1.已知下列方程组



用Jacobi迭代法，G-S迭代法，SOR方法（分别取松弛因子 ），最速下降法，共轭梯度法求解.要求精度 ，取最大迭代次数*N*=25.并比较集中迭代法的迭代步数与收敛速度.



1. Jacobi迭代法

A=[4 -1 0 -1 0 0;-1 4 -1 0 -1 0;

0 -1 4 0 0 -1;-1 0 0 4 -1 0;

0 -1 0 -1 4 -1;0 0 -1 0 -1 4];

b=[0;5;0;6;-2;6];

x0=[1.1 1.2 1.3 1.1 1.2 1.3];

t=6;

n=length(A);

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;

x(2:n+1,1)=x0;

for k = 1:t

for i = 1:n

if i==1

x(i+1,k+1)=-(1/A(i,i))\*(sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

if i~=1

x(i+1,k+1)=-(1/A(i,i))\*(sum(A(i,1:i-1)\*x(2:i,k))+sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

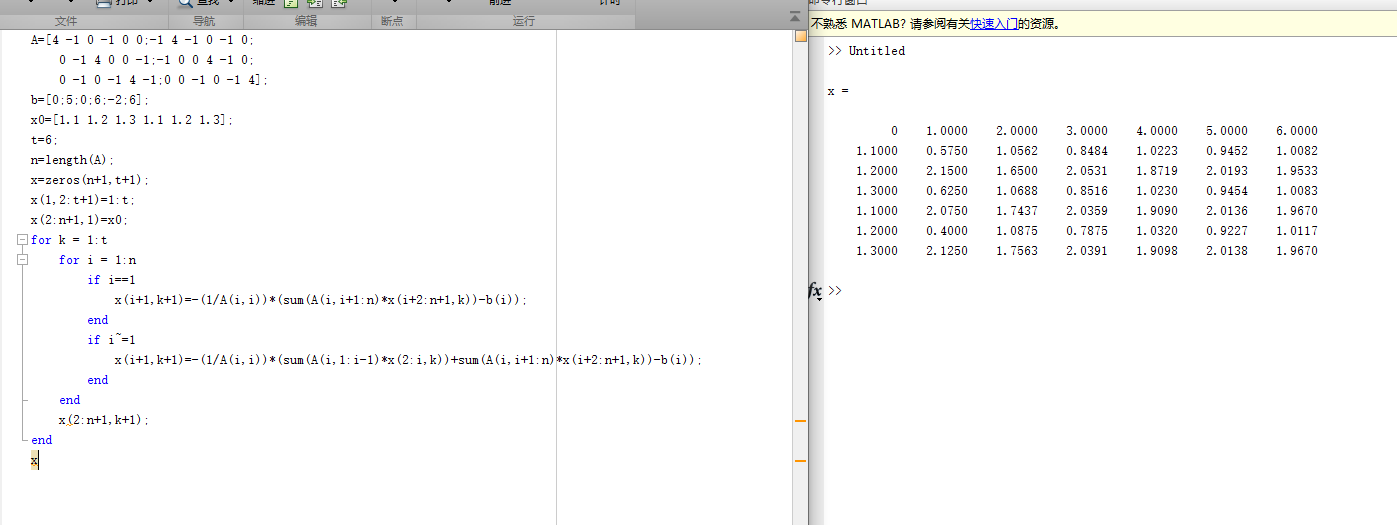
end

end

x(2:n+1,k+1);

end

x



x =

0 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000

1.1000 0.5750 1.0562 0.8484 1.0223 0.9452 1.0082

1.2000 2.1500 1.6500 2.0531 1.8719 2.0193 1.9533

1.3000 0.6250 1.0688 0.8516 1.0230 0.9454 1.0083

1.1000 2.0750 1.7437 2.0359 1.9090 2.0136 1.9670

1.2000 0.4000 1.0875 0.7875 1.0320 0.9227 1.0117

1.3000 2.1250 1.7563 2.0391 1.9098 2.0138 1.9670

1. G-S迭代法

A=[4 -1 0 -1 0 0;-1 4 -1 0 -1 0;

0 -1 4 0 0 -1;-1 0 0 4 -1 0;

0 -1 0 -1 4 -1;0 0 -1 0 -1 4];

b=[0;5;0;6;-2;6];

x0=[1.1 1.2 1.3 1.1 1.2 1.3];

t=6;

n=length(A);

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;

x(2:n+1,1)=x0;

for k = 1:t

for i = 1:n

if i==1

x(i+1,k+1)=-(1/A(i,i))\*(sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

if i~=1

x(i+1,k+1)=-(1/A(i,i))\*(sum(A(i,1:i-1)\*x(2:i,k+1))+sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

