# 数值分析大作业

# 程梓峻

# 2023年6月27日

# 目录

| 1        | $\operatorname{Pro}$ | gramming problems                                 | 1  |
|----------|----------------------|---|----|
|          | 1.1                  | Programming problem 1                             | 1  |
|          | 1.2                  | Programming problem 2                             | 5  |
|          | 1.3                  | Programming problem 3                             | 15 |
| <b>2</b> | Nui                  | merical results                                   | 23 |
|          | 2.1                  | One example from the paper of Lee and Greengard . | 23 |
|          | 2.2                  | Another two examples                              | 26 |
| 3        | The                  | eoretical questions                               | 27 |
|          | 3.1                  | Theoretical question 1                            | 27 |
|          | 3.2                  | Theoretical question 2                            | 29 |
|          | 3.3                  | Theoretical question 3                            | 30 |
|          |                      |   |    |

# 1 Programming problems

All code are aslo in GitHub with the link

# $1.1 \quad {\bf Programming \ problem \ 1}$

The code is as below shown:

```
1 a=0;
2 c=2;
3 Tol=10^(-4);
4 C=16;
5 f=0(x) 2;
6 p=0(x) -x;
7 q=0(x) 2;
8 GL=@(x) x;
9 GR=@(x) x-2;
10 s=2;
11 PhiL=@(x) (x-4)/2;
12 PhiR=@(x) x/2;
13 ui=@(x) 2*x;
14 F=0(x) 2-2*x;
15 A=zeros(1,101);
16 for i=1:length(A)
       A(i) = (i-1) *2/(length(A) -1);
18 end
19 K=10;
20 Ur=directsolver(A,GL,GR,s,PhiL,PhiR,ui,F,K);
21 plot(Ur(1,:),Ur(2,:),'k');
```

```
1 function Ur=directsolver(A,GL,GR,s,PhiL,PhiR,ui,F,K)
             n=length(A)-1;
2
             sigma=zeros(n*K,1);
3
             Ur=zeros(2,n*K);
             P=zeros(n*K,n*K);
6
            for i=1:n
                 sigma((i-1)*K+1:i*K)=Ch1([A(i),A(i+1)],K);
8
             end
             for j=1:n
9
                for i=1:K
                    t = (j-1) *K+i;
11
12
                     E=zeros(K,1);
13
                     E(i) = 1;
                     for T=1:n*K
14
                         if T<(j-1)*K+1
15
                            P(T,t) = PhiR(sigma(T)) *GR(sigma(t)) *...
16
17
                                 RIntegral1 (A(j), A(j), A(j+1), E, K);
                         elseif T>j*K
18
                                 P(T,t) = PhiL(sigma(T)) *GL(sigma(t)) *...
19
                                     LIntegral1(A(j),A(j+1),A(j+1),E,K);
20
                          else
21
22
                             P(T,t) = PhiR(sigma(T)) *GR(sigma(t)) *...
```

```
RIntegral1(A(j), sigma(T), A(j+1), E, K)...
23
                               +PhiL(sigma(T))*GL(sigma(t))*...
^{24}
25
                               LIntegral1 (A(j), sigma(T), A(j+1), E, K);
26
                           end
27
                      end
28
                  end
29
             end
             P=P+eye(n*K);
30
             Z=sigma;
31
              for i=1:n*K
32
                  Z(i) = F(sigma(i));
33
34
35
             Sigma=P \setminus Z;
36
             JL=zeros(1,n+1);
37
             JR=zeros(1,n+1);
             EL=zeros(n,K);
38
             ER=zeros(n,K);
39
             for j=1:n
40
                  for i=1:K
41
                      EL(j,i)=GL(sigma((j-1)*K+i));
42
43
                      ER(j,i)=GR(sigma((j-1)*K+i));
                  end
44
45
             end
             for j=1:n
46
                  JL(j+1)=JL(j)+LIntegrall(A(j),A(j+1),A(j+1),EL(j,:)'.*...
47
                      Sigma((j-1)*K+1:j*K),K);
48
                  JR(n+1-j) = JR(n+2-j) + RIntegrall(A(n+1-j), A(n+1-j), A(n+2-j), ...
49
                      ER(n+1-j,:)'.*Sigma((n-j)*K+1:(n-j+1)*K),K);
51
             end
52
             for j=1:n
53
                  for i=1:K
                      n=(j-1)*K+i;
54
                      Ur(2,n)=ui(sigma(n))+ER(j,i)/s*(JL(j)+...
55
                           LIntegrall(A(j), sigma(n), A(j+1), EL(j,:).*...
56
57
                           Sigma((j-1)*K+1:j*K),K))+...
                           EL(j,i)/s*(JR(j+1)+RIntegrall(A(j),sigma(n),...
58
59
                           A(j+1), ER(j,:).*Sigma((j-1)*K+1:j*K),K));
60
                  end
61
             end
62
             Ur(1,:)=sigma';
63
   end
```

```
CF=zeros(K,K);
3
             Theta=zeros(1,K);
             for i=1:K
5
                Theta(i) = (2*K-2*i+1)*pi/(2*K);
6
7
             end
8
             for i=1:K
9
               for j=1:K
                    CF(i,j) = 2 * cos((i-1) * Theta(j)) / K;
10
                 end
11
            CF(1,:) = CF(1,:)/2;
13
14
            L=zeros(K,K);
15
            for i=3:K
16
               L(1,i) = (-1)^{(i-1)} (i-1) (1/i-1/(i-2))/2;
17
                 L(i-1,i-2)=1/(2*(i-2));
                 L(i-1,i) = -1/(2*(i-2));
18
            end
19
20
            L(2,1)=1;
            L(1,1)=1;
            L(1,2) = -1/4;
22
23
            L(K,K-1)=1/(2*(K-1));
24
            L=(v-u)/2*L;
             J=L*CF*Z;
25
26
             for i=1:K
                 I=I+J(i)*cos((i-1)*acos((2*x-u-v)/(v-u)));
27
29 end
```

```
1 function I=RIntegral1(u,x,v,Z,K)
            I=0;
 2
            CF=zeros(K,K);
 3
            Theta=zeros(1,K);
 4
            for i=1:K
 5
                Theta(i) = (2*K-2*i+1)*pi/(2*K);
 6
 7
            for i=1:K
 8
 9
               for j=1:K
10
                    CF(i,j)=2*cos((i-1)*Theta(j))/K;
11
                 end
12
            CF(1,:)=CF(1,:)/2;
13
            R=zeros(K,K);
            for i=3:K
15
16
               R(1,i) = (1/i-1/(i-2))/2;
```

```
R(i-1,i-2) = -1/(2*(i-2));
17
                  R(i-1,i)=1/(2*(i-2));
18
19
             end
             R(2,1) = -1;
20
21
             R(1,1)=1;
22
             R(1,2)=1/4;
             R(K,K-1) = -1/(2*(K-1));
             R = (v-u)/2*R;
24
             J=R*CF*Z;
^{25}
             for i=1:K
26
                  I=I+J(i)*cos((i-1)*acos((2*x-u-v)/(v-u)));
27
28
29
   end
```

In these codes, the functions f, p, q, correspond to the problem  $u^{(2)} + pu^{(1)} + qu = f$ , and functions GL, GR, PhiL, PhiR, ui, F correspond to  $g_l, g_r, \psi_l, \psi_r, u_i, \widetilde{f}$  (in the paper of Lee and Greengard) respectively.s is just the constant  $g_l g_r^{(1)} - g_r g_l^{(1)}$ . And the code gives an example of these functions and variables at the beginning. The main part of the code is function (in matlab) "directsolver".

