-①\times2\to④ \\ ③+①\times1\to⑤ \end{matrix} \quad \begin{cases} 2x_1+x_2-x_3=1 & ① \\ 3x_2-x_3=-5 & ④ \\ 6x_2-3x_3=-7 & ⑤ \end{cases}$$

Step 2: Eliminate x_2 in ⑤

$$\begin{matrix} ①\to① \\ ④\to④ \\ ⑤-④\times2\to⑥ \end{matrix} \quad \begin{cases} 2x_1+x_2-x_3=1 & ① \\ 3x_2-x_3=-5 & ④ \\ -x_3=3 & ⑥ \end{cases}$$ easy to solve

Step 3: Solve the reduced linear equation

Solve ⑥, since it contains only x_3: $-x_3=3 \Rightarrow x_3=-3$

Solve ④, since it contains only x_2,x_3 and x_3 is solved. $3x_2-x_3=3x_2-(-3)=-5 \Rightarrow x_2=-\frac{8}{3}$

Solve ①, since x_2 and x_3 are already solved. $2x_1+x_2-x_3=2x_1-\frac{8}{3}-(-3)=1 \Rightarrow x_1=\frac{1}{3}$

Matrix Reformulation:

Row operations are equivalent to left-multiplications

$$\begin{array}{c|c} A & b \end{array} \quad \left[\begin{array}{ccc|c} 2&1&-1&1 \\ 4&5&-3&-3 \\ -2&5&-2&-8 \end{array}\right]$$

Step 1:

$$\underbrace{\begin{bmatrix}1&0&0\\-2&1&0\\1&0&1\end{bmatrix}}_{L_1}\left[\begin{array}{ccc|c} 2&1&-1&1 \\ 4&5&-3&-3 \\ -2&5&-2&-8 \end{array}\right] \to \left[\begin{array}{ccc|c} 2&1&-1&1 \\ 0&3&-1&-5 \\ 0&6&-3&-7 \end{array}\right]$$

Step 2:

$$\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\0&-2&1\end{bmatrix}}_{L_2}\left[\begin{array}{ccc|c} 2&1&-1&1 \\ 0&3&-1&-5 \\ 0&6&-3&-7 \end{array}\right] \to \left[\begin{array}{ccc|c} 2&1&1&1 \\ 0&3&-1&-5 \\ 0&0&-1&3 \end{array}\right]$$ easy to solve

Step 3: Solve

$$\left[\begin{array}{ccc|c} 2&1&1&1 \\ 0&3&-1&-5 \\ 0&0&-1&3 \end{array}\right] \to \begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}=\begin{pmatrix}\frac13\\-\frac83\\-3\end{pmatrix}$$ last to solve / first to solve

The coefficient matrix is upper triangular.

$$Ax=b \to L_1Ax=L_1b \to L_2L_1Ax=L_2L_1b$$

$$\begin{bmatrix}x&x&x\\x&x&x\\x&x&x\end{bmatrix} \quad \underbrace{\begin{bmatrix}1&0&0\\-2&1&0\\1&0&1\end{bmatrix}}_{L_1} \quad \begin{bmatrix}x&x&x\\0&x&x\\0&x&x\end{bmatrix} \quad \underbrace{\begin{bmatrix}1&0&0\\0&1&0\\0&-2&1\end{bmatrix}}_{L_2} \quad \begin{bmatrix}2&1&1\\0&3&-1\\0&0&-1\end{bmatrix}$$

Gaussian Elimination vs. LU decomposition

We first show that

L_1, L_2 就是 coefficient matrix

$$L_1^{-1}=\begin{bmatrix}1&0&0\\2&1&0\\-1&0&1\end{bmatrix}$$ ← 他即等於 Lower-triangle 正負調轉

Proof: Let $v_1=\begin{pmatrix}0\\2\\-1\end{pmatrix}$. Then $L_1=I-v_1e_1^T$

Then $(I-v_1e_1^T)(I+v_1e_1^T)=I-v_1e_1^T+v_1e_1^T-v_1\overset{\text{a scalar}}{e_1^Tv_1}e_1^T = I-(e_1^Tv_1)v_1e_1^T$

$e_1^Tv_1=(1\ 0\ 0)\begin{pmatrix}0\\2\\-1\end{pmatrix}=0$

$\Rightarrow (I-v_1e_1^T)(I+v_1e_1^T)=I$.

Similarly, $(I+v_1e_1^T)(I-v_1e_1^T)=I$.

So, $L_1^{-1}=I+v_1e_1^T$. ▨

Similarly,

$$L_2^{-1}=\begin{bmatrix}1&0&0\\0&1&0\\0&2&1\end{bmatrix}$$ (Let $v_2=\begin{pmatrix}0\\0\\2\end{pmatrix}$. Then $L_2=I-v_2e_2^T$, $L_2^{-1}=I+v_2e_2^T$)

Let

$$U=\begin{bmatrix}2&1&1\\0&3&-1\\0&0&-1\end{bmatrix}$$ (upper triangular)

Since $L_2L_1A=U$, we have $A=L_1^{-1}L_2^{-1}U$.