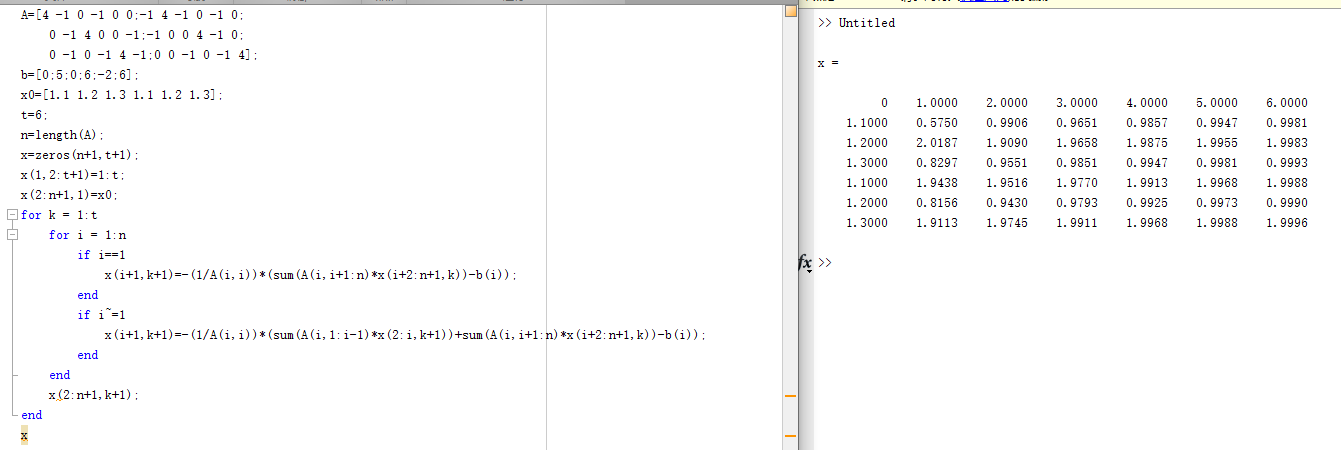
end

end

x(2:n+1,k+1);

end

x



x =

0 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000

1.1000 0.5750 0.9906 0.9651 0.9857 0.9947 0.9981

1.2000 2.0187 1.9090 1.9658 1.9875 1.9955 1.9983

1.3000 0.8297 0.9551 0.9851 0.9947 0.9981 0.9993

1.1000 1.9438 1.9516 1.9770 1.9913 1.9968 1.9988

1.2000 0.8156 0.9430 0.9793 0.9925 0.9973 0.9990

1.3000 1.9113 1.9745 1.9911 1.9968 1.9988 1.9996

1. SOR方法

A=[4 -1 0 -1 0 0;-1 4 -1 0 -1 0;

0 -1 4 0 0 -1;-1 0 0 4 -1 0;

0 -1 0 -1 4 -1;0 0 -1 0 -1 4];

b=[0;5;0;6;-2;6];

x0=[1.1 1.2 1.3 1.1 1.2 1.3];

omg=1.1;

t=6;

n=length(A);

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;

x(2:n+1,1)=x0;

for k = 1:t

for i = 1:n

if i==1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

if i~=1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,1:i-1)\*x(2:i,k))+sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

end

x(2:n+1,k+1);

end

x

omg=1.2;

t=6;

n=length(A);

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;

x(2:n+1,1)=x0;

for k = 1:t

for i = 1:n

if i==1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

if i~=1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,1:i-1)\*x(2:i,k))+sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

end

x(2:n+1,k+1);

end

x

omg=1.3;

t=6;

n=length(A);

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;

x(2:n+1,1)=x0;

for k = 1:t

for i = 1:n

if i==1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

end

if i~=1

x(i+1,k+1)=-omg\*(1/A(i,i))\*(sum(A(i,1:i-1)\*x(2:i,k))+sum(A(i,i+1:n)\*x(i+2:n+1,k))-b(i));

