EL 7133 (DSP II)
Spring 2016
Homework Assignment - Week 06

Name: **Amitesh Kumar Sah**
NYU ID: **N19714360**

Question B

B) As in HW 01, write a Matlab program to implement the forward and inverse wavelet transform with length-4 Daubechies filters. But this time, implement the transform using convolution, down-sampling, and up-sampling. You can use the Matlab function upfirdn. Numerically verify perfect reconstruction in Matlab. • To verify perfect reconstruction, plot the error signal. • Write the wavelet transform and inverse transform as two Matlab functions (each with input and output arguments). The user should be able to specify the number of wavelet levels. You can return the wavelet coefficients in a cell array to account for the subbands being of different lengths.

Program

```
clc;clear all;close all
x=[1 2 3 4 5 6 7 8 9 10 11 12];
%  x = dlmread('pwsmooth.txt');
 level=input('Level');
% level=4;
k=2;
 [c,d]=fordaub(x,level);

 m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 4 wavelet representation');

 y=invdaub(c,d,level);
 error=abs(x-y);

 figure,
 subplot(3,1,1),stem(x),title('Original signal');
 subplot(3,1,2),stem(y),title('Recovered signal');
 subplot(3,1,3),stem(error),title('Error signal');
```

Function for forward wavelet transform

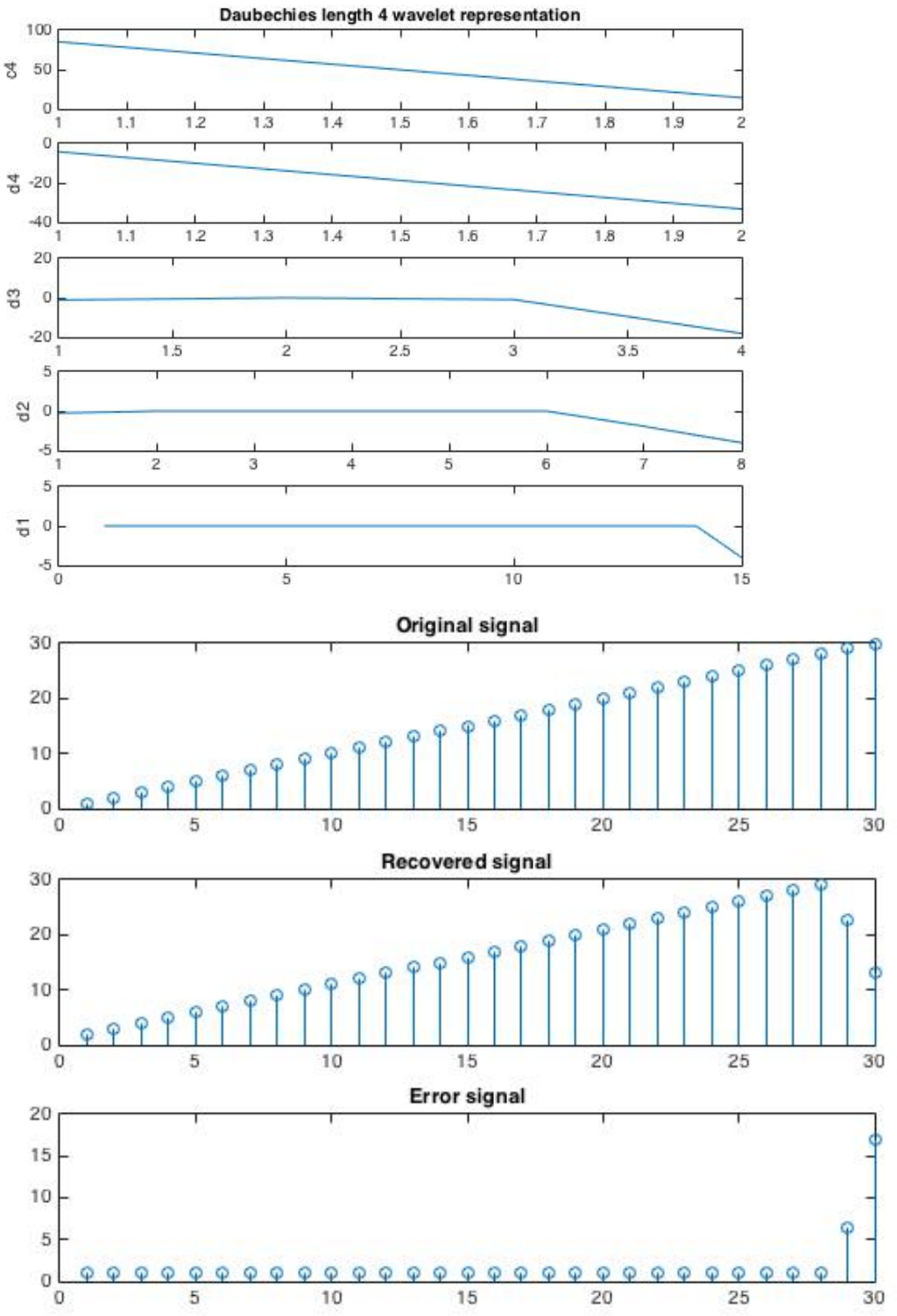
```
function [c,d]=fordaub(x,level)
% x=[1 2 3 4 5 6 7 8 9 10 11 12];
%  x = dlmread('pwsmooth.txt');
% level=4;

% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end

k=2;
```

```
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end

q=[-1/16 1/4 -1/16];
p=conv(p2,q);
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2))

c=cell(1,level);
d=cell(1,level);
xin=x;

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);
end

for i =1:level
c{1,i}=downsample(conv(xin,h,'same'),2);
d{1,i}=downsample(conv(xin,h1,'same'),2);
xin=c{1,i};
end
```

Function for Inverse Wavelet transform

```
function y=invdaub(c,d,level)
k=2;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end

q=[-1/16 1/4 -1/16];
p=conv(p2,q);
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2))

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);
end

h_inv=fliplr(h);
h1_inv=fliplr(h1);
for i =level:-1:1
y0{1,i}=conv(upsample(c{1,i},2),h_inv,'same');
y1{1,i}=conv(upsample(d{1,i},2),h1_inv,'same');
end

y=y0{1,1}+y1{1,1};
```

Result:



Question C

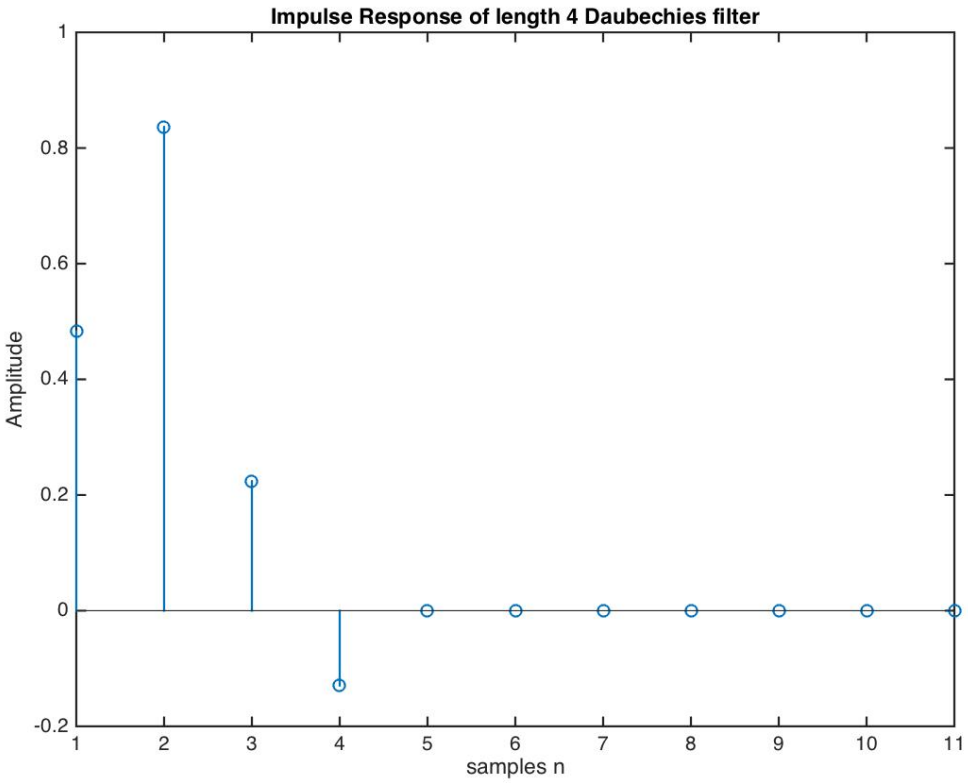
C) Construct Daubechies filters of lengths 4, 6, and 8. To obtain the low-pass filter, start with the symmetric half-band filters of the form Q(z) (z+1)^K from HW 03, then perform spectral factorization. Use K = 4, 6, and 8. To obtain the high-pass filter, use the relation in the notes (time-reverse and modulate). Verify that your length-4 filter is the same as the length-4 filter in the lecture notes from week 1. For each filter, show the impulse response, frequency response, and zero diagram. Numerically verify that your filters have the perfect reconstruction property when used in a filter bank or wavelet transform.