Let $L=L_1^{-1}L_2^{-1}=(I+v_1e_1^T)(I+v_2e_2^T)$
$=I+v_1e_1^T+v_2e_2^T+v_1e_1^Tv_2e_2^T$
$=I+v_1e_1^T+v_2e_2^T+v_1(e_1^Tv_2)e_2^T$ $\quad e_1^Tv_2=(1\ 0\ 0)\begin{pmatrix}0\\0\\2\end{pmatrix}=0$
$=I+v_1e_1^T+v_2e_2^T$
$[1\ 0\ 0]$

Denote $U=L_2L_1A$

Then $L_2L_1A=U$

$L_2^{-1}L_2L_1A=L_2^{-1}U$ (L_2 is invertible $\because \det(L_2)=1\neq0$)

$L_1A=L_2^{-1}U$ (similarly, L_1 is invertible)

$L_1^{-1}L_1A=L_1^{-1}L_2^{-1}U$

$A=L_1^{-1}L_2^{-1}U$

$L_1^{-1}L_2^{-1}=(I+v_1e_1^T)(I+v_2e_2^T)$

$=I+v_1e_1^T+v_2e_2^T+v_1 e_1^Tv_2 e_2^T$ ← 是 v_2 第一個數，0

$=I+v_1e_1^T+v_2e_2^T$

$=\begin{bmatrix}1&0&0\\2&1&0\\-1&2&1\end{bmatrix}$

$$\Rightarrow L=\begin{bmatrix}1&&\\2&1&0\\-1&2&1\end{bmatrix}$$

Therefore,

$$A=\underbrace{\begin{bmatrix}1&&\\2&1&\\-1&2&1\end{bmatrix}}_{\text{Lower triangular } L}\underbrace{\begin{bmatrix}2&1&1\\&3&-1\\&&-1\end{bmatrix}}_{\text{Upper triangular } U}$$

and $L^{-1}=L_2L_1$.

With these, the Gaussian Elimination can be viewed as:

- Step I: Compute $A=LU$, where L is lower triangular with diagonals 1, U is upper triangular.
  (so that $Ax=b \Leftrightarrow LUx=b \Leftrightarrow Ux=y$ and $Ly=b$.) (change of variable $Ly=b$, $Ux=y$)
- Step II: Solve $Ly=b$.
- Step III: Solve $Ux=y$.

all are easy to solve

3. LU decomposition and Triangular systems.

We will present Steps I, II, and III for general n.

We start with Steps II and III.

Forward Substitution for lower triangular system

A matrix $L\in\mathbb{R}^{n\times n}$ is called Lower Triangular if $l_{ij}=0$ $\forall i<j$.

i.e.

$$L=\begin{bmatrix}l_{11}&&&\\l_{21}&l_{22}&&\\\vdots&&\ddots&\\l_{n1}&l_{n2}&\cdots&l_{nn}\end{bmatrix}$$

上面 step II 中的 $Ly=b$, $y=L^{-1}b$, 而

$L^{-1}=L_1^{-1}L_2^{-1}$

$L=L_2L_1$

而 L_2L_1b 即是 guassian elimination 右手邊的操作。

To solve the linear equation $Lx=b$ where L is low triangular

We solve x_1 first, and then x_2, …, and x_n the last.

For x_1, use the first equation

$l_{11}x_1=b_1 \Rightarrow x_1=b_1/l_{11}$

For x_2, use the 2nd equation

$l_{21}x_1+l_{22}x_2=b_2 \Rightarrow x_2=(b_2-l_{21}x_1)/l_{22}$

⋮

For x_i, use the i-th equation

$\sum_{k=1}^{i-1}l_{ik}x_k+l_{ii}x_i=b_i \Rightarrow x_i=\left(b_i-\sum_{k=1}^{i-1}l_{ik}x_k\right)/l_{ii}$ (x_k, $k=1,\cdots,i-1$, already solved)

⋮

This procedure is called Forward Substitution.

Example:

$$\begin{bmatrix}1&&\\2&1&\\-1&2&1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}1\\-3\\-8\end{bmatrix}$$

$x_1=1$

$2x_1+x_2=-3 \Rightarrow x_2=-3-2x_1=-3-2\cdot1=-5$

$-x_1+2x_2+x_3=-8 \Rightarrow x_3=-8+x_1-2x_2=-8+1-2(-5)=3$

y: $\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}=\begin{pmatrix}1\\-5\\3\end{pmatrix}$

Pseudo code:

```
for i=1:n
    for k=1:i-1
        b_i = b_i - l_ik x_k
    end
    x_i = b_i / l_ii
end
```

最入的 loop 也只需用在 equation ／ 和第 i 個 equation. Other loops available (e.g., k-i-loop)

Computational complexity:

$$\sum_{i=1}^{n}\left(\left(\sum_{k=1}^{i-1}2\right)+1\right)=\sum_{i=1}^{n}(2(i-1)+1)=\sum_{i=1}^{n}(2i-1)=2\sum_{i=1}^{n}i-\sum_{i=1}^{n}1 = 2\cdot\frac{n(n+1)}{2}+n = n^2+2n \sim O(n^2)$$

Back substitution for upper triangular system

A matrix $U\in\mathbb{R}^{n\times n}$ is called Upper Triangular if $u_{ij}=0$ $\forall i>j$.

i.e.

$$U=\begin{bmatrix}u_{11}&u_{12}&\cdots&u_{1n}\\&u_{22}&\cdots&u_{2n}\\&&\ddots&\vdots\\&&&u_{nn}\end{bmatrix}$$

To solve the linear equation $Ux=b$ where U is upper triangular.

We solve x_n first, x_{n-1} the second, …, and x_1 the last.

For x_n, use the last equation

$u_{nn}x_n=b_n \Rightarrow x_n=b_n/u_{nn}$

For x_{n-1}, use the 2nd equation

$u_{n-1,n-1}x_{n-1}+u_{n-1,n}x_n=b_{n-1} \Rightarrow x_{n-1}=(b_{n-1}-u_{n-1,n}x_n)/u_{n-1,n-1}$

⋮

For x_i, use the i-th equation

$u_{ii}x_i+\sum_{k=i+1}^{n}u_{ik}x_k=b_i \Rightarrow x_i=\left(b_i-\sum_{k=i+1}^{n}u_{ik}x_k\right)/u_{ii}$ (x_k, $k=i+1,\cdots,n$, already solved)

⋮

This procedure is known as Back Substitution.

