

课题组组会-练习 3

王程

2023 年 10 月 24 日

一 练习及结果

1. 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 $u(0, t) = u(1, t) = 0$.

(1) 将空间离散格式改为 DG(P0P1)+DG(P0), 时间离散方式使用 a) 显式欧拉格式, b) TVD-RK3, c) BDF1 在均匀网格下进行求解。

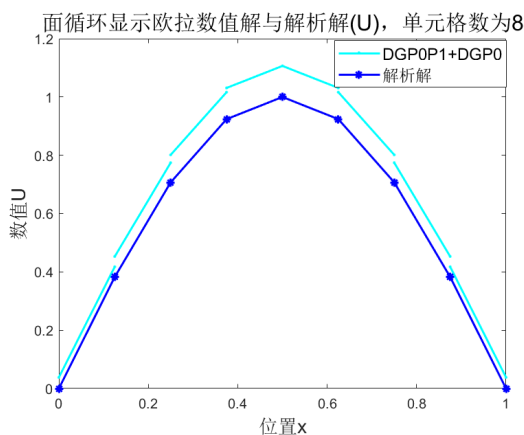
(2) 在网格生成时, 对内部点坐标进行随机扰动, 扰动范围为 $\pm 5\% \Delta x_{uniform}$, 重新对问题进行求解, 并测试精度。

2 在 $x \in [0, 1]$ 的均匀网格上尝试使用 Green-Gauss Reconstruction 对 $f(x)$ 及 $g(x)$ 进行 P1P2 重构, 其中 $f(x) = 1 + x + x^2, g(x) = \sin(\pi x)$, 测试重构精度。如果网格为不均匀网格呢?

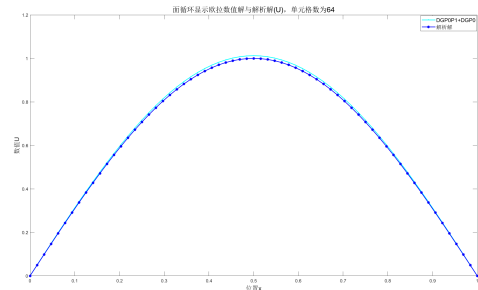
解:

(1) 更改空间离散格式, 下面给出三种时间离散格式下 (单元格数为 8, 64) 的方程数值解与解析解对比图以及空间精度。

A) Explicit Euler



(a) 8 elements



(b) 64 elements

图 1: Explicit Euler U

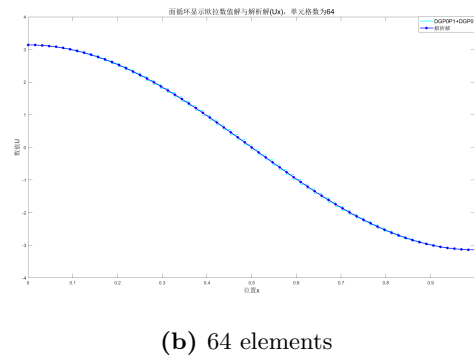
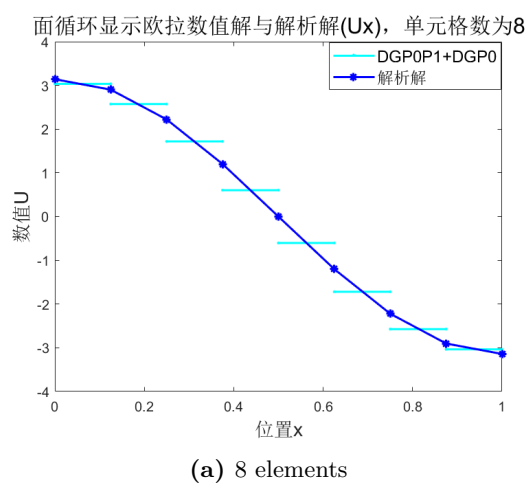


图 2: Explicit Euler U_x

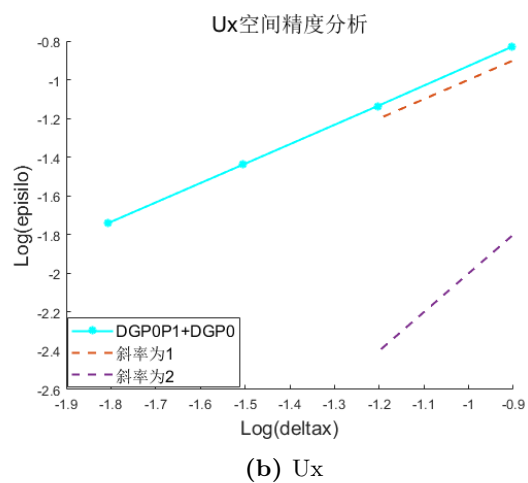
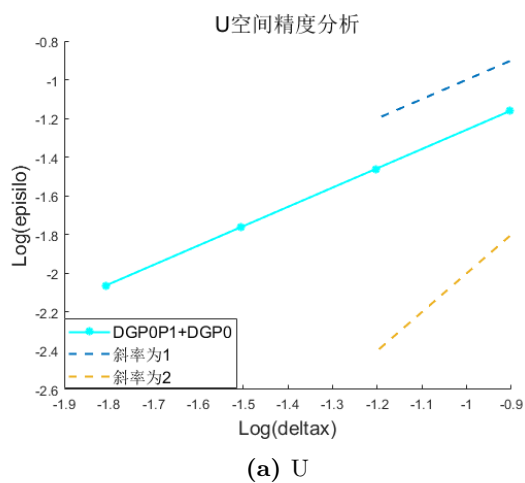


图 3: 空间精度

B) TVD-RK3

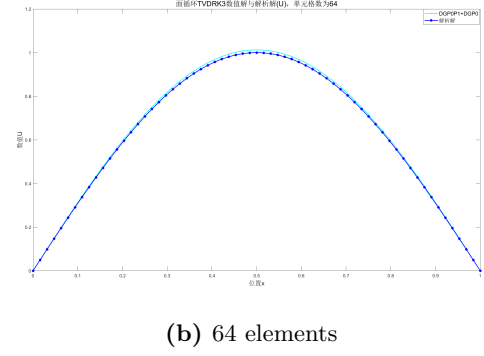
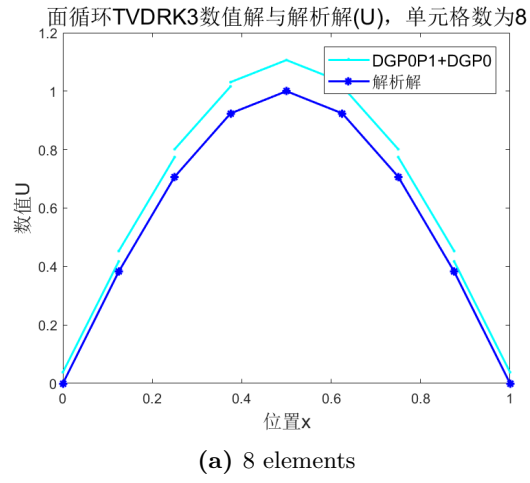