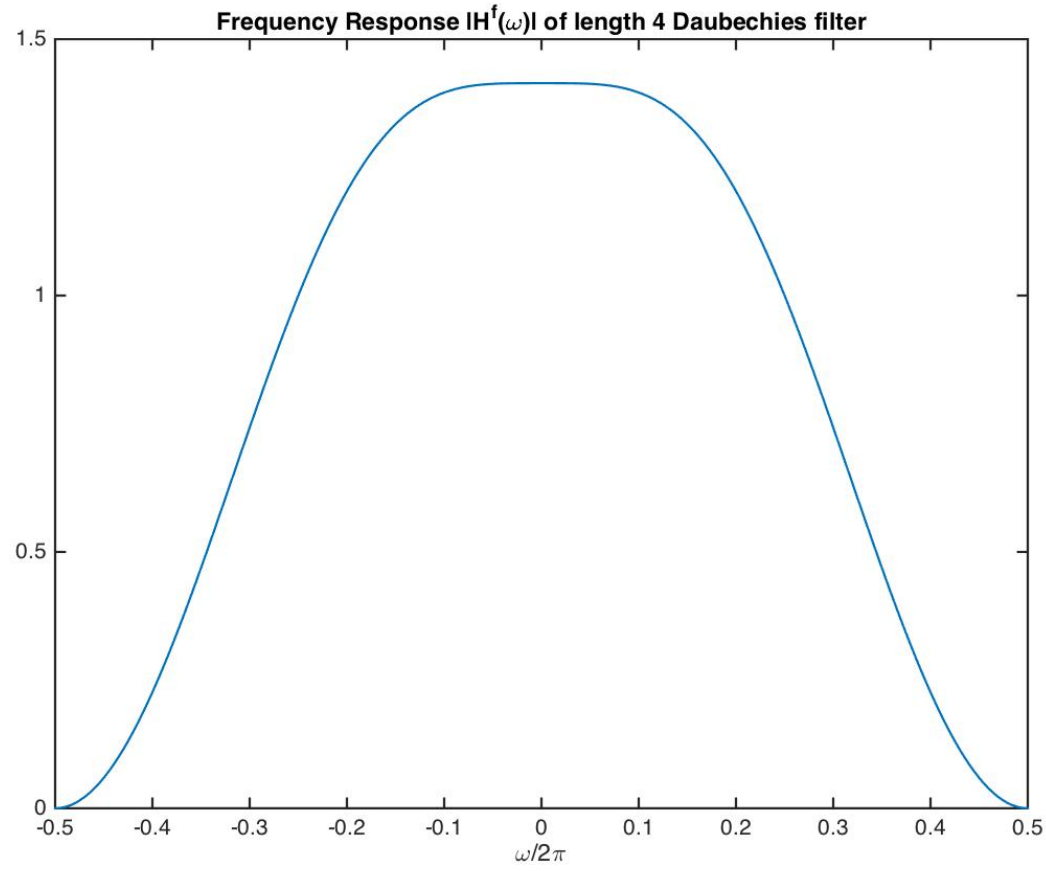
PROGRAM

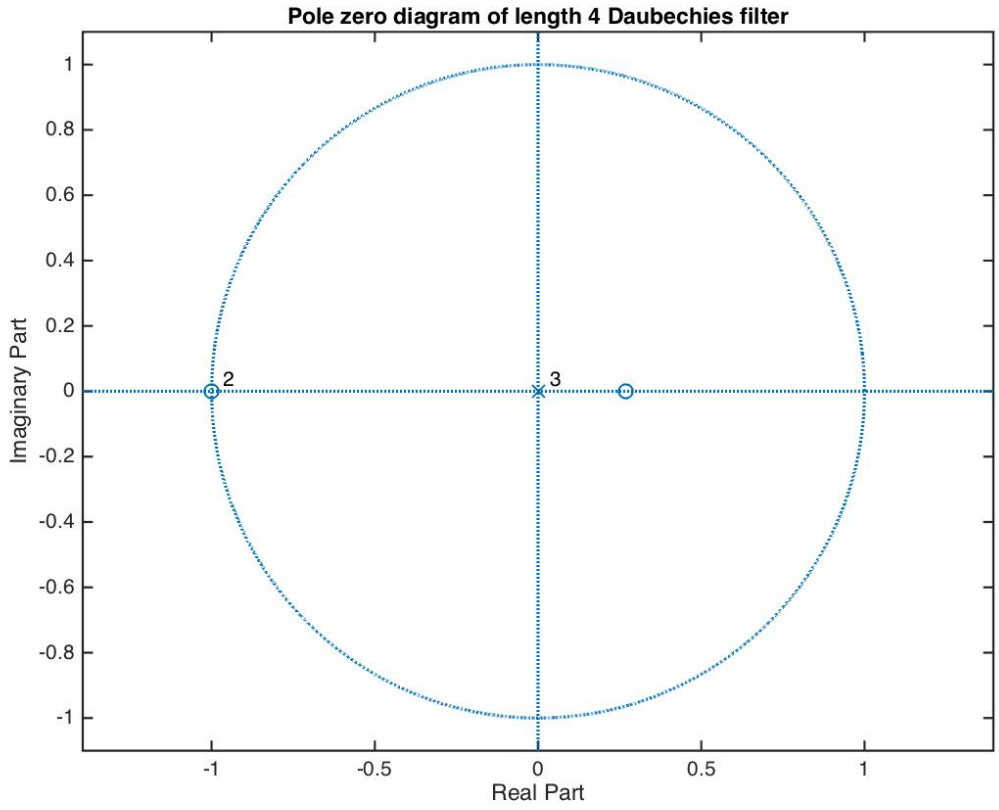
```
clc;clear all;close all

% Daubechies filters of lengths 4
k=2;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1)
end
q=[-1/16 1/4 -1/16];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

% Verification of length 4 filter
h0=(1+sqrt(3))/(4*sqrt(2));
h1=(3+sqrt(3))/(4*sqrt(2));
h2=(3-sqrt(3))/(4*sqrt(2));
h3=(1-sqrt(3))/(4*sqrt(2));

hv=[h0 h1 h2 h3];
T = table(['ho';'h1';'h2';'h3'],[h0;h1;h2;h3],h',...
    'VariableNames',{'Filter_coefficient' 'from_week_1' 'Obtained'})

% Plotting impulse response, frequency response and pole zero diagram
% Impulse Response
a=1;
b=h;
imp=[1 zeros(1,10)];
h=filter(b,a,imp);
figure,stem(h)
xlabel('samples n');
ylabel('Amplitude');
title('Impulse Response of length 4 Daubechies filter');
saveas(gcf,'Imp4.jpg')

% Frequency Response
j=sqrt(-1);
om=linspace(-pi,pi,200);
Hf=polyval(b,exp(j*om))./polyval(a,exp(j*om));
figure,plot(om/(2*pi),abs(Hf))
title('Frequency Response |H^f(\omega)| of length 4 Daubechies filter');
xlabel('\omega/2\pi');
saveas(gcf,'Fre4.jpg')

figure,
% Pole zero diagram
zplane(b,a)
title('Pole zero diagram of length 4 Daubechies filter');
saveas(gcf,'PZ4.jpg')
```

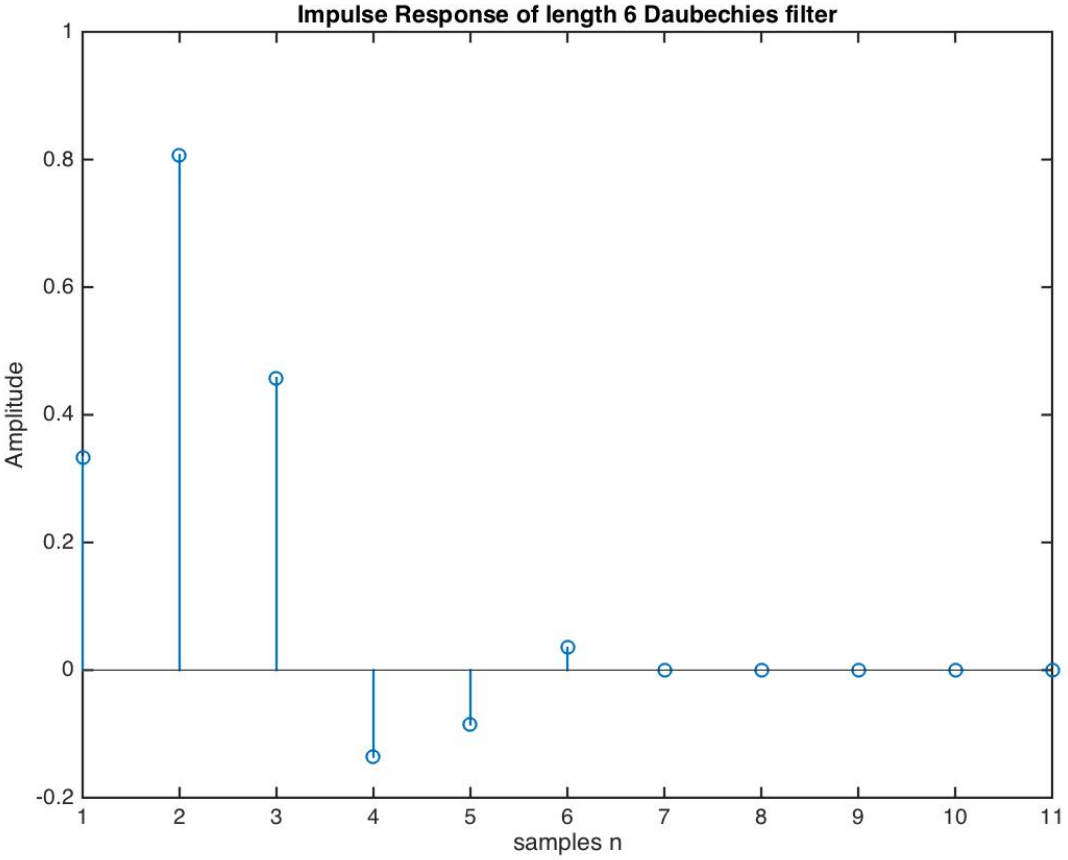
```
%%
% Daubechies filters of lengths 6
k=3;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1)
end
q=[0.0117 -0.0703 0.1484 -0.0703 0.0117];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

% Plotting impulse response, frequency response and pole zero diagram
% Impulse Response
a=1;
b=h;
imp=[1 zeros(1,10)];
h=filter(b,a,imp);
figure,stem(h)
xlabel('samples n');
ylabel('Amplitude');
title('Impulse Response of length 6 Daubechies filter');
saveas(gcf,'Imp6.jpg')

% Frequency Response
j=sqrt(-1);
om=linspace(-pi,pi,200);
Hf=polyval(b,exp(j*om))./polyval(a,exp(j*om));
figure,plot(om/(2*pi),abs(Hf))
title('Frequency Response |H^f(\omega)| of length 6 Daubechies filter');
xlabel('\omega/2\pi');
saveas(gcf,'Fre6.jpg')

figure,
% Pole zero diagram
zplane(b,a)
title('Pole zero diagram of length 6 Daubechies filter')
saveas(gcf,'PZ6.jpg')

%%
% Daubechies filters of lengths 8
k=4;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1)
end

q=[-0.0024 0.0195 -0.0640 0.1016 -0.0640 0.0195 -0.0024];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
```