### 1.2 Programming problem 2

The code is as below shown:

```
1 a=-1;

2 c=1;

3 Tol=10^(-5);

4 NN=5;

5 r=10^(-NN);

6 C=24;
```

```
7 f=@(x) 0;

8 p=@(x) 2*x/r;

9 q=@(x) 0;

10 GL=@(x) x+1;

11 GR=@(x) x-1;

12 s=2;

13 PhiL=@(x) x/r;

14 PhiR=@(x) x/r;

15 ui=@(x) x;

16 F=@(x) -2*x/r;

17 [Ur,eI]=linearsolver(a,c,Tol,C,GL,GR,s,PhiL,PhiR,ui,F,r);

18 plot(Ur(1,:),Ur(2,:),'k');
```

```
1 function [Ur,In]=linearsolver(a,c,Tol,C,GL,GR,s,PhiL,PhiR,ui,F,r)
2
             interval=struct('leftchild',0,'rightchild',0,'parent',0,...
                 'content',0,'exist',1);
             interval.content=struct('Interval',[0,0],'alphaL',0,'alphaR',...
4
                 0, 'betaL', 0, 'betaR', 0, '\DL', 0, '\DR', 0, 'niuL', 0, ...
 5
             'niuR',0,'niu',0);
 6
             interval(1).content.Interval=[a,c];
 7
             interval(1).content.niuL=0;
             interval(1).content.niuR=0;
 9
10
             interval(1).content.niu=1;
             N=1;
11
             Er=1;
12
             Mat2=zeros(4,3);
13
             Mat2(1,3)=0.5;
14
15
             while Er>Tol
                   S=zeros(1,N);
16
17
                    for i=1:N
                        if interval(i).leftchild==0 && interval(i).exist==1
18
                           S(i) = Evamonitor(interval, i, PhiL, PhiR, GL, GR, F);
19
20
                        end
21
                    end
                   M=N;
                    for i=1:N
23
24
                       if interval(i).leftchild==0 && interval(i).exist==1
25
                           if S(i)≥max(S)/C
                              x=interval(i).content.Interval(1);
26
                              y=interval(i).content.Interval(2);
27
28
                              interval(M) = struct('leftchild', 0, 'rightchild', ...
                                   0, 'parent', i, 'content', 0, 'exist', 1);
30
31
                              interval(M).content=struct('Interval',[x,...
```

```
(x+y)/2],'alphaL',0, ...
32
                               'alphaR',0,'betaL',0,'betaR',0, '△L',0,...
33
                               'ΔR',0,'niuL',0, ...
34
                               'niuR',0,'niu',0);
35
36
                               interval(i).leftchild=M;
37
                               M=M+1;
                               interval(M) = struct('leftchild', 0, 'rightchild', ...
38
                                    0, 'parent', i, 'content', 0, 'exist', 1);
39
                               interval(M).content=struct('Interval',[(x+y)/2,...
40
                                   y], 'alphaL', 0, ...
41
                               'alphaR',0,'betaL',0,'betaR',0, '△L',0,...
42
43
                               '\(\Delta\R', 0, '\niuL', 0, '\niuR', 0, '\niu', 0);
44
                               interval(i).rightchild=M;
45
                            elseif interval(i).parent>0
                                   n=interval(i).parent;
46
                                   m=interval(n).leftchild;
47
                                    if S(m) + S(m+1) < max(S) / (2^10) & ...
48
                                        interval(2*m+1-i).leftchild==0
49
                                       interval(m).exist=0;
50
                                       interval(m+1).exist=0;
51
52
                                       interval(n).leftchild=0;
                                       interval(n).rightchild=0;
53
54
                                   end
                             end
55
56
                        end
57
                    end
58
                    for i=1:M
                         if interval(i).leftchild==0 && interval(i).exist==1
60
                            X=ABD (interval, i, F, PhiL, PhiR, GL, GR);
61
                            interval(i).content.alphaL=X(1);
62
                            interval(i).content.alphaR=X(2);
                            interval(i).content.betaL=X(3);
63
                            interval(i).content.betaR=X(4);
64
                            interval(i).content.\DeltaL=X(5);
65
66
                            interval(i).content.\DeltaR=X(6);
67
                        end
68
                    end
69
                    for j=1:M
70
                        i=M+1-j;
                        n=interval(i).parent;
71
                        if n>0 && interval(i).exist==1
72
73
                            m=interval(n).leftchild;
                            U(1) = interval(m).content.alphaL;
74
75
                            U(2) = interval(m).content.alphaR;
                            U(3) = interval(m).content.betaL;
76
77
                            U(4)=interval(m).content.betaR;
```

```
78
                             U(5) = interval(m).content.\Delta L;
 79
                             U(6) = interval(m).content.\Delta R;
                             V(1) = interval (m+1).content.alphaL;
 80
                             V(2)=interval(m+1).content.alphaR;
 81
 82
                             V(3)=interval(m+1).content.betaL;
                             V(4)=interval(m+1).content.betaR;
 83
                             V(5) = interval(m+1).content.\Delta L;
                             V(6) = interval(m+1).content.\Delta R;
 85
                             Delta=1-U(3) \starV(2);
 86
                             interval(n).content.alphaL=(1-V(1))*(U(1)+...
 87
                                  Delta-1)/Delta+V(1);
 88
 89
                             interval(n).content.alphaR=V(2)*(1-U(4))*...
                                  (1-U(1))/Delta+U(2);
 90
 91
                             interval(n).content.betaL=U(3) * (1-V(4)) * ...
                                  (1-V(1))/Delta+V(3);
 92
                             interval(n).content.betaR=(1-U(4)) * (V(4)+...
 93
 94
                                  Delta-1)/Delta+U(4);
                             interval(n).content.\DeltaL=(1-V(1))*U(5)/...
 95
                                  Delta+V(5)+(V(1)-1)*U(3)*V(6)/Delta;
 96
                             interval(n).content.\Delta R = (1-U(4))*V(6)/...
 97
 98
                                  Delta+U(6) + (U(4) - 1) *V(2) *U(5) / Delta;
99
                          end
100
                     end
                      for i=1:M
101
                          n=interval(i).leftchild;
102
                          if n>0
                             x=interval(i).content.niuL;
104
105
                             y=interval(i).content.niuR;
106
                             z=interval(i).content.niu;
107
                             u1=interval(n).content.alphaL;
108
                             u3=interval(n).content.betaL;
                             u5=interval(n).content.\DeltaL;
109
                             v4=interval(n+1).content.betaR;
110
                             v2=interval(n+1).content.alphaR;
1111
112
                             v6=interval(n+1).content.\DeltaR;
113
                             interval(n).content.niuL=x;
114
                             interval(n+1).content.niuR=y;
115
                             interval(n).content.niu=z;
116
                             interval(n+1).content.niu=z;
                             NIU=[1, v2; u3, 1] \setminus [y*(1-v4)-z*v6; x*(1-u1)-z*u5];
117
118
                             interval(n).content.niuR=NIU(1);
                             interval(n+1).content.niuL=NIU(2);
119
120
                          end
121
                     end
122
                     X=zeros(1,M);
123
                      j=0;
```

```
124
                     for i=1:M
125
                          if interval(i).leftchild==0 && interval(i).exist==1
126
127
                             j=j+1;
128
                          end
129
                     end
130
                     Num=j;
131
                     Mat=[a-1;0;0;0];
                     for i=1:Num
132
133
                          K=Ch(interval(X(i)).content.Interval);
                         A=P (interval, X(i), PhiL, PhiR, GL, GR);
134
135
                         Z1=zeros(10,1);
136
                         Z2=zeros(10,1);
137
                          Z3=zeros(10,1);
138
                          for j=1:10
                              Z1(j) = F(K(j));
139
                              Z2(j) = PhiL(K(j));
140
                              Z3(j) = PhiR(K(j));
141
142
                          sigma=A\Z1+A\Z2*interval(X(i)).content.niuL+...
143
144
                              A\Z3*interval(X(i)).content.niuR;
145
                          Mat1=zeros(4,10);
                          for j=1:10
146
                              Mat1(1, j) = K(j);
