数值线性代数———
矩阵计算

潘建瑜

●直接法: LU 分解, Cholesky 分解, ...
●迭代法: 古典迭代法, Krylov 子空间迭代法
本章介绍直接法, 即Gauss 消去法 或PLU 分解
直接法优点: 稳定可靠 $\rightarrow$ 在工程界很受欢迎
直接法缺点: 运算量大 $O(n^3)$ $\rightarrow$ 不适合大规模稀疏线性方程组 (针对特殊结构矩阵的快速方法除外)

1.1 LU 分解

考虑线性方程组

$$Ax = b \tag{1}$$

其中 $A\mathbb{R}^{nn}$ 非奇异, $b\mathbb{R}^{n}$ 为给定的右端项.
Gauss 消去法本质上就是对系数矩阵 A 进行 LU 分解:

$$A = LU \tag{2}$$

其中 L 是单位下三角矩阵, U 为非奇异上三角矩阵.
分解 () 就称为 LU 分解

$$Ax = b \Longleftrightarrow \left\{ \begin{array}{l} Ly = b \\ Ux = y \end{array} \right| \Longrightarrow \text{只需求解两个三角方程组} \tag{3}$$

算法 1.1:	Gauss 消去法
1:	将 A 进行 LU 分解:
2:	$A = LU$, 其中 L 为单位下三角矩阵, U 为非奇异上三角矩阵;
3:	向前回代: 求解 $Ly = b$, 即得 $y = L^{-1}b$
4:	向后回代: 求解 $Ux = y$, 即得 $x = U^{-1}y = (LU)^{-1}b = A^{-1}b$.

需要指出的是: A 非奇异, 则解存在唯一, 但并不一定存在 LU 分解!

定理

(LU 分解的存在性和唯一性) 设 $A\mathbb{R}^{nn}$. 则存在唯一的单位下三角矩阵 L 和非奇异上三角矩阵 U, 使得 $A = LU$ 的充要条件是 A 的所有顺序主子矩阵 $A_k = A(1:k, 1:k)$ 都非奇异, $k = 1, 2, ..., n$.

给定一个矩阵

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n\times n}$$

- 第一步: 假定 $a_{11} \neq 0$, 构造矩阵

$$L_1 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \\ l_{n1} & 0 & 0 & \cdots & 1 \end{bmatrix}, \text{其中 } l_{i1} = \frac{a_{i1}}{a_{11}}, i = 2, 3, \ldots, n$$

易知 L_1 的逆为

$$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -l_{21} & 1 & 0 & \cdots & 0 \\ -l_{31} & 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & & \\ -l_{n1} & 0 & 0 & \cdots & 1 \end{bmatrix}$$

用 L_1^1 左乘 A, 并将所得到的矩阵记为 A(1) , 则

$$A^{(1)} = L_1^{-1}A \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{bmatrix}$$

即左乘 L_1^1 后, A 的第一列中除第一个元素外其它都变为 0.

- 第二步: 将上面的操作作用在 $A^{(1)}$ 的子矩阵 $A^{(1)}(2:n,2:n)$ 上, 将其第一列除第一个元素外都变为 0 : 假定 $a_{22}^{(1)} \neq 0$, 构造矩阵

$$L_2 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & l_{n2} & 0 & \cdots & 1 \end{bmatrix}, \text{其中 } l_{i2} = \frac{a_{i2}^{(1)}}{a_{22}^{(1)}}, i = 3, 4, \dots, n$$

用 L_2^{-1} 左乘 $A^{(1)}$, 并将所得到的矩阵记为 $A^{(2)}$, 则

$$A^{(2)} = L_2^{-1}A = L_2^{-1}L_1^{-1}A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & a_{n3}^{(2)} & \dots & a_{nn}^{(2)} \end{bmatrix}$$

- 依此类推, 假定 $a_{kk}^{(k-1)} \neq 0 (k = 3, 4, \dots, n-1)$, 则我们可以构造一系列的矩阵 $L_3, L_4, ..., L_{n1}$, 使得

$$L_{n-1}^{-1} \cdots L_2^{-1} L_1^{-1} A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & \cdots & a_{nn}^{(n-1)} \end{bmatrix} \triangleq U \to \text{上三角}$$

于是可得 $A = LU$ 其中

$$L = L_1 L_2 \cdots L_{n-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{bmatrix}$$

1.2 LU 分解的实现—矩阵初等变换

算法 1.2: LU 分解

```
for k = 1 to n - 1 do
    for i = k + 1 to n do
        l_ik = a_ik/a_kk % 计算 L 的第 k 列
    end for
    for j = k to n do
        u_kj = a_kj % 计算 U 的第 k 行
    end for
    end for i = k + 1 to n do
        for j = k + 1 to n do
            a_ij = a_ij l_ik u_kj % 更新 A(k + 1 : n, k + 1 : n)
        end for
    end for
end for
```

Gauss 消去法的运算量

由算法 1.2 可知, LU 分解的运算量 (加减乘除) 为

$$\sum_{i=1}^{n-1}\left(\sum_{j=i+1}^{n} 1+\sum_{j=i+1}^{n}\sum_{k=i+1}^{n} 2\right)=\sum_{i=1}^{n-1}\left(n-i+2(n-i)^2\right)=\frac{2}{3}n^3+O\left(n^2\right)$$

加上回代过程的运算量 $O\left(n^2\right)$, 总运算量为 $\frac{2}{3}n^3+O\left(n^2\right)$

Gauss 消去法的运算量

† 评价算法的一个主要指标是执行时间, 但这依赖于计算机硬件和编程技巧等, 因此直接给出算法执行时间是不太现实的. 所以我们通常是统计算法中算术运算 (加减乘除) 的次数.

† 在数值算法中, 大多仅仅涉及加减乘除和开方运算. 一般地, 加减运算次数与乘法运算次数具有相同的量级, 而除法运算和开方运算次数具有更低的量级.

† 为了尽可能地减少运算量, 在实际计算中, 数, 向量和矩阵做乘法运算时的先后执行次序为: 先计算数与向量的乘法, 然后计算矩阵与向量的乘法, 最后才计算矩阵与矩阵的乘法.

矩阵 L 和 U 的存储

当 A 的第 i 列被用于计算 L 的第 i 列后, 在后面的计算中不再被使用.
同样地, A 的第 i 行被用于计算 U 的第 i 行后, 在后面计算中也不再使用
为了节省存储空间, 在计算过程中将 L 的第 i 列存放在 A 的第 i 列, 将 U 的第 i 行存放在 A 的第 i 行, 这样就不需要另外分配空间存储 L 和 U.
计算结束后, A 的上三角部分为 U, 其绝对下三角部分为 L 的绝对下三角部分 (L 的对角线全部为 1, 不需要存储).

矩阵 L 和 U 的存储

算法 1.3: LU 分解

```
for k = 1 to n - 1 do
    for i = k + 1 to n do
        a_ik = a_ik / a_kk
        for j = k + 1 to n do
            a_ij = a_ij a_ik a_kj
        end for
    end for
end for
```

† 根据指标的循环次序, 算法 4.3 也称为 KIJ 型 LU 分解. 实际计算中一般不建议使用: 对指标 k 的每次循环, 都需要更新 A 的第 $k+1$ 至第 n 行, 这种反复读取数据的做法会使得计算效率大大降低. 对于按行存储的数据结构, 一般采用后面介绍的 IKJ 型 LU 分解.