1 Homework 5 (Solution)

1. A matlab program is attached.

2. Theoretical error for one source of amplitude 1 reexpansion (including its translation) is

$$error_p = \frac{2}{l \cdot 3^p}.$$

For N sources of amplitude 1 will be

$$error = \frac{2N}{l \cdot 3^p}.$$

In the problem considered, the amplitude of sources does not exceed $1/N$. So we have for the total error of SLFMM the following bound

$$error \leqslant \frac{2N}{l \cdot 3^p} \cdot \frac{1}{N} = \frac{2}{l \cdot 3^p}.$$

Since the size of the box changes with K as

$$l = \frac{L}{K},$$

we have

$$error \leqslant \frac{2K}{L \cdot 3^p}.$$

The truncation number that provides required accuracy ϵ is therefore

$$p = \left[\log_3 \frac{2K}{\epsilon L}\right].$$

For $K \sim 10$, $L = 10$, and $\epsilon = 10^{-5}$ we have

$$p \sim 11.$$

As the formula shows p is growing with K as $\log K$. For $K = O(N^{2/3})$ we also can see that it will grow as $\log N$, which is rather a week growth.We also expect that the actual error will be much less than the theoretical bound. So for further computations we took

$$p = 10,$$

and *always* checked the actual error to stay within the specified error bounds.

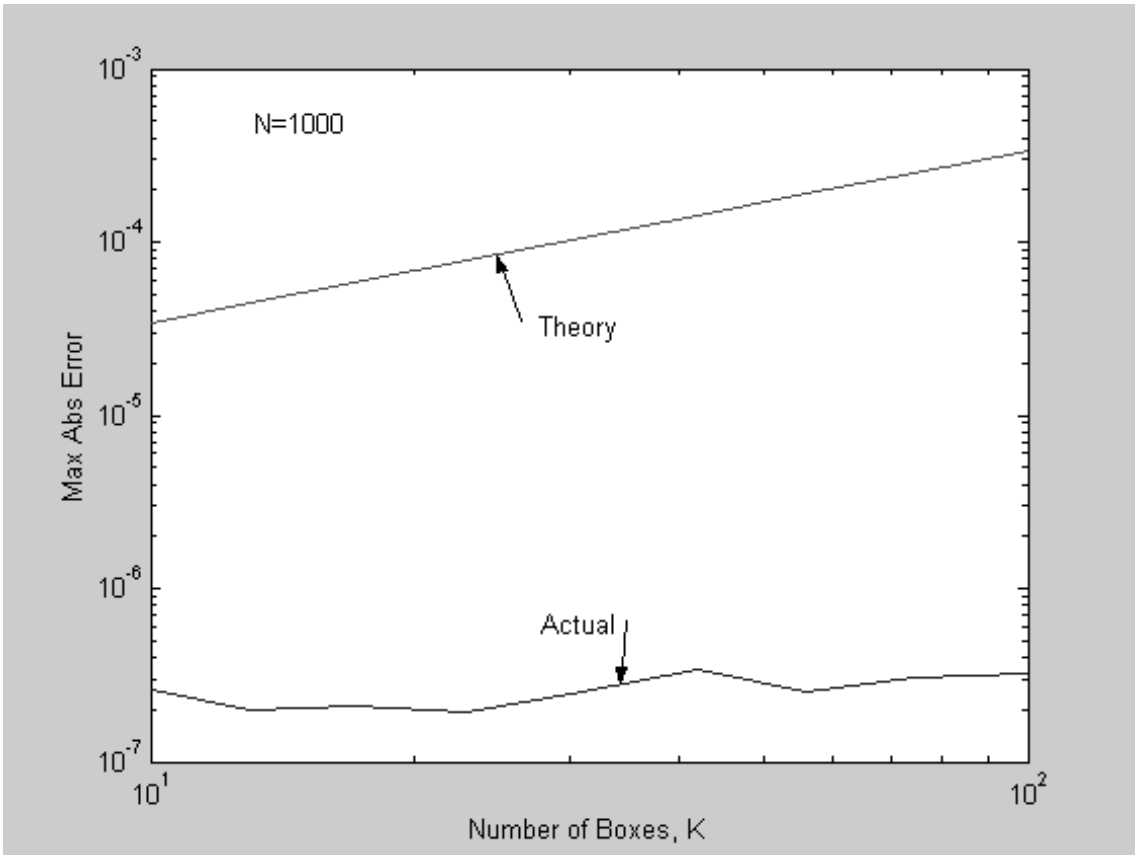
3. 