147
                              Mat1(2,j) = sigma(j);
148
149
                              Mat1(3, j) = X(i);
150
                          end
151
                          Mat=[Mat Mat1];
152
153
                     [\neg, idx] = sort(Mat(1,:));
154
                     Mat=Mat(:,idx);
                     JL=zeros(1,Num+1);
155
                     JR=zeros(1,Num+1);
156
                     for i=1:Num
157
158
                          n=Mat(3,10*i);
159
                          al=interval(n).content.alphaL;
160
                          b1=interval(n).content.betaL;
161
                          d1=interval(n).content.\DeltaL;
                          n1=interval(n).content.niuL;
162
163
                          n2=interval(n).content.niuR;
                          JL(i+1) = JL(i) + d1 + n1 * a1 + n2 * b1;
164
165
                          m=Mat(3,10*(Num+1-i));
                          a2=interval(m).content.alphaR;
166
167
                         b2=interval(m).content.betaR;
                         d2=interval(m).content.\Delta R;
168
169
                          m1=interval(m).content.niuL;
```

```
170
                         m2=interval(m).content.niuR;
171
                         JR (Num+1-i) = JR (Num+2-i) + d2+m1*a2+m2*b2;
172
173
                     for i=1:Num
174
                         Y=interval(Mat(3,10*i)).content.Interval;
175
                         K=Ch(Y);
176
                         Z1=zeros(10,1);
                         Z2=zeros(10,1);
177
                         for t=1:10
178
                              Z1(t)=GL(K(t));
                              Z2(t) = GR(K(t));
180
181
                         end
182
                         for j=1:10
183
                             ur=ui(K(j))+GR(K(j))/s*(JL(i)+ ...
184
                              LIntegral (Y(1), K(j), Y(2), Z1.*(Mat(2, 10*(i-1)+...
                              2:10*i+1))'))+GL(K(j))/s*(JR(i+1)+ ...
185
                              RIntegral(Y(1),K(j),Y(2),Z2.*(Mat(2,10*(i-1)+...
186
                              2:10*i+1))'));
187
188
                             Mat (4, 10 * (i-1) + 1 + j) = ur;
189
                         end
190
                     end
191
                     Er=Error(Mat,Mat2);
192
                     Mat2=Mat;
                     nn=size(Mat,2);
193
                     N=M;
194
195
              end
              Ur=zeros(2,nn-1);
196
197
              Ur(1,:) = Mat(1,2:nn);
198
              Ur(2,:) = Mat(4,2:nn);
199
              w=@(x) 2*pi^(-0.5)*exp(-x.^2);
200
              In=0;
              for i=1:Num
201
202
                   Y=interval(Mat(3,10*i)).content.Interval;
                   K=Ch(Y);
203
204
                   Z=zeros(10,1);
                   for j=1:10
205
206
                       G=integral(@(x) w(x), 0, K(j) *r^(-0.5))/...
                           integral (@(x) w(x), 0, r^{(-0.5)});
207
                       Z(j) = (Mat(4,10*(i-1)+j+1)-G)^2;
208
209
210
                   In=In+LIntegral(Y(1),Y(2),Y(2),Z);
211
212 end
```

```
1 function h=Evamonitor(str,n,phiL,phiR,gL,gR,F1)
            A=P(str,n,phiL,phiR,gL,gR);
2
            K=Ch(str(n).content.Interval);
3
            Z1=zeros(10,1);
4
5
            for i=1:10
6
                 Z1(i)=F1(K(i));
7
            end
            Theta=zeros(1,10);
8
            for i=1:10
9
10
                 Theta(i) = (20-2*i+1)*pi/20;
11
            end
            CF=zeros(10,10);
13
            for i=1:10
14
               for j=1:10
15
                    CF(i, j) = cos((i-1)*Theta(j))/5;
16
                 end
17
            end
            CF(1,:) = CF(1,:)/2;
18
            H=CF*(A\Z1);
            h=abs(H(9))+abs(H(10)-H(8));
20
21 end
```

```
1 function M=P(str,n,phiL,phiR,gL,gR)
             X=str(n).content.Interval;
2
             K=Ch(X);
3
             Theta=zeros(1,10);
4
             for i=1:10
5
 6
                 Theta(i) = (20-2*i+1)*pi/20;
7
             end
             CF=zeros(10,10);
             for i=1:10
9
                 for j=1:10
10
                     CF(i,j) = cos((i-1)*Theta(j))/5;
11
                 end
12
13
             end
             CB=5*CF';
14
15
             CF(1,:) = CF(1,:)/2;
16
             L=zeros(10,10);
             for i=3:10
17
               L(1,i) = (-1)^{(i-1)} (i-1) (1/i-1/(i-2))/2;
18
                 L(i-1,i-2)=1/(2*(i-2));
19
20
                 L(i-1,i) = -1/(2*(i-2));
21
            end
22
            L(2,1)=1;
```

```
23
              L(1,1)=1;
             L(1,2) = -1/4;
^{24}
              L(10, 9) = 1/18;
              L=(X(2)-X(1))/2*L;
26
27
              R=zeros(10,10);
28
              for i=3:10
                 R(1,i) = (1/i-1/(i-2))/2;
29
                  R(i-1,i-2) = -1/(2*(i-2));
30
                  R(i-1,i)=1/(2*(i-2));
31
32
              R(2,1) = -1;
33
34
              R(1,1)=1;
              R(1,2)=1/4;
35
36
              R(10, 9) = -1/18;
37
              R = (X(2) - X(1)) / 2 * R;
              IL=CB*L*CF;
38
              IR=CB*R*CF;
39
              DPL=eye(10);
40
41
              DPR=eye(10);
              DGL=eye(10);
42
43
              DGR=eye(10);
44
              for i=1:10
                  DPR(i,i) = phiR(K(i));
45
                  DGL(i,i)=gL(K(i));
46
                  DGR(i,i) = gR(K(i));
47
48
                  DPL(i,i)=phiL(K(i));
49
50
              M=eye(10)+DPL*IL*DGL+DPR*IR*DGR;
51 end
```

```
1 function K=Ch(X)
2     K=zeros(1,10);
3     for i=1:10
4         K(i)=(X(2)-X(1))/2*cos((20-2*i+1)*pi/20)+(X(2)+X(1))/2;
5     end
6 end
```

```
Z3=zeros(10,1);
 6
 7
              Z4=zeros(10,1);
              Z5=zeros(10,1);
8
              for i=1:10
9
10
                  Z1(i)=F1(K(i));
                  Z2(i)=phiL(K(i));
11
12
                  Z3(i) = phiR(K(i));
                  Z4(i) = gL(K(i));
13
                  Z5(i) = gR(K(i));
14
15
              A=P(str,n,phiL,phiR,gL,gR);
16
17
              Z(1) = LIntegral(X(1), X(2), X(2), Z4.*(A\Z2));
18
              Z(2) = LIntegral(X(1), X(2), X(2), Z5.*(A\Z2));
19
              Z(3) = LIntegral(X(1), X(2), X(2), Z4.*(A\Z3));
20
              Z(4) = LIntegral(X(1), X(2), X(2), Z5.*(A\Z3));
              Z(5) = LIntegral(X(1), X(2), X(2), Z4.*(A\Z1));
21
              Z(6) = LIntegral(X(1), X(2), X(2), Z5.*(A\Z1));
22
23 end
```

```
1 function er=Error(A,B)
             n=size(A,2);
             m=size(B,2);
3
              e=0;
              f=0;
5
              for i=3:m
6
                  for j=3:n
7
                       if A(1,j) \ge B(1,i)
8
9
                          a = (B(1, i) - A(1, j-1)) / (A(1, j) - A(1, j-1));
                          e=max(e, abs(a*A(4, j)+(1-a)*A(4, j-1)-B(4, i)));
10
11
                          f=max(f, abs(a*A(4, j)+(1-a)*A(4, j-1)+B(4, i)));
                          break
12
13
                       end
                  end
14
              end
15
16
              er=e/f;
17 end
```

```
Theta(i) = (20-2*i+1)*pi/20;
6
7
             end
             for i=1:10
8
                 for j=1:10
9
10
                  CF(i,j) = cos((i-1)*Theta(j))/5;
11
                 end
12
             end
             CF(1,:) = CF(1,:)/2;
13
            L=zeros(10,10);
14
15
             for i=3:10
               L(1,i) = (-1)^{(i-1)} (i-1) (1/i-1/(i-2))/2;
16
17
               L(i-1,i-2)=1/(2*(i-2));
                L(i-1,i) = -1/(2*(i-2));
18
19
             end
20
             L(2,1)=1;
            L(1,1)=1;
^{21}
            L(1,2) = -1/4;
22
            L(10,9)=1/18;
23
             L=(v-u)/2*L;
             J=L*CF*Z;
25
26
             for i=1:10
27
                 I=I+J(i)*cos((i-1)*acos((2*x-u-v)/(v-u)));
28
             end
29 end
```

```
1 function I=RIntegral(u,x,v,Z)
           I=0;
 2
3
            CF=zeros(10,10);
            Theta=zeros(1,10);
 4
            for i=1:10
                Theta(i) = (20-2*i+1)*pi/20;
 6
 7
            end
            for i=1:10
 8
                 for j=1:10
 9
                  CF(i,j)=2*cos((i-1)*Theta(j))/10;
11
                 end
12
            end
13
            CF(1,:) = CF(1,:)/2;
            R=zeros(10,10);
14
            for i=3:10
15
                R(1,i) = (1/i-1/(i-2))/2;
16
17
                 R(i-1,i-2)=-1/(2*(i-2));
                R(i-1,i)=1/(2*(i-2));
18
19
            end
```

```
20
              R(2,1) = -1;
              R(1,1)=1;
^{21}
              R(1,2)=1/4;
22
              R(10, 9) = -1/18;
23
24
              R = (v-u)/2*R;
25
              J=R*CF*Z;
26
              for i=1:10
27
                   I=I+J(i)*cos((i-1)*acos((2*x-u-v)/(v-u)));
28
              end
29
   end
```