图 4: TVD-RK3 U

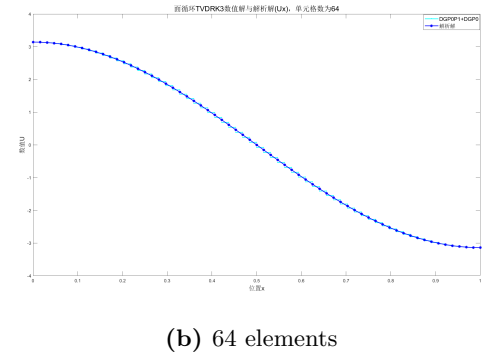
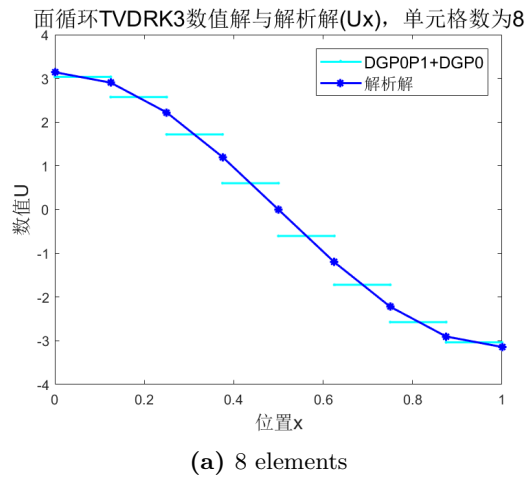


图 5: TVD-RK3 U_x

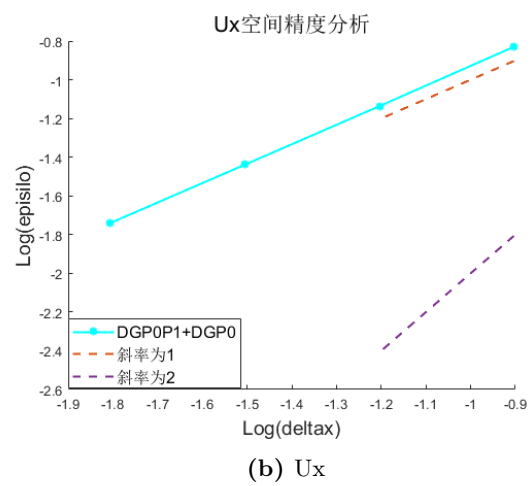
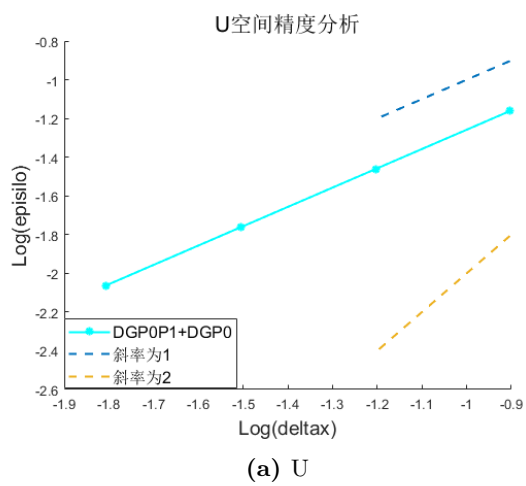


图 6: 空间精度

C)BDF1

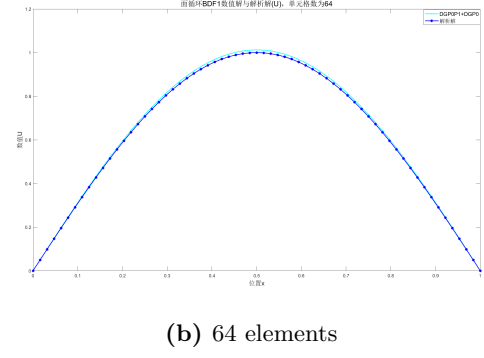
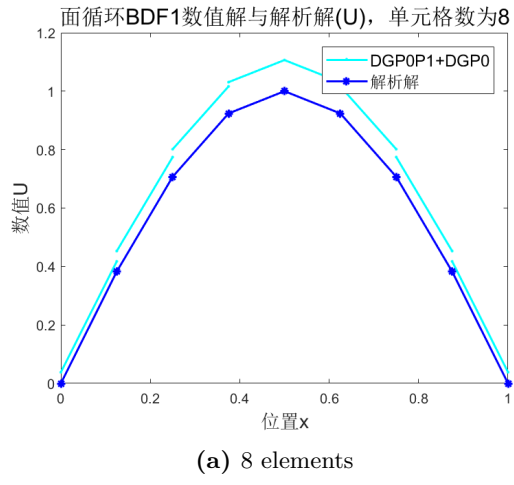


图 7: BDF1 U

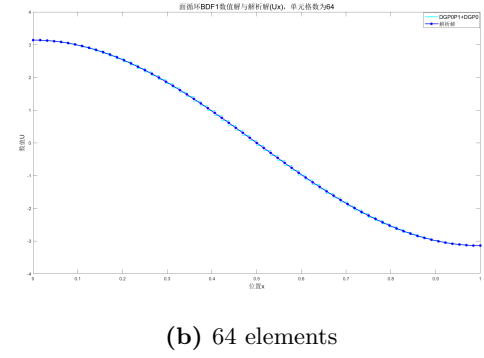
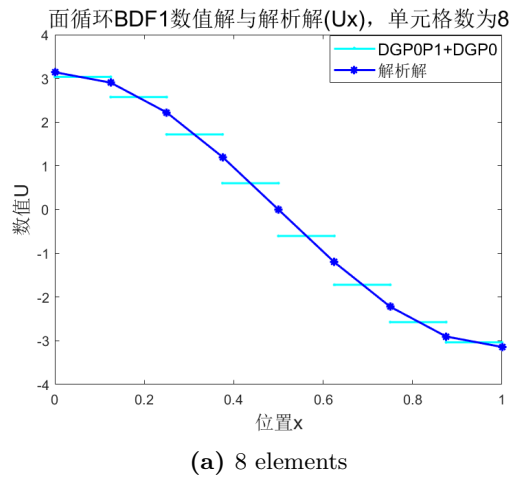