Figure 1:

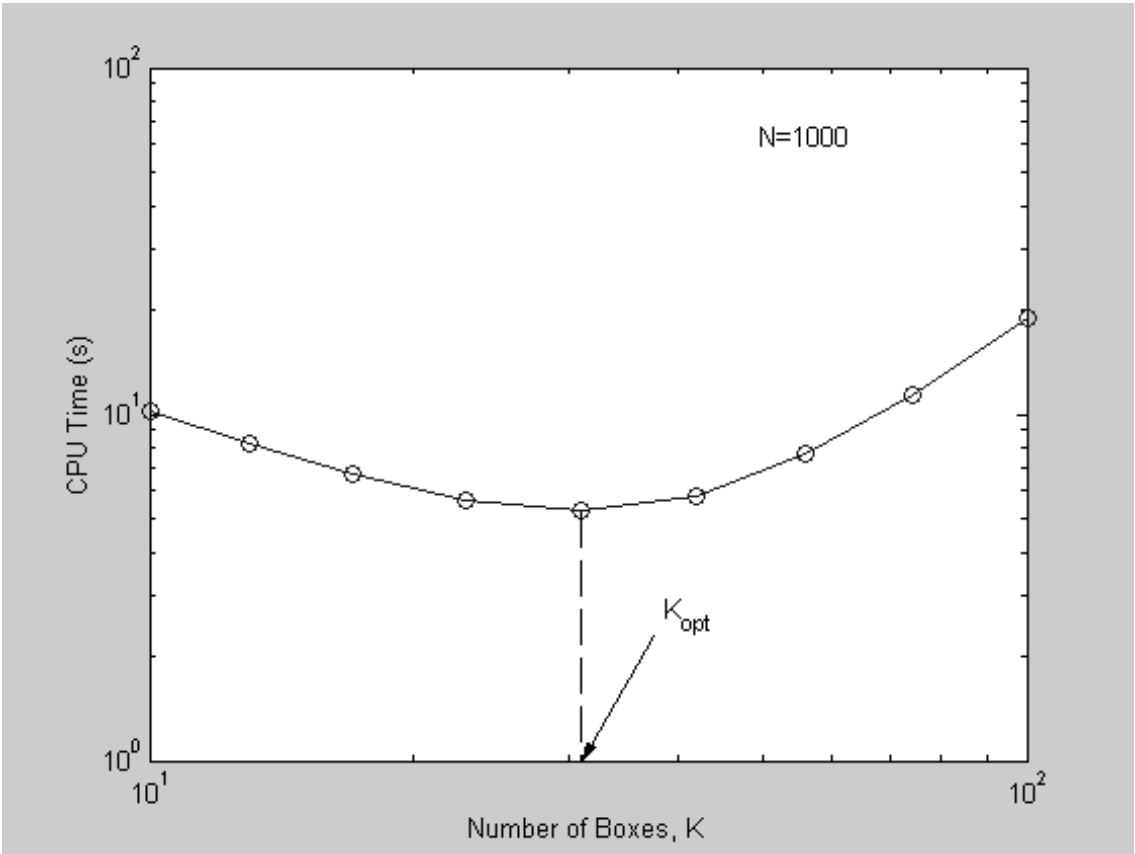


Figure 2:

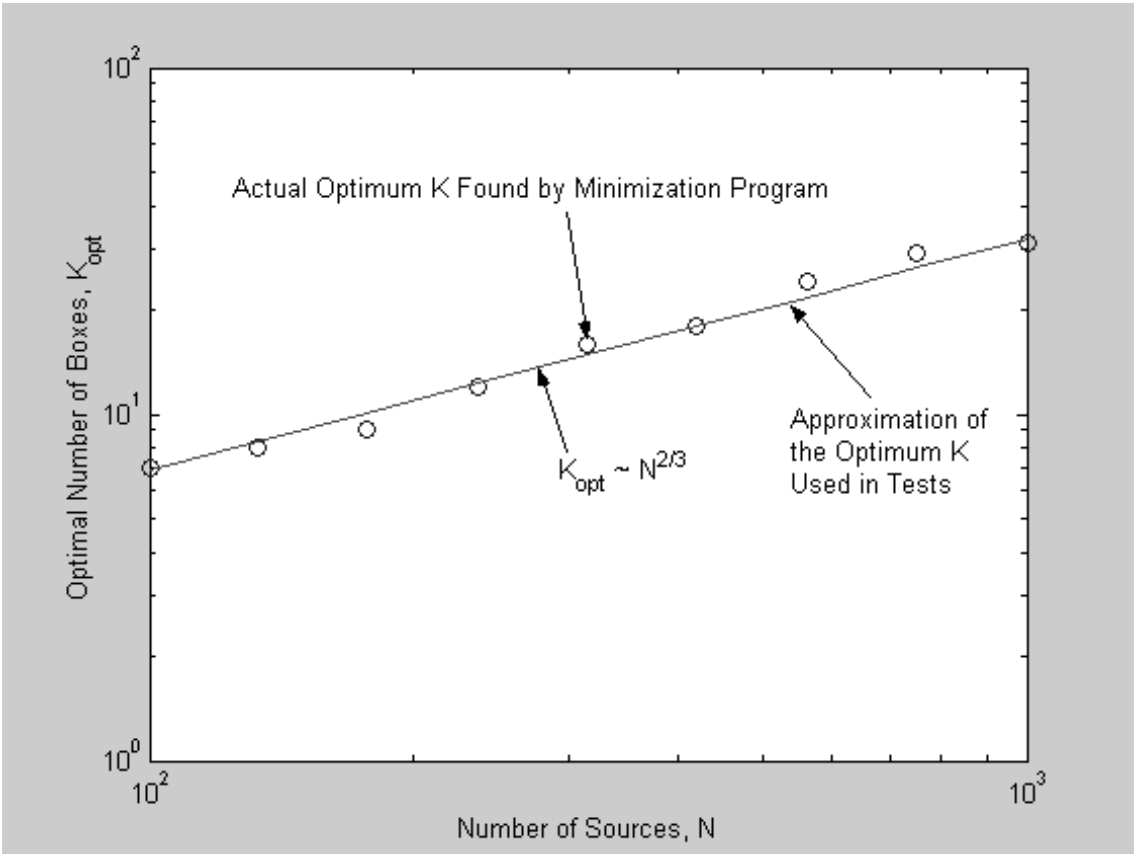


Figure 3:

4.-5.

