

课题组组会-练习 2

王程

2023 年 10 月 23 日

一 练习及结果

1. 已知方程组 $Ax = b$, 其中 $A \in \mathcal{R}^{20 \times 20}$, 定义为

$$\begin{bmatrix} 3 & -1/2 & -1/4 & & & \\ -1/2 & 3 & -1/2 & -1/4 & & \\ -1/4 & -1/2 & 3 & -1/2 & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & -1/4 \\ & & -1/4 & -1/2 & 3 & -1/2 \\ & & & -1/4 & -1/2 & 3 \end{bmatrix}$$

试通过迭代法求解此方程组, 认识迭代法收敛的含义以及迭代初值和方程组系数矩阵性质对收敛速度的影响。

实验要求:

(1) 选取不同的初始向量 $x^{(0)}$ 和不同的方程组右端项向量 b , 给定迭代误差要求, 用雅克比迭代法和高斯-赛德尔迭代法计算, 观测得到的迭代向量序列是否收敛? 若收敛, 记录迭代次数, 分析计算结果并给出你的结论。

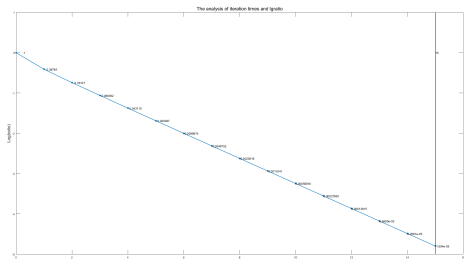
(2) 取定右端向量 b 和初始向量 $x^{(0)}$, 将 A 的主对角线元素成倍增长若干次, 非主对角线元素不变, 每次用雅克比迭代法计算, 要求迭代误差满足 $\|x^{(k+1)} - x^{(k)}\|_{\infty} < 10^{-5}$, 比较收敛速度, 分析现象, 并得出你的结论。

(3) 取定右端向量 b 和初始向量 $x^{(0)}$, 分别使用 a) 高斯-赛德尔迭代, b) 对称高斯-赛德尔迭代, c) 对以上两种方法分别加上松弛因子 ω 进行求解, 比较收敛速度, 分析现象, 得出结论。

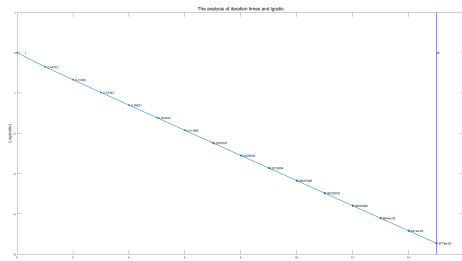
解:

(1) 采用随机数生成的方法构建初始向量 $x^{(0)}$, $x^{(0)}$ 中的元素为 $0 \sim 1$ 之间的随机数。同样的, 采用随机数生成的方法构建右端项向量 b , b 中的元素为 $0 \sim 100$ 之间的随机数, 要求迭代误差满足 $\|x^{(k+1)} - x^{(k)}\|_{\infty} < 10^{-5}$ 。以下分别给出两种迭代法的迭代次数与误差量级变化图, 其中高斯-赛德尔迭代分别采用了 Forward sweep 和 Backward sweep。