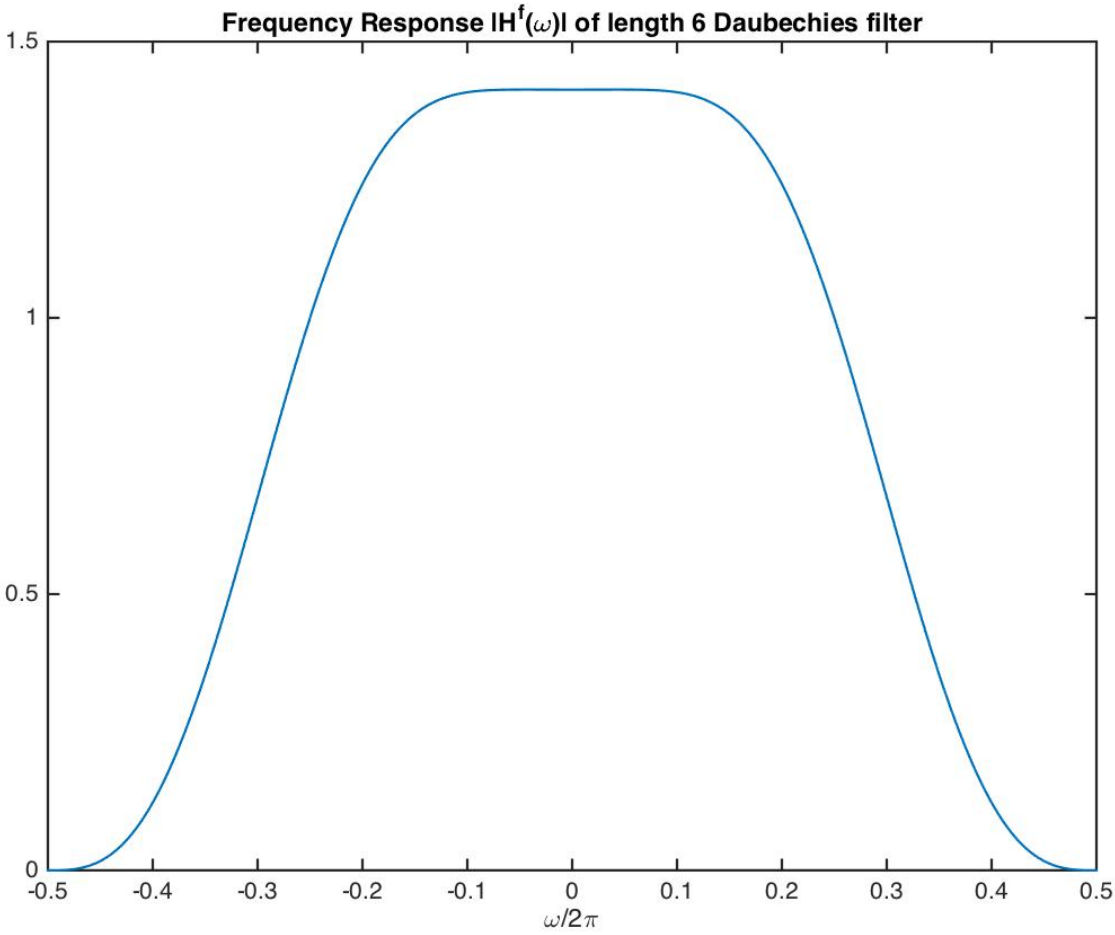
```
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

% Plotting impulse response, frequency response and pole zero diagram
% Impulse Response
a=1;
b=h;
imp=[1 zeros(1,10)];
h=filter(b,a,imp);
figure,stem(h)
xlabel('samples n');
ylabel('Amplitude');
title('Impulse Response of length 8 Daubechies filter');
saveas(gcf,'Imp8.jpg')

% Frequency Response
j=sqrt(-1);
om=linspace(-pi,pi,200);
Hf=polyval(b,exp(j*om))./polyval(a,exp(j*om));
figure,plot(om/(2*pi),abs(Hf))
title('Frequency Response |H^f(\omega)| of length 8 Daubechies filter');
xlabel('\omega/2\pi');
saveas(gcf,'Fre8.jpg')

figure,
% Pole zero diagram
zplane(b,a)
title('Pole zero diagram of length 8 Daubechies filter');
saveas(gcf,'PZ8.jpg')
```

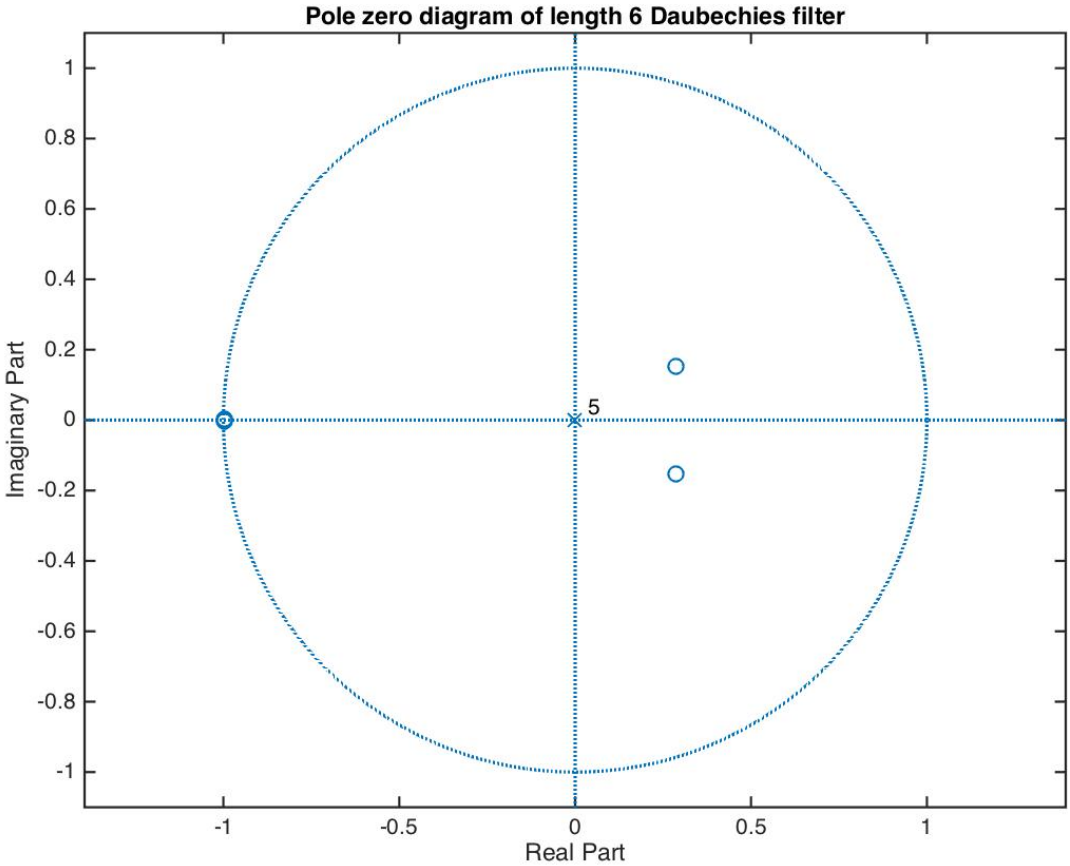

Impulse Response of length 4 Daubechies filter
Amplitude
samples n
1
0.8
0.6
0.4
0.2
0
-0.2
1
2
3
4
5
6
7
8
9
10
11