In these codes, similar to problem 1, the functions and variables at the beginning just give an example. And the main part of the code is function (in matlab) "linear solver". I use struct in matlab to build binary tree structure. And the number of chebyshev points of each leaf interval is always 10.

### 1.3 Programming problem 3

The code is as below shown:

```
1 Fun=struct('U',@(x) 0);
2 dFun=struct('V',@(x) 0);
   ddFun=struct('R',@(x) 0);
4 Fun(1)=struct('U',@(x) (x+2*exp(2)-2*exp(1))/(2*exp(2)-exp(1)));
   dFun(1) = struct('V', @(x) 1/(2*exp(2)-exp(1)));
   ddFun(1) = struct('R', @(x) 0);
7 f=0(x,y,z) z/(x*(y+1)^2)-y/(x^2*(y+1));
   f2=0(x,y,z) -y/(x^2*(y+1)^2)-2*z/(x*(y+1)^3);
   f3=0(x,y,z) 1/(x*(y+1)^2);
   X=\exp(1):(2*\exp(2)-\exp(1))/100:2*\exp(2);
11 Y=X;
   tol=10^(-4);
13
   er=1;
14 N=1;
   a=exp(1);
15
   c=2*exp(2);
   C=16;
   while er>tol
18
19
         e=0;
20
         f1=0;
```

```
p=0(x) -f3(x, Fun(N).U(x), dFun(N).V(x));
21
            q=0(x) -f2(x, Fun(N).U(x), dFun(N).V(x));
22
23
            F=@(x) -ddFun(N).R(x)+f(x,Fun(N).U(x),dFun(N).V(x));
24
           GL=0(x) x-exp(1);
25
            GR=@(x) x-2*exp(2);
26
           dGL=@(x) 1;
27
            dGR=@(x) 1;
28
            s=2*exp(2)-exp(1);
            PhiL=@(x) (p(x)+q(x)*GR(x))/s;
29
            PhiR=@(x) (p(x)+q(x)*GL(x))/s;
30
            [Ur, JL, JR, Leftendpoint] = linearsolver1(a, c, tol, C, GL, GR, s, PhiL, PhiR, F);
31
32
            d2v=0(x) ...
                 f3(x, Fun(N).U(x), dFun(N).V(x))*dv(x, Ur, JL, JR, Leftendpoint...
33
                 ,GL,GR,dGL,dGR,s,c)+ \dots
                      f2(x, Fun(N).U(x), dFun(N).V(x)) *v(x, Ur, JL, ...
                 JR, Leftendpoint, GL, GR, s, c) +F(x);
34
            Fun (N+1) = struct ('U', @(x) 0);
35
            \label{eq:fundamental} \texttt{Fun}\,(\texttt{N+1})\,. \texttt{U=@}\,(\texttt{x}) \quad \texttt{Fun}\,(\texttt{N})\,. \texttt{U}\,(\texttt{x})\,+ \texttt{v}\,(\texttt{x}, \texttt{Ur}, \texttt{JL}, \texttt{JR}, \texttt{Leftendpoint}, \texttt{GL}, \texttt{GR}, \texttt{s}, \texttt{c})\,;
36
            dFun(N+1) = struct('V', @(x) 0);
            dFun(N+1).V=0(x) dFun(N).V(x)+dv(x,Ur,JL,JR,Leftendpoint,GL,GR,...
38
39
                 dGL, dGR, s, c);
40
            ddFun(N+1) = struct('R', @(x) 0);
            ddFun(N+1).R=@(x) ddFun(N).R(x)+d2v(x);
41
            for i=1:101
42
                 e=max(e, abs(Fun(N+1).U(X(i))-Fun(N).U(X(i))));
43
                 f1=max(f1, abs(Fun(N+1).U(X(i))+Fun(N).U(X(i))));
45
            end
46
            er=e/f1;
47
           N=N+1;
48
            if N==5
49
               break
50
            end
51 end
52 for j=1:4
       for i=1:101
              Y(j,i) = Fun(j).U(X(i));
54
55
         end
56 end
57 for i=1:101
         Y(3,i) = lambertw(X(i));
58
59 end
60 ER=0;
61 for i=1:101
         ER=max(ER, abs(Y(4,i)-lambertw(X(i))));
64 plot (X,Y(1,:),'k',X,Y(2,:),'r',X,Y(3,:),'g')
```

```
65
 66
     function \Delta u=v(x,Ur,JL,JR,Leftendpoint,GL,GR,s,c)
 67
               n=length(Leftendpoint)+1;
               YY=zeros(1,n);
 68
 69
               for i=1:n-1
 70
                   YY(i)=Leftendpoint(i);
 71
               end
               YY(n)=c;
 72
               for i=1:n-1
 73
                   if x \ge YY(i) && YY(i+1) \ge x
                      K=Ch([YY(i),YY(i+1)]);
 75
 76
                       Z1=zeros(10,1);
 77
                       Z2=zeros(10,1);
 78
                       for j=1:10
 79
                           Z1(j)=GL(K(j));
                           Z2(j) = GR(K(j));
 80
 81
                       \Delta u=GR(x)/s*(JL(i)+LIntegral(YY(i),x,YY(i+1),Z1.*...
 82
 83
                            (Ur(2,10*(i-1)+1:10*i))')+GL(x)/s*(JR(i+1)+...
                               RIntegral (YY(i), x, YY(i+1), Z2.*(Ur(2, 10*(i-1)+...
 84
 85
                               1:10*i))'));
 86
                       break
 87
                   end
               end
 88
    end
 89
     function \Delta du=dv(x,Ur,JL,JR,Leftendpoint,GL,GR,dGL,dGR,s,c)
 91
 92
               n=length(Leftendpoint)+1;
 93
               YY=zeros(1,n);
 94
               for i=1:n-1
 95
                   YY(i) = Leftendpoint(i);
 96
               end
 97
               YY(n)=c;
               for i=1:n-1
 98
                   if x \ge YY(i) && YY(i+1) \ge x
                       K=Ch([YY(i),YY(i+1)]);
100
101
                       Z1=zeros(10,1);
102
                       Z2=zeros(10,1);
                       for j=1:10
103
104
                           Z1(j) = GL(K(j));
                           Z2(j) = GR(K(j));
105
106
                       \Delta du = dGR(x) / s * (JL(i) + LIntegral(YY(i), x, YY(i+1), Z1.*...
107
                            (Ur(2,10*(i-1)+1:10*i))')+dGL(x)/s*(JR(i+1)+...
108
                               RIntegral (YY(i), x, YY(i+1), Z2.* (Ur(2, 10*(i-1)+...
109
                               1:10*i))'));
110
```

```
111 break
112 end
113 end
114 end
```

```
1 function ...
         [Ur, JL, JR, Leftendpoint] = linear solver1 (a, c, Tol, C, GL, GR, s, PhiL, PhiR, F)
              interval=struct('leftchild',0,'rightchild',0,'parent',0,...
2
                  'content', 0, 'exist', 1);
3
              interval.content=struct('Interval',[0,0],'alphaL',0,'alphaR',...
4
                  0, 'betaL', 0, 'betaR', 0, '\Dalant', 0, '\Darant', 0, '\niuL', 0, \ldots \...
5
6
              'niuR', 0, 'niu', 0);
              interval(1).content.Interval=[a,c];
              interval(1).content.niuL=0;
9
              interval(1).content.niuR=0;
10
              interval(1).content.niu=1;
             N=1;
11
             Er=1;
^{12}
              Mat2=zeros(4,3);
13
             Mat2(1,3)=0.5;
14
              while Er>Tol
15
                    S=zeros(1,N);
16
17
                    for i=1:N
                         if interval(i).leftchild==0 && interval(i).exist==1
18
                            S(i) = Evamonitor(interval, i, PhiL, PhiR, GL, GR, F);
19
20
                         end
21
                    end
22
                    M=N;
                    for i=1:N
23
                         if interval(i).leftchild==0 && interval(i).exist==1
^{24}
                            if S(i)>max(S)/C
25
                                x=interval(i).content.Interval(1);
26
                               y=interval(i).content.Interval(2);
27
                               M=M+1;
28
                                interval(M) = struct('leftchild', 0, 'rightchild', ...
                                    0,'parent',i,'content',0,'exist',1);
30
31
                                interval(M).content=struct('Interval',[x,...
32
                                    (x+y)/2], 'alphaL', 0, ...
                                'alphaR',0,'betaL',0,'betaR',0, '\(\Delta\L',0,\dots\)
33
                                'ΔR',0,'niuL',0, ...
34
                                'niuR',0,'niu',0);
35
                                interval(i).leftchild=M;
36
37
38
                                interval(M) = struct('leftchild', 0, 'rightchild', ...
```

```
0, 'parent', i, 'content', 0, 'exist', 1);
39
                                interval(M).content=struct('Interval',[(x+y)/2,...
40
                                    y],'alphaL',0, ...
41
                                'alphaR',0,'betaL',0,'betaR',0, '\(\Delta\L',0,\dots\)
42
43
                                '\(\Delta\R', 0, '\niu\L', 0, '\niu\R', 0, '\niu', 0);
                                interval(i).rightchild=M;
44
45
                            elseif interval(i).parent>0
                                    n=interval(i).parent;
46
                                    m=interval(n).leftchild;
47
                                    if S(m) + S(m+1) < max(S) / (2^10) & ...
48
                                         interval(2*m+1-i).leftchild==0
49
50
                                       interval(m).exist=0;
51
                                       interval(m+1).exist=0;
52
