Chapter 5　线性方程组直接法

张亚楠*

November 16, 2023

引例

假设

$$Ax = b,\quad 其中\quad A \in \mathbb{R}^{n\times n},\quad b \in \mathbb{R}^n$$

唯一可解. 如何求解？ Gauss- Jordan消元法! 最容易求解的A 的类型有哪些?

(1) 单位矩阵；对角矩阵

(2) 上(下)三角形矩阵

(3) 正交矩阵

1) 如果系数矩阵A稠密且固定不变，需要针对1000个不同的 右端项 b, 反复求解上式；如何避免重复计算？ （预先分解系数矩阵）

2) 如果系数矩阵规模很大，例如$n = 10^6$， 同时又是稀疏矩阵（绝大多数元素都是零），如何有效求解？ （迭代法）

3) 不同算法的计算复杂度如何衡量?

- 直接法：Gauss消去及其变形：LU, Cholesky, LDL,　QR,
- 迭代法：Jacobi, Gauss-Seidel, SOR ;　CG, PCG, GMRES, 其它 Krylov子空间方法

1　预备知识

向量和矩阵 $\mathbb{R}^{m\times n}$ 表示全体$m \times n$实矩阵向量空间； $A \in \mathbb{R}^{m\times n}$, 即

$$A = (a_{ij}) = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

$\vec{x} \in \mathbb{R}^n$ 即 $\vec{x} = (x_1, x_2, \cdots, x_n)^\top$.

*ynzhang@suda.edu.cn　苏州大学数学科学学院

1. 矩阵的基本运算 矩阵加法和标量乘法；矩阵乘法；转置矩阵；矩阵行列式 det (A)

2. 矩阵的特征值和谱半径

定义 1 设 $A \in \mathbb{R}^{n\times n}$, 若存在数$\lambda$ 和非零向量$\vec{x} \in \mathbb{R}^n$, 使得

$$A\vec{x} = \lambda\vec{x}$$

称λ 为A的特征值，$\vec{x}$为对应特征值λ的特征向量， A 的全体特征值称为A 的谱，记作$\sigma(A) = \{\lambda_1, \lambda_2, ..., \lambda_n\}$ ， 记

$$\rho(A) = \max_{1\le i\le n} |\lambda_i|$$

为矩阵A的谱半径.

记I是恒同矩阵，则方程

$$(\lambda I - A) * \vec{x} = 0$$

有非零解；进而

$$\det(\lambda I - A) = 0$$

上式左端是关于未知量 λ的n次代数多项式，称为 A 的特征多项式，记为 $p(\lambda)$. 上式称为A 的特征方程. 由代数知识，$p(\lambda)$ 在复数域有n个根λ_j，则

$$p(\lambda) = (\lambda - \lambda_1)(\lambda - \lambda_2)...(\lambda - \lambda_n)$$

直接计算$p(\lambda)$的根是数值不稳定的，需要用到后文的专门计算矩阵 特征对的方法。

1.1 几类特殊矩阵: ”好矩阵”

1. 对角阵；$d-$对角阵；例如三对角阵；

$$\begin{bmatrix} 3 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 3 & 1 & 0 \\ 0 & 0 & 1 & 3 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix},$$

2. 对称阵 $A = A^T$, 对称正定阵 A; 注意对称正定矩阵也可能很坏， 例如Hilbert矩阵

3. 正交矩阵；$P^T \cdot P = I$, 或者$P^\top \cdot P =$ 对角矩阵, Fourier矩阵, 正弦矩阵，余弦矩阵

上三角矩阵 T 、 Hessenberg矩阵H

$$T = \begin{bmatrix} 2 & 4 & 3 & 5 & 3 \\ 0 & 1 & 4 & 4 & 4 \\ 0 & 0 & 4 & 5 & 3 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix}, \quad H = \begin{bmatrix} 2 & 4 & 3 & 5 & 3 \\ 2 & 1 & 4 & 4 & 4 \\ 0 & 2 & 4 & 5 & 3 \\ 0 & 0 & 5 & 2 & 2 \\ 0 & 0 & 0 & 5 & 4 \end{bmatrix}$$

例 1 任意给定的上三角矩阵 $T\in\mathbb{R}^n$ 、 上Hessenberg矩阵$H\in\mathbb{R}^n$, 则

$$T\cdot H,\quad H\cdot T$$

仍然是上Hessenberg矩阵。 另外，下三角乘以下三角仍然是下三角矩阵。

例 2 检验 （1）

$$L_{31}*\boldsymbol{a}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\l_{31}&0&1&0\\0&0&0&1\end{bmatrix}*\begin{bmatrix}a_1\\a_2\\a_3\\a_4\end{bmatrix}=\begin{bmatrix}a_1\\a_2\\l_{31}*a_1+a_3\\a_4\end{bmatrix}$$

（2）

$$L_{k1}*L_{m1}=L_{m1}*L_{k1}=\begin{bmatrix}1&&&&&&\\\vdots&\ddots&&&&&\\l_{k1}&\cdots&1&&&&\\\vdots&&&\ddots&&&\\l_{m1}&&&&1&&\\\vdots&&&&&\ddots&\\0&&&&&&1\end{bmatrix},\quad m\neq k$$

（3）

$$L_1*L_1^{-1}=\begin{bmatrix}1&&&\\l_{21}&1&&\\\vdots&&\ddots&\\l_{n1}&&&1\end{bmatrix}*\begin{bmatrix}1&&&\\-l_{21}&1&&\\\vdots&&\ddots&\\-l_{n1}&&&1\end{bmatrix}=I$$