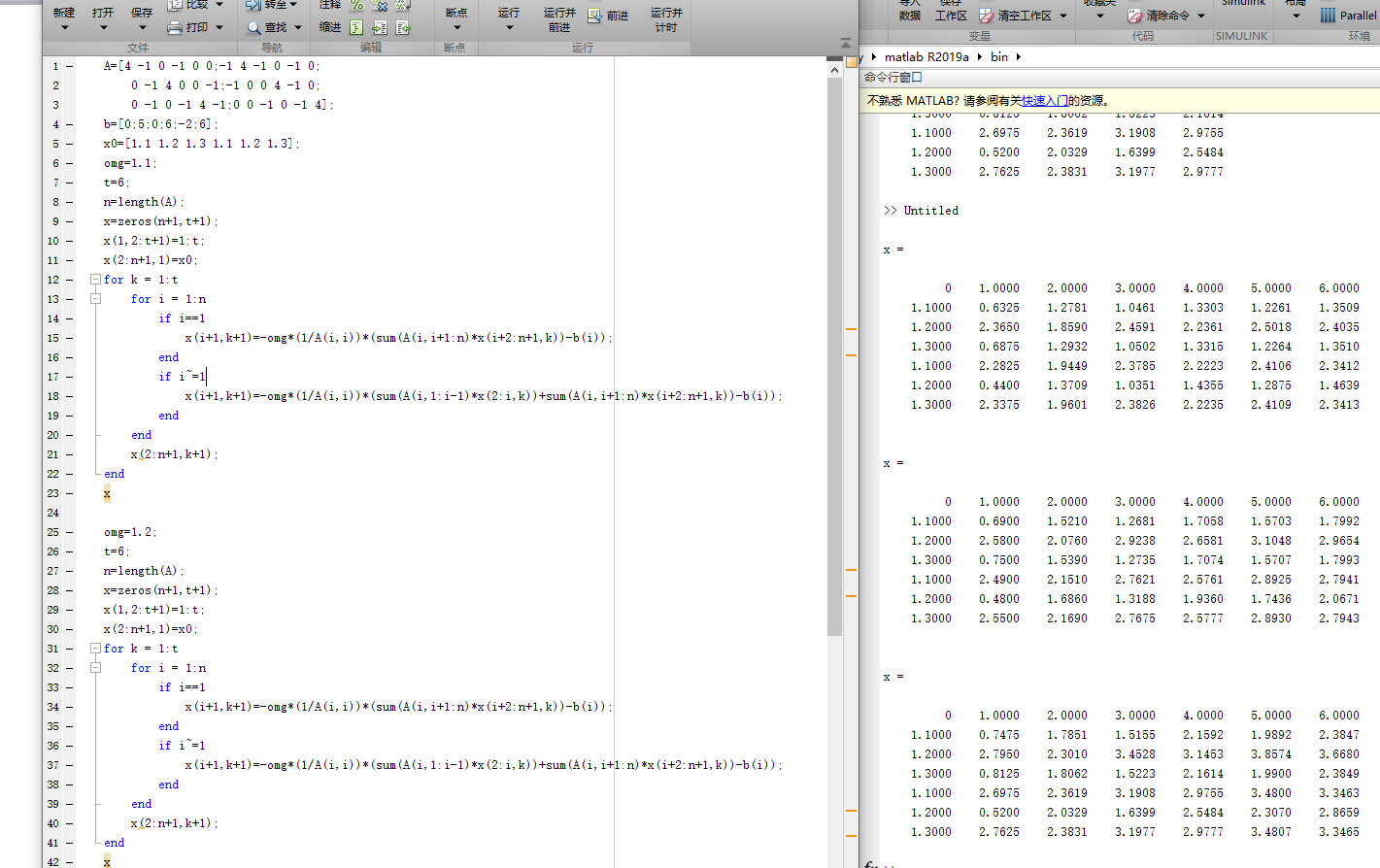
end

end

x(2:n+1,k+1);

end

x



松弛因子1.1

x =

0 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000

1.1000 0.6325 1.2781 1.0461 1.3303 1.2261 1.3509

1.2000 2.3650 1.8590 2.4591 2.2361 2.5018 2.4035

1.3000 0.6875 1.2932 1.0502 1.3315 1.2264 1.3510

1.1000 2.2825 1.9449 2.3785 2.2223 2.4106 2.3412

1.2000 0.4400 1.3709 1.0351 1.4355 1.2875 1.4639

1.3000 2.3375 1.9601 2.3826 2.2235 2.4109 2.3413

松弛因子1.2

x =

0 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000

1.1000 0.6900 1.5210 1.2681 1.7058 1.5703 1.7992

1.2000 2.5800 2.0760 2.9238 2.6581 3.1048 2.9654

1.3000 0.7500 1.5390 1.2735 1.7074 1.5707 1.7993

1.1000 2.4900 2.1510 2.7621 2.5761 2.8925 2.7941

1.2000 0.4800 1.6860 1.3188 1.9360 1.7436 2.0671

1.3000 2.5500 2.1690 2.7675 2.5777 2.8930 2.7943

松弛因子1.3

x =

0 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000

1.1000 0.7475 1.7851 1.5155 2.1592 1.9892 2.3847

1.2000 2.7950 2.3010 3.4528 3.1453 3.8574 3.6680

1.3000 0.8125 1.8062 1.5223 2.1614 1.9900 2.3849

1.1000 2.6975 2.3619 3.1908 2.9755 3.4800 3.3463

1.2000 0.5200 2.0329 1.6399 2.5484 2.3070 2.8659

1.3000 2.7625 2.3831 3.1977 2.9777 3.4807 3.3465

4.最速下降法

A=[4 -1 0 -1 0 0;-1 4 -1 0 -1 0;