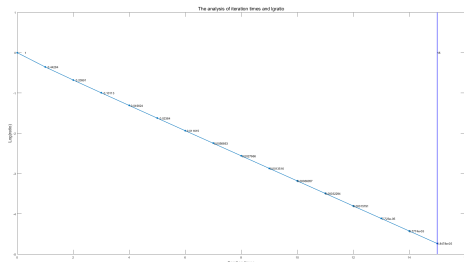
雅克比迭代法



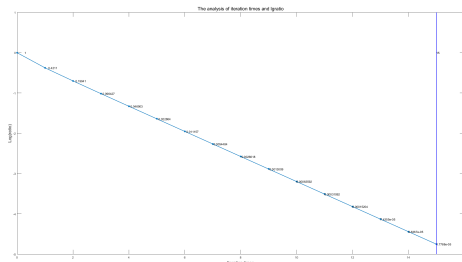
(a) random1



(b) random2



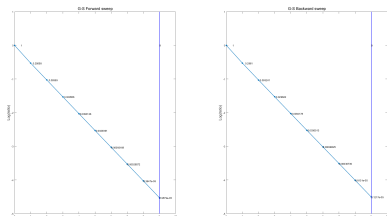
(c) random3



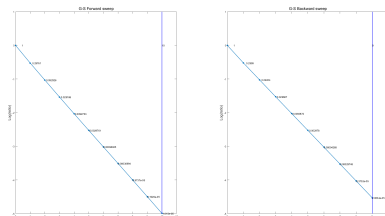
(d) random4

图 1: 雅克比迭代次数和误差量级图

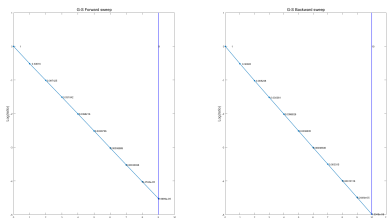
高斯-赛德尔迭代法



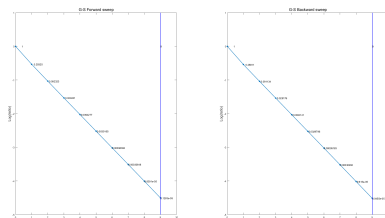
(a) random1



(b) random2



(c) random3

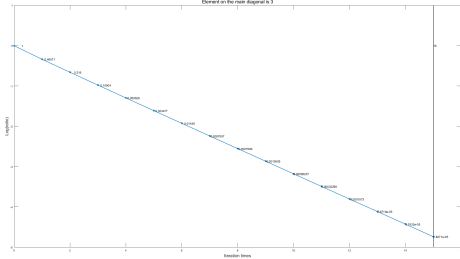


(d) random4

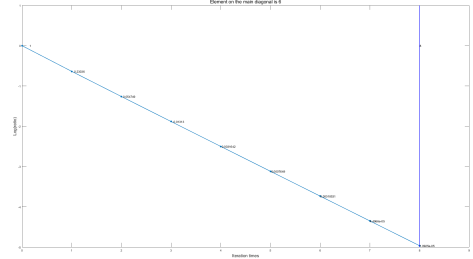
图 2: 高斯迭代次数和误差量级图

观察图 1 可以看出, 需要进行 15 次左右的雅克比迭代, 方程的解才能小于迭代误差限制。观察图 2 可以看出, 需要进行 9 次左右的高斯-赛德尔迭代, 方程的解才能小于迭代误差限制。高斯-赛德尔迭代的收敛速度快于雅克比迭代。

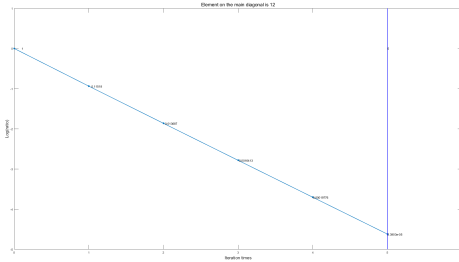
(2) 取右端向量 $b = [1, 2, \dots, 20]^T$, 初始向量 $x^{(0)} = 0$, 将 A 的主对角线元素成倍增长, 非主对角线元素不动, 得到下图:



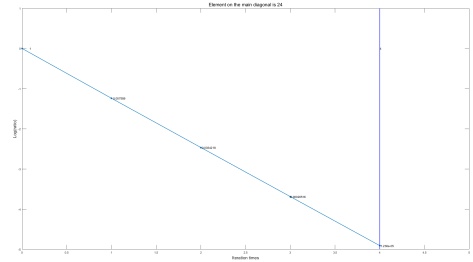
(a) element=3



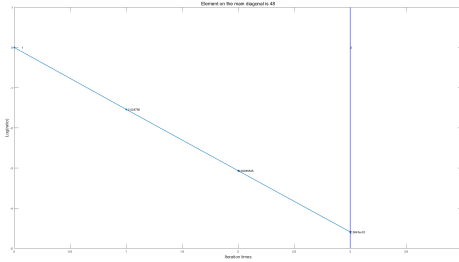
(b) element=6



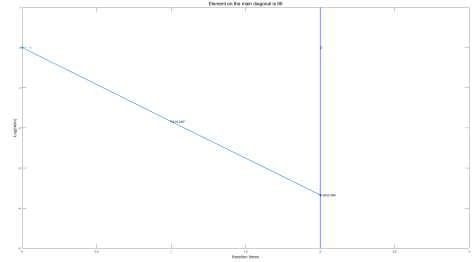
(c) element=12



(d) element=24



(e) element=48



(f) element=96

图 3: 主对角线元素与迭代次数图

观察图 3, 很明显主对角线元素越大, 雅克比迭代次数约少。原因可能是随着主对角线元素的增大, 矩阵 M 与 A 约接近, 导致迭代次数减小。

(3) 右端项和初始向量的取值与 (2) 保持一致, 给出不同 ω 下 G-S 迭代与 G-S(SOR) 的对比图, 以及 S-G-S 迭代与 S-G-S(SOR) 的对比图。