（4）

$$L_1*L_2=\begin{bmatrix}1&&&&\\l_{21}&1&&&\\l_{31}&&1&&\\\vdots&&&\ddots&\\l_{n1}&&&&1\end{bmatrix}*\begin{bmatrix}1&&&&\\0&1&&&\\0&l_{32}&1&&\\0&\vdots&&\ddots&\\0&l_{n2}&&&1\end{bmatrix}=\begin{bmatrix}1&&&&\\l_{21}&1&&&\\l_{31}&l_{32}&1&&\\\vdots&\vdots&&\ddots&\\l_{n1}&l_{n2}&&&1\end{bmatrix}\neq L_2*L_1$$

2 Gauss- Jordan消去和LU分解

引例：（消元方法）

$$\begin{aligned}2x+4y-2z&=2 & 2x+4y-2z&=2\\4x+9y-3z&=8 \quad\longrightarrow & 1y+1z&=4\\-2x-3y+7z&=10 & 4z&=8\end{aligned}$$

例 1 线性代数的方法：

$$A \to \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix} \to \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix} \to \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \to U$$

上述消元的过程保留消元因子即可得到LU分解

高斯消元相当于左乘初等矩阵；例如

$$E_{21} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -l_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}, \quad l_{21} = \frac{a_{21}}{a_{11}}$$

将恒同矩阵写在A的旁边，同时作用Gauss消元 $[A, I]$. 容易知道：对A的处理，完全体现在I身上.

$$\underline{E_{n,n-1} * E_{n,n-2} E_{n-1,n-2} * \ldots * E_{n1} \ldots E_{31} E_{21}} * A = U$$

上式横线部分的逆即是LU分解产生的下三角矩阵L. 根据矩阵求逆的规则和初等矩阵的逆可知 下三角L的元素即是Gauss消元过程中产生的乘子.

1) E_{21} 的逆矩阵(逆变换)

$$E_{21}^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

2) （同列的）初等矩阵乘法可交换 $E_{31} * E_{21} = E_{21} * E_{31} \ldots$

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ l_{31} & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \cdots & 0 \\ l_{31} & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

因此，可以记$E_1 = E_{n1} \ldots E_{21}$.

3) 由矩阵分块规则，第二列以后，如E_{42}, E_{32}， 有类似性质

$$E_{32} * E_{42} = E_{42} * E_{32} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & * & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & * & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & * & 1 & 0 \\ 0 & * & 0 & 1 \end{bmatrix}$$

类似记$E_2 = E_{n2} \ldots E_{32}$.

4) 由1), 2), 3) 分析可知：矩阵的变换过程可以写成

$$E_{n-1}E_{n-2}...E_2E_1A = U$$

其中E_k 以单位矩阵为基础，第k个主元以下为$l_{k+1,k}, l_{k+2,k}\cdots$ 的初等矩阵，其逆矩阵E_k^{-1}将E_k第k 列取负值, 形式如下：

$$\begin{bmatrix} 1 & & & k_{\downarrow} & & \\ & 1 & & & & \\ & & \ddots & & & \\ k\rightarrow & & & 1 & & \\ & & & * & \ddots & \\ & & & * & & 1 \end{bmatrix}$$

5) 计算 $L = (E_{n-1} * E_{n-2} * ... * E_1)^{-1} = E_1^{-1} * ... * E_{n-1}^{-1}$

$$\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{bmatrix}$$

注意（不同列的初等矩阵乘法不可交换）

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 11 & 4 & 1 \end{bmatrix}$$

下面说明如何计算 L

利用初等矩阵(变换)的意义或者分块矩阵乘法规则，可证明（单列初等矩阵满足）：

$$\begin{bmatrix} 1 & 0_{1\times(n-1)} \\ L_1 & I_{n-1} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0_{1\times(n-1)} \\ 0_{(n-1)\times 1} & L_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ L_1 & L_2 \end{bmatrix}$$

以及由分块矩阵乘法

$$\begin{bmatrix} I_\ell & 0_{\ell\times(n-\ell)} \\ 0_{(n-\ell)\times\ell} & A_{(n-\ell)\times(n-\ell)} \end{bmatrix} \cdot \begin{bmatrix} I_\ell & 0_{\ell\times(n-\ell)} \\ 0_{(n-\ell)\times\ell} & B_{(n-\ell)\times(n-\ell)} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & A*B \end{bmatrix}$$

也即

$$\begin{bmatrix} I & 0 & 0 \\ 0 & 1 & 0 \\ 0 & L_1 & I \end{bmatrix} \cdot \begin{bmatrix} I & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & L_2 \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & 1 & 0 \\ 0 & L_1 & L_2 \end{bmatrix}$$

上式说明消元过程中产生的乘子l_{ij}即是L的对应元素，变换过程中依次放在L的相应位置(i,j)即得

$$A = L * U$$

2.1 LU分解的MATLAB矩阵实现

如何尽量Matlab矩阵运算，避免循环

$$\left[\begin{array}{c|ccc} * & * & * & * \\ \hline * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{array}\right] = \begin{bmatrix} a_{11} & \boxed{\text{——}} \\ \boxed{\,|\,} & \boxed{A_{22}} \end{bmatrix}$$