图 8: BDF1 Ux

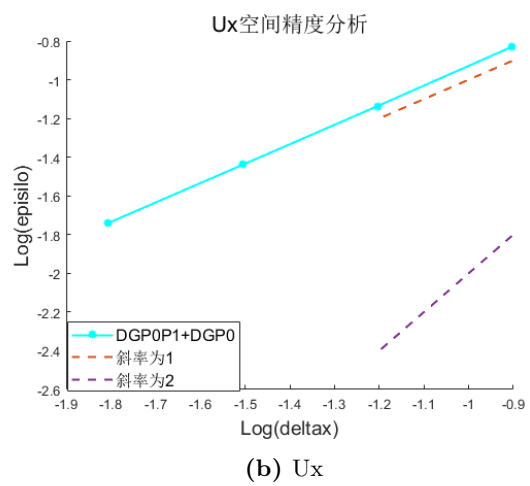
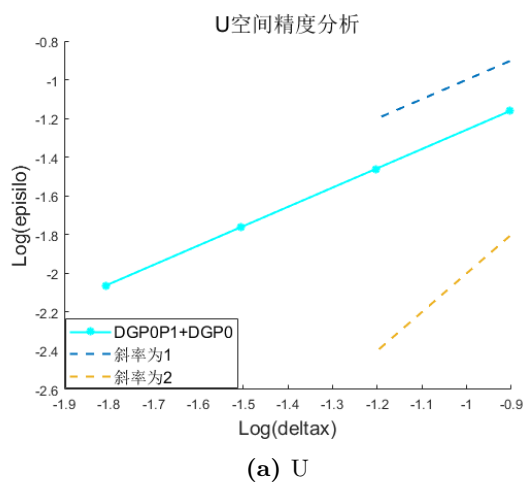


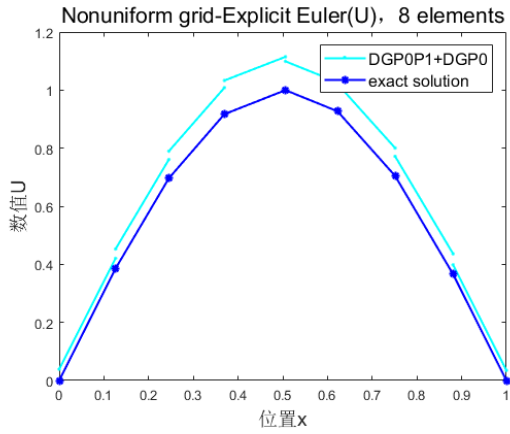
图 9: 空间精度

解：

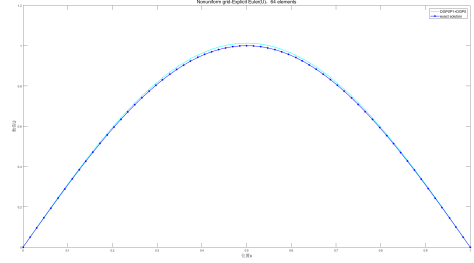
(2) 生成网格时对网格点进行扰动，这会影响 Δx ，进而会影响到每个单元上的 $\Delta \tau$ ，这里采用了两种向前推进方法：a) 取所有单元的最小 $\Delta \tau$ 进行推进 (如下面的 TVD-RK3)，b) 每个单元取自己对应的 $\Delta \tau$ ，最后终止的 $End\tau$ 取最大的那个 (如下面的 Obvious Euler 和 BDF1)。

以下展示非均匀网格下三种时间离散格式下 (单元格数为 8, 64) 的方程数值解与解析解对比图以及空间精度 (这里仍然采取的均匀网格的 Δx)，结果显示两种推进方式得到的 U 的精度相似，但方法 b) 得到的 U_x 精度较好。