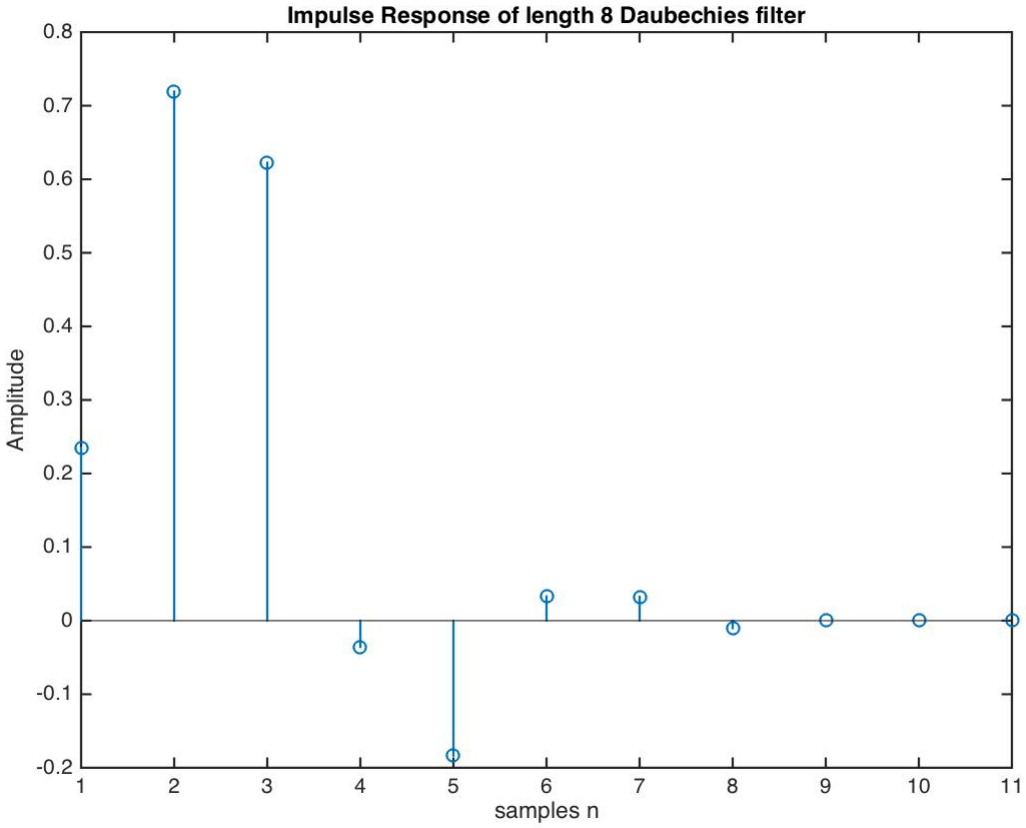

Frequency Response |H^f(ω)| of length 4 Daubechies filter
ω/2π
1.5
1
0.5
0
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5

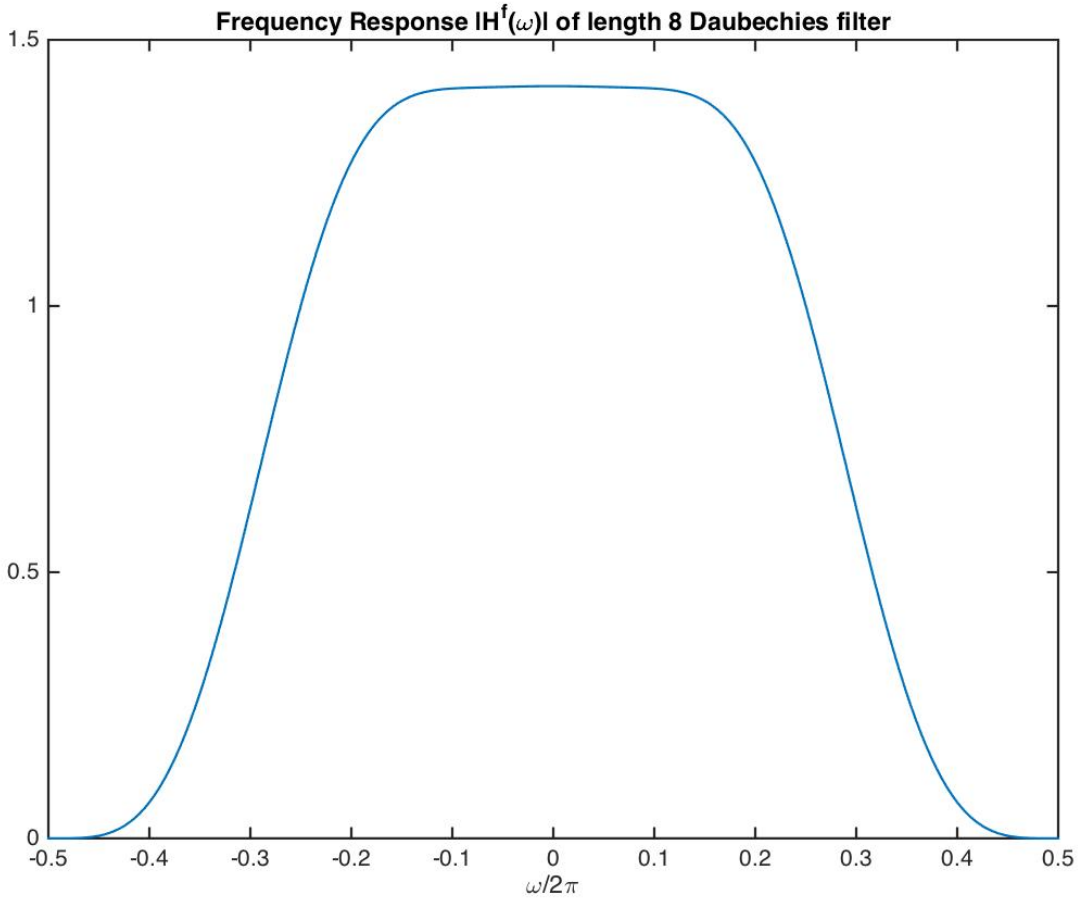

Pole zero diagram of length 4 Daubechies filter
Imaginary Part
Real Part
2
3
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1

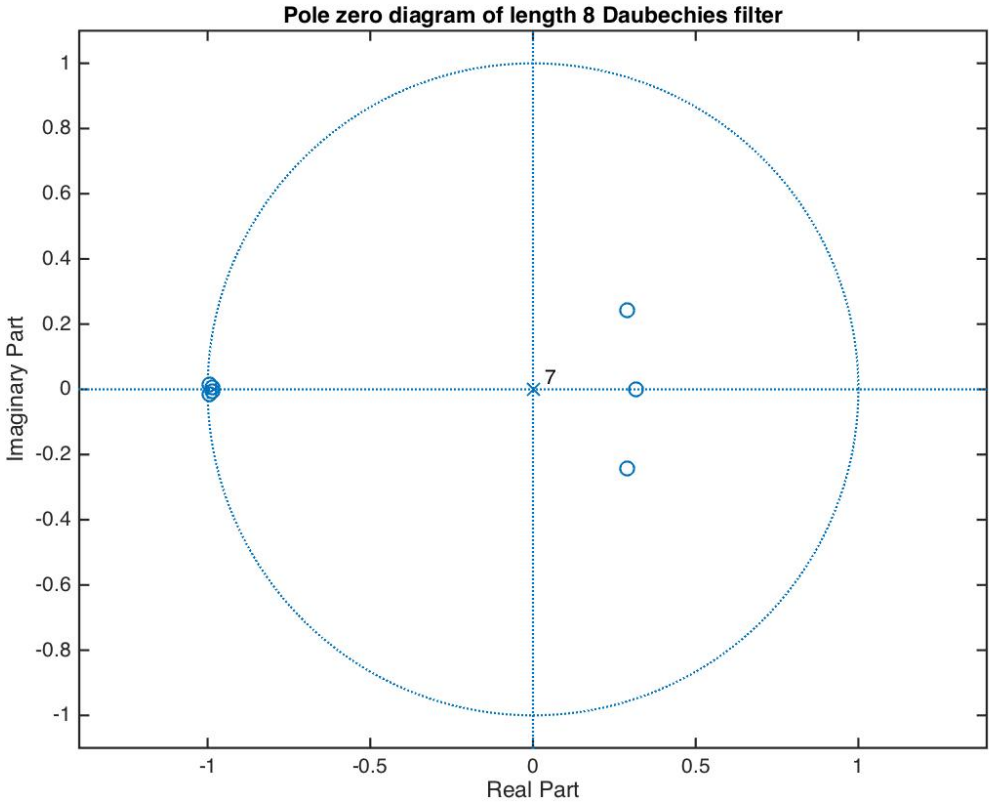

Impulse Response of length 6 Daubechies filter
Amplitude
samples n
1
0.8
0.6
0.4
0.2
0
-0.2
1
2
3
4
5
6
7
8
9
10
11


Frequency Response |H^f(ω)| of length 6 Daubechies filter
1.5
1
0.5
0
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
ω/2π


Pole zero diagram of length 6 Daubechies filter
Imaginary Part
Real Part
5
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


Impulse Response of length 8 Daubechies filter
Amplitude
samples n
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
1
2
3
4
5
6
7
8
9
10
11


Frequency Response |H^f(ω)| of length 8 Daubechies filter
ω/2π
1.5
1
0.5
0
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5



Verification of length 4 Daubechies filter

Filter_coefficient	from_week_1	Obtained
ho	0.48296	0.48295
h1	0.83652	0.83652
h2	0.22414	0.22416
h3	-0.12941	-0.12941

Question D

D) Using each of the Daubechies filters from (C), plot the wavelet representation of the signals bumps and pwsmooth. Use 5 levels of the wavelet transform.