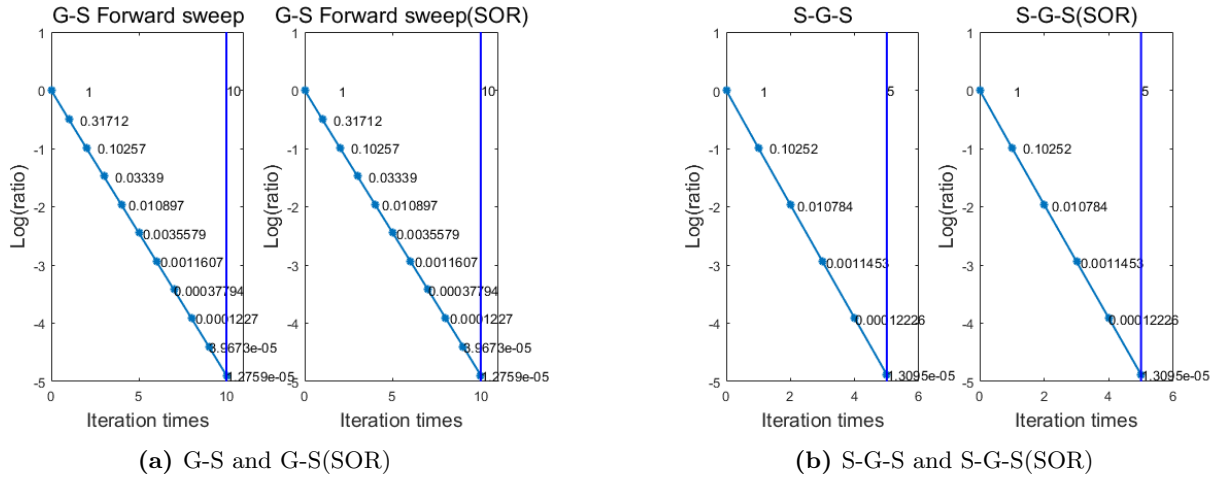


图 4: $\omega = 1$ 对比

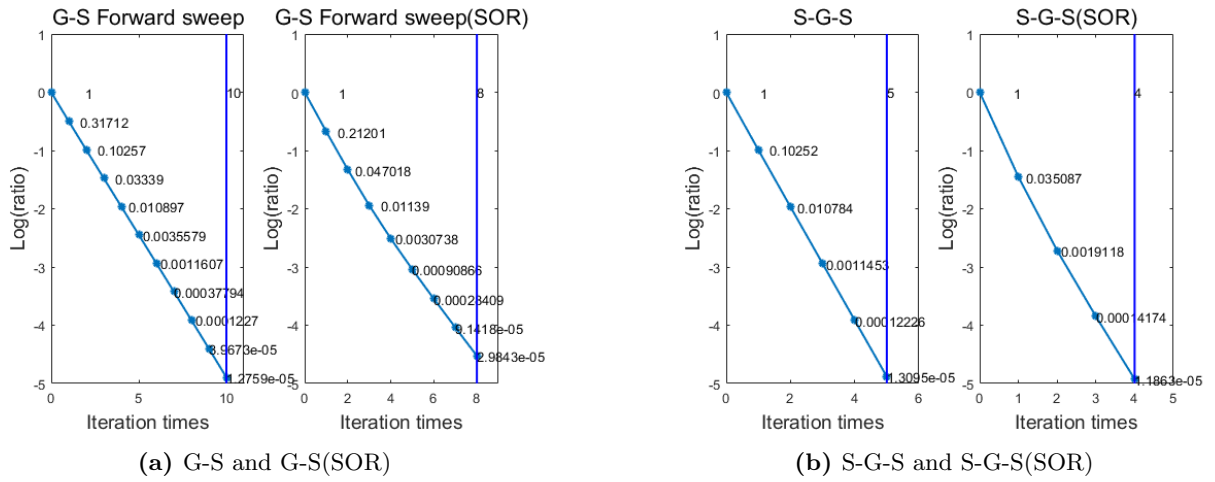


图 5: $\omega = 1.2$ 对比

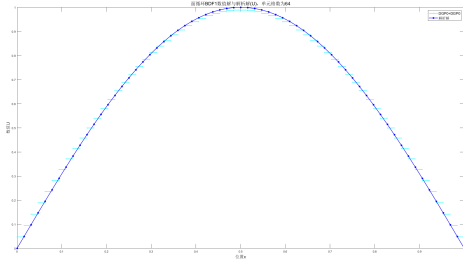
这里仅给出 $\omega = 1$ 与 $\omega = 1.2$ 的比较图, 改变松弛因子的数值, 最终发现当 $\omega = 1.2$ 左右时加速效果最好。

2. 对带源项的扩散方程 $u_t = u_{xx} + \pi^2 \sin(\pi x), x \in [0, 1], t \geq 0$, 满足以下初始条件 $u(x, 0) = x^2 - x$, 及边界条件 $u(0, t) = u(1, t) = 0$ 。

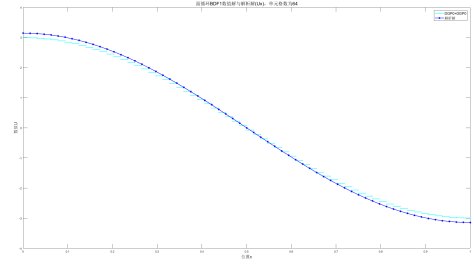
在练习 1 的基础上, 空间离散使用 DG(P0)+DG(P0) 格式, 时间离散格式使用 BDF1, 使用 LU-SGS 方法在均匀网格下进行求解。

解:

与课题组练习 1 类似, 仅改变迭代方法, 这里给出单元格数为 64, CFL=0.01 的情况下, U 与 U_x 的数值解与解析解的对比图, 以及 U 与 U_x 的空间精度分析。

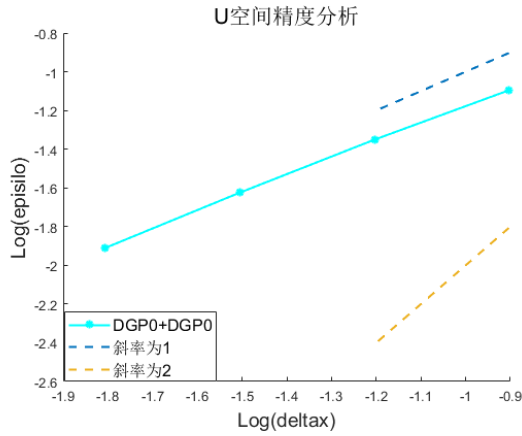


(a) U 的数值解与解析解

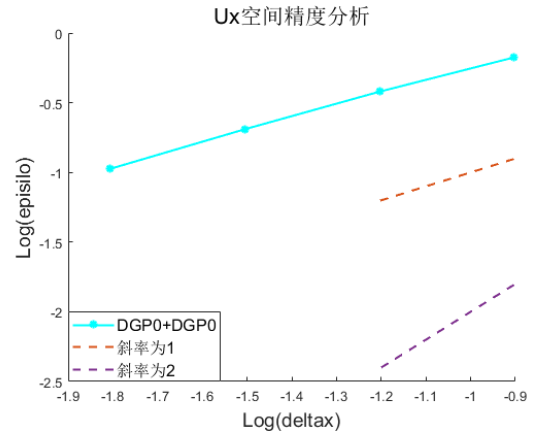


(b) U_x 的数值解与解析解

图 6: U 与 U_x 的数值解与解析解



(a) U 的空间精度分析



(b) U_x 的空间精度分析

图 7: U 与 U_x 的空间精度分析

二 附录 (代码)