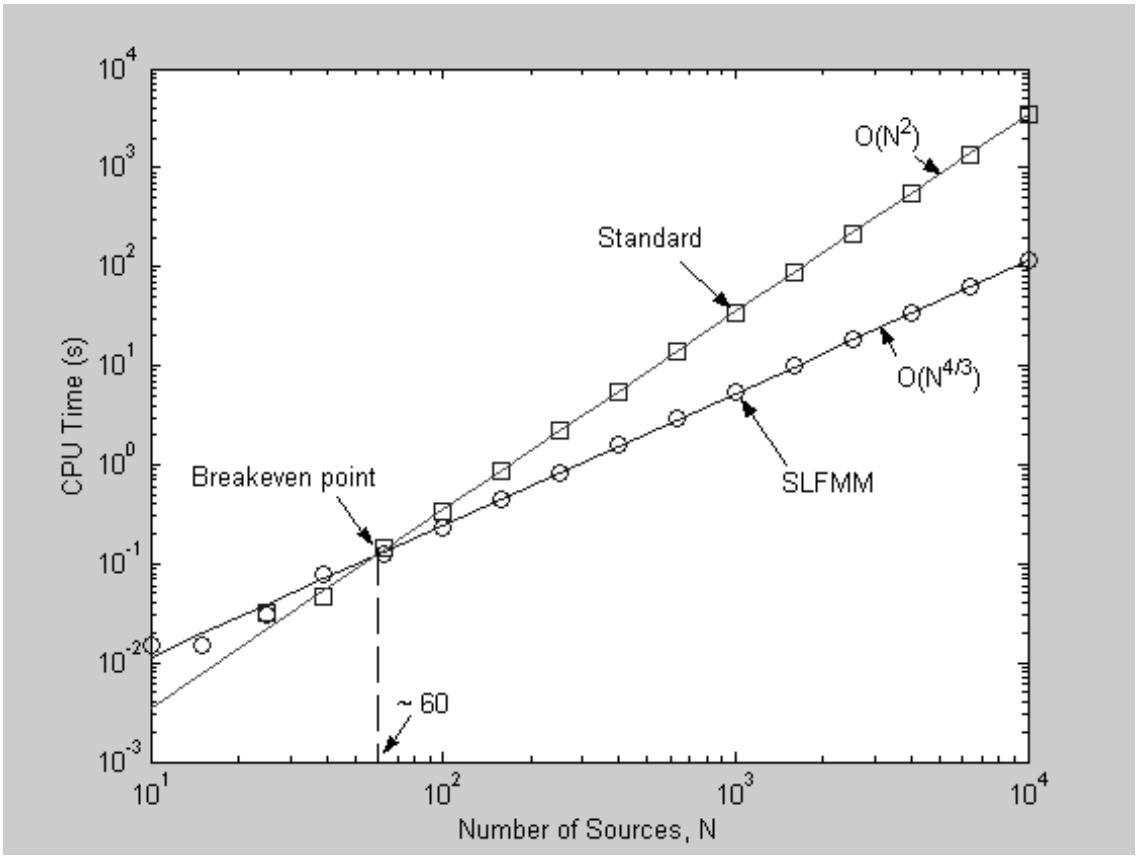


Figure 4:

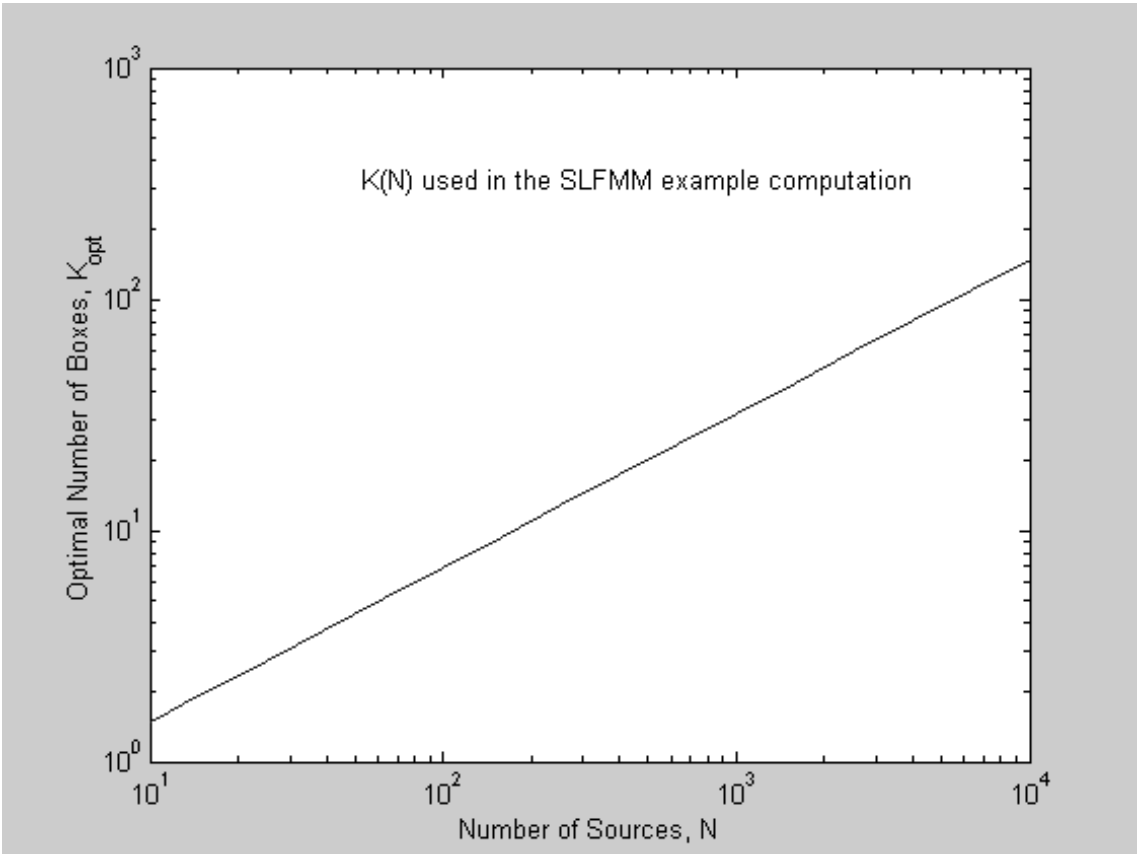


Figure 5:

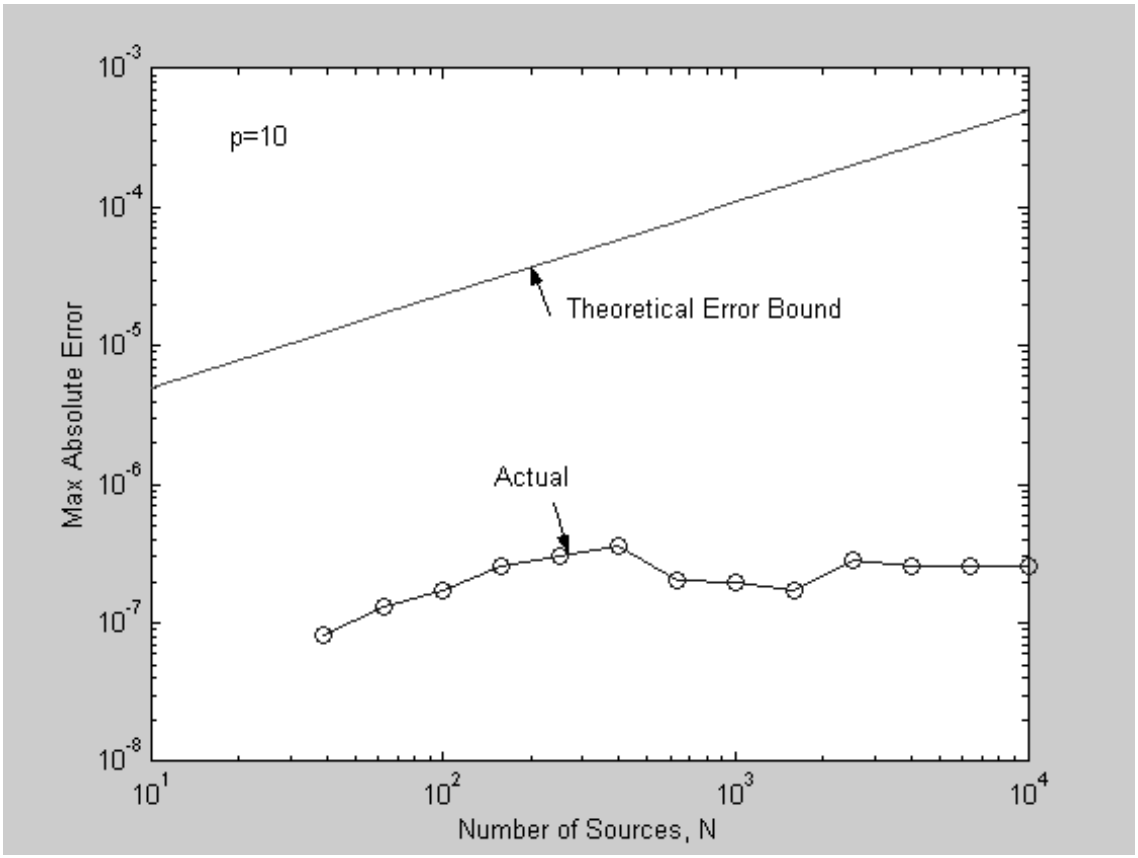
6. 

