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：

$$M_{n-1}\dots M_2M_1Ax = MA = Ux = M_{n-1}\dots M_2M_1b$$

$$A = M^{-1}U = {M_1}^{-1}\dots {M_{n-2}}^{-1}{M_{n-1}}^{-1}U = L_1\dots L_{n-2}L_{n-1}U = LU$$

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

$$\begin{bmatrix} 2 & -1 & 3 \\ 4 & 2 & 5 \\ 1 & 2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix}$$

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。