(1) 对n阶矩阵分块

(2) 计算下三角矩阵L的第一列元素

$$L_{j1} = \frac{A_{j1}}{A_{11}} \quad j = 2 \to n$$

(3) 更新A的剩余$(n-1)$阶矩阵

$$A_{new} = A_{22} - L(2:n,1) * A(1,2:n)$$

(4) 矩阵更新为$(n-1)$阶，重复第一步，直到循环结束。

Matlab 测试代码如下

```
A = rand(5); A = A'*A;
n = size(A,1);  U = A; L = eye(n);
% ===
for j = 1:n-1
    j1 = j+1:n;
    L(j1,j) = U(j1,j) / U(j,j);
    U(j1,j) = 0;
    U(j1,j1) = U(j1,j1) - L(j1,j)*U(j,j1);
end
% --- L, U,
err1 = abs(L*U - A); max(err1(:)),
```

Gauss 消去何时可用? 充分条件

定理 1 If A is

- positive definite : $x^{\mathrm{T}}Ax > 0;\ \forall x \in \mathbb{R}^n$
- diagonally dominate: $|a_{ii}| > \sum_{j=1}^{n} |a_{ij}|$

then Gauss elimination is stable.

2.2 Gauss消元的计算复杂度

对稠密矩阵执行Gauss消去所需乘除法的次数：

$$n^2+(n-1)^2+...+1\approx\frac{1}{3}n^3$$

如需多次求解同一个系数矩阵产生的方程组， 应当先进行三角形分解；避免每次重复消元。

$$Ax=b\Leftrightarrow LUx=b\Leftrightarrow\begin{cases}Ly=b\\Ux=y\end{cases}$$

除了第一步LU分解需要一次n^3次计算量，求解三角形方程组只要n^2次 计算量。

Gauss 消去何时好用?

带状矩阵：三对角矩阵，d-diagonal （d-banded）matrix D-对角矩阵的三角形分解，计算量为$\mathbb{O}(d^2\cdot N)$

3 列主元高斯消去

思考： Gauss消去是否对所有可逆矩阵可行？ 例如如下系数矩阵

$$\begin{bmatrix}0&1&2\\1&2&1\\3&1&1\end{bmatrix},\quad\begin{bmatrix}10^{-8}&1\\1&2\end{bmatrix}$$

衍生方法：列主元高斯消去；全主元高斯消去；...

$Ax=b$ 本身解存在唯一，但是消元的过程中，主元$a_{ii}^{(k)}=0$ 消元法无法继续； 或者 $a_{ii}^{(k)}\approx0$ 作为分母， 数值不稳定. 如何解决?

例 3 回忆线性代数课程中的解方程过程

$$\begin{bmatrix}0&2&1\\6&8&1\\4&4&1\end{bmatrix}\xrightarrow[\text{row change}]{(1,2)}\begin{bmatrix}6&8&1\\0&2&1\\4&4&1\end{bmatrix}\to\begin{bmatrix}6&8&1\\\boxed{0}&2&1\\\boxed{2/3}&-4/3&1/3\end{bmatrix}\to\begin{bmatrix}6&8&1\\\boxed{0}&2&1\\\boxed{2/3}&\boxed{-2/3}&1\end{bmatrix}$$

检验$PA=LU$

$$\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix}\begin{bmatrix}0&2&1\\6&8&1\\4&4&1\end{bmatrix}=\begin{bmatrix}1&0&0\\\boxed{0}&1&0\\\boxed{2/3}&\boxed{-2/3}&1\end{bmatrix}\begin{bmatrix}6&8&1\\0&2&1\\0&0&1\end{bmatrix}$$

寻找每一列的最大元然后交换两行，相对于左乘一个初等置换矩阵(permutation, 只交换两行)； 此过程写成矩阵形式如下

$$\underline{E_{n,n-1}P_{n-1}*E_{n,n-2}E_{n-1,n-2}P_{n-2}*...*E_{n1}...E_{31}E_{21}}P_1*A=U$$

类似于不选主元的LU分解，记每一列的初等行变换乘积为

$$E_k=E_{n,k}*E_{n-1,k}*E_{n-2,k}...*E_{k+1,k}$$

列主元Gauss消去可写为:

$$(E_{n-1}P_{n-1})...(E_2P_2)\cdot(E_1P_1)A = U$$

以$n=4$为例， 上式为

$$E_3P_3 * E_2P_2 * E_1P_1 * A = U$$

改写为

$$E_3 * (P_3E_2P_3) * (P_3P_2E_1P_2P_3) * (P_3P_2P_1) * A = U$$

进而

$$F_3F_2F_1PA = L^{-1} * PA = U$$

注意到E_k为正常的行变换矩阵，P_k 针对第k列，交换 $(k+1)$ 到 n之间的某两行, 满足:

$$P = P^T, \& \quad P * P = I$$

现分析 $L = F_3^{-1} * F_2^{-1} * F_1^{-1}$

(1) $F_3 = E_3$, $F_{n-1}^{-1} = E_{n-1}^{-1}$ 形如普通LU, 不参与换行.

(2) 现说明中间部分 F_k（其逆具有同样的形式）

$$F_k = P_{n-1}\cdots P_{k+1}E_kP_{k+1}\cdots P_{n-1}$$

(A)-1 $P_{k+1}E_kP_{k+1}$ 只交换 E_k对应第k列以下的某两行

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

当$k \neq 1$时，按照矩阵分块，将上式记为 A, B, C，有

$$\begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & C \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & ABC \end{bmatrix}$$

(A)-2 F_k 只交换 E_k对应第k列以下的元素，每次交换两行或者不换， 依赖于$P_{k+1}...P_{n-1}$, 即

$$F_k = \begin{bmatrix} 1 & 0 \\ P_{n-1}...P_{k+2}P_{k+1}\cdot\| & I \end{bmatrix}$$

(3) 每个F与E具有同样形式，可类似将乘子存于L的对应位置。且注意对应于第k列产生的E_k，以后$(k+1)$每次换行（P_j, $j>k$）均作用于E_k.