                                        interval(n).leftchild=0;
                                        interval(n).rightchild=0;
53
54
                                    end
                             end
55
56
                         end
58
                     for i=1:M
59
                         if interval(i).leftchild==0 && interval(i).exist==1
60
                            X=ABD(interval, i, F, PhiL, PhiR, GL, GR);
61
                            interval(i).content.alphaL=X(1);
                            interval(i).content.alphaR=X(2);
62
                            interval(i).content.betaL=X(3);
63
                            interval(i).content.betaR=X(4);
                            interval(i).content.\DeltaL=X(5);
65
66
                            interval(i).content.\DeltaR=X(6);
67
                         end
68
                    end
69
                    for j=1:M
                         i=M+1-j;
70
                         n=interval(i).parent;
71
                         if n>0 && interval(i).exist==1
72
73
                            m=interval(n).leftchild;
                            U(1) = interval (m).content.alphaL;
74
75
                            U(2) = interval(m).content.alphaR;
76
                            U(3) = interval(m).content.betaL;
                            U(4)=interval(m).content.betaR;
77
                            U(5) = interval(m).content.\Delta L;
78
                            U(6) = interval(m).content.\Delta R;
79
                            V(1)=interval(m+1).content.alphaL;
80
                            V(2) = interval (m+1).content.alphaR;
81
82
                            V(3)=interval(m+1).content.betaL;
                            V(4)=interval(m+1).content.betaR;
83
84
                            V(5) = interval(m+1).content.\Delta L;
```

```
85
                             V(6) = interval(m+1).content.\Delta R;
 86
                             Delta=1-U(3)*V(2);
                             interval(n).content.alphaL=(1-V(1))*(U(1)+...
 87
                                 Delta-1)/Delta+V(1);
 88
                             interval(n).content.alphaR=V(2)*(1-U(4))*...
                                 (1-U(1))/Delta+U(2);
 90
 91
                             interval(n).content.betaL=U(3)*(1-V(4))*...
92
                                 (1-V(1))/Delta+V(3);
 93
                             interval(n).content.betaR=(1-U(4))*(V(4)+...
                                 Delta-1)/Delta+U(4);
                             interval(n).content.\DeltaL=(1-V(1))*U(5)/...
 95
 96
                                 Delta+V(5)+(V(1)-1)*U(3)*V(6)/Delta;
97
                             interval(n).content.\Delta R = (1-U(4))*V(6)/...
 98
                                 Delta+U(6)+(U(4)-1)*V(2)*U(5)/Delta;
                         end
 99
100
                     end
101
                     for i=1:M
                         n=interval(i).leftchild;
102
103
                          if n>0
                             x=interval(i).content.niuL;
104
105
                             y=interval(i).content.niuR;
106
                             z=interval(i).content.niu;
107
                             u1=interval(n).content.alphaL;
                             u3=interval(n).content.betaL;
108
                             u5=interval(n).content.\DeltaL;
109
110
                             v4=interval(n+1).content.betaR;
                             v2=interval(n+1).content.alphaR;
111
112
                             v6=interval(n+1).content.\DeltaR;
113
                             interval(n).content.niuL=x;
114
                             interval(n+1).content.niuR=y;
115
                             interval(n).content.niu=z;
                             interval(n+1).content.niu=z;
116
                             NIU=[1, v2; u3, 1] \setminus [y*(1-v4)-z*v6; x*(1-u1)-z*u5];
117
                             interval(n).content.niuR=NIU(1);
1118
119
                             interval(n+1).content.niuL=NIU(2);
120
                         end
121
122
                     X=zeros(1,M);
                     j=0;
123
124
                         if interval(i).leftchild==0 && interval(i).exist==1
125
126
                            X(j+1)=i;
127
                             j = j + 1;
128
                         end
129
                     end
130
                     Num=j;
```

```
131
                     Mat=[a-1;0;0;0];
132
                     for i=1:Num
133
                          K=Ch(interval(X(i)).content.Interval);
134
                          A=P(interval, X(i), PhiL, PhiR, GL, GR);
135
                          Z1=zeros(10,1);
136
                          Z2=zeros(10,1);
137
                          Z3=zeros(10,1);
                          for j=1:10
138
                              Z1(j) = F(K(j));
139
                              Z2(j) = PhiL(K(j));
140
                              Z3(j) = PhiR(K(j));
141
142
143
                          sigma=A\Z1+A\Z2*interval(X(i)).content.niuL+...
144
                              A\Z3*interval(X(i)).content.niuR;
145
                          Mat1=zeros(4,10);
                          for j=1:10
146
                              Mat1(1, j) = K(j);
147
                              Mat1(2,j) = sigma(j);
148
149
                              Mat1(3, j) = X(i);
150
151
                          Mat=[Mat Mat1];
152
                     end
                     [\neg, idx] = sort(Mat(1,:));
153
154
                     Mat=Mat(:,idx);
                     JL=zeros(1,Num+1);
155
156
                     JR=zeros(1,Num+1);
                     for i=1:Num
157
158
                          n=Mat(3,10*i);
159
                          al=interval(n).content.alphaL;
160
                         b1=interval(n).content.betaL;
161
                          d1=interval(n).content.\DeltaL;
                          n1=interval(n).content.niuL;
162
163
                          n2=interval(n).content.niuR;
                          JL(i+1) = JL(i) + d1 + n1 * a1 + n2 * b1;
164
165
                          m=Mat(3,10*(Num+1-i));
166
                          a2=interval(m).content.alphaR;
167
                          b2=interval(m).content.betaR;
168
                          d2=interval(m).content.\Delta R;
                          m1=interval(m).content.niuL;
169
170
                          m2=interval(m).content.niuR;
                          JR (Num+1-i) = JR (Num+2-i) + d2+m1*a2+m2*b2;
171
172
                     for i=1:Num
173
174
                          Y=interval(Mat(3,10*i)).content.Interval;
175
                         K=Ch(Y);
                          Z1=zeros(10,1);
176
```

```
177
                          Z2=zeros(10,1);
178
                          for t=1:10
179
                               Z1(t)=GL(K(t));
                               Z2(t) = GR(K(t));
180
181
                          end
                          for j=1:10
182
                               ur=GR(K(j))/s*(JL(i)+ ...
183
                               LIntegral (Y(1), K(j), Y(2), Z1.*(Mat(2, 10*(i-1)+...
184
                               2:10*i+1))'))+GL(K(j))/s*(JR(i+1)+ ...
185
                               RIntegral(Y(1),K(j),Y(2),Z2.*(Mat(2,10*(i-1)+...
186
                               2:10*i+1))'));
187
188
                               Mat (4, 10 * (i-1) + 1 + j) = ur;
189
                          end
190
                      end
                      Er=Error(Mat, Mat2);
191
192
                     Mat2=Mat;
                     nn=size(Mat,2);
193
                     N=M;
194
195
               end
196
               Ur=zeros(2,nn-1);
197
               Ur(1,:) = Mat(1,2:nn);
198
               Ur(2,:)=Mat(2,2:nn);
199
               Leftendpoint=zeros(1, Num);
               for i=1:Num
200
                   Leftendpoint (i) = interval (Mat (3, 10 * i)).content.Interval (1);
201
202
203
    end
```

