

**计算方法上机报告**

第三次上机报告

06220143顾豪阳



2022-5-15

东南大学

电子科学与工程学院

# 一、题目简介

**P130数值实验4.2**

（1）请编写生成矩阵LU分解的程序，并把L，U都存储在A中；

（2）结合前面的向前向后代入程序，用你的程序求解线性方程组。

# 二、理论分析

借助基本消去矩阵，高斯消去法可以按照如下方式描述：

A通过与一系列基本消去矩阵相乘，得到便于计算的上三角矩阵U，在通过逆矩阵的性质可以得到矩阵A可以做如下分解：

这就是矩阵A的LU分解。

# 三、数值实验过程

用高斯消去法求解

并写出对应矩阵LU分解的L和U；

**解：**在消元过程中，A共需要消掉n-1个主元下面所有的元素，注意，第n个主元已经是矩阵的最后一个元素了，它的下面和右边都没有其他元素了，所以不存在说对第n个主元下面所有元素消去的情况。

这就获得了我们代码的第一个for循环，从第1行主元开始消元，一直到第n-1行主元。而在获得每一行主元过程中，需要对该行主元下面所有元素都消去，假如现在要获得第i行主元的话，就是说要对该主元所在列的第i+1行到第n行元素都消掉，那么这就获得了我们代码的第二个for循环，从消去第i+1个元素开始一直到第n个元素。前文说过，消掉第（j，i）个位置元素过程中，主元所乘系数就是L矩阵第（j，i）位置的元素，所以有L(j,i)=A(j,i)/A(i,i);。然后的话，就是把A矩阵第j行减去第j行乘以L（j，i），这样就可以消掉第（j，i）个元素了，就是这行代码A(j,:)=A(j,:)-(A(j,i)/A(i,i))\*A(i,:)。

最后，执行完两层for循环后，A矩阵就成为了U矩阵，L矩阵也从最初的单位阵成了L矩阵。

# 四、程序代码与结果

使用MATLAB编写计算程序，将运算结果与理论值进行对比：

定义LU分解的函数：

function [L,U,LU]=LUDecomposition(A)

[n,n]=size(A)

L=eye(n);

for i=1:n-1

for j=i+1:n

L(j,i)=A(j,i)/A(i,i);

% disp(L);

A(j,:)=A(j,:)-(A(j,i)/A(i,i))\*A(i,:);

end

end

U=A;

LU=L+U-eye(n)

end

在主函数中代入数值进行运算：

A=[2 -1 3;4 2 5;1 2 0];

[L,U,B]=LUDecomposition(A);

disp(L);

disp(U);

disp(B);

[n,n]=size(A);

d=[1,4,7];

y=1:3;

x=1:3;

y(1) = d(1)/L(1,1);

for i = 2 :n

for j = 1 :i - 1

d(i) = d(i) - L(i,j) \* y(j);

end

y(i) = d(i);

end

x(n) = y(1,n) / U(n,n);

for i = (n - 1 ): - 1 : 1

for j = n: - 1 :i + 1

y(i) = y(i) - U(i,j) \* x(j);

end

x(i) = y(i) / U(i,i);

end

disp(x);

运算结果为

L=

1.0000 0 0

2.0000 1.0000 0

0.5000 0.6250 1.0000

U=

2.0000 -1.0000 3.0000

0 4.0000 -1.0000

0 0 -0.8750

合并后B=

2.0000 -1.0000 3.0000

2.0000 4.0000 -1.0000

0.5000 0.6250 -0.8750

求解得x=

9 -1 -6

和理论值计算相吻合。

# 五、对实验的分析

LU分解分两步走：生成消去因子、更新矩阵数据，在求解方程组的过程中，可以看到，工作量还是主要来自于LU分解，它的量级在n的三次方，而在通过LU求解方程时使用的向前向后代入的运算量级在n的二次方，所以，随着问题规模的增大，工作量主要取决于LU分解。

而对于这个算法本身而言，当A中出现0时，使用LU分解会造成很大的误差，而此时的解决方法也比较简单，只需要从该0主元下面所有元素中找到一个非0元素，然后将其所在的行与该0主元所在的行进行交换就行了（当然这里的0是一种特殊情况，如果这个数值很小效果也一样），也就是说，对角线上的元素应该选取最大的值，即需要一个矩阵P来变换矩阵的次序，使得对角线的绝对值为最大值，这就是PLU分解。

代码如下：

function AdvanceLUDecomposition(A,n)

D=A;

L=zeros(n);

P=eye(n);

for i=1:n-1

for j=i+1:n

if A(i,i)==0

for k=n:-1:i+1

if A(k,i)~=0

L([i k],:)=L([k i],:);

A([i k],:)=A([k i],:);

P([i k],:)=P([k i],:);

break;

end

end

end

L(j,i)=A(j,i)/A(i,i);

A(j,:)=A(j,:)-(A(j,i)/A(i,i))\*A(i,:);

end

end

这里仅给出PLU分解的函数代码，主函数中运行过程和结果与上方结果一致，相较于上方的LU分解的函数代码，最大的区别是进行了行的交换：A([i k],:)=A([k i],:);选择对角元非0。当然，此时代码的工作量也会增加很多。

# 六、结论和感想

通过这个算例，我们可以得出，对于一个矩阵，可以分解为一个下三角矩阵和一个上三角矩阵的乘积，且二者都不需要额外存储，当然在运算时需要考虑主元非零的问题，如果主元过小，趋于0的话，需要进行PLU分解。

实际应用中，这样的方法也是非常有意义的，拿我们的电子专业来说，在处理信号时，A矩阵相当于系统里的各种滤波和变换操作，x相当于系统的输入，b相当于系统的输出，我们一般是获得了输出b，然后想求得输入x，只要系统不变，那么知道b，又知道了L和U矩阵，我们只需要对每一个新的b执行n^2次乘法/除法和n^2-n次加法/减法就可以获得b对应的输入x了，正因为这样，LU分解在实际应用中用的也是非常广泛。