(4) 排列矩阵$P = P_{n-1}...P_2P_1$的计算只需要存储一个数组即可。

$$p = (1, \underline{2,...j,...,k...,n}), \to\ p_1 = (1, \underline{2,...,k,...,j,...,n}) \to ... \to p_{n-1}$$

P可通过交换单位矩阵的行来实现

$$P = I(p_{n-1},:)$$

(5) 若顺带计算A的行列式，可记录符号: sign = 1;

```
if  rows exchange
    sign = -sign
endif
```

定理 2 如果A为非奇异矩阵，则存在置换矩阵P, 使得

$$PA = LU$$

Matlab Code: lutx.m & mylulect3.m

```
A = rand(5);
n = size(A,1); p = (1:n)'; sign = 1;  U = A;
% ===
for j = 1 : n
% --- exchange rows
    [~,Ind] = max(abs(U(j:n,j)));
    Ind = Ind + j - 1;
    if U(j,j) / U(Ind,j) < 1
    p([j, Ind]) = p([Ind, j]);  sign = - sign;
    U([j, Ind],:) = U([Ind,j],:);
    end
% --- eliment
    j1 = j+1:n;
    mj = U(j1,j)./U(j,j); U(j1,j) = mj;
    U(j1,j1) = U(j1,j1) - mj*U(j,j1);
end
% === output  perm mat  % L & U
P = eye(n); P(1:n,:) = P(p,:),
L = tril(U,-1) + eye(n);  U = triu(U);
det(A) - sign*prod(diag(U)),
err = abs(P*A - L*U); max(err(:))
```

4 对称正定矩阵的三角分解

设B是可逆矩阵（或列满秩），则$A = B^T B$对称正定.

Hint: 对称性显然满足。任给$x \in R^n$,

$$x^T A x = x^T B^T B x = (Bx)^T (Bx) \geq 0$$

且等号成立时，$Bx=0 \to x=0$. 实际上，只要B的列向量线性无关，上述结果成立。

反过来，给定对称正定矩阵A

Q-(1) 是否存在合适的B(例如三角形矩阵)使之满足$A=B^T*B$？

Q-(2) 如果存在，如何找到？ Cholesky分解

思考： 一般地，没有规律的稠密矩阵可以进行LU分解，现在矩阵有对称性, 存储量可以减半；计算量是否也应当减半?

提取上三角矩阵的对角元素组成对角阵，则U_0是单位上三角矩阵；由对称性可得

$$A=LU=L*D*U_0=U_0^{\mathrm{T}}*(D*L^{\mathrm{T}})=L_1U_1$$

下面说明：L 与$L_1=U_0^{\mathrm{T}}$相等！ 由于 L, L_1都是单位下三角矩阵；由分解的唯一性可知

$$LU=L_1U_1 \Leftrightarrow L^{-1}LUU_1^{-1}=L^{-1}L_1U_1U_1^{-1} \Leftrightarrow UU_1^{-1}=L^{-1}L_1=I \Rightarrow L=U_0^{\mathrm{T}}$$

其中用到下三角乘以下三角仍是下三角，下三角的逆仍是下三角。 进而

$$A=LDL^{\mathrm{T}}=(L\sqrt{D})*(\sqrt{D}L^{\mathrm{T}})=\tilde{L}*\tilde{L}^{\mathrm{T}}$$

例 4 证明：若A对称正定，则对角线元素D均大于0.

提示：(反证法)假设D某个对角元素≤ 0, 可以证明存在向量x 满足$x^{\top}Ax\leq 0$

4.1 Cholesky 分解

将$A=L*L^{\mathrm{T}}$ 写成分块形式

$$\begin{pmatrix} a_{11} & A_{21}^{\mathrm{T}} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix}\begin{pmatrix} l_{11} & L_{21}^{\mathrm{T}} \\ 0 & L_{22}^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} l_{11}^2 & l_{11}L_{21}^{\mathrm{T}} \\ l_{11}L_{21} & L_{21}L_{21}^{\mathrm{T}}+L_{22}L_{22}^{\mathrm{T}} \end{pmatrix}$$

matlab code: mycholect.m

```
 %  === chol  A  = L*L'
A = [25 15 -5; 15 18 0; -5  0  11];
% A = rand(5); A = A'*A;
n1 = size(A,1); L = tril(A);
for j = 1:n1
     j1 = j+1:n1;
     L(j,j) = sqrt(L(j,j));
     mj = L(j1,j)./L(j,j);
     L(j1,j) = mj;
     L(j1,j1) = L(j1,j1) - mj*mj';
end
L = tril(L); e1 = abs(A - L*L'); max(e1(:)),
```

Algorithm 1 Cholesky 分解

Input: $A \in \boldsymbol{R}^{n\times n}$

Output: 下三角 L, 满足$A = L * L'$

1: 计算对角主元 l_{11} 和第一列L_{21}: (一次开方，n-1次乘法)

$$l_{11} = \sqrt{a_{11}}, \quad L_{21} = \frac{1}{l_{11}} A_{21}$$

2: compute L_{22} from: ($(n-1)^2$次乘法,对称矩阵减半 $(n-1)^2/2$)

$$A_{22} - L_{21} * L_{21}^{\mathrm{T}} = L_{22} L_{22}^{\mathrm{T}} = A_{new}$$

3: n阶矩阵变成了(n-1)阶矩阵

$$\tilde{A} = \tilde{L} * \tilde{L}^{\mathrm{T}}$$

4: 重复1，2，过程. 每次作用$\frac{n^2}{2}$ 次乘法，共计$n^3/6$

例 5 利用Cholesky分解，计算对称矩阵的三角形分解.