Program

```
% Length 4 Daubechies filter level 5 , Pwsmooth input file
clc;clear all;close all
x = dlmread('pwsmooth.txt');
level=5;

% padding of zero if the signal is odd length
if mod(length(x),2)~=0
```

```
    x(1,end+1)=0;
end

k=2;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end
q=[-1/16 1/4 -1/16];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 4 wavelet representation for pwsmooth');
     saveas(gcf,'pwsmooth4.jpg')


%%
 % Length 4 Daubechies filter level 5 , bumps input file
clear all
x = dlmread('bumps.txt');
level=5;
% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end

k=2;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end
q=[-1/16 1/4 -1/16];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
```

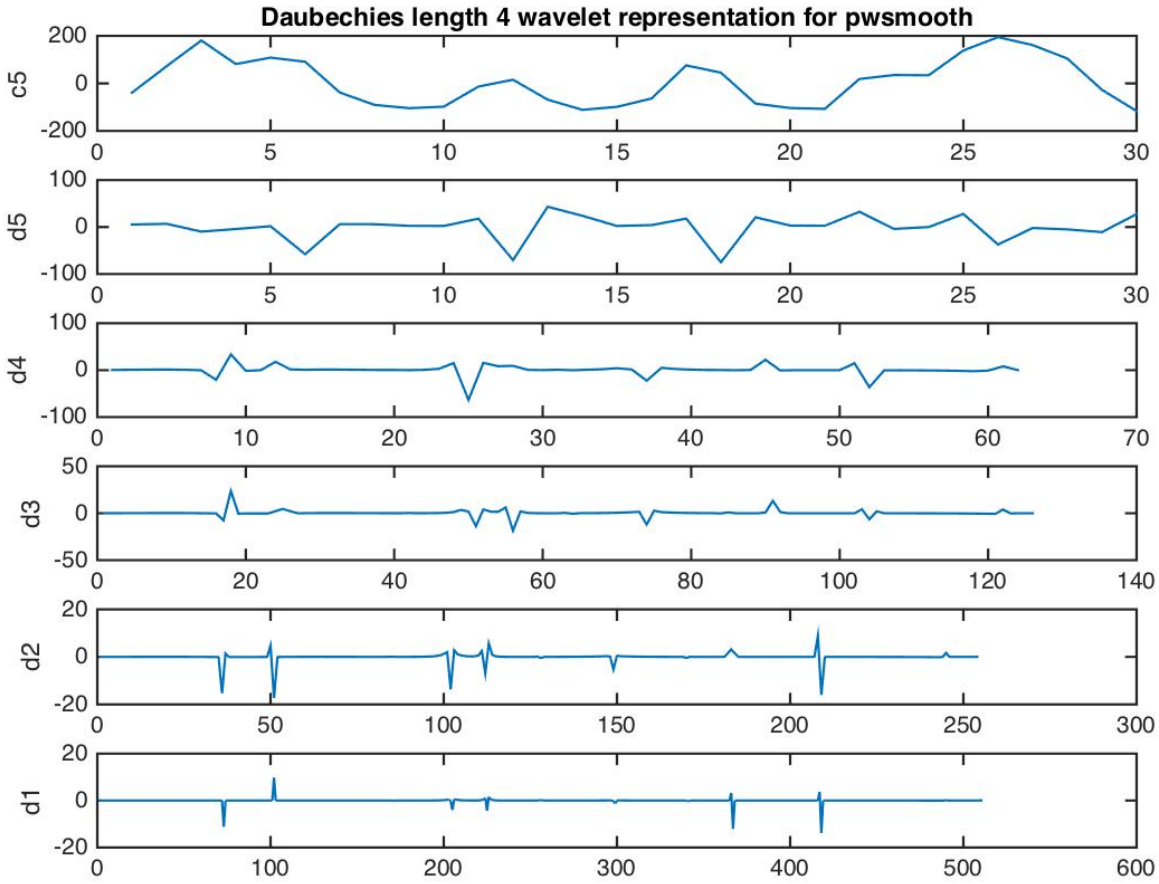
```
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 4 wavelet representation for bumps');
     saveas(gcf,'bumps4.jpg')

%%
 % Length 6 Daubechies filter level 5 , Pwsmooth input file
 clear all;
 clear all
x = dlmread('pwsmooth.txt');
level=5;
% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end

k=3;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end
q=[0.0117 -0.0703 0.1484 -0.0703 0.0117];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
```

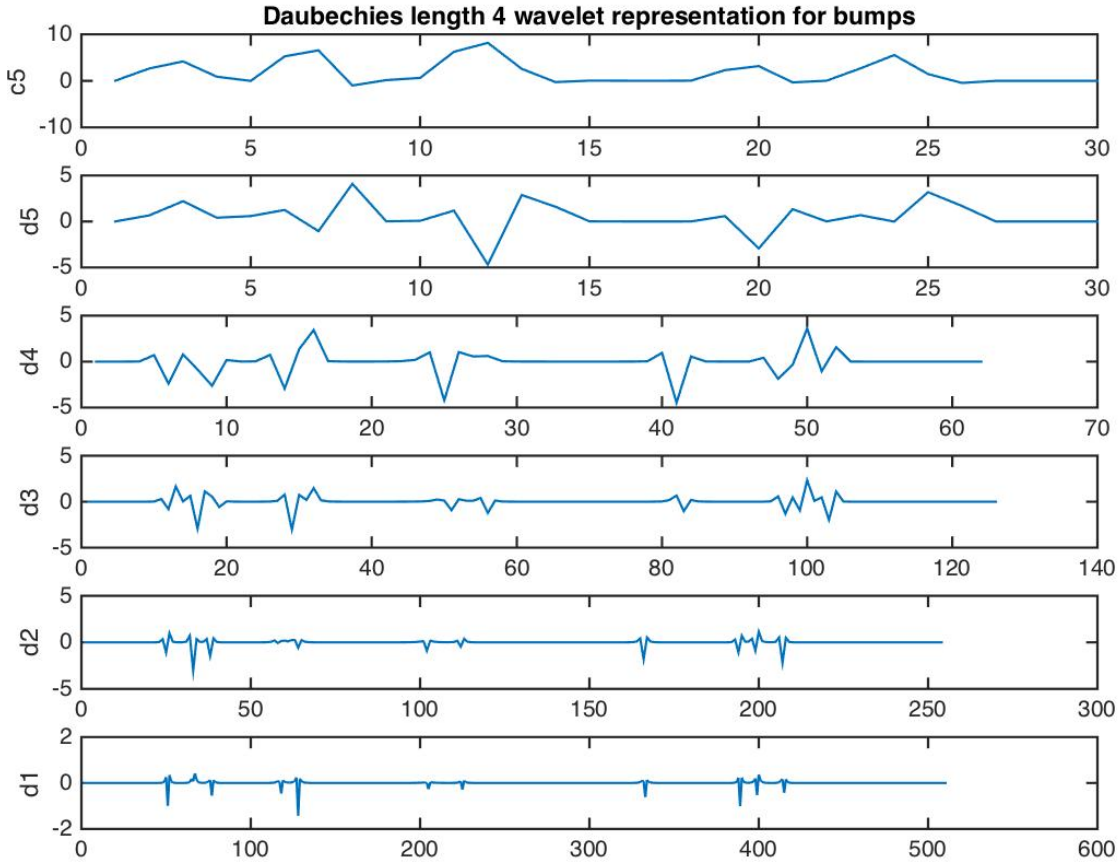
```
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 6 wavelet representation for pwsmooth');
     saveas(gcf,'pwsmooth6.jpg')

 %%
 % Length 6 Daubechies filter level 5 , Bumps input file
 clear all;
 clear all
x = dlmread('bumps.txt');
level=5;
% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end



k=3;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end
q=[0.0117 -0.0703 0.1484 -0.0703 0.0117];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
```