JacobiIteration

```
1  clc
2  close all
3  clear all
4  %Pre-proceeding
5  D=3*eye(20);
6  L=sparse(2:20,1:19,-1/2,20,20)+sparse(3:20,1:18,-1/4,20,20);
7  U=L'; b=100*rand(20,1); X0=rand(20,1); Xold=zeros(20,1); Xnew=zeros(20,1);
8  tol=10^(-5); endtimes=100; epsilon=10^(-12);
9  ratioR=[0:endtimes; zeros(1, endtimes+1)];
10 %Proceeding
11 Xold=X0; resnorm0=sqrt(sum(((D+L+U)*X0-b).^2))+epsilon;
12 resnorm=sqrt(sum(((D+L+U)*Xold-b).^2));
13 ratio=resnorm/resnorm0; ratioR(2,1)=ratio;
14 for times=2:endtimes
15 Xnew=-D\((L+U)*Xold+D\b;
16 resnorm=sqrt(sum(((D+L+U)*Xnew-b).^2));
17 ratio=resnorm/resnorm0;
18 if ratio<tol
19 break
20 end
21 ratioR(2,times)=ratio;
22 Xold=Xnew;
23 end
24 ind=find(ratioR(2,:),1,'last');
25 ratioR(:,ind+1:endtimes+1)=[];
26 plot(ratioR(1,:),log10(ratioR(2,:)),'-*','linewidth',1.5)
27 xlabel('Iteration times','fontsize',14)
28 ylabel('Log(ratio)','fontsize',14)
29 title('The analysis of iteration times and lgratio','fontsize',16)
30 hold on
31 str1=num2str(ratioR(2,ind)); text(ratioR(1,ind),log10(ratioR(2,ind)),str1,'
    linewidth',1.5);
32 line([ratioR(1,ind),ratioR(1,ind)], [-5,1], 'color', 'b','linewidth'
    ,1.5);
33 str2=num2str(ind-1);
34 text(ind-1,0,str2,'linewidth',1.5);
35 xlim([0,ind])
```

Gauss-seidelIteration

```
1  clc
2  close all
3  clear all
4  %Pre-proceeding
5  n=20;
6  D=3*eye(n);
7  L=sparse(2:n,1:n-1,-1/2,n,n)+sparse(3:n,1:n-2,-1/4,n,n);
8  U=L';
9  A=D+L+U;
10 b=100*rand(n,1);b1=b;
11 X0=rand(n,1);XF=zeros(n,1);XB=zeros(n,1);
12 tol=10^(-5);endtimes=100;epsilon=10^(-12);
13 ratioRF=[0:endtimes;zeros(1,endtimes+1)];
14 ratioRB=[0:endtimes;zeros(1,endtimes+1)];
15 XF=X0;XB=X0;
16 resnorm0=sqrt(sum((A*X0-b).^2))+epsilon;
17 resnorm=sqrt(sum((A*XF-b).^2));
18 ratio=resnorm/resnorm0;ratioRF(2,1)=ratio;ratioRB(2,1)=ratio;
19
20 %Proceeding
21 %Forward sweep
22 for times=2:endtimes
23 %calculate X
24 for i=1:n
25 for j=1:i-1
26 b(i)=b(i)-A(i,j)*XF(j);
27 end
28 for j=i+1:n
29 b(i)=b(i)-A(i,j)*XF(j);
30 end
31 XF(i)=b(i)/A(i,i);
32 end
33 b=b1;
34 resnorm=sqrt(sum((A*XF-b).^2));
35 ratio=resnorm/resnorm0;
36 if ratio<tol
37 break
```

```

38 end
39 ratioRF(2,times)=ratio;
40 end
41 ind=find(ratioRF(2,:),1,'last');
42 ratioRF(:,ind+1:endtimes+1)=[];
43 subplot(1,2,1)
44 plot(ratioRF(1,:),log10(ratioRF(2,:)),'-*','linewidth',1.5)
45 xlabel('Iteration times','fontsize',14)
46 ylabel('Log(ratio)','fontsize',14)
47 title('G-S Forward sweep','fontsize',16)
48 hold on
49 str1=num2str(ratioRF(2,:));text(ratioRF(1,:),log10(ratioRF(2,:)),str1,'
    linewidth',1.5);
50 line([ratioRF(1,ind),ratioRF(1,ind)], [-5,1], 'color', 'b','linewidth'
    ,1.5);
51 str2=num2str(ind-1);
52 text(ind-1,0,str2,'linewidth',1.5);
53 xlim([0,ind])
54
55
56 %Backward sweep
57 for times=2:endtimes
58 %calculate X
59 for i=n:-1:1
60 for j=n:-1:i+1
61 b(i)=b(i)-A(i,j)*XB(j);
62 end
63 for j=i-1:-1:1
64 b(i)=b(i)-A(i,j)*XB(j);
65 end
66 XB(i)=b(i)/A(i,i);
67 end
68 b=b1;
69 resnorm=sqrt(sum((A*XB-b).^2));
70 ratio=resnorm/resnorm0;
71 if ratio<tol
72 break
73 end
74 ratioRB(2,times)=ratio;
75 end

