Numerical Analysis HW3

ID：3200101135 Name：李坤林

Problem 1

> The following linear systems $A\mathbf{x}=\mathbf{b}$ have $\mathbf{x}$ as the actual solution and $\tilde{\mathbf{x}}$ as an approximate solution. Compute $||\mathbf{x}-\tilde{\mathbf{x}}||_\infty$.

a.

$$||\mathbf{x}-\tilde{\mathbf{x}}||_\infty = ||(0.2, 0.5, 0.4)^T|| = 0.5$$
$$||A\tilde{\mathbf{x}}-\mathbf{b}||_\infty = ||(0, -0.3, -0.2)^T|| = 0.3$$

b.

$$||\mathbf{x}-\tilde{\mathbf{x}}||_\infty = ||(0.33, 0.9, -0.8)^T|| = 0.9$$
$$||A\tilde{\mathbf{x}}-\mathbf{b}||_\infty = ||(0.27, 0.16, 0.21)^T|| = 0.27$$

Problem 2

> Show that if A is symmetric, then $||A||_2 = \rho(A)$

证明:

记：$D = diag\{\lambda_1, \lambda_2, \lambda_3, \cdots, \lambda_n\}$，满足：$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$

则：$||\lambda_1||$为A的谱半径

令：x_1为λ_1对应的右特征向量，满足：$Ax_1 = \lambda_1 x_1$

则必有：

满足：$\frac{||Ax_1||_2}{||x_1||_2} = |\lambda_1| \leq ||A||_2 \quad (2.1)$

令y_1为A的2范数对应的单位向量，即：$||y_1||_2 = 1$ 且 $||A||_2 = ||Ay_1||_2$

而：y_1可以被Q线性表出为：$y_1 = Qz_1$，且z_1也为单位向量

不难得出：$||A||_2 = ||Ay_1||_2 = ||AQz_1||_2 = ||Dz_1||_2 \leq |\lambda_1| \quad (2.2)$

综上2.1&2.2可得：$||A||_2 = |\lambda_1|$

$Q.E.D.$

Problem 3

> Implement the algorithm of Gaussian elimination with scaled partial pivoting, and solve the following linear systems.

a.

见"HW3\Code\gauss.m"

```
clear all;
clc;
A = [0.03 58.9;5.31 -6.10];
b = [59.2 47.0];
x= GaussianSolver0(A,b)

function x=GaussianSolver0(A,b)
[n,~] = size(A);
x=zeros(n,1);

for j = 1:n-1
    for i = j+1:n %矩阵的维数
        mul = A(i,j)/A(j,j);
        A(i,:)= A(i,:)-mul*A(j,:);
        b(i)= b(i)-mul*b(j);
    end
end%这个循环可以使矩阵A的第一列元素全为0

for i=n:-1:1
    sum=0;
    for j=n:-1:i+1
        sum=sum+x(j)*A(i,j);
    end
    x(i)=(b(i)-sum)/A(i,i);
end
end
%以下为命令行输入:
%>> format long
%>> x
%命令行输出:
```

```
%x=10.000000000000142
%   1.000000000000000
```

	x_1	x_2
准确值	10	1
计算值	10.000000000000142	1.000000000000000

b.

见"HW3\Code\gauss_2.m"

```
clear

A=input('输入系数矩阵A: ');
b=input('输入b向量（按行向量）: ');
B=[A b'];
n=length(b);
RA=rank(A);
RB=rank(B);
zhica=RB-RA;
if zhica>0,
   disp('此方程组无解.\n')
    return
end

if RA==RB
    if RA==n
       fprintf('此方程组有唯一解.\n',n)
       X=zeros(n,1);
       for p=1:n-1
           t=find(abs(B(p:end,p))==max(abs(B(p:end,p))))+p-1;
            if abs(B(t,p))~=abs(B(p,p))
                 l=B(t,:);
                 B(t,:)=B(p,:);
                 B(p,:)=l;
            end  %列主元判断

            for k=p+1:n
                 m= B(k,p)/ B(p,p);
                 B(k,p:n+1)= B(k,p:n+1)-m* B(p,p:n+1);
            end
       end
       %把方程组系数矩阵A化为同解的上三角矩阵

       b=B(1:n,n+1);
       A=B(1:n,1:n);
       X(n)=b(n)/A(n,n);
       for q=n-1:-1:1
          X(q)=(b(q)-sum(A(q,q+1:n)*X(q+1:n)))/A(q,q);
       end
       %从xn至x1逐个求解上三角方程组

```

```
    else
      disp('此方程组有无穷多解.')
       return
    end
end

disp('方程组的解为: ');X

%以下为命令行提示及输入
%>> gauss_2
%输入系数矩阵A: [3.03,-12.1,14;-3.03,12.1,-7;6.11,-14.2,21]
%输入b向量 (按行向量) : [-119,120,-139]
%命令行输出
%X =                  0
%   10.000000000000000
%    0.142857142857143
```

	x_1	x_2	x_2
准确解	0	10	$\frac{7}{1}$
数值解	0	10.000000000000000	0.142857142857143

Problem 4

> Implement the Jacobi iterative method and list the first three iteration results when solving the following linear systems, using $\mathbf{x}^{(0)} = \mathbf{0}$.

a.