A) Nonuniform grid explicit euler

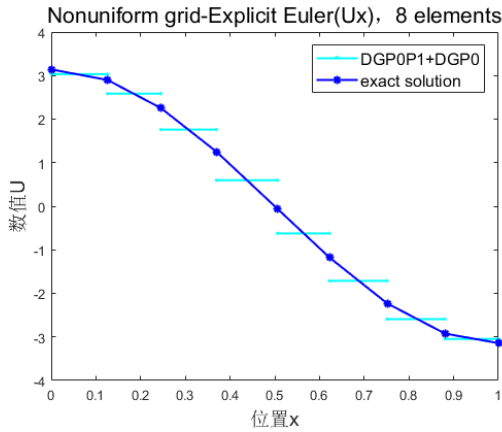


(a) 8 elements

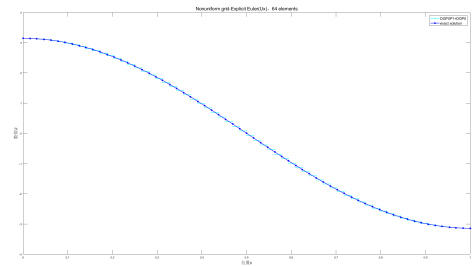


(b) 64 elements

图 10: Nonuniform grid Explicit euler U

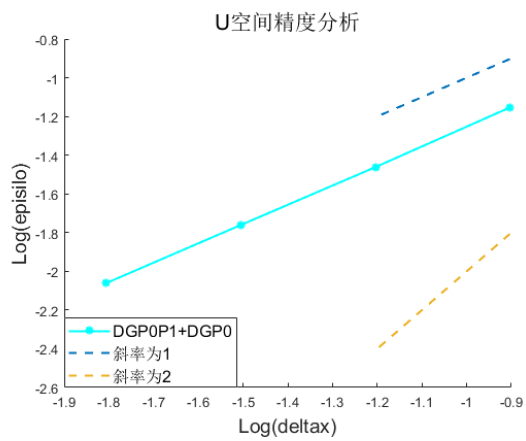


(a) 8 elements

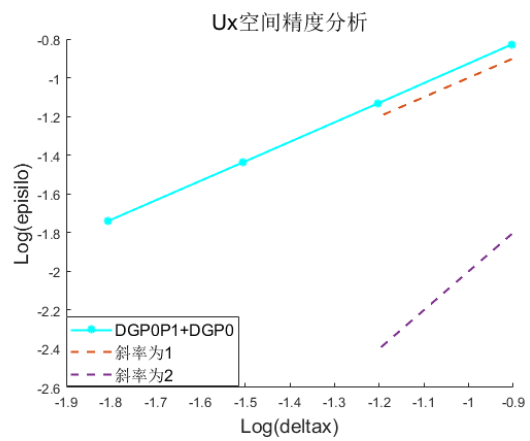


(b) 64 elements

图 11: Nonuniform grid Explicit euler Ux



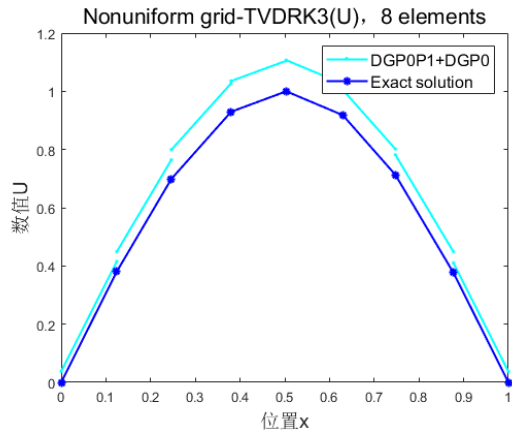
(a) U



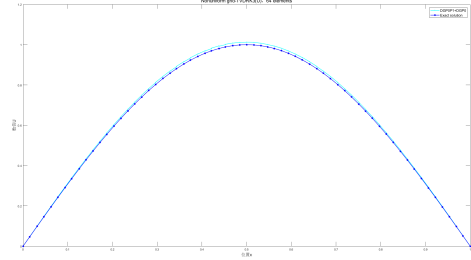
(b) Ux

图 12: 空间精度

B) Nonuniform grid TVDRK3

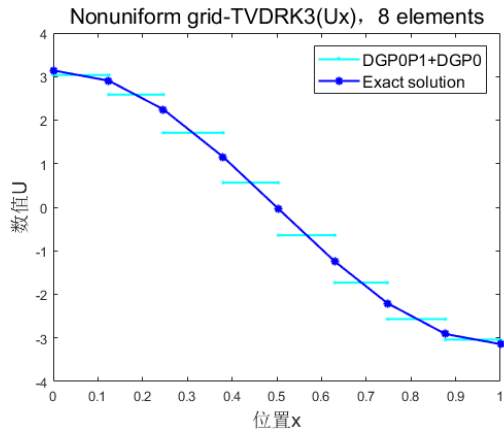


(a) 8 elements

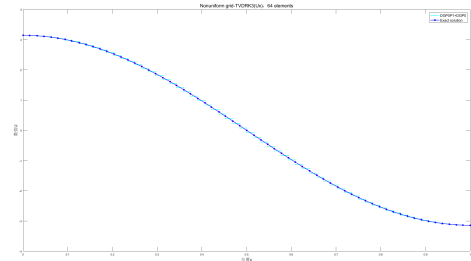


(b) 64 elements

图 13: Nonuniform grid TVDRK3 U



(a) 8 elements



(b) 64 elements

图 14: Nonuniform grid TVDRK3 Ux

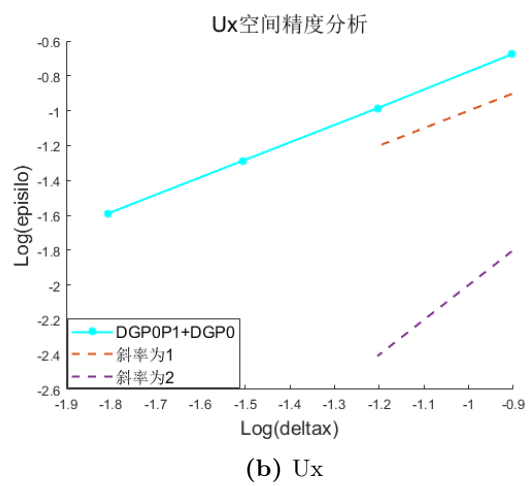
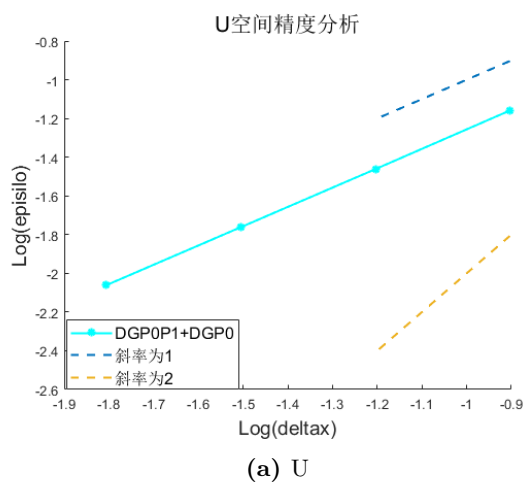
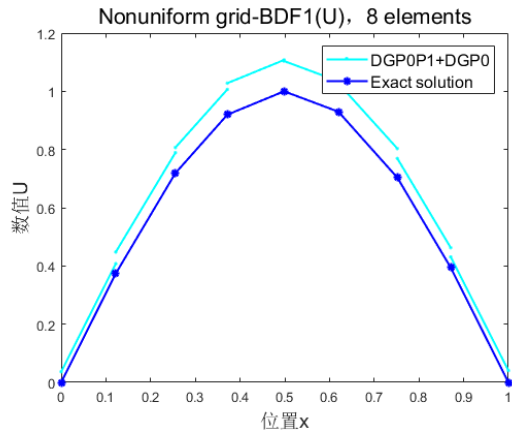
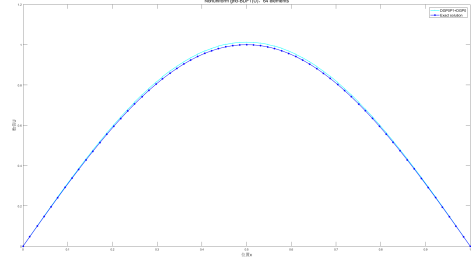