Figure 6:

```
%homework 5
% SLFMM

% Initialization
clear all;
close all;

global X Y U V L
global    SourceBoxIndex    NonEmptySourceBox    EvaluationBoxIndex
NonEmptyEvaluationBox BoxSize p

npoint=1;
kpoint=9;
%NN=floor(logspace(1,4,npoint));
NN=[1000];
epsilon=1e-5;
L=10;

for ii=1:npoint             % loop for varying N

    % Generate data points;
    N=NN(ii)
    M=N-1;
    X=L*sort(rand(1,N));
    U=rand(1,N)/N;
    V=zeros(1,N);
    for j=1:N-1,
        Y(j)=.5*(X(j)+X(j+1));
    end;

    [ts(ii),Vs]=h5_standard(U);

    N43(ii)=N^(1.333333333);
    N2(ii)=N*N;

    p=10;
    KK=floor(logspace(1,2,kpoint));

    % module for computations with varying K
    for kk=1:kpoint % loop for varying K
        K=KK(kk)
        BoxSize=L/K;
        tm(kk)=h5_SLFMM_K(K);
        err(kk)=max(abs(V-Vs));
        err_theory(kk)=h5_GetTheoreticalError(p,N,K,BoxSize);
    end;
```

```
    %optimization module:
    %   x = fminbnd(@h5_SLFMM_K,1,N,optimset('Display','iter'))
    %   Kopt(ii)=round(x);
    %   tm(ii)=h5_SLFMM_K(Kopt(ii));

    %module for computations of timing and error with optimal K
    %     K_opt=0.69*(N*N/p)^(0.3333333333);
    %     Kopt(ii)=K_opt;
    %     tm(ii)=h5_SLFMM_K(K_opt);
    %     err(ii)=max(abs(V-Vs));
    %
err_theory(ii)=h5_GetTheoreticalError(p,N,K_opt,BoxSize);

end;

% figure plot for varying K
figure;
loglog(KK,tm,'o');

figure;
loglog(KK,err_theory,'r');
hold on;
loglog(KK,err,'b');

% figure plot for varying N with optimum K

% figure;
% loglog(NN,tm,'o');
% hold on;
% loglog(NN,ts,'s');
% loglog(NN,0.00053*N43,'b');
% loglog(NN,0.000035*N2,'r');
%
% figure;
% loglog(NN,Kopt,'b');
%
% figure;
% loglog(NN,err_theory,'r');
% hold on;
% loglog(NN,err,'b');
%-----------------------------------------
function box_number=h5_GetBox(x,BoxSize);
box_number=floor(x/BoxSize)+1;
%-----------------------------------------
function box_center=h5_GetBoxCenter(box_number,box_size);
box_center=(box_number-.5)*box_size;
%-----------------------------------------
```

```
function [StartIndex,EndIndex]=h5_GetE2(Box,BoxIndex,NonEmptyBox)

K=length(NonEmptyBox);
dum=0;
[NumberNonemptyBoxes,dum]=size(BoxIndex);

if Box==1
    left_neighbor=Box;
else
    left_neighbor=Box-1;
end;
if Box==K
    right_neighbor=Box;
else
    right_neighbor=Box+1;
end;

left=NonEmptyBox(left_neighbor);
self=NonEmptyBox(Box);
right=NonEmptyBox(right_neighbor);

StartIndex=0;
EndIndex=0;

if left+self+right ~=0

    if left ~=0
        StartIndex=BoxIndex(left,1);
    else
        if self ~=0
            StartIndex=BoxIndex(self,1);