The Newton method in the paper of Starr and Rokhlin views 2-order ODE about function with image in real number as 1-order ODE about function with image in real vectors. But to decrease the complexcity of computation, I use another "Newton method" which is a little different from that in the paper of Starr and Rokhlin. Below is my method:

To solve u for  $u^{(2)}(x) = f(x, u, u^{(1)})$ , we can define  $\mathcal{F}(u) := u^{(2)} - f(x, u, u^{(1)})$ , then we only need to solve the "zero" of  $\mathcal{F}$ . And formally Newton's method (in its pristine form) is

$$u_{n+1} = u_n - \frac{\mathcal{F}(u_n)}{\mathcal{F}^{(1)}(u_n)}$$

Where  $\mathcal{F}^{(1)}$  is just "formal dervative". Let  $v = u_{n+1} - u_n$ , then

$$v\mathcal{F}^{(1)}(u_n) = -\mathcal{F}(u_n)$$

But by definition of deverivative, we have

$$v\mathcal{F}^{(1)}(u_n) \simeq \mathcal{F}(u_n + v) - \mathcal{F}(u_n) \simeq v^{(2)} - v\partial_u f(x, u_n, u_n^{(1)}) - v^{(1)}\partial_{u^{(1)}} f(x, u_n, u_n^{(1)})$$

So  $u_{n+1} = n_n + v$  and we only need to solve  $v^{(2)} - v\partial_u f(x, u_n, u_n^{(1)}) - v^{(1)}\partial_{u^{(1)}}f(x, u_n, u_n^{(1)}) = -\mathcal{F}(u_n)$ , which can be solved by direct solver. The initial guess  $u_1$  need to satisfies the boundary condition, then in each step, we only need to solve v with homogeneous boundary condition.