图 15: 空间精度

C) Nonuniform grid BDF1

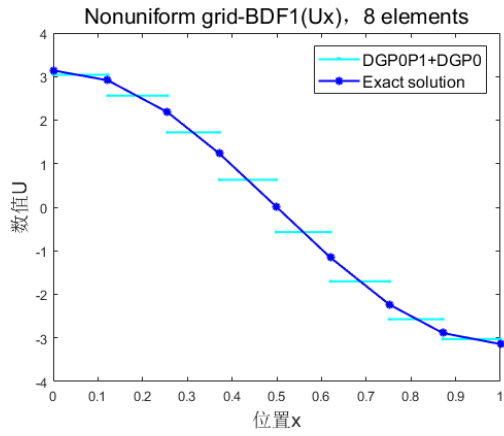


(a) 8 elements

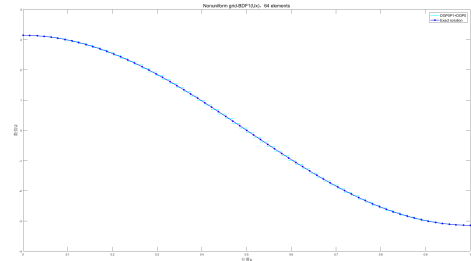


(b) 64 elements

图 16: Nonuniform grid BDF1 U



(a) 8 elements



(b) 64 elements

图 17: Nonuniform grid BDF1 Ux

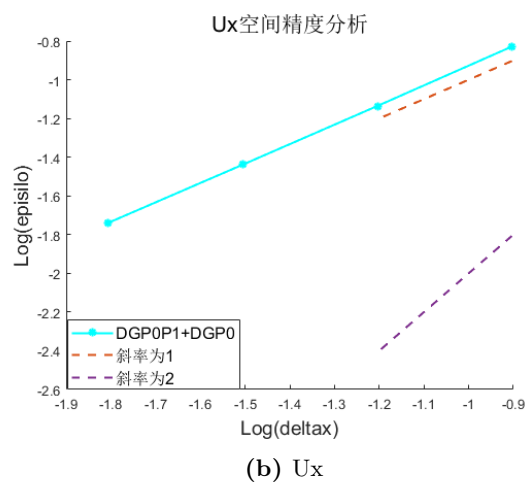
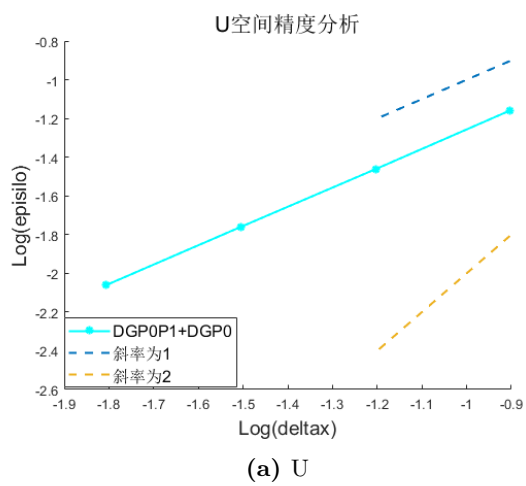
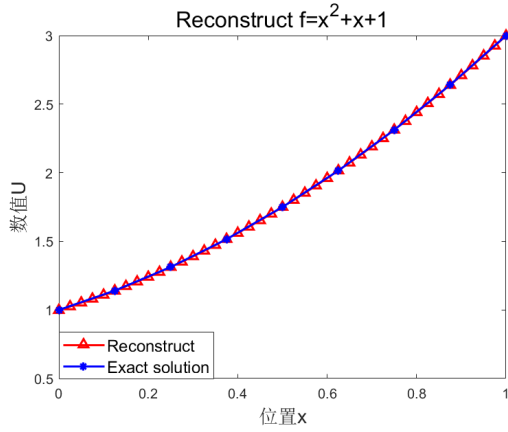


图 18: 空间精度

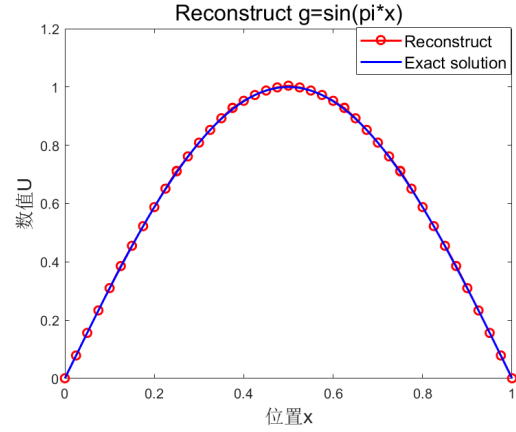
解:

2. 以下对 $f(x), g(x)$ 进行均匀网格与非均匀网格下的 P1P2 Green-Gauss Reconstruction, 并给出非均匀网格下的重构精度图 (仍然采取均匀网格的 Δx).

A) Uniform grid



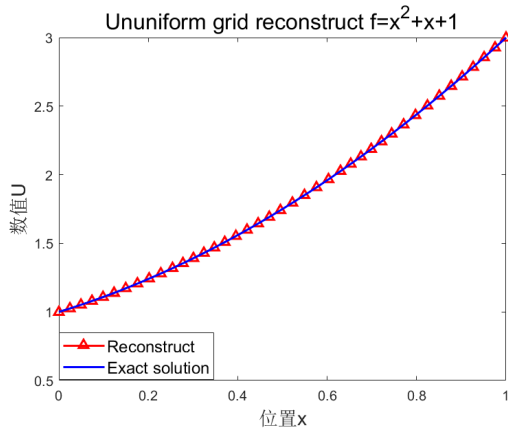
(a) f 重构



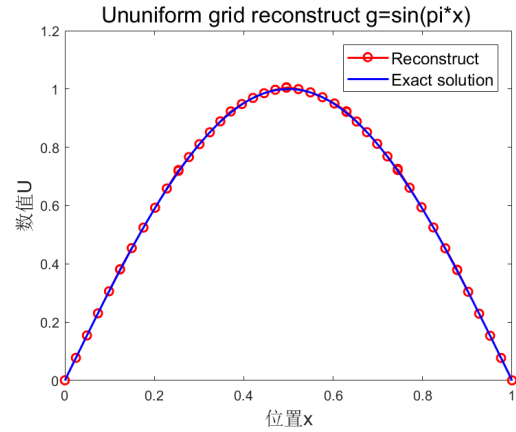
(b) g 重构

图 19: Uniform grid Green gauss Reconstruction

B) Nonuniform grid



(a) f 重构



(b) g 重构

图 20: Nonuniform grid Green gauss Reconstruction

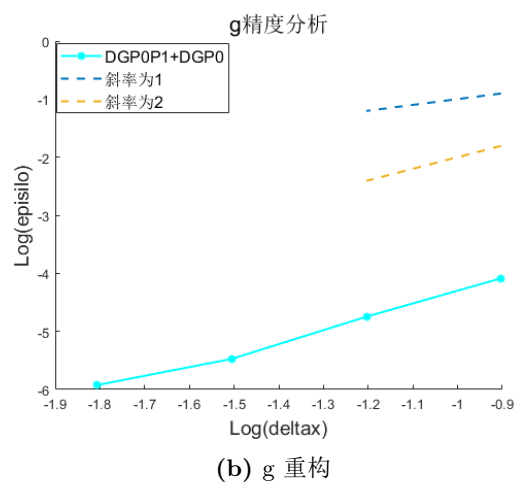
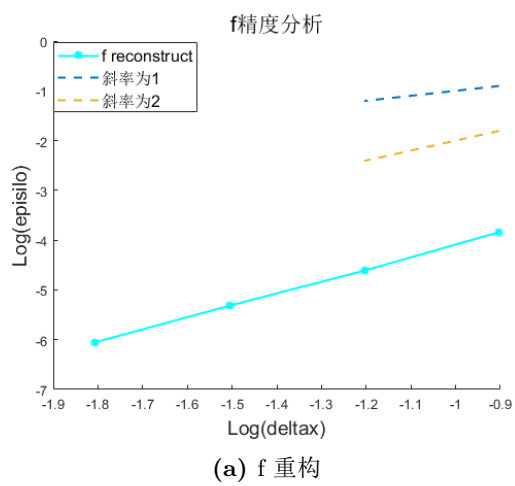


图 21: 精度分析

二 附录 (代码, 仅展示部分)