        else
            StartIndex=BoxIndex(right,1);
        end;
    end;

    if right ~=0
        EndIndex=BoxIndex(right,2);
    else
        if self ~=0
            EndIndex=BoxIndex(self,2);
        else
            EndIndex=BoxIndex(left,2);
        end;
    end;
end;
%-----------------------------------------
```

```
function R=h5_GetRBasis(y,p)
R=[1:p];
R(1)=1;
for i=1:p-1
    R(i+1)=R(i)*y;
end;
%-----------------------------------------
function B=h5_GetSEexpansionCoefficients(xi,xc,p)
B=h5_GetRBasis(xi-xc,p);
%-----------------------------------------
function epsilon=h5_GetTheoreticalError(p,N,K,BoxSize)
epsilon=2/(BoxSize*3^p);
%-----------------------------------------
function phi=h5_Phi(y,x)
phi=1/(y-x);
%-----------------------------------------
function [time,V]=h5_SLFMM(U)
% One-dimensional Single Level FMM

global  X  Y  SourceBoxIndex  NonEmptySourceBox  EvaluationBoxIndex
NonEmptyEvaluationBox BoxSize p

time=cputime;

dum=0;
[NumberOfNonemptySourceBoxes,dum]=size(SourceBoxIndex);
[NumberOfNonemptyEvaluationBoxes,dum]=size(EvaluationBoxIndex);
NumberOfSources=length(X);
NumberOfEvaluations=length(Y);
V=[1:NumberOfEvaluations];

%Step 1. Generation of S-expansion coefficients for each nonempty
box

C=zeros(NumberOfNonemptySourceBoxes,p);
B=[1:p];

for n=1:NumberOfNonemptySourceBoxes
    StartIndex=SourceBoxIndex(n,1);
    EndIndex=SourceBoxIndex(n,2);
    xc=h5_GetBoxCenter(h5_GetBox(X(StartIndex),BoxSize),BoxSize);
    for i=StartIndex:EndIndex,
        B=h5_GetSEexpansionCoefficients(X(i),xc,p);
        C(n,1:p)=C(n,1:p)+U(i)*B(1:p);
    end;
end;
```

```
%Step 2. S|R-translation of the expansion coefficients

D=zeros(NumberOfNonemptyEvaluationBoxes,p);
A=[1:p];

% SetSource_nonempty=[1:NumberOfNonemptySourceBoxes];

for m=1:NumberOfNonemptyEvaluationBoxes
    StartIndex=EvaluationBoxIndex(m,1);
    yc=h5_GetBoxCenter(h5_GetBox(Y(StartIndex),BoxSize),BoxSize);
%     Count=0;
%     nn=0;
    for n=1:NumberOfNonemptySourceBoxes
        StartIndex=SourceBoxIndex(n,1);

xc=h5_GetBoxCenter(h5_GetBox(X(StartIndex),BoxSize),BoxSize);
        t=yc-xc;
        if abs(t) > 1.5*BoxSize
            B(:)=C(n,:);
            A=(h5_SR_translation(B,t))';
            D(m,1:p)=D(m,1:p)+A(1:p);
%             Count=Union(Count,n);
%         else
%             nn=nn+1;
        end;
    end;
%     NonCounted(m)=setdiff(SetSource_nonempty,Count);
%    Nn(m)=nn;
end;

%Step 3. Final Summation

for m=1:NumberOfNonemptyEvaluationBoxes
    StartIndex=EvaluationBoxIndex(m,1);