0 -1 4 0 0 -1;-1 0 0 4 -1 0;

0 -1 0 -1 4 -1;0 0 -1 0 -1 4];

b=[0 5 0 6 -2 6];

x0=[1.1 1.2 1.3 1.1 1.2 1.3];

t=6;

n=6;

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t; %第1行第2到t+1列

x(2:n+1,1)=x0; %第2行到n+1行第1列

r(:,1)=b'-A\*x(2:n+1,1);%第1列 第2到n+1行第1列

for i = 1:t

a(i)=dot(r(:,i),r(:,i))/dot(A\*r(:,i),r(:,i)); %第i列 步长

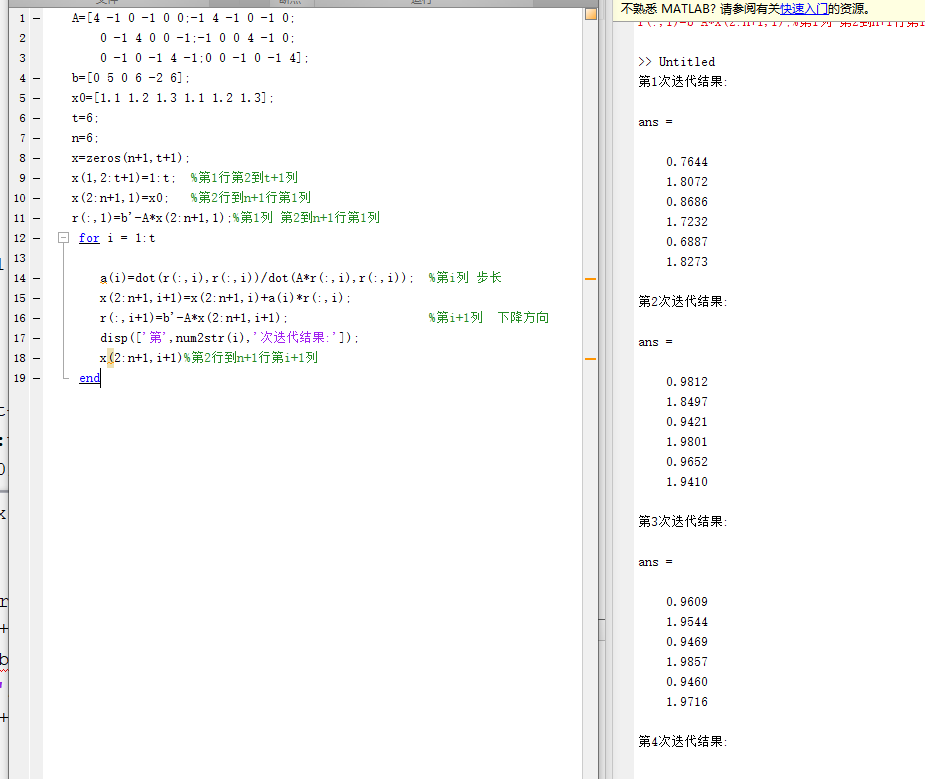
x(2:n+1,i+1)=x(2:n+1,i)+a(i)\*r(:,i);

r(:,i+1)=b'-A\*x(2:n+1,i+1); %第i+1列 下降方向

disp(['第',num2str(i),'次迭代结果:']);

x(2:n+1,i+1)%第2行到n+1行第i+1列

end



第1次迭代结果:

ans =

0.7644

1.8072

0.8686

1.7232

0.6887

1.8273

第2次迭代结果:

ans =

0.9812

1.8497

0.9421

1.9801

0.9652

1.9410

第3次迭代结果:

ans =

0.9609

1.9544

0.9469

1.9857

0.9460

1.9716

第4次迭代结果:

ans =

0.9869

1.9641

0.9841

1.9760

0.9804

1.9733

第5次迭代结果:

ans =

0.9852

1.9861

0.9843

1.9907

0.9785

1.9898

第6次迭代结果:

ans =

0.9949

1.9871

0.9948

1.9909

0.9927

1.9908

1. 共轭梯度法

A=[4 -1 0 -1 0 0;-1 4 -1 0 -1 0;