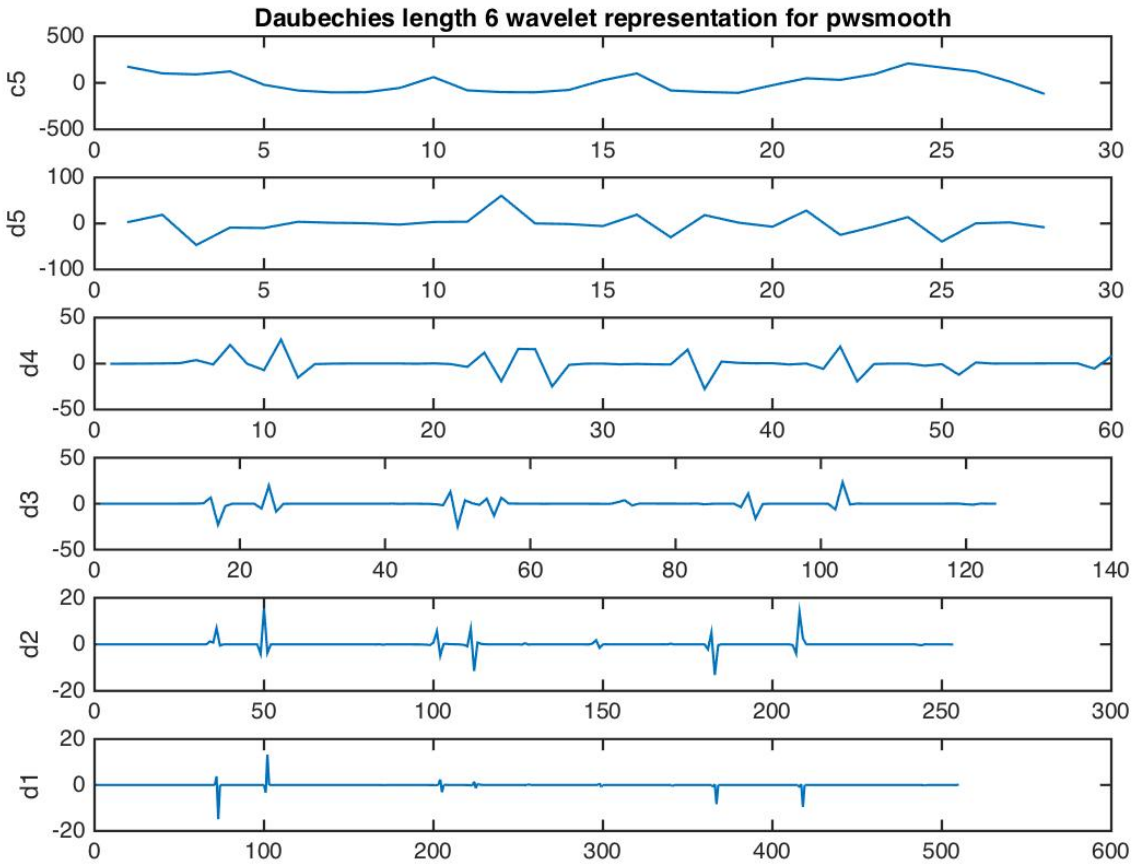
```
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 6 wavelet representation for bumps');
     saveas(gcf,'bumps6.jpg')
     %%

 % Length 8 Daubechies filter level 5 , Pwsmooth input file
 clear all;
 clear all
x = dlmread('pwsmooth.txt');
level=5;
% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end

k=4;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end

q=[-0.0024 0.0195 -0.0640 0.1016 -0.0640 0.0195 -0.0024];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 8 wavelet representation for pwsmooth');
     saveas(gcf,'pwsmooth8.jpg')
    %%
 % Length 8 Daubechies filter level 5 , Bumps input file
 clear all;
 clear all
x = dlmread('bumps.txt');
level=5;
% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
```

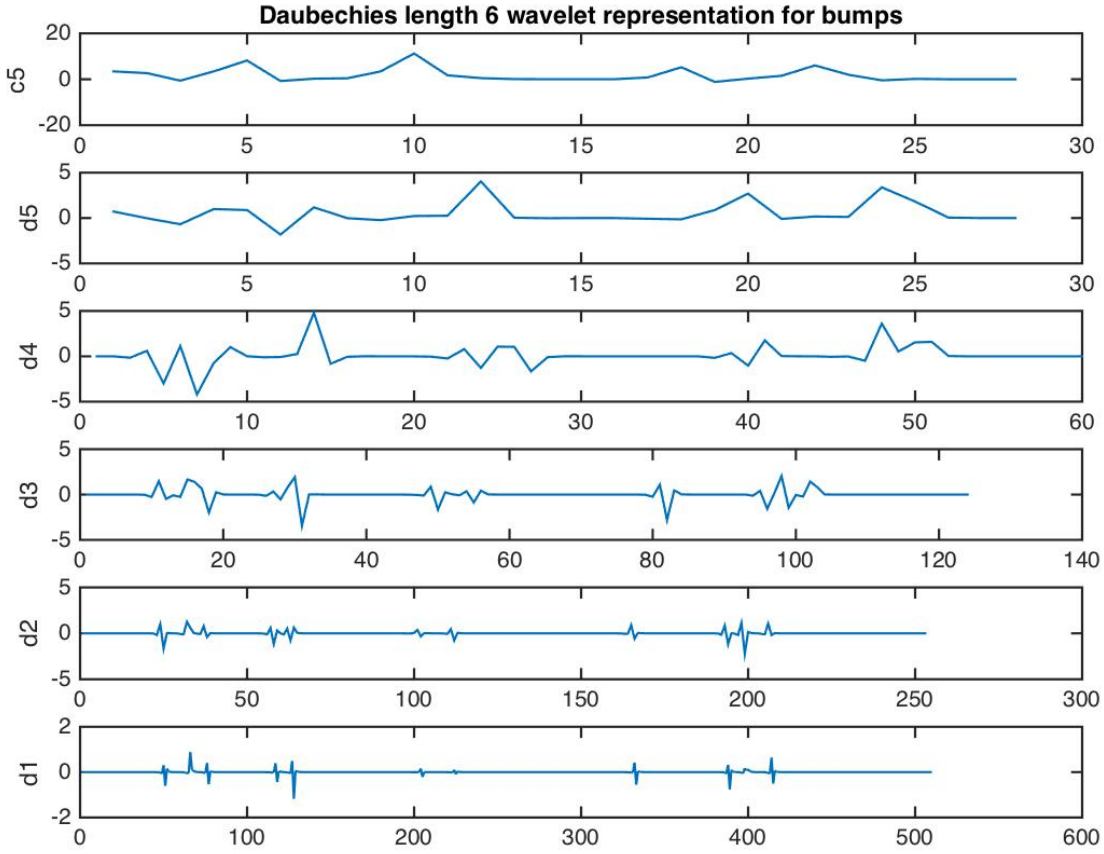
```
end

k=4;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end

q=[-0.0024 0.0195 -0.0640 0.1016 -0.0640 0.0195 -0.0024];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)) % h is a low pass filter

l=length(h);
l1=1;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

xin=x;
for i =1:level
c{1,i}=downsample(conv(xin,h,'valid'),2);
d{1,i}=downsample(conv(xin,h1,'valid'),2);
xin=c{1,i};
end

m=level;
 figure,
for j=1:level
    x_axis=1:1:length(d{1,m});
    subplot(level+1,1,j+1),plot(x_axis,d{1,m}),ylabel(['d',num2str(m)]);
    m=m-1;
end
    x_axis=1:1:length(c{1,level});
     subplot(level+1,1,1),plot(x_axis,c{1,level}),ylabel(['c',num2str(level)]);
     title('Daubechies length 8 wavelet representation for bumps');
     saveas(gcf,'bumps8.jpg')
```

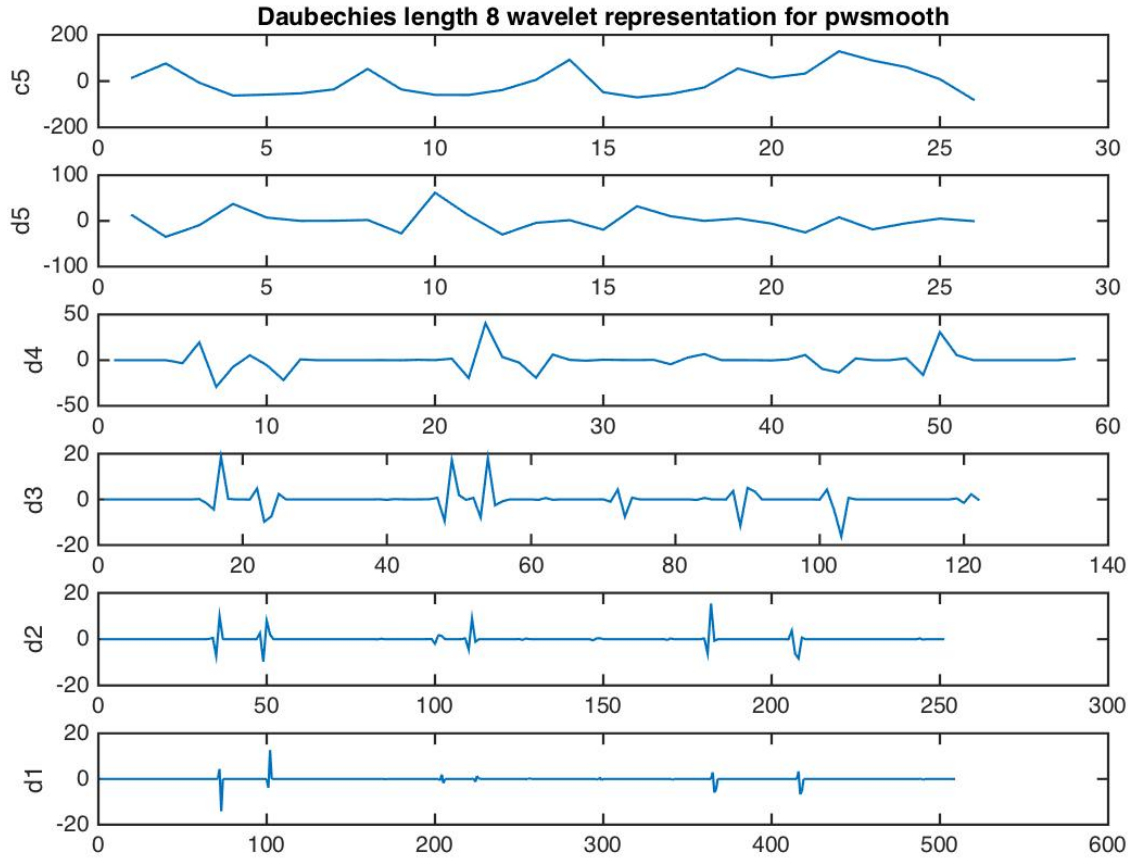
Output:

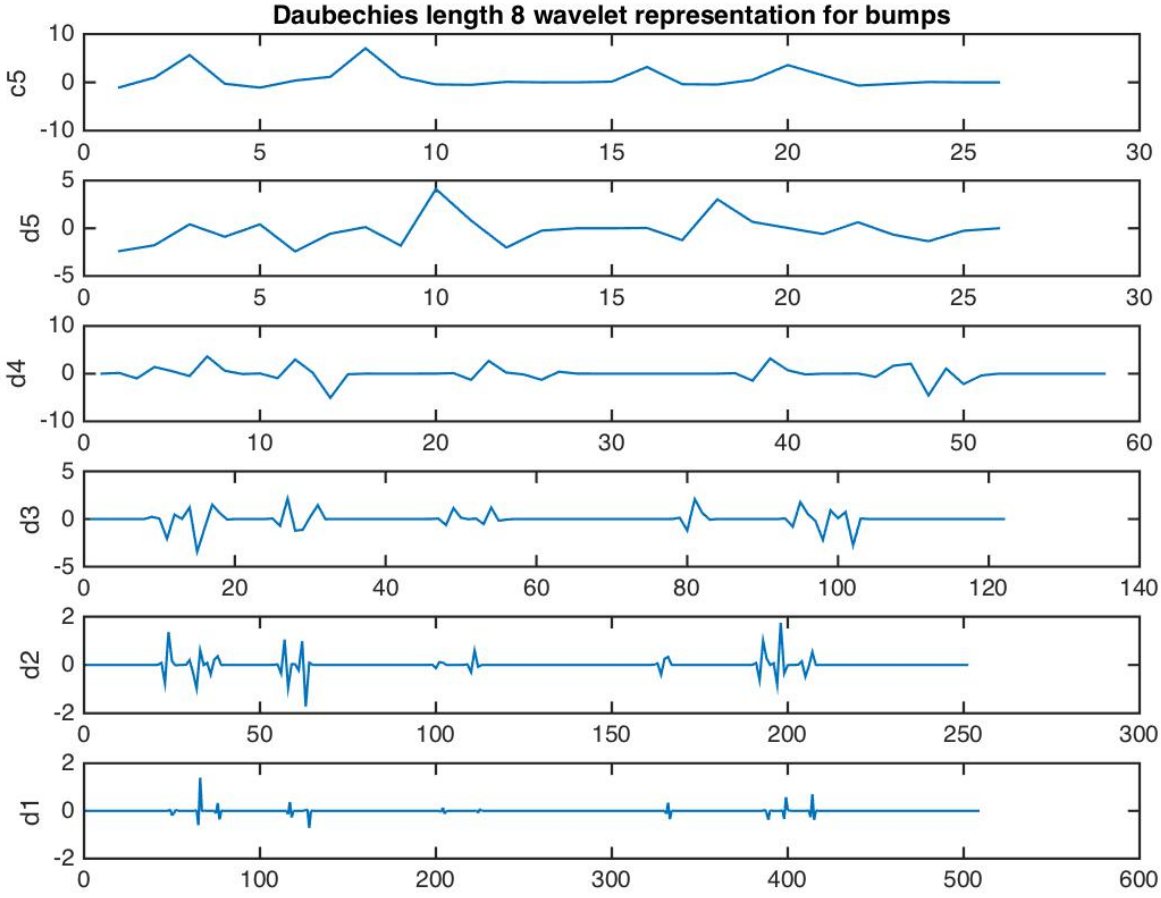

Daubechies length 4 wavelet representation for pwsmooth
c5
d5
d4
d3
d2
d1


Daubechies length 4 wavelet representation for bumps
c5
d5
d4
d3
d2
d1


Daubechies length 6 wavelet representation for pwsmooth
c5
d5
d4
d3
d2
d1


Daubechies length 6 wavelet representation for bumps
c5
d5
d4
d3
d2
d1


Daubechies length 8 wavelet representation for pwsmooth
c5
d5
d4
d3
d2
d1


Daubechies length 8 wavelet representation for bumps
c5
d5
d4
d3
d2
d1

Question E

E) Wavelet functions. Using Noble Identities, the path from the input to the high-pass output at each level of a wavelet transform is equivalent to --> F(z) --> down(M) -->. See Exercise 11.28 in the DSP Exercise packet assigned in HW 4 for the case of a 3-level transform. 1.) Identify F(z) and M (general formulas for any level). 2.) For a length-4 Daubechies filter, plot the impulse response f(n) for each level up to level 10. I suggest to use subplot(5, 2, n) and 'orient tall' in Matlab to place the plots on a single page. 3.) Repeat 2.) for length-6 Daubechies filter. 4.) Repeat 2.) for length-8 Daubechies filter. Comment on your observations

```
%% 1.) Identify F(z) and M (general formulas for any level).
% Haar filter
clc;clear all;close all
x=1:1:20;
% x = dlmread('pwsmooth.txt');
level=input('Enter the level of haar transform coeff: ');

% padding of zero if the signal is odd length
if mod(length(x),2)~=0
    x(1,end+1)=0;
end

h=[0.5 0.5];
l=length(h);
l1=3;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

% determining number of low pass and high pass filter used
nl=level-1;
nh=1;
nd=power(2,level);

% Equivalent filter F(z)
u=2;
if level==1
    f=h1
else
   for i=2:nl
       hi=upsample(h,u);
       h=conv(h,hi);
       u=power(u,2);
   end
   f=conv(h,upsample(h1,power(2,nl)))
end
figure, stem(f);
disp(['down(M) = ',num2str(nd)]);
```

Output:

Enter the level of haar transform coeff: 1
f = -0.5000 0.5000
down(M) = 2

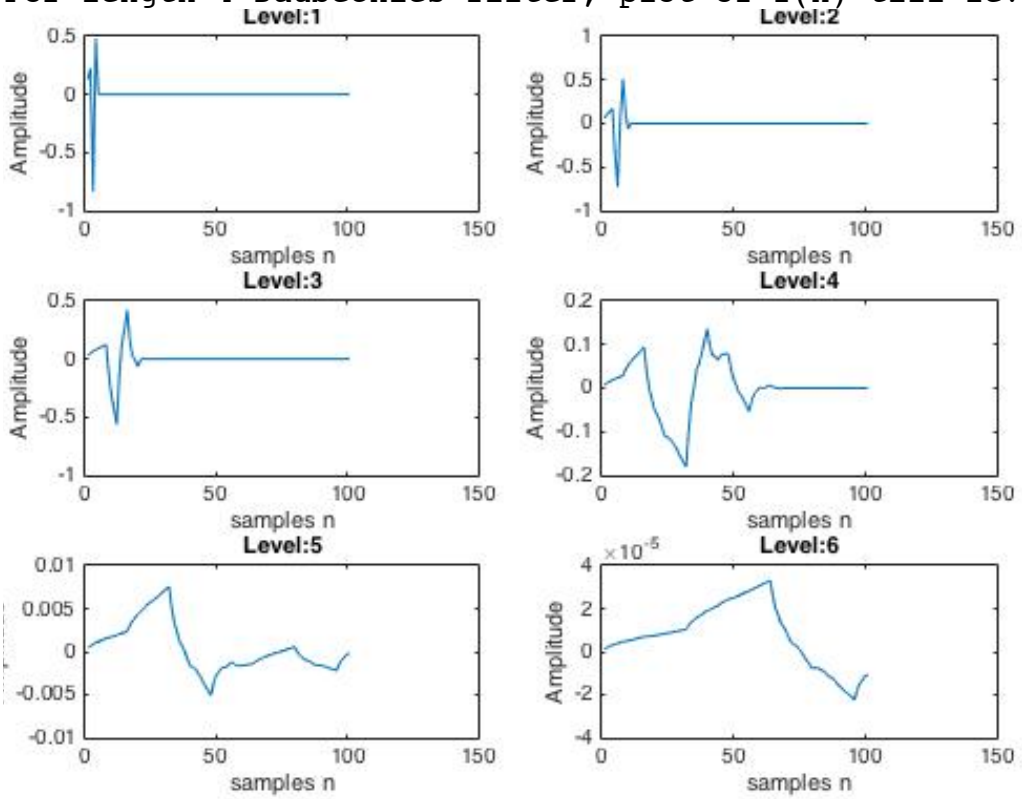
Enter the level of haar transform coeff: 2
f =-0.2500 -0.2500 0.2500 0.2500 0
down(M) = 4