Example:

$$\begin{bmatrix}2&1&1\\&3&-1\\&&-1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}1\\-5\\3\end{bmatrix}$$ ← Use the previous y.

For x_3: $-x_3=3 \Rightarrow x_3=-3$

For x_2: $3x_2-x_3=-5 \Rightarrow x_2=(-5+x_3)/3=(-5+(-3))/3=-\frac83$

For x_1: $2x_1+x_2+x_3=1 \Rightarrow x_1=(1-x_2-x_3)/2=(1-(-\frac83)+3)/2=\frac{10}{3}$

pseudo code:

```
for i=n:-1:1
    for k=i+1:n
        b_i = b_i - u_ik x_k
    end
    x_i = b_i / u_ii
end
```

Other implementation available.

Computational cost:

$$\sum_{i=n}^{1}\left(\left(\sum_{k=i+1}^{n}2\right)+1\right)=\sum_{i=n}^{1}(2(n-i)+1)=2\sum_{i=n}^{1}(n-i)+\sum_{i=n}^{1}1 = 2\sum_{i=1}^{n}i+n = 2\frac{n(n+1)}{2}+n = n^2+2n \sim O(n^2)$$

Now let's present Step I — LU decomposition.

LU decomposition

Let $A\in\mathbb{R}^{n\times n}$ be an $n\times n$ matrix. Then there exist a decomposition,

$$A=LU,$$

where $L\in\mathbb{R}^{n\times n}$ is lower triangular with diagonals 1, and $U\in\mathbb{R}^{n\times n}$ is upper triangular.

$$L=\begin{bmatrix}1&&&\\l_{21}&1&&\\\vdots&&\ddots&\\l_{n1}&l_{n2}&\cdots&1\end{bmatrix} \quad U=\begin{bmatrix}u_{11}&u_{12}&\cdots&u_{1n}\\&u_{22}&\cdots&u_{2n}\\&&\ddots&\vdots\\&&&u_{nn}\end{bmatrix}$$

上述的 step II, III 講完，現在才回到 step I.

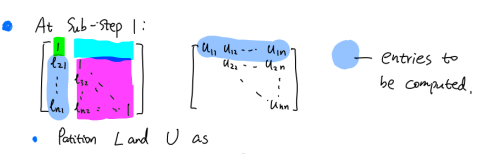
How to find LU decomposition?

The Gaussian Elimination provide an algorithm.

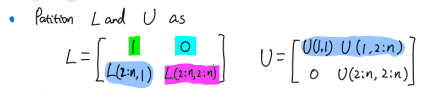
Here we derive the LU decomposition directly, where L is obtained column-by column and U is obtained row by row.

For this purpose, we first introduce some notations.

we used the following notations for sub-matrices. (matlab notation)



- At Sub-step 1:



- Partition L and U as

$$L=\begin{bmatrix}1&0\\L(2:n,1)&L(2:n,2:n)\end{bmatrix} \quad U=\begin{bmatrix}U(1,1)&U(1,2:n)\\0&U(2:n,2:n)\end{bmatrix}$$

Then,

$$LU=\begin{bmatrix}U(1,1)&U(1,2:n)\\U(1,1)L(2:n,1)&L(2:n,1)U(1,2:n)+L(2:n,2:n)U(2:n,2:n)\end{bmatrix}$$ $(n-1)$

Comparing it with

$$A=\begin{bmatrix}A(1,1)&A(1,2:n)\\A(2:n,1)&A(2:n,2:n)\end{bmatrix}$$

LU 逐步 term by term 對比.

$$\begin{bmatrix}A_{11}&\cdots&A_{1n}\\A_{21}&A_{22}&\\\vdots&&\ddots\\A_{n1}&\cdots&A_{nn}\end{bmatrix}$$

We obtain

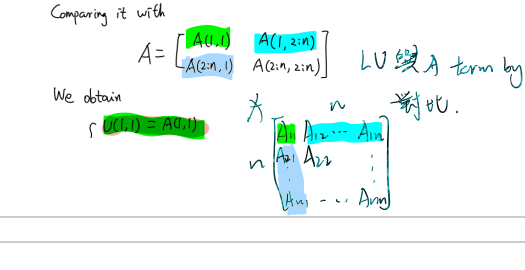
$U(1,1)=A(1,1)$

$U(1,2:n)=A(1,2:n)$

$U(1,1)L(2:n,1)=A(2:n,1) \Rightarrow L(2:n,1)=A(2:n,1)/U(1,1)$

⋮

- At sub-step k:



- Partition L and U as

$$L=\begin{bmatrix}L(1:k-1,1:k-1)&0&0\\L(k,1:k-1)&1&0\\L(k+1:n,1:k-1)&L(k+1:n,k)&L(k+1:n,k+1:n)\end{bmatrix}$$

$$U=\begin{bmatrix}U(1:k-1,1:k-1)&U(1:k-1,k)&U(1:k-1,k+1:n)\\0&U(k,k)&U(k,k+1:n)\\0&0&U(k+1:n,k+1:n)\end{bmatrix}$$

Then, by direct calculation, (don't care)

$$LU=\begin{bmatrix}\times&\times&\times\\\times&L(k,1:k-1)U(1:k-1,k)+U(k,k)&L(k,1:k-1)U(1:k-1,k+1:n)+U(k,k+1:n)\\\times&L(k+1:n,1:k-1)U(1:k-1,k)+U(k,k)L(k+1:n,k)&\times\end{bmatrix}$$

