第八章、MATLAB解方程

一、线性方程组求解

1、 直接解法：

利用左除运算符的直接解法

对于线性方程组***Ax=b***，可以利用左除运算符求解：

x=A\b

$$\begin{cases} 12x_1 - 3x_2 + 3x_3 = 15 \\ -18x_1 + 3x_2 - x_3 = -15 \\ x_1 + x_2 + x_3 = 6 \end{cases}$$

```
A=[12 -3 3; -18 3 -1; 1 1 1];
b=[15;-15;6]
x=A\b
```

2、 利用矩阵的分解求解线性方程组

矩阵分解是指根据一定的原理用某种算法将一个矩阵分解成若干个矩阵的乘积。常见的矩阵分解有**LU**分解、**QR**分解、**Cholesky**分解，奇异值分解等。

（1）LU分解

矩阵的**LU**分解就是将一个矩阵表示为一个交换下三角矩阵和一个上三角矩阵的乘积形式。线性代数中已经证明，只要方阵**A**是非奇异的，**LU**分解总是可以进行的。**LUX=b**

调用格式为：**[L,U]=lu(A)**

功能：产生一个上三角阵**U**和一个变换形式的下三角阵**L**(行交换)，使之满足**X=LU**。注意，这里的矩阵**A**必须是方阵。

调用格式为：**[L,U,P]=lu(A)**：

功能：产生一个上三角阵**U**和一个下三角阵**L**以及一个置换矩阵**P**，使之满足**PA=LU**。当然矩阵**A**同样必须是方阵。

实现**LU**分解后，线性方程组**Ax=b**的解**x=U\(L\b)**或**x=U\(L\Pb)**，这样可以大大提高运算速度。

置换矩阵是一种系数只由0和1组成的方块矩阵。置换矩阵的每一行和每一列都恰好有一个1，其余的系数都是0。在线性代数中，每个n阶的置换矩阵都代表了一个对n个元素（n维空间的基）的置换。当一个矩阵乘上一个置换矩阵时，所得到的是原来矩阵的横行（置换矩阵在左）或纵列（置换矩阵在右）经过置换后得到的矩阵。

对应于置换π = (1 4 2 5 3)的置换矩阵P_π 是

$$P_\pi = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \mathbf{e}_{\pi(3)} \\ \mathbf{e}_{\pi(4)} \\ \mathbf{e}_{\pi(5)} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_4 \\ \mathbf{e}_2 \\ \mathbf{e}_5 \\ \mathbf{e}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

给定一个向量 **g**，

$$P_\pi \mathbf{g} = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \mathbf{e}_{\pi(3)} \\ \mathbf{e}_{\pi(4)} \\ \mathbf{e}_{\pi(5)} \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_4 \\ g_2 \\ g_5 \\ g_3 \end{bmatrix}.$$

第一种调用格式：

```
A=[12 -3 3; -18 3 -1; 1 1 1];
b=[15;-15;6]
[L,U]=lu(A);
x=U\(L\b)
```

采用LU分解的第二种格式，命令如下：

```
[L,U ,P]=lu(A);
x=U\(L\P*b)
```

（2）QR分解

对矩阵A进行QR分解，就是把X分解为一个正交矩阵Q和一个上三角矩阵R的乘积形式。QR分解只能对方阵进行。MATLAB的函数qr可用于对矩阵进行QR分解，

调用格式：[Q,R]=qr(A)

功能：产生一个正交矩阵Q和一个上三角矩阵R，使之满足A=QR。

调用格式：[Q,R,E]=qr(A)

功能：产生一个正交矩阵Q、一个上三角矩阵R以及一个置换矩阵E，使之满足AE=QR。

实现QR分解后，线性方程组Ax=b的解x=R\(Q\b)或x=E(R\(Q\b))。

采用QR分解的第1种格式，命令如下

```
A=[2,1,-5,1;1,-5,0,7;0,2,1,-1;1,6,-1,-4];
b=[13,-9,6,0]';
[Q,R]=qr(A);
x=R\(Q\b)
```

采用QR分解的第2种格式，命令如下：

```
[Q,R,E]=qr(A);
x=E*(R\(Q\b))
```

（3）Cholesky分解

如果矩阵X是对称正定的，则Cholesky分解将矩阵X分解成一个下三角矩阵和上三角矩阵的乘积。设上三角矩阵为R，则下三角矩阵为其转置，即A=R'R。

MATLAB函数chol(A)用于对矩阵X进行Cholesky分解。

调用格式为：R=chol(A)