```



```

76 ind=find(ratioRB(2,:),1,'last');
77 ratioRB(:,ind+1:endtimes+1)=[];
78 subplot(1,2,2)
79 plot(ratioRB(1,:),log10(ratioRB(2,:)),'-*','linewidth',1.5)
80 xlabel('Iteration times','fontsize',14)
81 ylabel('Log(ratio)','fontsize',14)
82 title('G-S Backward sweep','fontsize',16)
83 hold on
84 str1=num2str(ratioRB(2,:));text(ratioRB(1,:),log10(ratioRB(2,:)),str1,'
    linewidth',1.5);
85 line([ratioRB(1,ind),ratioRB(1,ind)],[-5,1],'color','b','linewidth'
    ,1.5);
86 str2=num2str(ind-1);
87 text(ind-1,0,str2,'linewidth',1.5);
88 xlim([0,ind])

```

S-G(SOR) and S-G-S(SOR)

```

1  clc
2  close all
3  clear all
4  %Pre-proceeding
5  n=20;
6  omega=1;%松弛因子
7  %构建矩阵
8  D=3*eye(n);
9  L=sparse(2:n,1:n-1,-1/2,n,n)+sparse(3:n,1:n-2,-1/4,n,n);
10 U=L';
11 A=D+L+U;
12 b=[1:n]';b1=b;
13 X0=zeros(n,1);XF=zeros(n,1);Xsgs=zeros(n,1);Xold=zeros(n,1);
14 %终止条件等
15 tol=10^(-5);endtimes=100;epsilon=10^(-12);
16 ratioRF=[0:endtimes;zeros(1,endtimes+1)];
17 ratioRsgs=[0:endtimes;zeros(1,endtimes+1)];
18 ratioRSORFGS=[0:endtimes;zeros(1,endtimes+1)];
19 ratioRSORSGS=[0:endtimes;zeros(1,endtimes+1)];
20 XF=X0;Xsgs=X0;Xsorfgs=X0;Xsorsgs=X0;
21 resnorm0=sqrt(sum((A*X0-b).^2))+epsilon;
22 resnorm=sqrt(sum((A*XF-b).^2));
23 ratio=resnorm/resnorm0;ratioRF(2,1)=ratio;ratioRsgs(2,1)=ratio;

```

```

ratioRSORFGS(2,1)=ratio;ratioRSORSGS(2,1)=ratio;
24
25 % Proceeding
26 %G-S-Forward sweep
27 for times=2:endtimes
28 %calculate X
29 for i=1:n
30 for j=1:i-1
31 b(i)=b(i)-A(i,j)*XF(j);
32 end
33 for j=i+1:n
34 b(i)=b(i)-A(i,j)*XF(j);
35 end
36 XF(i)=b(i)/A(i,i);
37 end
38 b=b1;
39 resnorm=sqrt(sum((A*XF-b).^2));
40 ratio=resnorm/resnorm0;
41 if ratio<tol
42 break
43 end
44 ratioRF(2,times)=ratio;
45 end
46 ind=find(ratioRF(2,:),1,'last');
47 ratioRF(:,ind+1:endtimes+1)=[];
48 subplot(1,2,1)
49 plot(ratioRF(1,:),log10(ratioRF(2,:)),'-*','linewidth',1.5)
50 xlabel('Iteration times','fontsize',14)
51 ylabel('Log(ratio)','fontsize',14)
52 title('G-S Forward sweep','fontsize',16)
53 hold on
54 str1=num2str(ratioRF(2,:));text(ratioRF(1,:),log10(ratioRF(2,:)),str1,'
    linewidth',1.5);
55 line([ratioRF(1,ind),ratioRF(1,ind)], [-5,1], 'color', 'b','linewidth'
    ,1.5);
56 str2=num2str(ind-1);
57 text(ind-1,0,str2,'linewidth',1.5);
58 xlim([0,ind])
59
60 %SOR-FGS

```

```

61 R=b1-A*X0;
62 for times=2:endtimes
63 %calculate deltaX
64 Xold=Xsorfgs;
65 ie=1;
66 Xsorfgs(ie)=R(ie)/(D(ie,ie)/omega);
67 ie=2;
68 R(ie)=R(ie)-Xsorfgs(ie-1)*(L(ie,ie-1)/omega);
69 Xsorfgs(ie)=R(ie)/(D(ie,ie)/omega);
70 for ie=3:n
71 R(ie)=R(ie)-Xsorfgs(ie-1)*(L(ie,ie-1)/omega)-Xsorfgs(ie-2)*(L(ie,ie-2)/
    omega);
72 Xsorfgs(ie)=R(ie)/(D(ie,ie)/omega);
73 end
74 Xsorfgs=Xsorfgs+Xold;
75 resnorm=sqrt(sum((A*Xsorfgs-b1).^2));
76 ratio=resnorm/resnorm0;
77 if ratio<tol
78 break
79 end
80 R=b1-A*Xsorfgs;
81 ratioRSORFGS(2,times)=ratio;
82 end
83 ind=find(ratioRSORFGS(2,:),1,'last');
84 ratioRSORFGS(:,ind+1:endtimes+1)=[];
85 subplot(1,2,2)
86 plot(ratioRSORFGS(1,:),log10(ratioRSORFGS(2,:)),'-*','linewidth',1.5)
87 xlabel('Iteration times','fontsize',14)
88 ylabel('Log(ratio)','fontsize',14)
89 title('G-S Forward sweep(SOR)','fontsize',16)
90 hold on
91 str1=num2str(ratioRSORFGS(2,:));text(ratioRSORFGS(1,:),log10(
    ratioRSORFGS(2,:)),str1,'linewidth',1.5);
92 line([ratioRSORFGS(1,ind),ratioRSORFGS(1,ind)],[-5,1],'color','b','
    linewidth',1.5);
93 str2=num2str(ind-1);
94 text(ind-1,0,str2,'linewidth',1.5);
95 xlim([0,ind])
96
97 %SGS