Nonuniform grid BDF1

```
1  clc
2  clear all
3  close all
4  % Pre-processing
5  Unit=64;CFL=0.01;endtau=2;endx=1;deltax=1/Unit;numberx=endx/deltax+1;
6  %记录内点位置
7  Uexasolution=zeros(2,numberx);
8  UDGP0P1plusDGP0=zeros(2,numberx-1);
9  Unumsolution1=zeros(1,2);
10 Unumsolution2=zeros(2,numberx-1);
11 Acc=zeros(3,4);a1=[1/8,1/16,1/32,1/64];a2=[1/8,1/16];
12 %solve the question
13 %solve the numsolution
14 [UDGP0P1plusDGP0,Grid,endtau01plus0]=BDF1subDGP0P1plusDGP0(Unit,CFL,
    endtau);%acquire u and
    ux
15
16 %solve the exasolution
17 x=0;
18 for k=1:numberx
19 x=Grid(1,k);
20 Uexasolution(1,k)=sin(pi*x);
21 Uexasolution(2,k)=pi*cos(pi*x);
22 end
23 %post-processing
24 %plot the u
25 %plot the DGP0P1plusDGP0
26 figure
27 Unumsolution1(1,1)=UDGP0P1plusDGP0(1,1)-UDGP0P1plusDGP0(2,1)*(Grid(2)-
    Grid(1))/2;Unumsolution1(1,2)=UDGP0P1plusDGP0(1,1)+UDGP0P1plusDGP0
    (2,1)*(Grid(2)-Grid(1))/2;
28 plot([Grid(1,1),Grid(1,2)],Unumsolution1,'-c.','linewidth',1.5);hold on
29 H1=plot([Grid(1,1),Grid(1,2)],Unumsolution1,'-c.','linewidth',1.5);hold
    on
30 for i=2:numberx-1
31 Unumsolution1(1,1)=UDGP0P1plusDGP0(1,i)-UDGP0P1plusDGP0(2,i)*(Grid(i+1)-
    Grid(i))/2;Unumsolution1(1,2)=UDGP0P1plusDGP0(1,i)+UDGP0P1plusDGP0(2,
```

```

    i)*(Grid(i+1)-Grid(i))/2;
32 plot([Grid(1,i),Grid(1,i+1)],Unumsolution1,'-c.','linewidth',1.5)
33 end
34
35 %plot the exact
36 y=Grid;
37 plot(y,Uexasolution(1,:),'-b*','linewidth',1.5)
38 H2=plot(y,Uexasolution(1,:),'-b*','linewidth',1.5);hold on
39 lgd=legend([H1,H2],'DGP0P1+DGP0','Exact solution');
40 lgd.FontSize=12;
41 xlabel('位置x','fontsize',14)
42 ylabel('数值U','fontsize',14)
43 title('Ununiform grid-BDF1(U), 64 element','fontsize',16)
44 hold off
45
46
47 %plot the ux
48 %DGP0P1plusDGP0
49 figure
50 Unumsolution1(1,1)=UDGP0P1plusDGP0(2,1);Unumsolution1(1,2)=
    UDGP0P1plusDGP0(2,1);
51 plot([Grid(1,1),Grid(1,2)],Unumsolution1,'-c.','linewidth',1.5);hold on
52 H1=plot([Grid(1,1),Grid(1,2)],Unumsolution1,'-c.','linewidth',1.5);hold
    on
53 for i=2:numberx-1
54 Unumsolution1(1,1)=UDGP0P1plusDGP0(2,i);Unumsolution1(1,2)=
    UDGP0P1plusDGP0(2,i);
55 plot([Grid(1,i),Grid(1,i+1)],Unumsolution1,'-c.','linewidth',1.5)
56 end
57
58 %exact
59 y=Grid;
60 plot(y,Uexasolution(2,:),'-b*','linewidth',1.5)
61 H2=plot(y,Uexasolution(2,:),'-b*','linewidth',1.5);hold on
62 lgd=legend([H1,H2],'DGP0P1+DGP0','Exact solution');
63 lgd.FontSize=12;
64 xlabel('位置x','fontsize',14)
65 ylabel('数值U','fontsize',14)
66 title('Ununiform grid-BDF1(Ux), 64 element','fontsize',16)
67 hold off

```



```

68
69 %determine the accuracy of space U
70 [U,G,~]=BDF1subDGP0P1plusDGP0(8,CFL,endtau);
71 Acc(1,1)=AccuracyU(8,U,G);
72 [U,G,~]=BDF1subDGP0P1plusDGP0(16,CFL,endtau);
73 Acc(1,2)=AccuracyU(16,U,G);
74 [U,G,~]=BDF1subDGP0P1plusDGP0(32,CFL,endtau);
75 Acc(1,3)=AccuracyU(32,U,G);
76 [U,G,~]=BDF1subDGP0P1plusDGP0(64,CFL,endtau);
77 Acc(1,4)=AccuracyU(64,U,G);
78
79 figure
80 hold on
81 plot(log10(a1),log10(Acc(1,:)),'-c*','linewidth',1.5)
82 H1=plot(log10(a1),log10(Acc(1,:)),'-c*','linewidth',1.5);
83
84 H2=plot(log10(a2),1*log10(a2),'--','linewidth',1.5);
85 plot(log10(a2),2*log10(a2),'--','linewidth',1.5)
86 H3=plot(log10(a2),2*log10(a2),'--','linewidth',1.5);
87 lgd=legend([H1,H2,H3],'DGP0P1+DGP0','斜率为1','斜率为2');
88 lgd.FontSize=12;
89 xlabel('Log(deltax)','fontsize',14)
90 ylabel('Log(episilo)','fontsize',14)
91 title('U空间精度分析','fontsize',16)
92
93 %determine the accuracy of space Ux
94 %DGP0P1
95 [U,G,~]=BDF1subDGP0P1plusDGP0(8,CFL,endtau);
96 Acc(1,1)=AccuracyUx(8,U,G);
97 [U,G,~]=BDF1subDGP0P1plusDGP0(16,CFL,endtau);
98 Acc(1,2)=AccuracyUx(16,U,G);
99 [U,G,~]=BDF1subDGP0P1plusDGP0(32,CFL,endtau);
100 Acc(1,3)=AccuracyUx(32,U,G);
101 [U,G,~]=BDF1subDGP0P1plusDGP0(64,CFL,endtau);
102 Acc(1,4)=AccuracyUx(64,U,G);
103
104 figure
105 hold on
106 plot(log10(a1),log10(Acc(1,:)),'-c*','linewidth',1.5)
107 H1=plot(log10(a1),log10(Acc(1,:)),'-c*','linewidth',1.5);

```

```

108
109 plot(log10(a2),1*log10(a2),'--','linewidth',1.5)
110 H2=plot(log10(a2),1*log10(a2),'--','linewidth',1.5);
111 plot(log10(a2),2*log10(a2),'--','linewidth',1.5)
112 H3=plot(log10(a2),2*log10(a2),'--','linewidth',1.5);
113 lgd=legend([H1,H2,H3],'DGP0P1+DGP0','斜率为1','斜率为2');
114 lgd.FontSize=12;
115 xlabel('Log(deltax)','fontsize',14)
116 ylabel('Log(episilo)','fontsize',14)
117 title('Ux空间精度分析','fontsize',16)