$$\begin{pmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{pmatrix} = \begin{pmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ l_{31} & l_{32} & l_{33} \end{pmatrix} * \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ & l_{22} & l_{23} \\ & & l_{33} \end{pmatrix}$$

- $k = 1$ 时， 计算 L 的第一列; 并生成下一步待分解$(n-1)$ 阶矩阵

$$\begin{pmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{pmatrix} = \begin{pmatrix} 5 & & \\ 3 & l_{22} & \\ -1 & l_{32} & l_{33} \end{pmatrix} * \begin{pmatrix} 5 & 3 & -1 \\ & l_{22} & l_{23} \\ & & l_{33} \end{pmatrix}$$

$$\begin{pmatrix} 18 & 0 \\ 0 & 11 \end{pmatrix} - \begin{pmatrix} 3 \\ -1 \end{pmatrix} * \begin{pmatrix} 3 & -1 \end{pmatrix} = \begin{pmatrix} 9 & 3 \\ 3 & 10 \end{pmatrix}$$

- 针对$(n-1)$阶矩阵，重复上一步骤，得到L 的第二列

$$\begin{pmatrix} 9 & 3 \\ 3 & 10 \end{pmatrix} = \begin{pmatrix} l_{22} & \\ l_{32} & l_{33} \end{pmatrix} * \begin{pmatrix} l_{22} & l_{23} \\ & l_{33} \end{pmatrix} = \begin{pmatrix} 3 & 0 \\ 1 & l_{33} \end{pmatrix} * \begin{pmatrix} 3 & 1 \\ 0 & l_{33} \end{pmatrix}$$

  以及$(n-2)$矩阵 $10 - 1 = 9$

- $l_{33}^2 = 9 \to l_{33} = 3$; 计算完毕。

标注 1 以上代码没有充分利用矩阵的对称性，其中第十行$mj * mj'$ 计算量应该减半。但是如果利用循环改写这句代码，效率反而变慢。因为Matlab计算矩阵更高效。下节给出Cholesky分解计算效率更高的推导和代码写法。

4.2 Cholesky分解 II

按照矩阵乘法$A = L * L^T$, 即

$$A_{ij} = \sum_{k=1}^{n} L_{ik} * L_{jk} = \sum_{k=1}^{j} L_{ik} * L_{jk} = \sum_{k=1}^{j-1} L_{ik} * L_{jk} + L_{ij} * L_{jj}$$

按列计算可得

$$\begin{cases} A_{jj} = \sum_{k=1}^{j-1} L_{jk} * L_{jk} + L_{jj} * L_{jj} & \to L_{jj} = \sqrt{A_{jj} - \sum_{k=1}^{j-1} L_{jk}^2} \\ A_{ij} = \sum_{k=1}^{j-1} L_{ik} * L_{jk} + L_{ij} * L_{jj} & \to L_{ij} = \left(A_{ij} - \sum_{k=1}^{j-1} L_{ik} * L_{jk}\right) / L_{jj} \end{cases}$$

上述过程也可按照分块矩阵乘法描述, 记

$$j_0 = 1 : j-1, \; j_1 = j+1 : n$$

为两个指标向量, 则 $A = L * L'$形式如下

$$\begin{bmatrix} A(j_0, j_0) & * & * \\ A(j, j_0) & A(j,j) & * \\ A(j_1, j_0) & A(j_1, j) & A(j_1, j_1) \end{bmatrix}$$

$$= \begin{bmatrix} L(j_0, j_0) & & \\ L(j, j_0) & L(j,j) & \\ L(j_1, j_0) & L(j_1, j) & L(j_1, j_1) \end{bmatrix} \cdot \begin{bmatrix} L(j_0, j_0)' & L(j, j_0)' & L(j_1, j_0)' \\ & L(j,j) & L(j_1, j)' \\ & & L(j_1, j_1)' \end{bmatrix}$$

进而

$$A(j,j) = L(j, j_0) * L(j, j_0)' + L(j,j) * L(j,j)$$

$$A(j1, j) = L(j_1, j_0) * L(j, j_0)' + L(j_1, j) * L(j,j)$$

```
1  % === chol A = L*L', % A = rand(10); A = A'*A;
2  A = [25 15 -5; 15 18 0; -5  0  11];
3  n1 = size(A,1); L = eye(n1);
4  % ---
5  for j = 1:n1
6       j0 = 1:j-1;  j1 = j+1:n1;
7       L(j,j) = sqrt(A(j,j) -  L(j,j0)*L(j,j0)');
8       L(j1,j) = (A(j1,j) - L(j1,j0)*L(j,j0)')/L(j,j);
9  end
10 e2 = abs(A - L*L'); max(e2(:))
```

matlab code: mycholect2.m 代码中计算量最大发生在第8行, $\boxed{L(j1, j0) * L(j, j0)'}$ 对固定的j, 需要$(j-1)*(n-j+1)$ 次乘法, 共计 $\sum_{j=1}^{n} j(n-j) \approx n^3/6$