Compare it with

$$A=\begin{bmatrix}A(1:k-1,1:k-1)&A(1:k-1,k)&A(1:k-1,k+1:n)\\A(k,1:k-1)&A(k,k)&A(k,k+1:n)\\A(k+1:n,1:k-1)&A(k+1:n,k)&A(k+1:n,k+1:n)\end{bmatrix}$$

We get

① $\begin{cases}L(k,1:k-1)U(1:k-1,k)+U(k,k)=A(k,k)\\L(k,1:k-1)U(1:k-1,k+1:n)+U(k,k+1:n)=A(k,k+1:n)\end{cases}$, i.e., 可以合併

估計 $\to L(k,1:k-1)U(1:k-1,k:n)+U(k,k:n)=A(k,k:n)$, which implies 未知

$U(k,k:n)=A(k,k:n)-L(k,1:k-1)U(1:k-1,k:n)$

② $L(k+1:n,1:k-1)U(1:k-1,k)+U(k,k)L(k+1:n,k)=A(k+1:n,k)$, which gives

$L(k+1:n,k)=[A(k+1:n,k)-L(k+1:n,1:k-1)U(1:k-1,k)]/U(k,k)$

⋮

In summary, the LU decomposition is done by

```
for k=1,2,…,n
    U(k,k:n) = A(k,k:n) - L(k,1:k-1) U(1:k-1,k:n)
    L(k+1:n,k) = [A(k+1:n,k) - L(k+1:n,1:k-1) U(1:k-1,k)]/U(k,k)
end
```

In kth step, we only get kth column of L and kth row of U

In practice, a_{ij} can be overwritten by $\begin{cases}u_{ij} & \text{if } i\leq j\\ l_{ij} & \text{if } i>j\end{cases}$

pseudo-code for LU decomposition.

```
for k=1,2,…,n
    A(k,k:n) = A(k,k:n) - A(k,1:k-1) A(1:k-1,k:n)
    A(k+1:n,k) = (A(k+1:n,k) - A(k+1:n,1:k-1) A(1:k-1,k))/A(k,k)
end
```

Remarks:

原來都是用 A 的 U 部分，?Jordan memory

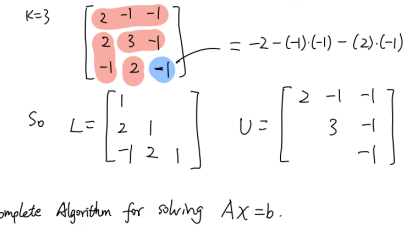
P.S. 這裏 derive 的跟 Gaussian 的 … 不同.

P.S. LU Decomposition 有 recursive form，但老師在這裏沒有寫的出來.

1. Due to the overwriting, the output of the above code is

$$L=\begin{pmatrix}1&&&\\a_{21}&1&&\\\vdots&&\ddots&\\a_{n1}&a_{n2}&\cdots&1\end{pmatrix} \text{ and } U=\begin{pmatrix}a_{11}&a_{12}&\cdots&a_{1n}\\&a_{22}&\cdots&a_{2n}\\&&\ddots&\vdots\\&&&a_{nn}\end{pmatrix}$$

2. There are other implementations. (略!)
3. Graphical illustration:


Assume k is in step k: a small loop from k to n.
column k
row k
← blue 部份是 assume k 在 increase to n.

Computational complexity:

In the first line:

$A(k,k:n)=A(k,k:n)-A(k,1:k-1)A(1:k-1,k:n)$ $\quad 1\times(k-1) \quad (k-1)\times(n-k+1)$

So the complexity is $2(k-1)(n-k+1)+(n-k+1)=(2k-1)(n-k+1)$ (for matrix-matrix product, for subtraction)

In the 2nd line:

$A(k+1:n,k)=(A(k+1:n,k)-A(k+1:n,1:k-1)A(1:k-1,k))/A(k,k)$ $\quad (n-k)\times(k-1) \quad (k-1)\times1$

So the complexity is $2(k-1)(n-k)+2(n-k)=2k(n-k)$ (for m-v product, for div and subtraction)

The total complexity is

$$\sum_{k=1}^{n}[(2k-1)(n-k+1)+2k(n-k)] = \sum_{k=1}^{n}(4nk-4k^2+3k-n-1) = 4n\sum_{k=1}^{n}k-4\sum_{k=1}^{n}k^2+3\sum_{k=1}^{n}k-n\sum_{k=1}^{n}1-\sum_{k=1}^{n}1 = 4n\frac{n(n+1)}{2}-4\cdot\frac{n(n+1)(2n+1)}{6}+3\frac{n(n+1)}{2}-n^2-n = \frac23n^3+\text{lower order terms (e.g. } n^2, n, \text{constant)} \sim O(n^3)$$

Example: Find LU Decomposition of

$$A=\begin{bmatrix}2&1&-1\\4&5&-3\\-2&5&-2\end{bmatrix}$$

k=1: $\begin{bmatrix}2&1&-1\\2&5&-3\\-1&5&-2\end{bmatrix}$ $\quad$ k=2: $\begin{bmatrix}2&1&-1\\2&3&-1\\-1&2&-2\end{bmatrix}$

k=3: $\begin{bmatrix}2&-1&-1\\2&3&-1\\-1&2&-1\end{bmatrix}$ $=-2-(-1)(-1)-(2)(-1)$

So $L=\begin{bmatrix}1&&\\2&1&\\-1&2&1\end{bmatrix}$ $\quad U=\begin{bmatrix}2&-1&-1\\&3&-1\\&&-1\end{bmatrix}$