代码见"HW3\Code\jacobi.m"

```
clear;
A=input('请输入线性方程组的系数矩阵:');
b=input('请输入线性方程组的常向量:');
x1=input('请输入解向量的初始值:');
n=numel(b);

e_max=1e6;        %%前后之差
while e_max>=1e-6
    e_max=0;
    for i=1:n
        s=0;            %初始化变量
        for j=1:n
            if j~=i

                s=s+A(i,j)*x1(j);
            end
        end
        x2(i) = (b(i)-s)/A(i,i);
        e = abs(x2(i)-x1(i));
        if e > e_max
```

```
            e_max = e;
        end
    end
    x1=x2       %观察每步迭代结果
end

%以下为命令行提示及输入
%>> jacobi_a
%请输入线性方程组的系数矩阵:[4,1,-1;-1,3,1;2,2,5]
%请输入线性方程组的常向量:[5,-4,1]
%请输入解向量的初始值:[0,0,0]
%以下为命令行输出
%x1 =1.250000000000000  -1.333333333333333   0.200000000000000
%x1 =1.633333333333333  -0.983333333333333   0.233333333333333
%x1 =1.554166666666667  -0.866666666666667  -0.060000000000000
%x1 =1.451666666666667  -0.795277777777778  -0.075000000000000
%x1 =1.430069444444444  -0.824444444444445  -0.062555555555556
%x1 =1.440472222222222  -0.835791666666667  -0.042250000000000
%x1 =1.448385416666667  -0.839092592592593  -0.041872222222222
%x1 =1.449305092592593  -0.836580787037037  -0.043717129629630
%x1 =1.448215914351852  -0.835659259259259  -0.045089722222222
%x1 =1.447642384259259  -0.835564787808642  -0.045022662037037
%x1 =1.447635531442901  -0.835778317901235  -0.044831038580247
%x1 =1.447736819830247  -0.835844476658951  -0.044742885416667
%x1 =1.447775397810571  -0.835840098251029  -0.044756937268519
%x1 =1.447770790245628  -0.835822554973637  -0.044774119823817
%x1 =1.447762108787455  -0.835818363310185  -0.044779294108796
%x1 =1.447759767300347  -0.835819532367916  -0.044777498190908
%x1 =1.447760508544252  -0.835820911502915  -0.044776093972972
%x1 =1.447761204382486  -0.835821132494258  -0.044775838816535
```

	x_1	x_2	x_3
$\mathbf{x}^{(1)}$	1.250000000000000	-1.333333333333333	0.200000000000000
$\mathbf{x}^{(2)}$	1.633333333333333	-0.983333333333333	0.233333333333333
$\mathbf{x}^{(3)}$	1.554166666666667	-0.866666666666667	-0.060000000000000

b.

代码同上，下面给出命令行操作

```
%>> jacobi_a
%请输入线性方程组的系数矩阵:[-2,1,0.5;1,-2,-0.5;0,1,2]
%请输入线性方程组的常向量:[4,-4,0]
%请输入解向量的初始值:[0,0,0]

%以下为命令行输出
%x1 =-2      2      0
%x1 =-1      1     -1
%x1 =-1.750000000000000   1.750000000000000  -0.500000000000000
%x1 =-1.250000000000000   1.250000000000000  -0.875000000000000
%x1 =-1.593750000000000   1.593750000000000  -0.625000000000000
```

```
%x1 =-1.359375000000000   1.359375000000000  -0.796875000000000
%x1 =-1.519531250000000   1.519531250000000  -0.679687500000000
%x1 =-1.410156250000000   1.410156250000000  -0.759765625000000
%x1 =-1.484863281250000   1.484863281250000  -0.705078125000000
%x1 =-1.433837890625000   1.433837890625000  -0.742431640625000
%x1 =-1.468688964843750   1.468688964843750  -0.716918945312500
%x1 =-1.444885253906250   1.444885253906250  -0.734344482421875
%x1 =-1.461143493652344   1.461143493652344  -0.722442626953125
%x1 =-1.450038909912109   1.450038909912109  -0.730571746826172
%x1 =-1.457623481750488   1.457623481750488  -0.725019454956055
%x1 =-1.452443122863770   1.452443122863770  -0.728811740875244
%x1 =-1.455981373786926   1.455981373786926  -0.726221561431885
%x1 =-1.453564703464508   1.453564703464508  -0.727990686893463
%x1 =-1.455215319991112   1.455215319991112  -0.726782351732254
%x1 =-1.454087927937508   1.454087927937508  -0.727607659995556
%x1 =-1.454857951030135   1.454857951030135  -0.727043963968754
%x1 =-1.454332015477121   1.454332015477121  -0.727428975515068
%x1 =-1.454691236140206   1.454691236140206  -0.727166007738560
%x1 =-1.454445883864537   1.454445883864537  -0.727345618070103
%x1 =-1.454613462585257   1.454613462585257  -0.727222941932268
%x1 =-1.454499004190438   1.454499004190438  -0.727306731292629
%x1 =-1.454577180727938   1.454577180727938  -0.727249502095219
%x1 =-1.454523785159836   1.454523785159836  -0.727288590363969
%x1 =-1.454560255011074   1.454560255011074  -0.727261892579918
%x1 =-1.454535345639442   1.454535345639442  -0.727280127505537
%x1 =-1.454552359056663   1.454552359056663  -0.727267672819721
%x1 =-1.454540738676599   1.454540738676599  -0.727276179528332
%x1 =-1.454548675543784   1.454548675543784  -0.727270369338299
%x1 =-1.454543254562683   1.454543254562683  -0.727274337771892
%x1 =-1.454546957161631   1.454546957161631  -0.727271627281342
%x1 =-1.454544428239520   1.454544428239520  -0.727273478580816
%x1 =-1.454546155525444   1.454546155525444  -0.727272214119760
%x1 =-1.454544975767218   1.454544975767218  -0.727273077762722
%x1 =-1.454545781557071   1.454545781557071  -0.727272487883609
```