4.3 改进的平方根方法

当矩阵为负定或者不定矩阵时, Cholesky分解不可直接使用。而改进的平方根法可用, 且与LU分解比较计算量减半。

$$A = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \vdots & \ddots & \\ l_{n1} & l_{n2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} 1 & l_{21} & \cdots & l_{n1} \\ & 1 & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}$$

依据矩阵乘法规则 $A=(LD)*L^{\mathrm{T}}$ 即

$$a_{ij}=\sum_{k=1}^{n}(LD)_{ik}(L^{\mathrm{T}})_{kj}=\sum_{k=1}^{n}\left(\sum_{l=1}^{n}l_{il}d_{lk}\right)l_{jk}\xrightarrow{\text{D is diag}}\sum_{k=1}^{n}\left(l_{ik}d_k\right)l_{jk}$$

由于$l_{jk}=0$, for $j<k$

$$a_{ij}=\sum_{k=1}^{j}\left(l_{ik}d_k\right)l_{jk}=\sum_{k=1}^{j-1}\left(l_{ik}d_k\right)l_{jk}+l_{ij}d_j\underline{l_{jj}=1}$$

即

$$a_{ij}=\sum_{k=1}^{j-1}l_{ik}*\left(d_k l_{jk}\right)+l_{ij}d_j$$

对 i = 1,2,...,n; 注意 L为单位下三角，对角线元素为1，$l_{ij}=0,\ i<j$

$$a_{jj}=\sum_{k=1}^{j-1}l_{jk}*\left(d_k l_{jk}\right)+d_j\ \rightarrow\ d_j=a_{jj}-\sum_{k=1}^{j-1}l_{jk}*\left(d_k l_{jk}\right)$$

$$l_{ij}=\left[a_{ij}-\sum_{k=1}^{j-1}l_{ik}*\left(d_k l_{jk}\right)\right]/d_j,\qquad i>j$$

编程时可采用下三角矩阵存储， 针对 $j=1\rightarrow n-1$ 按列计算。记$j_0=1:j-1,\ j_1=j+1:n$

A-(1) 计算(j-1)维向量 $DL=D(j_0).*L(j,j_0)$ 待用

A-(2) 令 $i=j$, 计算 $d_j=a_{jj}-L(j,j_0)*DL'$

A-(3) 计算$L(j_1,j)=[l_{i+1,j},...,l_{n,j}]^T=A(j1,j)-L(j_1,j_0)*DL'$

- i = j = 1 时， $d_1=a_{11}$, $i\geq 2$时,

  $$l_{i1}=a_{i1}/d_1,\quad i\geq 2$$

- $j\geq 2$时， 当 $k<j$， d_k, l_{ik}, l_{jk} 已知, 则

  $$d_j=a_{jj}-\sum_{k=1}^{j-1}l_{jk}*\left(d_k l_{jk}\right)$$

  当 i>j时, $l_{i,1},...l_{i,j-1}$已知， 则

  $$l_{ij}=\left[a_{ij}-\sum_{k=1}^{j-1}l_{ik}*\left(d_k l_{jk}\right)\right]\Big/d_j,\quad i>j$$

```
A = [25, 15, -5; 15, 18, 0; -5,  0 , 11];
n1 = size(A,1); L = eye(n1); Dd = zeros(n1,1);
% ---
for j = 1:n1
    j0 = 1:j-1; j1 = j+1:n1;
    DL = L(j,j0)'.*Dd(j0); % j,size(DL,1), % col vec
```

```
    Dd(j) = A(j,j) -  L(j,j0)*DL;
    L(j1,j) = (A(j1,j) - L(j1,j0)*DL)/Dd(j);
end
D = diag(Dd); e2 = abs(A - L*D*L'); max(e2(:))
```

Tips: 圆括号部分作为一个整体可避免重复计算，运算量为$\frac{1}{6}n^3$. Cholesky 分解同样适用于稀疏矩阵，且一般情况下 工作量大大降低。 matlab code: mycholect3.m 对有些问题 通过行交换改变矩阵的pattern可以进一步的降低工作量。 矩阵分解内涵相当丰富，可自行阅读《数值代数》相关资料.

5 追赶法(Thomas Algorithm)

矩阵的三角分解工作量是$\mathbb{O}(n^3)$量级的乘除法，对称矩阵工作量可以减半； 即使如此，对大型的稀疏矩阵，人们更倾向于后文的迭代法。 但是有一类特殊的、 且在实际问题中经常出现的线性方程组非常适合于直接法(Gauss消去或者三角分解)

三对角或者d对角线性方程组

$$\begin{pmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}$$

此时对上述方程$Ax = f$进行LU分解，得到如下形式

$$A = \begin{pmatrix} 1 & & & & \\ \alpha_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \alpha_{n-2} & 1 & \\ & & & \alpha_{n-1} & 1 \end{pmatrix} * \begin{pmatrix} \beta_1 & c_1 & & & \\ & \beta_2 & c_2 & & \\ & & \ddots & & \\ & & & \beta_{n-1} & c_{n-1} \\ & & & & \beta_n \end{pmatrix}$$

Algorithm 2 追赶法Thomas algorithm solve [0 a b c 0]*x = d;

Input: input 矩阵A的三条对角线向量 a, b, c 以及右端项 d

Output: 向量 x

初始数据 N = length(d); alp = a; bet = b; gam = c; x = d;
for k = 1:N-1 **do**
    μ = a(k)/bet(k); alp(k) = μ;
    bet(k+1) = bet(k+1) - μ*c(k);
return alp, bet