```

```

98  for times=2:endtimes
99  %calculate X Forward sweep
100  for i=1:n
101  for j=1:i-1
102  b(i)=b(i)-A(i,j)*Xsgs(j);
103  end
104  for j=i+1:n
105  b(i)=b(i)-A(i,j)*Xsgs(j);
106  end
107  Xsgs(i)=b(i)/A(i,i);
108  end
109  b=b1;
110  %Backward sweep
111  for i=n:-1:1
112  for j=n:-1:i+1
113  b(i)=b(i)-A(i,j)*Xsgs(j);
114  end
115  for j=i-1:-1:1
116  b(i)=b(i)-A(i,j)*Xsgs(j);
117  end
118  Xsgs(i)=b(i)/A(i,i);
119  end
120  b=b1;
121  resnorm=sqrt(sum((A*Xsgs-b).^2));
122  ratio=resnorm/resnorm0;
123  if ratio<tol
124  break
125  end
126  ratioRsgs(2,times)=ratio;
127  end
128  ind=find(ratioRsgs(2,:),1,'last');
129  ratioRsgs(:,ind+1:endtimes+1)=[];
130  figure
131  subplot(1,2,1)
132  plot(ratioRsgs(1,:),log10(ratioRsgs(2,:)),'-*','linewidth',1.5)
133  xlabel('Iteration times','fontsize',14)
134  ylabel('Log(ratio)','fontsize',14)
135  title('S-G-S','fontsize',16)
136  hold on
137  str1=num2str(ratioRsgs(2,:));text(ratioRsgs(1,:),log10(ratioRsgs(2,:)),

```

```

    str1,'linewidth',1.5);
138 line([ratioRsgs(1,ind),ratioRsgs(1,ind)], [-5,1], 'color', 'b','
    linewidth',1.5);
139 str2=num2str(ind-1);
140 text(ind-1,0,str2,'linewidth',1.5);
141 xlim([0,ind])
142 hold off
143
144
145 %SOR-SGS
146 R=omega*(2-omega)*(b1-A*X0);
147 for times=2:endtimes
148 %calculate deltaX
149 %Forward sweep
150 Xold=Xsorgsgs;
151 ie=1;
152 Xsorgsgs(ie)=R(ie)/D(ie,ie);
153 ie=2;
154 R(ie)=R(ie)-Xsorgsgs(ie-1)*(L(ie,ie-1)*omega);
155 Xsorgsgs(ie)=R(ie)/D(ie,ie);
156 for ie=3:n
157 R(ie)=R(ie)-Xsorgsgs(ie-1)*(L(ie,ie-1)*omega)-Xsorgsgs(ie-2)*(L(ie,ie-2)*
    omega);
158 Xsorgsgs(ie)=R(ie)/D(ie,ie);
159 end
160 %Backward sweep
161 R=D*Xsorgsgs;
162 ie=n;
163 Xsorgsgs(ie)=R(ie)/D(ie,ie);
164 ie=n-1;
165 R(ie)=R(ie)-Xsorgsgs(ie+1)*(U(ie,ie+1)*omega);
166 Xsorgsgs(ie)=R(ie)/D(ie,ie);
167 for ie=n-2:-1:1
168 R(ie)=R(ie)-Xsorgsgs(ie+1)*(U(ie,ie+1)*omega)-Xsorgsgs(ie+2)*(U(ie,ie+2)*
    omega);
169 Xsorgsgs(ie)=R(ie)/D(ie,ie);
170 end
171 %此时 Xsorgsgs 里面存储的是 deltaX
172 Xsorgsgs=Xsorgsgs+Xold;
173 resnorm=sqrt(sum((A*Xsorgsgs-b1).^2));

```

```

174 ratio=resnorm/resnorm0;
175 if ratio<tol
176 break
177 end
178 R=omega*(2-omega)*(b1-A*Xsorsgs);
179 ratioRSORSGS(2,times)=ratio;
180 end
181 ind=find(ratioRSORSGS(2,:),1,'last');
182 ratioRSORSGS(:,ind+1:endtimes+1)=[];
183 subplot(1,2,2)
184 plot(ratioRSORSGS(1,:),log10(ratioRSORSGS(2,:)),'-*','linewidth',1.5)
185 xlabel('Iteration times','fontsize',14)
186 ylabel('Log(ratio)','fontsize',14)
187 title('S-G-S(SOR)','fontsize',16)
188 hold on
189 str1=num2str(ratioRSORSGS(2,:));text(ratioRSORSGS(1,:),log10(
    ratioRSORSGS(2,:)),str1,'linewidth',1.5);
190 line([ratioRSORSGS(1,ind),ratioRSORSGS(1,ind)],[-5,1],'color','b','
    linewidth',1.5);
191 str2=num2str(ind-1);
192 text(ind-1,0,str2,'linewidth',1.5);
193 xlim([0,ind])

```

subDG(P0)+DG(P0) L-U Iteration

```

1 function [Unumsolution,n]=subDGP0plusDGP0(Unit,CFL,endtau)
2 %Some basic paramater
3 endx=1;deltax=endx/Unit;numberx=endx/deltax+1;
4 tol=10^(-10);
5 nu=1;Lr=1/(2*pi);Tr=Lr^2/nu;
6 abslambda=sqrt(nu/Tr);deltatau=CFL*deltax/abslambda;%伪时间变量
7 B1=1;
8 C=[B1,0;0,B1/deltax];Mtau=[deltax,0;0,1/deltax];%此为推导出的 U=CV 中的 C
9 A=[abslambda,0;0,abslambda];epsilon=10^(-12);
10 R=zeros(2*Unit,1);
11 Rd=zeros(2,numberx-1);
12 Rb=zeros(2,numberx-1);
13 Fn=zeros(2,numberx);
14 X=zeros(2*Unit,1);
15
16