0 -1 4 0 0 -1;-1 0 0 4 -1 0;

0 -1 0 -1 4 -1;0 0 -1 0 -1 4];

b=[0 5 0 6 -2 6];

x0=[1.1 1.2 1.3 1.1 1.2 1.3];

t=6;

n=6;

x=zeros(n+1,t+1);

x(1,2:t+1)=1:t;%第1行第2到t+1列

x(2:n+1,1)=x0;%第2行到n+1行第1列

r(:,1)=b'-A\*x(2:n+1,1);%第1列 第2到n+1行第1列

p(:,1)=r(:,1);%第1列 第1列 是共轭梯度法的区别

for i = 1:t

a(i)=dot(r(:,i),p(:,i))/dot(A\*p(:,i),p(:,i)); %第i列 步长

x(2:n+1,i+1)=x(2:n+1,i)+a(i)\*p(:,i);%第2行到n+1行第i+1列 第2行到n+1行第i列 第i列

r(:,i+1)=b'-A\*(x(2:n+1,i)+a(i)\*p(:,i));%第i+1列 第2行到n+1行第i列 第i列 方向

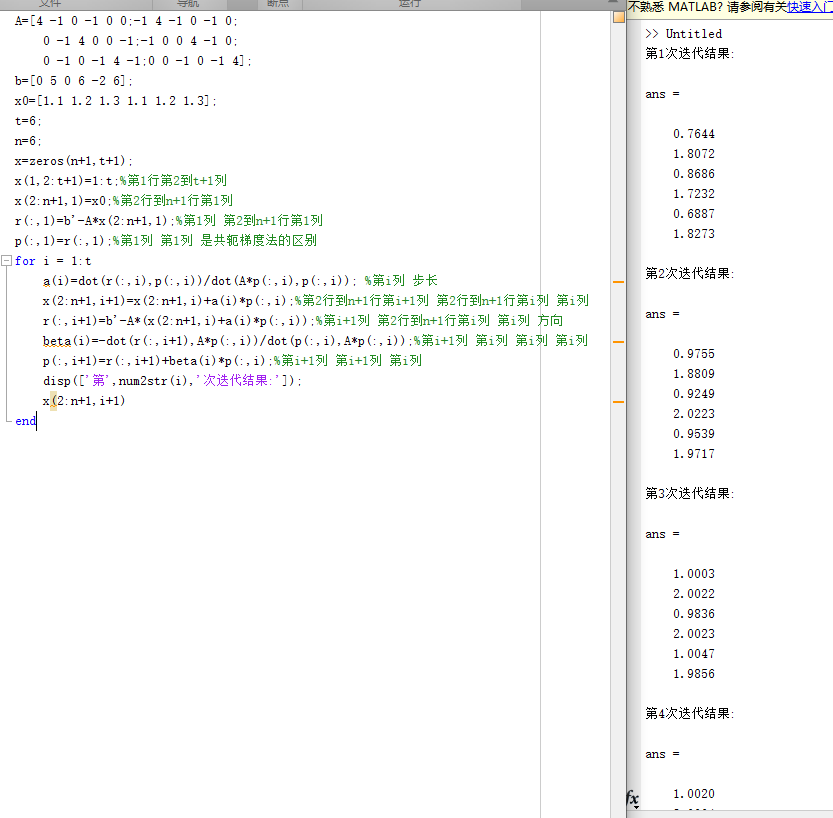
beta(i)=-dot(r(:,i+1),A\*p(:,i))/dot(p(:,i),A\*p(:,i));%第i+1列 第i列 第i列 第i列

p(:,i+1)=r(:,i+1)+beta(i)\*p(:,i);%第i+1列 第i+1列 第i列

disp(['第',num2str(i),'次迭代结果:']);

x(2:n+1,i+1)

end



第1次迭代结果:

ans =

0.7644

1.8072

0.8686

1.7232

0.6887

1.8273

第2次迭代结果:

ans =

0.9755

1.8809

0.9249

2.0223

0.9539

1.9717

第3次迭代结果:

ans =

1.0003

2.0022

0.9836

2.0023

1.0047

1.9856

第4次迭代结果:

ans =

1.0020

2.0004

0.9999

2.0006

0.9985

1.9985

第5次迭代结果:

ans =

1.0000

2.0000

1.0000

2.0000

1.0000

2.0000

第6次迭代结果:

ans =

1

2

1

2

1

2

结论：SOR方法最慢，G-S迭代法其次，jacobi迭代法和最速下降法再次，共轭梯度法最快