Enter the level of haar transform coeff: 3
f =-0.1250 -0.1250 -0.1250 -0.1250 0.1250 0.1250 0.1250 0.1250 0 0 0 0
down(M) = 8

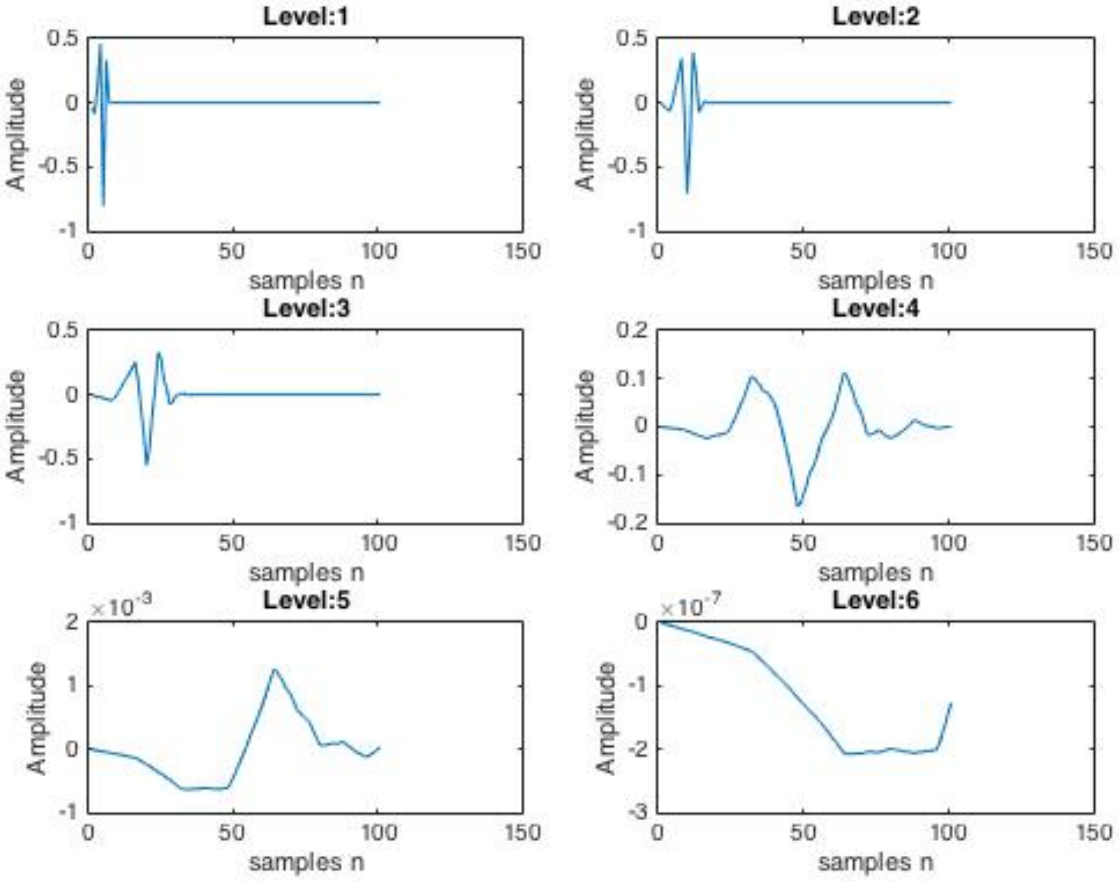
```
%% 2.) For a length-4 Daubechies filter, plot the impulse response f(n) for each level
up to level 6.

% length-4 Daubechies filter
clc;clear all;close all
k=2;
p1=[1 1];
p2=p1;
for i=1:2*k-1
    p2=conv(p2,p1);
end
q=[-1/16 1/4 -1/16];
p=conv(p2,q)
r=seprts(p);
h=poly(r);
h=real(h); % (discard zero imag part)
h=h*sqrt(max(p)/sum(abs(h).^2)); % h is a low pass filter

l=length(h);
l1=3;
h1=h(end:-1:1);
for n=1:l
h1(n)=power(-1,n-l1+1)*h1(n);     %h1 is a highpass filter
end

% Equivalent filter F(z)
for j=1:6
    level=j;
u=2;
h2=h;
if level==1
    f=h1;
else
   for i=2:level-1
       hi=upsample(h,u);
       h=conv(h,hi);
       u=power(u,2);
   end
   f=conv(h,upsample(h1,power(2,level-1)));
end

a=1;
b=f;
imp=[1 zeros(1,10)];
f1=filter(b,a,imp);
subplot(3,2,j),stem(f1);
xlabel('samples n');
ylabel('Amplitude');
title(['Level:',num2str(level)]);
clearvars h;
clearvars f
clearvars hi
clearvars b
h=h2;
end
```

```
saveas(gcf,'1.Dau4.jpg')
```

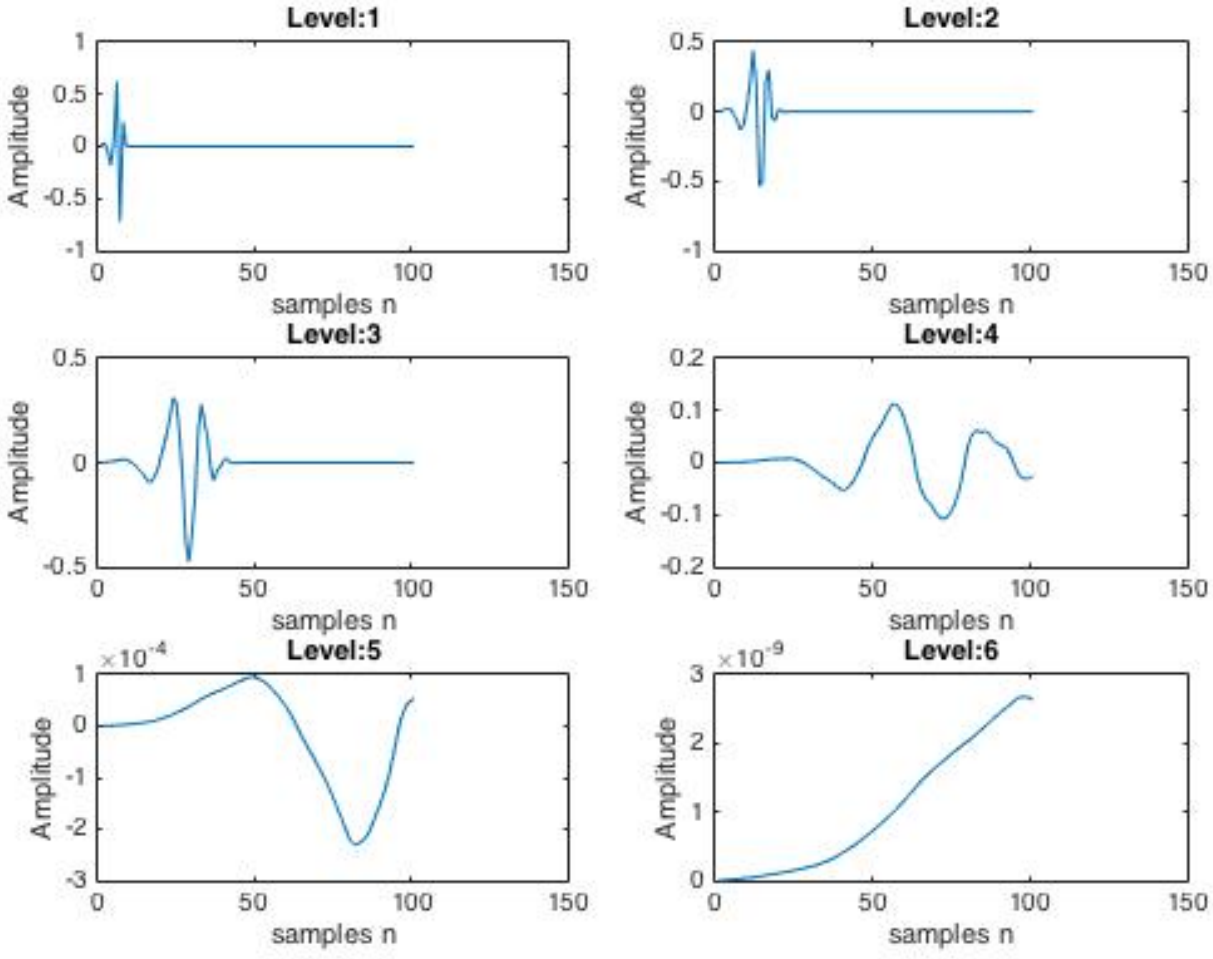
For length-4 Daubechies filter, plot of f(n) till level 6



For length-6 Daubechies filter, plot of f(n) till level 6



For length-8 Daubechies filter, plot of f(n) till level 6



For length 4
K=2,
Q=[-1/16 1/4 -1/16];

For length 6
K=3,
Q=[0.0117 -0.0703 0.1484 -0.0703 0.0117];

For length 8
K=4
Q=[-0.0024 0.0195 -0.0640 0.1016 -0.0640 0.0195 -0.0024];

Observation:
As the level increases, it appears that the filter is getting stretched. It stretches to a long samples . However, its magnitude decreases as the level increases.