Complete Algorithm for solving $Ax=b$.

```
for k=1,2,…,n
    A(k,k:n) = A(k,k:n) - A(k,1:k-1) A(1:k-1,k:n)
    A(k+1:n,k) = (A(k+1:n,k) - A(k+1:n,1:k-1) A(1:k-1,k))/A(k,k)
```

k=1: $\begin{bmatrix}2&1&-1\\2&5&-3\\-1&5&-2\end{bmatrix}$ $\quad$ k=2: $\begin{bmatrix}2&1&-1\\2&5-2&-3\\-1&5&-2\end{bmatrix}$

k=3

$$\begin{bmatrix}2&1&-1\\2&3&-1\\-1&2&-2\end{bmatrix} \quad -2-(-1\ 2)\begin{pmatrix}-1\\-1\end{pmatrix}=-1$$

output $=\begin{bmatrix}2&1&-1\\2&3&-1\\-1&2&-1\end{bmatrix}$ $\quad L\begin{bmatrix}1&&\\2&1&\\-1&2&1\end{bmatrix}$

$U\begin{bmatrix}2&1&-1\\&3&-1\\&&-1\end{bmatrix}$

Advantage of LU decomposition (over GE formulation)

1. No need to calculate $A=LU$ again if we solve linear systems of the same coefficient matrix A.
2. There are other usages of LU decomposition
   - Fast inversion of A.
     ① $A=LU$
     ② $A^{-1}=U^{-1}L^{-1}$
   - Determinant of A
     ① $A=LU$
     ② $\det(A)=\det(LU)=\det(L)\det(U)=\det(U)=u_{11}u_{22}\cdots u_{nn}$ ($\det(L)=1$)
3. we can find a very efficient implementation
4. LU can help on the theoretical analysis of GE. (understand the stability of GE).
5. Can utilize the structure of A to obtain more efficient algorithms than the original GE.

Complete Algorithm for solving $Ax=b$.

```
for k=1,2,…,n
    A(k,k:n) = A(k,k:n) - A(k,1:k-1) A(1:k-1,k:n)
    A(k+1:n,k) = (A(k+1:n,k) - A(k+1:n,1:k-1) A(1:k-1,k))/A(k,k)
end
for i=1:n
    b_i = b_i - A(i,1:i-1) b(1:i-1)
end
for i=n:-1:1
    b_i = b_i - A(i,i+1:n) b(i+1:n)
end
```

LU decomposition $A=LU$, where a_{ij} is overwritten by $\begin{cases}l_{ij} & \text{if } i>j\\ u_{ij} & \text{if } i\leq j\end{cases}$. So the result is $L=\begin{pmatrix}1&\\ a_{21}&\ddots\end{pmatrix}$, $U=\begin{pmatrix}a_{11}&\cdots&a_{1n}\\&\ddots&\vdots\\&&a_{nn}\end{pmatrix}$

Back substitution for solving $Ux=y$, where y is overwritten by x, i.e., $x=\begin{pmatrix}b_1\\\vdots\\b_n\end{pmatrix}$ — the output of the solution of $Ax=b$.

forward substitution for solving $Ly=b$, where b is overwritten by y. So $y=\begin{pmatrix}b_1\\\vdots\\b_n\end{pmatrix}$

Example: Solve

$$\begin{bmatrix}2&1&-1\\4&-1&-5\\2&7&9\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}0\\0\\4\end{bmatrix}$$

Step 1: LU decomposition.

$$\begin{bmatrix}2&1&-1\\4&-1&-5\\2&7&9\end{bmatrix}\to\begin{bmatrix}2&1&-1\\2&-1&-5\\1&7&9\end{bmatrix}\to\begin{bmatrix}2&1&-1\\2&-3&-3\\1&2&9\end{bmatrix}\to\begin{bmatrix}2&1&-1\\2&-3&-3\\1&-2&4\end{bmatrix}$$

So

$$L=\begin{bmatrix}1&&\\2&1&\\1&-2&1\end{bmatrix} \quad U=\begin{bmatrix}2&1&-1\\&-3&-3\\&&4\end{bmatrix}$$

Step 2: Forward substitution

$$\begin{bmatrix}1&&\\2&1&\\1&-2&1\end{bmatrix}\begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}=\begin{bmatrix}0\\0\\4\end{bmatrix}$$

$y_1=0$

$2y_1+y_2=0 \Rightarrow y_2=0-2y_1=0$

$y_1-2y_2+y_3=4 \Rightarrow y_3=4-y_1+2y_2=4$

Step 3: Back substitution

$$\begin{bmatrix}2&1&-1\\&-3&-3\\&&4\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}0\\0\\4\end{bmatrix}$$

$4x_3=4 \Rightarrow x_3=1$

$-3x_2-3x_3=0 \Rightarrow x_2=(0+3x_3)/(-3)=-1$

$2x_1+x_2-x_3=0 \Rightarrow x_1=(0+x_3-x_2)/2=1$

Pivoting

In the k-th sub-step, $A(k,k)$ is used as a denominator.

- If $A(k,k)=0$, then the LU decomposition cannot continue, even if A is invertible.

Example: $A=\begin{bmatrix}0&1\\1&0\end{bmatrix}$

- Also, if $A(k,k)$ (at step k) is very small, then the division by $A(k,k)$ will make L and U contain large entries. Consequently, we may get large error in the solution due to round-off error of digital computers.