In the codes above, the function f correspond to the problem  $u^{(2)}(x) = f(x, u, u^{(1)})$ , and functions GL, GR, dGL, dGR, PhiL, PhiR, F correspond to  $g_l, g_r, g_l^{(1)}, g_r^{(1)}, \psi_l, \psi_r, u_i, \widetilde{f}(= -\mathcal{F}(u_n))$  (in the paper of Lee and Greengard) respectively.s is just the constant  $g_l g_r^{(1)} - g_r g_l^{(1)}$ .

## 2 Numerical results

#### 2.1 One example from the paper of Lee and Greengard

My choice is example 1 (Viscous shock) in the paper of Lee and Greengard,i.e.

$$u^{(2)}(x) + \frac{2x}{\epsilon}u^{(1)}(x) = 0$$

For  $\epsilon=10^{-5}$ , the algorithm requires nine levels of mesh refinement. As figure 1 shown.

And we also have the results about relative  $L^2$  norm and CPU time, as the table below shown:

And the final graph corresponding to the table above is figure 2.

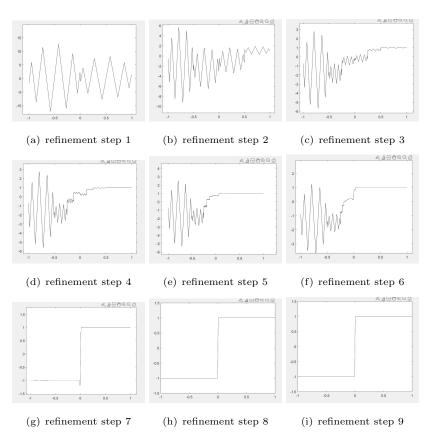


图 1: refinement process for the case  $\epsilon=10^{-5}$ 

| $\epsilon$        | $10^{-4}$            | $10^{-5}$            | $10^{-6}$            | $10^{-7}$            | $10^{-8}$            | $10^{-9}$            |
|-------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| C                 | 16                   | 16                   | 24                   | 24                   | 24                   | 32                   |
| $L^2$ -error      | $6.53 \cdot 10^{-6}$ | $1.08 \cdot 10^{-7}$ | $7.98 \cdot 10^{-9}$ | $1.81 \cdot 10^{-4}$ | $2.33 \cdot 10^{-6}$ | $2.38 \cdot 10^{-7}$ |
| CPU time (second) | 0.478                | 0.505                | 0.567                | 1.22                 | 0.743                | 1.42                 |

表 1: results about relative  $L^2$  norm and CPU time

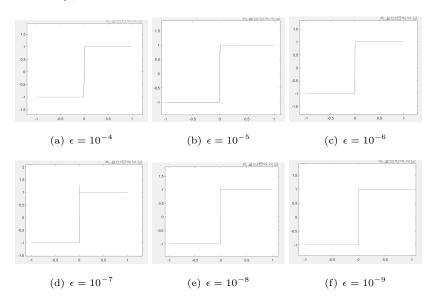
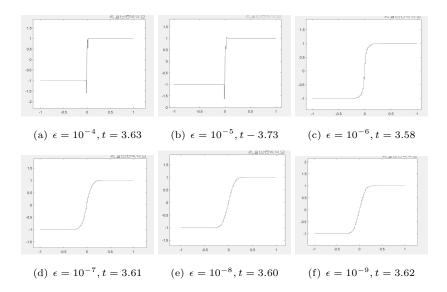


图 2: the final graph corresponding to table1

Here  $\log_2 C$  is just the refinement parameter in the paper of Lee and Greengard(i.e. $S_{div} = C^{-1} \max_{i=1}^{M}$ ). By these results, we get that the algorithm has well numerical performance on aspects of speed, accuracy and adaptivity.

However, if we consider the direct solver, then the result is as fig3 shown:

By these results, we get the fast algorithm in problem 2 is more adaptive, faster, more accurate than the direct solver in problem 1.



 $\boxtimes$  3: the graph corresponding to direct solver, where there is 100 equidistant intervals on [-1,1] and 10 chebychev points on each small interval. And t is CPU time (second)

### 2.2 Another two examples

Example  $1:u^{(2)}-xu^{(1)}+2u=2$  boundary conditions are u(0)=0,u(2)=4. Then the solution is  $u(x)=x^2$ . And for fast solver,we choose C=16, then the CPU time is 0.101s, and the error (corresponding to  $L^1$  norm) is  $O(10^{-16})$ . For the direct solver, we have the following table 2.

| n                 | 20    | 40     | 60    | 80                   | 100                |
|-------------------|-------|--------|-------|----------------------|--------------------|
| $L^1$ -error      | 0.009 | 0.0024 | 0.001 | $6.09 \cdot 10^{-4}$ | $3.91\cdot10^{-4}$ |
| CPU time (second) | 0.163 | 0.593  | 1.29  | 2.27                 | 3.55               |

表 2: results about relative  $L^1$  norm and CPU time.n is the number of equisdistant intervals,the number of chewbyshev points in each subinterval is stil 10

From the table, we can get for simple ODE, fast solver still performs much better than the direct solver.

Example2:For nonlinear eqn:  $u^{(2)}(x) = -\frac{u(x)}{x^2(u(x)+1)} + \frac{u^{(1)}(x)}{x(u(x)+1)^2}$ ,  $u(e) = 1, u(2e^2) = 2$ . The solution is Lambert-W function. We choose  $u_1 = \frac{x+2e^2-2e}{2e^2-e}$  as initial guess, and the result is as fig4 shown.

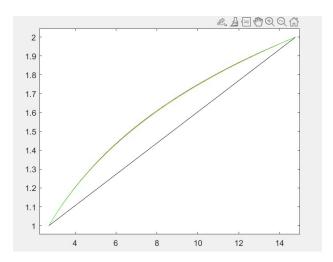


图 4: black line is initial guess  $u_1$ ,red line is  $u_2$ ,and green line is Lambert-W function

From the figure,we get after one iteration,what we get is very close to Lambert-W function. Moreover, if we iterate three times, then for  $u_4$ , the  $L^1$ -error is  $O(10^{-6})$ , and the CPU time is 2.43s. So the nonlinear solver performs well.