```

BDF1subDGP0P1plusDGP0

```

1 function [Unumsolution,Grid,N]=subDGP0P1plusDGP0(Unit,CFL,endtau)
2 endx=1;deltax=endx/Unit;tol=10^(-10);
3 nu=1;Lr=1/(2*pi);Tr=Lr^2/nu;
4 abslambda=sqrt(nu/Tr);
5 numberx=Unit+1;
6 %记录内点位置
7 Grid=zeros(1,numberx);
8 for i=2:numberx-1
9 Grid(1,i)=(i-1)*deltax+(0.1*rand(1)-0.05)*deltax;
10 end
11 Grid(1,numberx)=endx;
12
13 deltatau=zeros(1,numberx-1);
14 for i=1:numberx-1
15 deltatau(i)=CFL*(Grid(1,i+1)-Grid(1,i))/abslambda;%伪时间变量
16 end
17
18 %Ucurrent=zeros(2,numberx-1);
19 %Unext=zeros(2,numberx-1);
20
21 B1=1;
22 %C=[B1,0;0,B1/deltax];
23 %Mtau=[deltax,0;0,deltax/12+1/deltax];A=[abslambda,0;0,abslambda];
24 R=zeros(2*Unit,1);
25 Rd=zeros(2,numberx-1);
26 Rb=zeros(2,numberx-1);
27 Fn=zeros(2,numberx);
28

```

```

29
30 %solve the question
31 %构建 LHS
32 %Mtau/deltatau
33 LHS1=zeros(2*Unit,2*Unit);
34 for k=1:numberx-1
35 LHS1(2*k-1,2*k-1)=(Grid(k+1)-Grid(k))/deltatau(k);
36 LHS1(2*k,2*k)=((Grid(k+1)-Grid(k))/12+1/(Grid(k+1)-Grid(k)))/deltatau(k)
    ;
37 end
38 %Rdomain
39 for k=1:numberx-1
40 LHS2(2*k,2*k)=(nu+1/Tr)/(Grid(k+1)-Grid(k));
41 end
42
43 %Rboundary
44 LHS3=zeros(2*Unit,2*Unit);
45 for iface=2:numberx-1
46 ieL=iface-1;
47 ieR=iface;
48 CL=[B1,1/2;0,B1/(Grid(ieL+1)-Grid(ieL))];
49 CR=[B1,-1/2;0,B1/(Grid(ieR+1)-Grid(ieR))];
50 %diag
51 LHS3(2*ieL-1:2*ieL,2*ieL-1:2*ieL)=LHS3(2*ieL-1:2*ieL,2*ieL-1:2*ieL)+CL
    '*[abslambda/2,-nu/2;-1/(2*Tr),abslambda/2]*CL;
52 LHS3(2*ieR-1:2*ieR,2*ieR-1:2*ieR)=LHS3(2*ieR-1:2*ieR,2*ieR-1:2*ieR)-CR
    '*[-abslambda/2,-nu/2;-1/(2*Tr),-abslambda/2]*CR;
53 %upper
54 LHS3(2*ieL-1:2*ieL,2*ieR-1:2*ieR)=LHS3(2*ieL-1:2*ieL,2*ieR-1:2*ieR)+CL
    '*[-abslambda/2,-nu/2;-1/(2*Tr),-abslambda/2]*CR;
55 %lower
56 LHS3(2*ieR-1:2*ieR,2*ieL-1:2*ieL)=LHS3(2*ieR-1:2*ieR,2*ieL-1:2*ieL)-CR
    '*[abslambda/2,-nu/2;-1/(2*Tr),abslambda/2]*CL;
57 end
58 CL=[B1,1/2;0,B1/(Grid(numberx)-Grid(numberx-1))];
59 CR=[B1,-1/2;0,B1/(Grid(2)-Grid(1))];
60 LHS3(2*1-1:2*1,2*1-1:2*1)=LHS3(2*1-1:2*1,2*1-1:2*1)-CR'*([abslambda/2,-
    nu/2;-1/(2*Tr),abslambda/2]*[0,0;0,1/deltax]+[-abslambda/2,-nu
    /2;-1/(2*Tr),-abslambda/2]*CR);
61 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))+CL'*([
        abslambda/2,-nu/2;-1/(2*Tr),abslambda/2]+[-abslambda/2,-nu/2;-1/(2*Tr
        ),-abslambda/2]*[0,0;0,1])*CL;
62 LHS=LHS1+LHS2+LHS3;
63
64 %取出我们所需要的 D
65 D=zeros(2*Unit,2*Unit);
66 for iface=2:numberx
67 ieL=iface-1;
68 D(2*ieL-1:2*ieL,2*ieL-1:2*ieL)=LHS(2*ieL-1:2*ieL,2*ieL-1:2*ieL);
69 end
70 %为循环所预设的一些量
71 Ucurrent=zeros(2,numberx-1);
72 Unext=zeros(2*Unit,1);
73 %initial condition set up
74
75 for k=1:numberx-1
76 x=Grid(k);
77 Ucurrent(1,k)=(x+(Grid(k+1)-Grid(k))/2)^2-(x+(Grid(k+1)-Grid(k))/2);
78 Ucurrent(2,k)=(2*(x+(Grid(k+1)-Grid(k))/2)-1)*(Grid(k+1)-Grid(k));%
    Ucurrent(2,:) 存储的是
    Ux*deltx
79 end
80
81 %Rdomain
82 for k=1:numberx-1
83 Rd(1,k)=gauss1(Grid(k),Grid(k+1));
84 %Rd(1,k)=pi*(cos(pi*x)-cos(pi*(x+deltax)));
85 %Rd(2,k)=-nu*Ucurrent(2,k)/deltax+(-pi/deltax*(deltax/2*cos(pi*(x+deltax))-
    deltax/2)*cos(pi*x)-1/pi*(sin(pi*(x+deltax))-sin(pi*x)))-Ucurrent(2,k)/(Tr*deltax);
86 Rd(2,k)=-nu*Ucurrent(2,k)/(Grid(k+1)-Grid(k))-Ucurrent(2,k)/(Tr*(Grid(k
    +1)-Grid(k)))+gauss2(Grid(k),Grid(k+1));
87 end
88
89 %Rboundary
90 for iface=2:numberx-1
91 ieL=iface-1;
92 ieR=iface;
93 B2L=1/2;
94 B2R=-1/2;