```

```

17 %构建 LHS
18 %Mtau/deltatau
19 LHS1=sparse(1:2:2*Unit-1,1:2:2*Unit-1,deltax/deltatau,2*Unit,2*Unit);
20 LHS1=LHS1+sparse(2:2:2*Unit,2:2:2*Unit,1/(deltax*deltatau),2*Unit,2*Unit
    );
21 %Rdomain
22 LHS2=-sparse(2:2:2*Unit,2:2:2*Unit,-1/(Tr*deltax),2*Unit,2*Unit);
23 %Rboundary
24 LHS3=zeros(2*Unit,2*Unit);
25 for iface=2:numberx-1
26     ieL=iface-1;
27     ieR=iface;
28     %diag
29     LHS3(2*ieL-1:2*ieL,2*ieL-1:2*ieL)=LHS3(2*ieL-1:2*ieL,2*ieL-1:2*ieL)+C'*[
        abslambda/2,-nu/(2*deltax);-1/(2*Tr),abslambda/(2*deltax)];
30     LHS3(2*ieR-1:2*ieR,2*ieR-1:2*ieR)=LHS3(2*ieR-1:2*ieR,2*ieR-1:2*ieR)-C
        '*[-abslambda/2,-nu/(2*deltax);-1/(2*Tr),-abslambda/(2*deltax)];
31     %upper
32     LHS3(2*ieL-1:2*ieL,2*ieR-1:2*ieR)=LHS3(2*ieL-1:2*ieL,2*ieR-1:2*ieR)+C
        '*[-abslambda/2,-nu/(2*deltax);-1/(2*Tr),-abslambda/(2*deltax)];
33     %lower
34     LHS3(2*ieR-1:2*ieR,2*ieL-1:2*ieL)=LHS3(2*ieR-1:2*ieR,2*ieL-1:2*ieL)-C'*[
        abslambda/2,-nu/(2*deltax);-1/(2*Tr),abslambda/(2*deltax)];
35 end
36 LHS3(2*1-1:2*1,2*1-1:2*1)=LHS3(2*1-1:2*1,2*1-1:2*1)-C'*([abslambda/2,-nu
    /2;-1/(2*Tr),abslambda/2]*[0,0;0,1]+[-abslambda/2,-nu/2;-1/(2*Tr),-
    abslambda/2])*C;
37 LHS3(2*(numberx-1)-1:2*(numberx-1),2*(numberx-1)-1:2*(numberx-1))=LHS3
    (2*(numberx-1)-1:2*(numberx-1),2*(numberx-1)-1:2*(numberx-1))+C'*([
    abslambda/2,-nu/2;-1/(2*Tr),abslambda/2]+[-abslambda/2,-nu/2;-1/(2*Tr
    ),-abslambda/2]*[0,0;0,1])*C;
38
39 LHS=LHS1+LHS2+LHS3;
40
41 %取出我们所需要的 D
42 D=zeros(2*Unit,2*Unit);
43 for iface=2:numberx
44     ieL=iface-1;
45     D(2*ieL-1:2*ieL,2*ieL-1:2*ieL)=LHS(2*ieL-1:2*ieL,2*ieL-1:2*ieL);
46 end

```

```

47 %取出我们所需要的 L
48 L=zeros(2*Unit,2*Unit);
49 for iface=2:numberx-1
50 ieR=iface;
51 ieL=iface-1;
52 L(2*ieR-1:2*ieR,2*ieL-1:2*ieL)=LHS(2*ieR-1:2*ieR,2*ieL-1:2*ieL);
53 end
54
55 %取出我们所需要的 U
56 U=zeros(2*Unit,2*Unit);
57 for iface=2:numberx-1
58 ieR=iface;
59 ieL=iface-1;
60 U(2*ieL-1:2*ieL,2*ieR-1:2*ieR)=LHS(2*ieL-1:2*ieL,2*ieR-1:2*ieR);
61 end
62
63
64 %为循环所预设的一些量
65
66 Ucurrent=zeros(2,numberx-1);
67 Unext=zeros(2*Unit,1);
68 %initial condition set up
69 x=0;
70
71 for k=1:numberx-1
72 Ucurrent(1,k)=(x+deltax/2)^2-(x+deltax/2);
73 Ucurrent(2,k)=(2*(x+deltax/2)-1)*deltax;
74 x=x+deltax;
75 end
76
77 %Rdomain
78 x=0;
79 for k=1:numberx-1
80 Rd(1,k)=pi*(cos(pi*x)-cos(pi*(x+deltax)));
81 Rd(2,k)=-Ucurrent(2,k)/(Tr*deltax);
82 x=x+deltax;
83 end
84 %Rboundary
85 for iface=2:numberx-1
86 ieL=iface-1;