Example: $\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{pmatrix}1\\2\end{pmatrix}$

Assume the computation is on a computer with 3 correct digits. The exact solution is $x_{true}=\begin{pmatrix}1.0001\cdots\\0.9999\cdots\end{pmatrix} \xrightarrow{\text{keep 3 digits}} \begin{pmatrix}1\\0.999\end{pmatrix}$

LU decomposition:

$$\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\to\begin{bmatrix}10^{-4}&1\\10^4&1\end{bmatrix}\to\begin{bmatrix}10^{-4}&1\\10^4&(1-10^4)\end{bmatrix}$$ $=-9999 \xrightarrow{\text{keep 3 digits}} -9990$

So the computed LU decomposition is

$L\approx\begin{bmatrix}1&\\10^4&1\end{bmatrix}$ $\quad U\approx\begin{bmatrix}10^{-4}&1\\&-9990\end{bmatrix}$

Forward substitution:

$$\begin{bmatrix}1&\\10^4&1\end{bmatrix}\begin{pmatrix}y_1\\y_2\end{pmatrix}=\begin{pmatrix}1\\2\end{pmatrix} \Rightarrow \begin{cases}y_1=1\\y_2=(2-10^4)=-9998 \overset{\text{keep 3 digits}}{=} -9990\end{cases}$$

Back substitution:

$$\begin{pmatrix}10^{-4}&1\\&-9990\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}=\begin{pmatrix}1\\-9990\end{pmatrix} \Rightarrow \begin{cases}x_2=\frac{-9990}{-9990}=1\\x_1=(1-x_2)/10^{-4}=\frac{1-1}{10^{-4}}=0\end{cases}$$

So the computed solution is $x_{computed}=\begin{pmatrix}x_1\\x_2\end{pmatrix}=\begin{pmatrix}0\\1\end{pmatrix}$,

which has a relative error (in 2-norm)

$$\frac{\|x_{computed}-x_{true}\|_2}{\|x_{true}\|_2}=\frac{\|\binom{0}{1}-\binom{1}{0.999}\|_2}{\|\binom{1}{0.999}\|_2}\approx\frac{1}{\sqrt2}\approx 70.7\%$$

To overcome these, we introduce a technique called "pivoting".

We call $A(k,k)$ in the denominator in the k-th substep "pivot".

Partial Pivoting:

- permute the equations so that the pivot is as large as possible, i.e.,

switch order of equations.

P: reordering of rows of A.

we find a permutation matrix P s.t. the LU decomposition of PA

$$PA=LU,$$

where the "pivot" is as large as possible → decrease the error.

- We say $P\in\mathbb{R}^{n\times n}$ is a permutation matrix if there is only exactly a "1" in each row and each column and all other entries are 0.

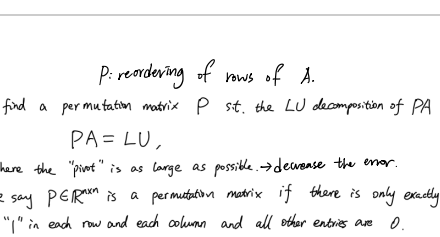
For example:

$\begin{bmatrix}0&1\\1&0\end{bmatrix}$ (we have $\begin{bmatrix}0&1\\1&0\end{bmatrix}\begin{bmatrix}a_1\\a_2\end{bmatrix}=\begin{bmatrix}a_2\\a_1\end{bmatrix}$, i.e., it permutes $\binom{a_1}{a_2}$ to $\binom{a_2}{a_1}$)

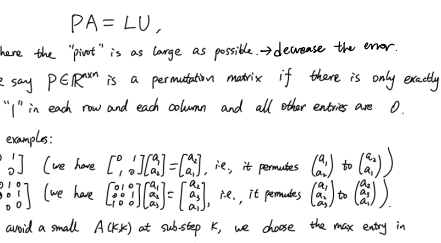
$\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$ (we have $\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}\begin{bmatrix}a_1\\a_2\\a_3\end{bmatrix}=\begin{bmatrix}a_2\\a_3\\a_1\end{bmatrix}$, i.e., it permutes $\begin{pmatrix}a_1\\a_2\\a_3\end{pmatrix}$ to $\begin{pmatrix}a_2\\a_3\\a_1\end{pmatrix}$)

- To avoid a small $A(k,k)$ at substep k, we choose the max entry in $A(k:n,k)$ in magnitude as "pivot".

In the original LU:

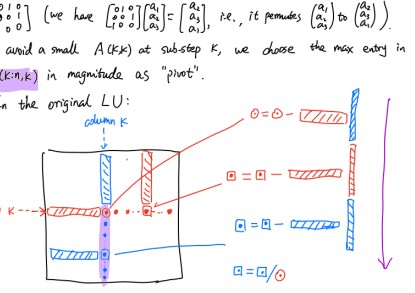


Original LU is the same as:



因為如果計完 k row 再 switch 是就計算時與都要考慮.

LU with partial pivoting: (adding row switch before modify the row)



The modification of $A(k,k:n)$ has to be done after row switching

```
P = I
for k=1,2,…,n
    A(k,k:n) = A(k,k:n) - A(k,1:k-1) A(1:k-1,k:n)   (*)
    j_k = arg max_{k≤i≤n} |A(i,k)|
    A(k,1:n) <-> A(j_k,1:n)
    P(k,1:n) <-> P(j_k,1:n)           } changing row ordering.
    A(k,k+1:n) = A(k,k+1:n) - A(k,1:k-1) A(1:k-1,k+1:n)
    A(k+1:n,k) = A(k+1:n,k)/A(k,k)
end
```

Complexity (?)

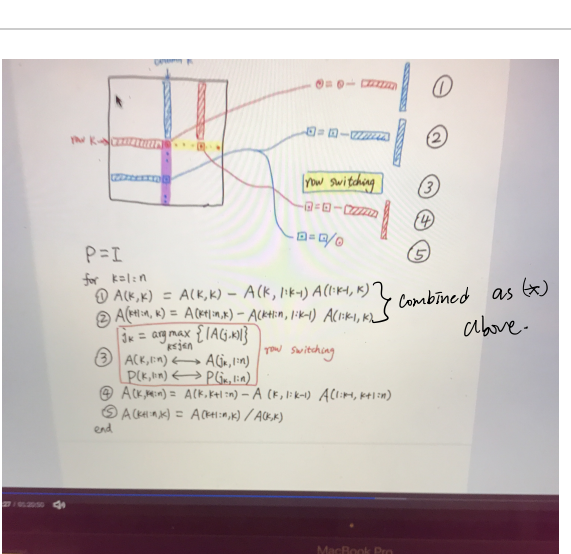
We will get $PA=LU$. Then,

$PAx=Pb \Leftrightarrow LUx=Pb \Leftrightarrow Ly=Pb$ and $Ux=y$.