功能：产生一个上三角阵R，使*R'R=A*。若A为非对称正定，则输出一个出错信息。

实现Cholesky分解后，线性方程组Ax=b变成R‘Rx=b，所以x=R\(R’\b)。

3、迭代解法

（1）Jacobi迭代法

对于线性方程组**Ax=b**，如果**A**为非奇异方阵，即**aii≠0(i=1,2,…,n)**，则可将**A**分解为**A=D-L-U**，其中**D**为对角阵，其元素为**A**的对角元素，**L**与**U**为**A**的下三角阵和上三角阵，于是**Ax=b**化为：

$$(D-L-U)x=b$$

$$Dx=b+Lx+Ux$$

$$x=D^{-1}(b+Lx+Ux)$$

$$x=D^{-1}(L+U)x+D^{-1}b$$

与之对应的迭代公式为：$x(k+1)=D^{-1}(L+U)x(k)+D^{-1}b$

这就是**Jacobi**迭代公式。如果序列**{x(k+1)}**收敛于**x**，则**x**必是方程**Ax=b**的解。

Jacobi迭代法的MATLAB函数文件Jacobi.m如下：

```
function [y,n]=jacobi(A,b,x0,eps)
if nargin==3
    eps=1.0e-6;
elseif nargin<3
    error
    return
end
D=diag(diag(A));    %求A的对角矩阵
L=-tril(A,-1);      %求A的下三角矩阵
U=-triu(A,1);       %求A的上三角矩阵
B=D\(L+U);
f=D\b;
y=B*x0+f;
n=1;                %迭代次数
while abs(y-x0)>=eps
    x0=y;
    y=B*x0+f;
    n=n+1;
end
```

用Jacobi迭代法求解下列线性方程组。设迭代初值为0，迭代精度为10-6。

在命令中调用函数文件Jacobi.m，命令如下：

```
A=[10,-1,0;-1,10,-2;0,-2,10];
b=[9,7,6]';
[x,n]=jacobi(A,b,[0,0,0]',1.0e-6)
```

（2）Gauss-Serdel迭代法

在Jacobi迭代过程中的迭代公式$Dx(k+1)=(L+U)x(k)+b$

可以改进为$Dx(k+1)=Lx(k+1)+Ux(k)+b$

于是得到：$x(k+1)=(D-L)^{-1}Ux(k)+(D-L)^{-1}b$

该式即为Gauss-Serdel迭代公式。和Jacobi迭代相比，

Gauss-Serdel迭代用新分量代替旧分量，精度会高些。

Gauss-Serdel迭代法的MATLAB函数文件gauseidel.m如下：

```
function [y,n]=gauseidel(A,b,x0,eps)
if nargin==3
    eps=1.0e-6;
elseif nargin<3
    error
    return
end
D=diag(diag(A));    %求A的对角矩阵
L=-tril(A,-1);      %求A的下三角矩阵
U=-triu(A,1);       %求A的上三角矩阵
G=(D-L)\U;
f=(D-L)\b;
y=G*x0+f;
n=1;                %迭代次数
while abs(y-x0)>=eps
    x0=y;
    y=G*x0+f;
    n=n+1;
end
```

用Gauss-Serdel迭代法求下列线性方程组。迭代初值为0，迭代精度为10-6。

在命令中调用函数文件gauseidel.m，命令如下：

```
A=[10,-1,0;-1,10,-2;0,-2,10];
b=[9,7,6]';
[x,n]=gauseidel(A,b,[0,0,0]',1.0e-6)
```

二、非线性方程组的求解

对于非线性方程组**F(X)=0**，用**fsolve**函数求其数值解。**fsolve**函数的调用格式为：

X=fsolve('fun',X0)

其中**X**为返回的解，

fun是用于定义需求解的非线性方程组的函数文件名，

X0是求根过程的初值，

求下列非线性方程组在(0.5,0.5) 附近的数值解。

(1) 建立函数文件myfun.m。

```
function q=myfun(p)
x=p(1);
y=p(2);
q(1)=x-0.6*sin(x)-0.3*cos(y);
q(2)=y-0.6*cos(x)+0.3*sin(y);
```

(2) 在给定的初值x0=0.5，y0=0.5下，调用fsolve函数求方程的根。

```
x=fsolve('myfun',[0.5,0.5]')
x =
    0.6354
    0.3734
```

将求得的解代回原方程，可以检验结果是否正确，命令如下：

```
q=myfun(x)
q =
     1.0e-009 *
    0.2375    0.2957
```

可见得到了较高精度的结果。

fplot

三、 函数极值

MATLAB提供了基于单纯形算法求解函数极值的函数fmin,其调用格式为：

x=fminsearch('fname',x1,x2)

其中fminsearch函数用于求单变量函数的最小值点。fname是被最小化的目标函数名，x1和x2限定自变量的取值范围。

MATLAB没有专门提供求函数最大值的函数，但只要注意到-f(x)在区间(a,b)上的最小值就是f(x)在(a,b)的最大值，所以fmin(f,x1,x2)返回函数f(x)在区间(x1,x2)上的最大值。

求f(x)=x3-2x-5在[0,5]内的最小值点。

(1) 建立函数文件mymin.m。

```
function fx=mymin(x)
fx=x.^3-2*x-5;
```

(2) 调用fmin函数求最小值点。

```
x=fminsearch('mymin',0,5)
x=
   0.8165
```