```



```

87 ieR=iface;
88 Fn(:,iface)=0.5*([-nu*Ucurrent(2,ieL)/deltax;-Ucurrent(1,ieL)/Tr]+[-nu*
    Ucurrent(2,ieR)/deltax;-Ucurrent(1,ieR)/Tr])-0.5*A*([Ucurrent(1,ieR);
    Ucurrent(2,ieR)/deltax]-[Ucurrent(1,ieL);Ucurrent(2,ieL)/deltax]);
89 Rb(:,ieL)=Rb(:,ieL)-C'*Fn(:,iface);
90 Rb(:,ieR)=Rb(:,ieR)+C'*Fn(:,iface);
91 end
92 Fn(:,1)=0.5*([-nu*Ucurrent(2,1)/deltax;0]+[-nu*Ucurrent(2,1)/deltax;-
    Ucurrent(1,1)/Tr])-0.5*A*([Ucurrent(1,1);Ucurrent(2,1)/deltax]-[0;
    Ucurrent(2,1)/deltax]);
93 Fn(:,numberx)=0.5*([-nu*Ucurrent(2,numberx-1)/deltax;-Ucurrent(1,numberx
    -1)/Tr]+[-nu*Ucurrent(2,numberx-1)/deltax;0])-0.5*A*([0;Ucurrent(2,
    numberx-1)/deltax]-[Ucurrent(1,numberx-1);Ucurrent(2,numberx-1)/
    deltax]);
94 Rb(:,1)=Rb(:,1)+C'*Fn(:,1);
95 Rb(:,numberx-1)=Rb(:,numberx-1)-C'*Fn(:,numberx);
96
97 %R 组装
98 for k=1:numberx-1
99 R(2*k-1:2*k,1)=Rd(:,k)+Rb(:,k);
100 end
101 %进行必要的向量等价转变
102 for k=1:numberx-1
103 Unext(2*k-1:2*k,1)=Ucurrent(:,k);
104 end
105
106 %循环迭代
107 for n=deltatau:deltatau:endtau
108 X=0;
109 b=R;
110 %Forward sweep
111 ie=1;
112 X(ie:ie+1,1)=D(ie:ie+1,ie:ie+1)\R(ie:ie+1,1);
113 for ie=2:numberx-1
114 R(2*ie-1:2*ie,1)=R(2*ie-1:2*ie,1)-L(2*ie-1:2*ie,2*(ie-1)-1:2*(ie-1))*X
    (2*(ie-1)-1:2*(ie-1),1);
115 X(2*ie-1:2*ie,1)=D(2*ie-1:2*ie,2*ie-1:2*ie)\R(2*ie-1:2*ie,1);
116 end
117 %Backward sweep
118 for ie=1:numberx-1

```

```

119 R(2*ie-1:2*ie,1)=D(2*ie-1:2*ie,2*ie-1:2*ie)*X(2*ie-1:2*ie,1);
120 end
121
122 ie=numberx-1;
123 X(2*ie-1:2*ie)=D(2*ie-1:2*ie,2*ie-1:2*ie)\R(2*ie-1:2*ie,1);
124 for ie=numberx-2:-1:1
125 R(2*ie-1:2*ie,1)=R(2*ie-1:2*ie,1)-U(2*ie-1:2*ie,2*(ie+1)-1:2*(ie+1))*X
    (2*(ie+1)-1:2*(ie+1),1);
126 X(2*ie-1:2*ie,1)=D(2*ie-1:2*ie,2*ie-1:2*ie)\R(2*ie-1:2*ie,1);
127 end
128 if max(X)<tol
129 break
130 end
131 Unext=Unext+X;
132 Rd=zeros(2,numberx-1);
133 Rb=zeros(2,numberx-1);
134 for k=1:numberx-1
135 Ucurrent(:,k)=Unext(2*k-1:2*k,1);
136 end
137 %Rdomain
138 x=0;
139 for k=1:numberx-1
140 Rd(1,k)=pi*(cos(pi*x)-cos(pi*(x+deltax)));
141 Rd(2,k)=-Ucurrent(2,k)/(Tr*deltax);
142 x=x+deltax;
143 end
144 %Rboundary
145 for iface=2:numberx-1
146 ieL=iface-1;
147 ieR=iface;
148 Fn(:,iface)=0.5*([-nu*Ucurrent(2,ieL)/deltax;-Ucurrent(1,ieL)/Tr]+[-nu*
    Ucurrent(2,ieR)/deltax;-Ucurrent(1,ieR)/Tr])-0.5*A*([Ucurrent(1,ieR);
    Ucurrent(2,ieR)/deltax]-[Ucurrent(1,ieL);Ucurrent(2,ieL)/deltax]);
149 Rb(:,ieL)=Rb(:,ieL)-C'*Fn(:,iface);
150 Rb(:,ieR)=Rb(:,ieR)+C'*Fn(:,iface);
151 end
152
153 Fn(:,1)=0.5*([-nu*Ucurrent(2,1)/deltax;0]+[-nu*Ucurrent(2,1)/deltax;-
    Ucurrent(1,1)/Tr])-0.5*A*([Ucurrent(1,1);Ucurrent(2,1)/deltax]-[0;
    Ucurrent(2,1)/deltax]);

```

```

154 Fn(:, numberx)=0.5*([-nu*Ucurrent(2, numberx-1)/deltax;-Ucurrent(1, numberx
    -1)/Tr]+[-nu*Ucurrent(2, numberx-1)/deltax;0]) -0.5*A*([0; Ucurrent(2,
    numberx-1)/deltax]-[Ucurrent(1, numberx-1); Ucurrent(2, numberx-1)/
    deltax]);
155 Rb(:, 1)=Rb(:, 1)+C'*Fn(:, 1);
156 Rb(:, numberx-1)=Rb(:, numberx-1)-C'*Fn(:, numberx);
157
158 %R 组装
159 for k=1:numberx-1
160 R(2*k-1:2*k, 1)=Rd(:, k)+Rb(:, k);
161 end
162
163 for k=1:numberx-1
164 Unext(2*k-1:2*k, 1)=Ucurrent(:, k);
165 end
166 end
167 Unumsolution(1, :)=Ucurrent(1, :); Unumsolution(2, :)=Ucurrent(2, :)/deltax;
168 end

```