1 Solution (Homework 1)

1. 

\[ v_i = \sum_{j=1}^{N} u_j (x_i - x_j)^n = \sum_{j=1}^{N} u_j \sum_{k=0}^{n} (-1)^k \binom{n}{k} x_i^{n-k} x_j^k = \]

\[ = \sum_{k=0}^{n} (-1)^k \binom{n}{k} x_i^{n-k} \sum_{j=1}^{N} u_j x_j^k = \sum_{k=0}^{n} (-1)^k c_k \binom{n}{k} x_i^{n-k}, \]

where

\[ c_k = \sum_{j=1}^{N} u_j x_j^k. \]  

2. Matlab codes are attached.

3. See Figure 1.

4. See Figure 2. This graph shows the performance of two codes. Squares and circles shows the results of computations and lines show the linear (for “Fast”) and quadratic (for “Standard”) dependences in the logarithmic coordinates.

5. See continuous lines on the previous graph. They are close to the computational results. So the standard method behaves as \( O(N^2) \) algorithm, while the “Fast” method behaves as \( O(N) \).
Figure 1:

Figure 2:
clear all;

npoint=9;
NN=floor(logspace(2,4,npoint));
n=5;

for i=1:npoint
    N=NN(i);
    X=rand(1,N);
    U=rand(1,N);
    [tm(i),Vm]=h1_fast_2003(X,U,n);

    if N<=1000
        [ts(i),Vs]=h1_standard_2003(X,U,n);
        Nst(i)=N;
        Nst2(i)=N*N;
        err(i)=max(abs(Vs-Vm));
    end;
end;

loglog(NN,tm,'o');
hold on;
loglog(NN,0.00015*NN,'r');
loglog(Nst,ts,'s');
loglog(Nst,0.000015*Nst2,'r');
figure;
loglog(Nst,err,'-*');
%ylim([1e-10,1]);
function [time,V]=h1_standard_2003(X,U,n)

%standard matrix vector multiplication
%homework 1

t=cputime;

N=length(U);

for i=1:N
    V(i)=U(i);
    for j=1:N,
        V(i)=V(i)+U(j)*(X(i)-X(j))^n;
    end;
end;

time=cputime-t;
function [time,V]=h1_fast_2003(X,U,n)

%middleman matrix vector multiplication 
%homework 1

t=cputime;
N=length(U);

% find coefficients c(k)

nf=factorial(n);

signum=-1;

for k=0:n 
    signum=-signum;
    sum=0;
    for j=1:N
        sum=sum+U(j)*X(j)^k;
    end;
    binomial=nf/(factorial(k)*factorial(n-k));
    c(k+1)=signum*binomial*sum;
end;

% final summation

for i=1:N
    V(i)=U(i);
    for k=0:n 
        V(i)=V(i)+c(k+1)*X(i)^(n-k);
    end;
end;

time=cputime-t;