```

```

95 Fn(:, iface)=0.5*([-nu*Ucurrent(2, ieL)/(Grid(ieL+1)-Grid(ieL));-(Ucurrent
    (1, ieL)+B2L*Ucurrent(2, ieL))/Tr]+[-nu*Ucurrent(2, ieR)/(Grid(ieR+1)-
    Grid(ieR));-(Ucurrent(1, ieR)+B2R*Ucurrent(2, ieR))/Tr]) -0.5*A*([
    Ucurrent(1, ieR)+B2R*Ucurrent(2, ieR); Ucurrent(2, ieR)/(Grid(ieR+1)-Grid
    (ieR))]-[Ucurrent(1, ieL)+B2L*Ucurrent(2, ieL); Ucurrent(2, ieL)/(Grid(
    ieL+1)-Grid(ieL))]);
96 Rb(:, ieL)=Rb(:, ieL)-[1, B2L; 0, 1/(Grid(ieL+1)-Grid(ieL))]'*Fn(:, iface);
97 Rb(:, ieR)=Rb(:, ieR)+[1, B2R; 0, 1/(Grid(ieR+1)-Grid(ieR))]'*Fn(:, iface);
98 end
99 Fn(:, 1)=0.5*([-nu*Ucurrent(2, 1)/(Grid(1+1)-Grid(1)); 0]+[-nu*Ucurrent
    (2, 1)/(Grid(1+1)-Grid(1));-(Ucurrent(1, 1)+B2R*Ucurrent(2, 1))/Tr])
    -0.5*A*([Ucurrent(1, 1)+B2R*Ucurrent(2, 1); Ucurrent(2, 1)/(Grid(1+1)-
    Grid(1))]-[0; Ucurrent(2, 1)/(Grid(1+1)-Grid(1))]);
100 Fn(:, numberx)=0.5*([-nu*Ucurrent(2, numberx-1)/(Grid(numberx)-Grid(
    numberx-1));-(Ucurrent(1, numberx-1)+B2L*Ucurrent(2, numberx-1))/Tr]+[-
    nu*Ucurrent(2, numberx-1)/(Grid(numberx)-Grid(numberx-1)); 0]) -0.5*A
    *([0; Ucurrent(2, numberx-1)/(Grid(numberx)-Grid(numberx-1))]-[Ucurrent
    (1, numberx-1)+B2L*Ucurrent(2, numberx-1); Ucurrent(2, numberx-1)/(Grid(
    numberx)-Grid(numberx-1))]);
101 Rb(:, 1)=Rb(:, 1)+[1, B2R; 0, 1/(Grid(2)-Grid(1))]'*Fn(:, 1);
102 Rb(:, numberx-1)=Rb(:, numberx-1)-[1, B2L; 0, 1/(Grid(numberx)-Grid(numberx
    -1))]'*Fn(:, numberx);
103
104 %R 组装
105 for k=1:numberx-1
106 R(2*k-1:2*k, 1)=Rd(:, k)+Rb(:, k);
107 end
108
109 %进行必要的向量等价转变
110 for k=1:numberx-1
111 Unext(2*k-1:2*k, 1)=Ucurrent(:, k);
112 end
113
114 %循环迭代
115 for n=1:floor(endtau/max(deltatau))
116 X=D\R;
117 if max(X)<tol
118 N=max(n*deltatau);
119 break
120 end

```

```

121 Unext=Unext+X;
122 Rd=zeros(2,numberx-1);
123 Rb=zeros(2,numberx-1);
124 for k=1:numberx-1
125 Ucurrent(:,k)=Unext(2*k-1:2*k,1);
126 end
127 %Rdomain
128 for k=1:numberx-1
129 Rd(1,k)=gauss1(Grid(k),Grid(k+1));
130 %Rd(1,k)=pi*(cos(pi*x)-cos(pi*(x+deltax)));
131 %Rd(2,k)=-nu*Ucurrent(2,k)/deltax+(-pi/deltax*(deltax/2*cos(pi*(x+deltax))-(-
    deltax/2)*cos(pi*x)-1/pi*(sin(pi*(x+deltax))-sin(pi*x))))-Ucurrent(2,k)/(Tr*deltax);
132 Rd(2,k)=-nu*Ucurrent(2,k)/(Grid(k+1)-Grid(k))-Ucurrent(2,k)/(Tr*(Grid(k
    +1)-Grid(k)))+gauss2(Grid(k),Grid(k+1));
133 end
134
135 %Rboundary
136 for iface=2:numberx-1
137 ieL=iface-1;
138 ieR=iface;
139 B2L=1/2;
140 B2R=-1/2;
141 Fn(:,iface)=0.5*([-nu*Ucurrent(2,ieL)/(Grid(ieL+1)-Grid(ieL));-(Ucurrent
    (1,ieL)+B2L*Ucurrent(2,ieL))/Tr]+[-nu*Ucurrent(2,ieR)/(Grid(ieR+1)-
    Grid(ieR));-(Ucurrent(1,ieR)+B2R*Ucurrent(2,ieR))/Tr])-0.5*A*([
    Ucurrent(1,ieR)+B2R*Ucurrent(2,ieR);Ucurrent(2,ieR)/(Grid(ieR+1)-Grid
    (ieR))]-[Ucurrent(1,ieL)+B2L*Ucurrent(2,ieL);Ucurrent(2,ieL)/(Grid(
    ieL+1)-Grid(ieL))]);
142 Rb(:,ieL)=Rb(:,ieL)-[1,B2L;0,1/(Grid(ieL+1)-Grid(ieL))]'*Fn(:,iface);
143 Rb(:,ieR)=Rb(:,ieR)+[1,B2R;0,1/(Grid(ieR+1)-Grid(ieR))]'*Fn(:,iface);
144 end
145 Fn(:,1)=0.5*([-nu*Ucurrent(2,1)/(Grid(1+1)-Grid(1));0]+[-nu*Ucurrent
    (2,1)/(Grid(1+1)-Grid(1));-(Ucurrent(1,1)+B2R*Ucurrent(2,1))/Tr])
    -0.5*A*([Ucurrent(1,1)+B2R*Ucurrent(2,1);Ucurrent(2,1)/(Grid(1+1)-
    Grid(1))]-[0;Ucurrent(2,1)/(Grid(1+1)-Grid(1))]);
146 Fn(:,numberx)=0.5*([-nu*Ucurrent(2,numberx-1)/(Grid(numberx)-Grid(
    numberx-1));-(Ucurrent(1,numberx-1)+B2L*Ucurrent(2,numberx-1))/Tr]+[-
    nu*Ucurrent(2,numberx-1)/(Grid(numberx)-Grid(numberx-1));0])-0.5*A
    *([0;Ucurrent(2,numberx-1)/(Grid(numberx)-Grid(numberx-1))]-[Ucurrent
    (1,numberx-1)+B2L*Ucurrent(2,numberx-1);Ucurrent(2,numberx-1)/(Grid(

```

```

    numberx)-Grid (numberx-1)))]);
147 Rb(:,1)=Rb(:,1)+[1,B2R;0,1/(Grid(2)-Grid(1))]*Fn(:,1);
148 Rb(:,numberx-1)=Rb(:,numberx-1)-[1,B2L;0,1/(Grid(numberx)-Grid(numberx
    -1))]*Fn(:,numberx);
149
150 %R 组装
151 for k=1:numberx-1
152 R(2*k-1:2*k,1)=Rd(:,k)+Rb(:,k);
153 end
154 end
155 if n==floor(endtau/max(deltatau))
156 N=max(n*deltatau);
157 end
158 Unumsolution(1,:)=Ucurrent(1,:);
159 for k=1:numberx-1
160 Unumsolution(2,k)=Ucurrent(2,k)/(Grid(k+1)-Grid(k));
161 end
162 end

```