# 3 Theoretical questions

### 3.1 Theoretical question 1

W.L.O.G. assume the partition of interval [a, c] is  $a = a_1 < \cdots < a_{n+1} = c$ ,and assume  $\forall 1 \leq i \leq n$ , there are K Chebyshev nodes  $\{\tau_i^j\}_{j=1}^K$  on interval  $[a_i, a_{i+1}]$ , then let  $S := \{\tau_i^j | 1 \leq i \leq n, 1 \leq j \leq K\}, V = \{f : S \to \mathbb{R}\}$ , then the integral operator P

can be viewed as a linear map from V to V which satisfies that  $P(\sum_{1 \leq i \leq n, 1 \leq j \leq K} a_{ij} e_{ij}) = Int(\tau_i^j) e_{ij}$ , where  $a_{ij} \in \mathbb{R}, e_{ij} \in V$  s.t.  $\forall 1 \leq i_0 \leq n, 1 \leq j_0 \leq K, e_{ij}(\tau_{i_0}^{j_0}) = \delta_{i,i_0} \delta_{j,j_0}$ , and

$$Int: [a,c] \to \mathbb{R} \quad x \mapsto \Phi(x) + \psi_l(x) \int_a^x g_l(t) \Phi(t) dt + \psi_r(x) \int_r^c g_r(t) \Phi(t) dt$$

Here  $\forall 1 \leq i \leq n, \Phi|_{[a_i,a_{i+1}]}$  is just the Chebyshev interpolant of  $(\tau_i^j,a_{ij})$   $(1 \leq j \leq K)$  on  $[a_i,a_{i+1}]$ . Then  $e_{ij}$  is a basis of V wrt which the matrix of P is  $I_{nK} + \psi_L L g_L + \psi_R R g_R$ , where  $\psi_L,g_L,\psi_R,g_R$  are diagonal matrix s.t.

$$\psi_L(e_{ij}) = \psi_l(\tau_i^j) e_{ij}, \psi_R(e_{ij}) = \psi_r(\tau_i^j) e_{ij}$$
$$q_L(e_{ij}) = q_l(\tau_i^j) e_{ij}, q_R(e_{ij}) = q_r(\tau_i^j) e_{ij}$$

And  $L(\sum_{1 \leq i \leq n, 1 \leq j \leq K} a_{ij} e_{ij}) = \sum_{1 \leq i \leq n, 1 \leq j \leq K} (\int_a^{\tau_i^j} \Phi(t) dt) e_{ij}, R(\sum_{1 \leq i \leq n, 1 \leq j \leq K} a_{ij} e_{ij}) = \sum_{1 \leq i \leq n, 1 \leq j \leq K} (\int_{\tau_i^j}^b \Phi(t) dt) e_{ij}$ , where definition of  $\Phi$  is the same as above. So for any fixed i,

$$L(\sum_{j=1}^{K} r_j e_{ij}) = L_i(\sum_{j=1}^{K} r_j e_{ij}) + (\int_{a_i}^{a_{i+1}} F(t)dt) \sum_{k>i,1 \le j \le K} e_{kj}$$

Here F is the Chebyshev interpolant of  $(\tau_i^j, r_j)$   $(1 \leq j \leq K)$  on  $[a_i, a_{i+1}]$ , and  $L_i$  is just the left integration operator (this is metioned in the paper of Lee and Greengard) on  $[a_i, a_{i+1}]$ . So let  $\tau_i^j$  be the ((i-1)K+j)th basis element, then assume  $L = (L_{ij})_{1 \leq i,j \leq n}$ , where  $L_{ij}$  are  $K \times K$  matrix. Then  $\forall X = (r_1, \dots, r_K)^t \in \mathbb{R}^K, \forall k < i, L_{k,i}X = 0; \forall k > i, L_{k,i}X = (\int_{a_i}^{a_{i+1}} F(t)dt, \dots, \int_{a_i}^{a_{i+1}} F(t)dt)^t$ . So

$$L = \begin{pmatrix} L_{11} & & & \\ L_{21} & L_{22} & & \\ \vdots & \ddots & \ddots & \\ L_{n1} & \cdots & L_{n-1,n} & L_{nn} \end{pmatrix}$$

Here 
$$L_{ii}=L_{i}$$
 and for any  $i< n$ ,  
all rows of 
$$\begin{pmatrix} L_{i+1,i} \\ \vdots \\ L_{ni} \end{pmatrix}$$
 are the same.  
And  $\exists X=(r_{1},\cdots,r_{K})^{t}\in\mathbb{R}^{K}$  s.t.  $\int_{a_{i}}^{a_{i+1}}F(t)dt\neq 0$ ,  
so  $rk\begin{pmatrix} L_{i+1,i} \\ \vdots \\ L_{ni} \end{pmatrix})=1$ .  
Then  $\psi_{L}Lg_{L}=(\tilde{L}_{ij})_{1\leq i,j\leq n}$ ,  
where  $\tilde{L}_{ij}$  are  $K\times K$  matrices and  $\forall 1\leq i< j\leq n, \tilde{L}_{ij}=0$  and  $rk\begin{pmatrix} \tilde{L}_{i+1,i} \\ \vdots \\ \tilde{L}_{ni} \end{pmatrix})\leq 1$ .  
Similarly  $\psi_{R}Rg_{R}=\begin{pmatrix} \tilde{R}_{i,j} \\ \vdots \\ \tilde{R}_{i-1,i} \end{pmatrix}$  are  $K\times K$  matrices and  $\forall 1\leq j< i\leq n, \tilde{R}_{ij}=0$  and  $rk\begin{pmatrix} \tilde{R}_{1,i} \\ \vdots \\ \tilde{R}_{i-1,i} \end{pmatrix} \geq 1$ .  
So the matrix  $P=(P_{ij})_{1\leq i,j\leq n}$ ,  
where for all  $i< j,rk\begin{pmatrix} P_{i+1,i} \\ \vdots \\ P_{ni} \end{pmatrix} \leq 1$  and  $rk\begin{pmatrix} P_{1,j} \\ \vdots \\ P_{j-1,j} \end{pmatrix} \leq 1$ .  
And  $P_{ii}$  is just the local integral operator  $P_{[a_{i},a_{i+1}]}$  on  $[a_{i},a_{i+1}]$ .  
This notation is the same as the one in the paper of Lee and Greengard).  
And when  $a_{i+1}-a_{i}$  is small enough, the norm of  $R_{i}$  and  $L_{i}$  tends to zero,  
so  $P_{ii}-Id=\tilde{R}_{i}+\tilde{L}_{i}=O(R_{i})+O(L_{i})=o(Id)$ ,  
hence  $rk(P_{ii})=K$  if  $a_{i+1}-a_{i}$  is small enough. All in all,  
we get the rank structure of  $P$ 

(P is just the matrix A in Theoretical question 1).Generally, when the number of chewbyshev points of  $[a_i, a_{i+1}]$  are distinct, use the same method, we can get the similar answer.

#### 3.2 Theoretical question 2

Newton's method (in its pristine form) is very sensitive to the choice of initial guess. But my "Newton method" in Programming problem 3 avoid function whose image is vectoror matrix, and in my method the inintial guess must satisfies boundary condition, so in the process of iteration we only need to solve ODE with trival boundary condition. Then such method and initial guess can help us get better iteration results. If we want to get better initial guess, we can sketch the graph of the solution roughly and choose an initial guess which is close to the graph and satisfies boundary condition.

#### 3.3 Theoretical question 3