Consequently, the forward substitution will solve $Ly=Pb$ and the back substitution still solve $Ux=y$.

- Example 1: $\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}1\\2\end{bmatrix}$ on a 3-digits computer

LU decomposition: $\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\to\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\to\begin{bmatrix}1&1\\10^{-4}&1\end{bmatrix}\to\begin{bmatrix}1&1\\10^{-4}&1-10^{-4}\end{bmatrix}=0.9999$



P = I

for k=1:n

① $A(k,k)=A(k,k)-A(k,1:k-1)A(1:k-1,k)$

② $A(k+1:n,k)=A(k+1:n,k)-A(k+1:n,1:k-1)A(1:k-1,k)$ } Combined as (*) above.

$j_k=\arg\max_{k\leq i\leq n}|A(i,k)|$

③ $A(k,1:n)\longleftrightarrow A(j_k,1:n)$, $P(k,1:n)\longleftrightarrow P(j_k,1:n)$ row switching

④ $A(k,k+1:n)=A(k,k+1:n)-A(k,1:k-1)A(1:k-1,k+1:n)$

⑤ $A(k+1:n,k)=A(k+1:n,k)/A(k,k)$

end

Step 1: Compute $PA=LU$

Step 2: Compute $Ly=Pb$ by forward substitution

Step 3: Compute $Ux=y$ by backward substitution

we have confidence that $\frac{A(k+1:n,k)}{A(k,k)}$ in Step ⑤ is always ≤ 1. So, the lower triangular matrix will be always smaller than 1.

$$\begin{bmatrix}10^{-4}&1\\1&1\end{bmatrix}\xrightarrow{P}\begin{bmatrix}1&1\\10^{-4}&1\end{bmatrix}\to\begin{bmatrix}1&1\\10^{-4}&1\end{bmatrix}\to\begin{bmatrix}1&1\\10^{-4}&1-10^{-4}\end{bmatrix}=0.9999$$

$P=\begin{bmatrix}0&1\\1&0\end{bmatrix}$ $\quad \xrightarrow{\text{keep 3 digits}} \begin{bmatrix}1&1\\10^{-4}&0.999\end{bmatrix}$

So $P=\begin{bmatrix}0&1\\1&0\end{bmatrix}$, $L=\begin{bmatrix}1&\\10^{-4}&1\end{bmatrix}$, $U=\begin{bmatrix}1&1\\&0.999\end{bmatrix}$ and $PA=LU$.

$LU=\begin{bmatrix}1&1\\10^{-4}&0.9991\end{bmatrix}$ 已經很接近 A

Forward substitution:

$$\begin{bmatrix}1&\\10^{-4}&1\end{bmatrix}\begin{bmatrix}y_1\\y_2\end{bmatrix}=\begin{bmatrix}0&1\\1&0\end{bmatrix}\begin{bmatrix}1\\2\end{bmatrix}=\begin{bmatrix}2\\1\end{bmatrix}$$

$\Rightarrow \begin{cases}y_1=2\\y_2=1-10^{-4}y_1=1-2\cdot10^{-4}=0.9998 \overset{\text{keep 3 digits}}{=} 0.999\end{cases}$

Back substitution:

$$\begin{bmatrix}1&1\\&0.999\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}=\begin{bmatrix}2\\0.999\end{bmatrix}$$

$\Rightarrow \begin{cases}x_2=0.999\\x_1=2-x_2=2-0.999=1.001 \overset{\text{keep 3 digits}}{=} 1.00\end{cases}$

So the computed solution is $x_{computed}=\begin{pmatrix}1.00\\0.999\end{pmatrix}$, and the relative error is $\frac{\|x_{computed}-x_{true}\|_2}{\|x_{true}\|_2}=\frac{\|\binom{1.00}{0.999}-\binom{1}{0.999}\|_2}{\|\binom{1}{0.999}\|_2}=0\%$

- Example 2: Find the partially pivoted LU decomposition of

$$A=\begin{bmatrix}2&1&-1\\4&5&-3\\-2&5&-2\end{bmatrix}$$

(Ignore the round-off errors)

$$\begin{bmatrix}2&1&-1\\④&5&-3\\-2&5&-2\end{bmatrix}\to\begin{bmatrix}4&5&-3\\2&1&-1\\-2&5&-2\end{bmatrix}\to\begin{bmatrix}4&5&-3\\\frac12&1&-1\\-\frac12&5&-2\end{bmatrix}\to$$

$P=\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix}$ 第一第二行調轉.

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

$$\begin{bmatrix}4&5&-3\\\frac12&-\frac32&-1\\-\frac12&\frac{15}{2}&-2\end{bmatrix}\to\begin{bmatrix}4&5&-3\\-\frac12&\frac{15}{2}&-2\\\frac12&-\frac32&-1\end{bmatrix}\to\begin{bmatrix}4&5&-3\\-\frac12&\frac{15}{2}&-2\\\frac12&-\frac15&-1\end{bmatrix}\to\begin{bmatrix}4&5&-3\\-\frac12&\frac{15}{2}&-\frac72\\\frac12&-\frac15&-\frac{6}{5}\end{bmatrix}$$

$P=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$ 第二第三行調轉，同時轉 P.