可以验算上述三角分解过程需要约2n次乘除法， 一旦分解完成，可按照如下方式计算三对角

方程组

$$Ax = f \Leftrightarrow LUx = f \Leftrightarrow \begin{cases} Ly = f \\ Ux = y \end{cases}$$

求解上述方程只要约 3n次乘除法. 若要反复求解上述方程，也应当先进行分解. matlab code: tridisolve.m& tridilu.m 三对角或者D-对角线性方程组不可执行行交换(row exchage)；如果破坏这种紧凑性， 运算量会从最低的 $\mathbf{O}(n)$ 急剧上升到 $\mathbf{O}(n^3)$.

```
n1 = 21;
a = ones(n1-1,1); b = -2*ones(n1,1); c = a;
A = diag(b) + diag(a,-1) + diag(c,1); figure; spy(A)
r = 0*b+1;     % -- test problem
% [a/b/c] --> L = [alp // 1], U = [bet // c]
for j = 1:n1-1
    a(j) = a(j) / b(j);
    b(j+1) = b(j+1) - a(j)*c(j);
end
% --- solve eqn, Ax = r by L*Ux =r
y = r;
for j = 2:n1 % --- Ly = r
    y(j) = y(j) - a(j-1)*y(j-1);
end
% ---- Ux = y
x1 = y; x1(n1) = y(n1) / b(n1);
for j = n1-1:-1:1
    x1(j) = (y(j) - x1(j+1)*c(j))/b(j);
end
e1 = norm(A*x1 - r, inf)
figure; plot(x1,'ro-','linewidth', 1 );
```

若不在乎$2n$次的重复计算，或者只需要计算一次三对角线性方程组，以下代码也是高效的.

```
 function U = Thomas1(a,b,c,r)
%    A SIMPLE CODE for Tridiag sysm : A*U = r;
%    a = diag(A); b = diag(A,-1); c = diag(A,1); r = rhs
%    Tips: sparse + "\" is more efficient than "this code" and "sine trans"
    % Elimination
    n = length(r);
    for ii = 2 : n
    a(ii) = a(ii) - b(ii-1)/a(ii-1)*c(ii-1);
    r(ii) = r(ii) - b(ii-1)/a(ii-1)*r(ii-1);
    end
```

```
    % backward Substitute
    U(n) = r(n) / a(n);
    for ii = n-1 : -1 : 1
    U(ii) = (r(ii) - c(ii)*U(ii+1))/a(ii);
    end
```

6 Gram-Schmidt正交化和QR分解

第三章针对最小二乘问题的计算，我们给出了超定方程系数矩阵的 QR分解，也称为不完全QR分解。 本节对于可逆矩阵，同样的方法可得到其完全QR分解。 后文特征值计算也会涉及QR分解，而QR分解本身也可以用来求解线性方程组。

例 6 给定非奇异矩阵

$$A = \begin{bmatrix} | & | & \cdots & | \\ a_1 & a_2 & \cdots & a_n \\ | & | & \cdots & | \end{bmatrix}$$

如何根据A 的列向量得到空间

$$\mathrm{span}(A) = \mathbb{R}^n = span\{a_1, a_2, ..., a_n\}$$

的一组标准正交基?

Gram-Schmidt正交化方法本质上是将A分解为

$$A = QR, \quad \& \quad Q^T Q = I, \quad R(i,j) = 0, \text{ for } i > j$$

```
A = rand(8);     % A = hilb(8)
n1 = size(A,1); R = zeros(n1); Q = A;
for j = 1:n1
    v = A(:,j);
    for k = 1:j-1
        R(k,j) = Q(:,k)'*v;
        v = v - R(k,j)*Q(:,k);
    end
    R(j,j) = norm(v);
    Q(:,j) = v / R(j,j);
end
A - Q*R,    Q'*Q - eye(n1)   %  check error
% this code is not correct to illconditoin matrix like: hilbert
```

一旦矩阵A的QR分解完成， 求解$Ax=b$ 可按照以下两步进行

$$\begin{cases} y = Q^T b \\ Rx = y \end{cases}$$

读者可检验计算复杂度并与LU分解比较； 实际利用Matlab计算时，矩阵乘以向量可能比解三角形方程组更快。

7 Krylov子空间 0

直接法对 于三对角矩阵和三角形矩阵表现良好， Hessenberg矩阵是这两者结合而成，LU分解效果如何？ 分析H矩阵进行LU分解需要的乘除法次数以及分解后的矩阵形式

$$H = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ + & 1 & & & \\ 0 & + & 1 & & \\ 0 & 0 & + & 1 & \\ 0 & 0 & 0 & + & 1 \end{bmatrix} \cdot \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}$$

假定计算 $Ax=b$ 时，（计算机）有以下限制

- 不可执行矩阵分解（lu, QR都不行）
- 可以算乘法：$\forall v \in \mathbb{R}^n$, 可以得到$Av$
- 可以求解系数矩阵为Hessenberg矩阵的线性方程组

在以上限制条件下求解$Ax=b$？

记Krylov矩阵

$$\mathbb{K}_n = \left[b, Ab, A^2 b, \ldots, A^{n-1} b\right]_{n\times n}$$

为推导过程简单，我们假设$\mathbb{K}_n$是满秩的。

现考虑对$\mathbb{K}$进行QR分解，得到

$$\mathbb{K}_n = Q\cdot R = Q\cdot \begin{bmatrix} r_{11} & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix} = Q * \left[r_{11} * e_1, \left|\boxed{R_{(:,2:n)}}\right|\right]$$