	x_1	x_2	x_3
$\mathbf{x}^{(1)}$	-2	2	0
$\mathbf{x}^{(2)}$	-1	1	-1
$\mathbf{x}^{(3)}$	-1.750000000000000	1.750000000000000	-0.500000000000000

Problem 5

> Use the Jacobi method and Gauss-Seidel method to solve the following linear systems, with TOL= 0.001 in the $\mathbf{L}_\infty$ norm.

a、b两题所用Gauss-Seidel方法代码相同，所以先将代码置顶，解答中给出命令行操作，代码详见"HW3\Code\Gauss_Seidel.m"

Jacobi方法代码与上题相同，解答仅给出命令行操作

```
function x=Gauss_Seidel(A,b,x0,ep,N)
%A为系数矩阵，b为右端向量，x0为初始向量（默认零向量）
%ep为精度（1e-6），N为最大迭代次数（默认500次），x返回近似解向量
n=length(b);
if nargin<5
    N=500;
end

if nargin<4
    ep=1e-6;
end

if nargin<3
    x0=zeros(n,1);
end

x=zeros(n,1);
k=0;

while k<N
    for i=1:n
        if i==1
            x(1)=(b(1)-A(1,2:n)*x0(2:n))/A(1,1);
        elseif i==n
                x(n)=(b(n)-A(n,1:n-1)*x(1:n-1))/A(n,n);
        else
                x(i)=(b(i)-A(i,1:i-1)*x(1:i-1)-A(i,i+1:n)*x0(i+1:n))/A(i,i);
        end
    end

    if norm(x-x0,inf)<ep
        break;
    end

    x0=x;

    k=k+1;
end

if k==N
    Warning('已到达迭代次数上限！');
end

disp(['k=',num2str(k)])
```

a.

Gauss_Seidel方法

```
%命令行操作
>> B=[3,-1,1;3,6,2;3,3,7];
>> b2=[1;0;4];
>> x0=[0;0;0];
>> ep=1e-3;
>> N=500;
>> x=Gauss_Seidel(B,b2,x0,ep,N)
%命令行输出
x =0.035351068284877
  -0.236788626595343
   0.657758953561628
```

Jacobi方法

```
%命令行操作
>> jacobi_a
请输入线性方程组的系数矩阵:[3,-1,1;3,6,2;3,3,7]
请输入线性方程组的常向量:[1,0,4]
请输入解向量的初始值:[0,0,0]
%命令行输出
%由于直接使用Problem4的代码，所以中途打印了迭代过程值，此处删去
x1 =0.035087737436061
   -0.236841911943220
    0.657894952890508
```

	Gauss_Seidel方法	Jacobi方法
L_∞	0.657758953561628	0.657894952890508

b.

Gauss_Seidel方法

```
%命令行操作
>> B=[10,-1,0;-1,10,-2;0,-2,10];
>> b2=[9;7;6];
>> x0=[0;0;0];
>> ep=1e-3;
>> N=500;
>> x=Gauss_Seidel(B,b2,x0,ep,N)
%命令行输出
x =0.995747500000000
   0.957873750000000
   0.791574750000000
```

Jacobi方法

```
%命令行操作
>> jacobi_a
请输入线性方程组的系数矩阵:[10,-1,0;-1,10,-2;0,-2,10]
请输入线性方程组的常向量:[9,7,6]
请输入解向量的初始值:[0,0,0]
%命令行输出
%由于直接使用Problem4的代码，所以中途打印了迭代过程值，此处删去
x1 =0.995789443750000
    0.957894656250000
    0.791578887500000
```

	Gauss_Seidel方法	Jacobi方法
L_∞	0.995747500000000	0.995789443750000