Therefore,

$$\underbrace{\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}}_{P}\underbrace{\begin{bmatrix}2&1&-1\\4&5&-3\\-2&5&-2\end{bmatrix}}_{A}=\underbrace{\begin{bmatrix}1&&\\-\frac12&1&\\\frac12&-\frac15&1\end{bmatrix}}_{L}\underbrace{\begin{bmatrix}4&5&-3\\&\frac{15}{2}&-\frac72\\&&-\frac65\end{bmatrix}}_{U}$$

Remark: $A=LU$ 變 $PA=LU$ 中的 L 和 U 與是不同的!

- There are other pivoting strategies, e.g., fully pivoting, where permutation on both equations and unknowns are imposed. ($PAQ=LU$, Q 在 column 那邊)

Remark: The strict lower triangular of L in partially pivoted LU decomposition always has an absolute value smaller than 1.

Advantage of LU decomposition

- No need to calculate $A=LU$ decomposition again if we solve linear systems of the same coefficient matrix A.
- Fast inversion of A.
  ① Compute $A=LU$.
  ② then $A^{-1}=U^{-1}L^{-1}$

The computational cost LU decomposition can be reduced if

- A is tridiagonal

$$A=\begin{bmatrix}\times&\times&&\\\times&\times&\times&\\&\times&\ddots&\ddots\\&&\times&\times\end{bmatrix} \quad \text{— } O(n^2)$$

- A is symmetric positive definite

$A=LL^T$ — Half of standard LU decomposition.

Matrix decomposition/factorization is the main theme of matrix computation, and there are other decompositions that can be used to solve $Ax=b$.

Cholesky Decomposition of Symmetric Positive Definite Matrices

Definition: A matrix $A\in\mathbb{R}^{n\times n}$ is symmetric positive semi-definite (SPSD) if:

① A is symmetric, i.e., $A=A^T$ and

② $\forall x\in\mathbb{R}^n$ and $x\neq0$, $x^TAx\geq0$.

If the "≥0" in ② is replaced by ">0", then A is called symmetric positive definite (SPD)

Facts:

(a) $A=C^TC$ for some $C\in\mathbb{R}^{m\times n}$ is SPSD.

(b) If A is SPD, then A is nonsingular.

(c) If A is SPSD, then $A(i_1:i_2,i_1:i_2)$ is SPSD for any $i_2\geq i_1$.

Examples of SPSD/SPD matrices:

(a) The Hessian of a convex function is SPSD

(b) The covariance matrix is SPSD.

(c) $\begin{bmatrix}2&1\\1&2\end{bmatrix}$ is SPD.

Cholesky decomposition: If $A\in\mathbb{R}^{n\times n}$ is SPD, then there exists a decomposition

$$A=LL^T,$$

where $L\in\mathbb{R}^{n\times n}$ is lower triangular.

Algorithms for Cholesky decomposition.

Sub-step 1: Partition L as $L=\begin{bmatrix}L(1,1)&0\\L(2:n,1)&L(2:n,2:n)\end{bmatrix}$

Then $LL^T=\begin{bmatrix}(L(1,1))^2&(L(1,1)L(2:n,1))^T\\L(1,1)L(2:n,1)&*\end{bmatrix}$

Compare it with $A=\begin{bmatrix}A(1,1)&(A(2:n,1))^T\\A(2:n,1)&*\end{bmatrix}$

we get

$L(1,1)^2=A(1,1) \Rightarrow L(1,1)=(A(1,1))^{1/2}$

$L(1,1)L(2:n,1)=A(2:n,1) \Rightarrow L(2:n,1)=A(2:n,1)/L(1,1)$

⋮

Sub-step k: Partition L as $L=\begin{bmatrix}L(1:k-1,1:k-1)&0&0\\L(k,1:k-1)&L(k,k)&0\\L(k+1:n,1:k-1)&L(k+1:n,k)&L(k+1:n,k+1:n)\end{bmatrix}$

Then

$$LL^T=\begin{bmatrix}*&*&*\\*&L(k,1:k-1)(L(k,1:k-1))^T+L(k,k)^2&*\\*&L(k+1:n,1:k-1)(L(k,1:k-1))^T+L(k+1:n,k)L(k,k)&*\end{bmatrix}$$

Compare it with $A=\begin{bmatrix}*&*&*\\*&A(k,k)&*\\*&A(k+1:n,k)&*\end{bmatrix}$

We obtain,

$L(k,1:k-1)(L(k,1:k-1))^T+(L(k,k))^2=A(k,k)$

$\Rightarrow L(k,k)=\left(A(k,k)-L(k,1:k-1)(L(k,1:k-1))^T\right)^{1/2}$ — The SPD-ness of A ensures this number is positive

$L(k+1:n,1:k-1)(L(k,1:k-1))^T+L(k+1:n,k)L(k,k)=A(k+1:n,k)$

$\Rightarrow L(k+1:n,k)=\left(A(k+1:n,k)-L(k+1:n,1:k-1)(L(k,1:k-1))^T\right)/L(k,k)$

⋮

So, the full Cholesky decomposition algorithm is

```
for k=1,2,…,n
    L(k,k) = (A(k,k) - L(k,1:k-1)(L(k,1:k-1))^T)^(1/2)
    L(k+1:n,k) = (A(k+1:n,k) - L(k+1:n,1:k-1)(L(k,1:k-1))^T)/L(k,k)
end
```

Remarks:

1. The computational cost is $\frac13n^3$ + lower order terms, which is about half of the standard LU decomposition.
2. The SPD-ness of A implies that $A(k,k)-L(k,1:k-1)(L(k,1:k-1))^T$ is positive. Therefore, no pivoting is not necessary.