进而

$$A * \mathbb{K}_n = \left[\mathbb{K}_{(:,2:n)}, \left| A^n b \right.\right] = \left[Q * R_{(:,2:n)}, \left| A^n b\right.\right] = Q * \left[R_{(:,2:n)}, \left| Q^{\mathrm{T}} A^n b\right.\right]$$

也即

$$A*Q*R=Q*\left[Hessenberg\right] \Longrightarrow A*Q=Q*\left[Hessenberg\right]*R^{-1}=Q*H$$

上式中H是Hessenberg矩阵。 由$H=Q^{\mathrm{T}}AQ$, 即

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & h_{32} & h_{33} \end{bmatrix} = \begin{bmatrix} q_1^{\mathrm{T}} \\ q_2^{\mathrm{T}} \\ q_3^{\mathrm{T}} \end{bmatrix} * \left[Aq_1,\ Aq_2,\ Aq_2\right] = \begin{bmatrix} q_1^{\mathrm{T}}Aq_1 & q_1^{\mathrm{T}}Aq_2 & q_1^{\mathrm{T}}Aq_3 \\ q_2^{\mathrm{T}}Aq_1 & q_2^{\mathrm{T}}Aq_2 & q_2^{\mathrm{T}}Aq_3 \\ q_3^{\mathrm{T}}Aq_1 & q_3^{\mathrm{T}}Aq_2 & q_3^{\mathrm{T}}Aq_3 \end{bmatrix}$$

由正交性$q_3^{\mathrm{T}}(Aq_1)$为零。 Arnoldi算法代码如下。

```
function [Q,H] = myArnoldi0(A,v)
% % basic Arnoldi by Gram-Schmidt method generate A,Q,H
% Full Orthogonalization Method: A*Q = Q*H
% for example: n = 7; A = hilb(n); v = ones(n,1);
%      [Q,H] = myArnoldi0(A,v);  norm(A*Q - Q*H), norm(Q'*Q-eye(n))
m = size(A,1);
Q = zeros(m,m);     H = zeros(m,m);
v = v./norm(v);     Q(:,1) = v;
for k = 1:m
    w = A*Q(:,k);
    H(1:k,k) = Q(:,1:k)'*w;     % projection coefs
    w = w - Q(:,1:k)*H(1:k,k); % w orth to Q(:,1:k)
    H(k+1,k) = norm(w,2);
    Q(:,k+1) = w / H(k+1,k);
end
Q = Q(:,1:m); H = H(1:m,1:m);
```

两个事实

1. 给定A, v, 依次给出$A^j v$并执行Gram-Schmidt正交化得到正交矩阵Q 是容易的, 但不实用
2. Arnoldi算法可在给出Q的同时给出Hessenberg矩阵.

借助于 Arnoldi 算法得到

$$A*Q=Q*H \to \quad A=QHQ^{\mathrm{T}}$$

计算$Ax=b$ 转化为

$$QH(Q^{\mathrm{T}}x)=b \to \begin{cases} Hy=Q^{\mathrm{T}}b \\ x=Qy \end{cases}$$

标注 2 上述过程更厉害的用处在于: A真的可以被封装甚至A根本不是矩阵, 同样可以产生H, Q. 这对非线性方程组的Inexact Newton 法是非常重要的。 另外, 我们推导过程选择了$\mathbb{K}$是列满秩

的， 实际上，对大型稀疏矩阵且$m \gg n$时，$\mathbb{K}_n(m \gg n)$即是广泛应用的Krylov子空间。 进一步讨论放在下一章。

标注 3 上述 测试代码采用的是基本的Gram-Schmidt 正交化方法，这对于条件数小的矩阵没有问题；但是当矩阵是病态时 该代码极度不稳定，因此实用的代码是modified Gram-Schmidt正交化方法， 以及两个更优的方法：重复正交化(reOrthogonal) 和 Householder变换. 下一章将给出modified Gram-Schmidt正交化的算法代码. 另外，当时A是对称矩阵时，$H = Q^{\top}AQ$ 是Hessenberg矩阵同时也是对称矩阵， 则H是三对角矩阵。此时，Arnoldi算法只要在前两个向量上投影即可，计算量大大降低， 称为Lanczos算法。

上机练习

1. 对如下5对角n阶方阵进行LU分解，并计算该过程需要乘除法的次数.

$$A = \begin{bmatrix} 35 & -16 & 1 & 0 & 0 & \dots & 0 \\ -16 & 35 & -16 & 1 & 0 & \dots & 0 \\ 1 & -16 & 35 & -16 & 1 & \dots & 0 \\ & & \ddots & & \ddots & & \\ 0 & \dots & 1 & -16 & 35 & -16 & 1 \\ 0 & \dots & 0 & 1 & -16 & 35 & -16 \\ 0 & \dots & 0 & 0 & 1 & -16 & 35 \end{bmatrix}_{n\times n}$$

进一步考虑，由于矩阵具有对称性，上述分解过程的工作量能否减半?

2. 利用追赶法计算三对角$Ax = b, \quad n = 15$；并画出向量x的图像

$$\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & & \ddots & \\ & & -1 & 2 \end{bmatrix}_{n\times n} * \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{n\times 1}$$

3. 对矩阵A进行LU分解 以及QR分解； 并对$A^{\mathrm{T}}A$进行Cholesky分解

$$\begin{bmatrix} 8 & -3 & 2 \\ 4 & 11 & -1 \\ 6 & 3 & 12 \end{bmatrix}$$