    EndIndex=EvaluationBoxIndex(m,2);
    Box=h5_GetBox(Y(StartIndex),BoxSize);
    yc=h5_GetBoxCenter(Box,BoxSize);

[StartIndexX,EndIndexX]=h5_GetE2(Box,SourceBoxIndex,NonEmptySource
Box);
    for j=StartIndex:EndIndex
        R=h5_GetRBasis(Y(j)-yc,p);
        A(:)=D(m,:);
        V(j)=dot(A,R);
       if StartIndexX ~= 0,
         for i=StartIndexX:EndIndexX
%           for mm=1:Nn(m);
```

```
%               m_noncount=NonCounted(mm);
%               StartIndexX=SourceBoxIndex(m_noncount,1);
%               EndIndexX=SourceBoxIndex(m_noncount,2);
%
                V(j)=V(j)+U(i)*h5_Phi(Y(j),X(i));
            end;
        end;
    end;
end;

time=cputime-time;
%-----------------------------------------
function y=h5_SLFMM_K(x)
%treats SLFMM as function of K for use in fmin

global U V
global  X  Y  SourceBoxIndex  NonEmptySourceBox  EvaluationBoxIndex
NonEmptyEvaluationBox BoxSize L

K=floor(x);

BoxSize=L/K;
[SourceBoxIndex,NonEmptySourceBox]=h5_SetDataStructure(X,BoxSize,K
);
[EvaluationBoxIndex,NonEmptyEvaluationBox]=h5_SetDataStructure(Y,B
oxSize,K);
[y,V]=h5_SLFMM(U);
%-----------------------------------------
function a=h5_SR_translation(b,t)

p=length(b);          a=linspace(0,0,p);          c=linspace(0,0,p);
SR_matrix=zeros(p,p);

%recursively form SR matrix
invt=1/t; a(1)=invt; SR_matrix(1,1)=invt; invt2=invt*invt;
nminus=1;
for n=2:p
    nminus=-nminus;
    a(n)=a(n-1)*invt;        SR_matrix(1,n)=a(n);        c(1)=a(n);
SR_matrix(n,1)=nminus*a(n);
    mminus=nminus;
    for m=2:n-1
        mminus=-mminus;
        c(m)=-(1+(n-1)/(m-1))*c(m-1)*invt;
        SR_matrix(m,n)=c(m);
        SR_matrix(n,m)=mminus*c(m);
    end;
```

```
    SR_matrix(n,n)=-2*invt2*(2-1/(n-1))*SR_matrix(n-1,n-1);
end;

a=SR_matrix*b';
%-----------------------------------------
function
[BoxIndex,NonEmptyBox]=h5_SetDataStructure(SortedData,BoxSize,K)
N=length(SortedData);
BoxIndex=zeros(K,2);
NonEmptyBox=linspace(0,0,K);
i=1; j=0; box0=0;
while i<=N
    box1=h5_GetBox(SortedData(i),BoxSize);
    if box1 ~= box0
        j=j+1;
        box0=box1;
        BoxIndex(j,1)=i;      %bookmark for data
        NonEmptyBox(box1)=j;
    end;
    i=i+1;
end;
BoxIndex=BoxIndex(1:j,1:2);
for jj=1:j-1
    BoxIndex(jj,2)=BoxIndex(jj+1,1)-1;
end;
BoxIndex(j,2)=N;
%-----------------------------------------
function [time,V]=h5_standard(U)
%standard matrix vector multiplication
%homework 5

global X Y

time=cputime;

N=length(X);
M=length(Y);

for j=1:M
    V(j)=0;
    for i=1:N,
        V(j)=V(j)+U(i)*h5_Phi(Y(j),X(i));
    end;
end;

time=cputime-time;
```