Machine Learning for Signal Processing and Pattern Classification.
DISCRETE WAVELET TRANSFORM (DWT) – (Reconstruction)
SIGNAL REPRESNTATION USING WAVLET AND FOURIER COEFICIENTS

SIMRAN

1.

AIM:

To explore the implementation of wavelet decomposition, reconstruction and to give our understanding of the process involved in the wavelet decomposition and reconstruction.

CODE 1:

```
clc;
clear all;
close all;
x=[4 2 6 -2 4 6 2 2];

% Wavelet Decomposition
[c s]=wavedec(x,1,'haar'); % c is wavelet coefficients(length
will be same as sequence legnth
c
% 1 is the level of decompostion.
[W1, W2]=wfilters('haar','d'); %W1 is low pass filter and W2 is
high pass filter
c1=conv(x,(W1)); %conv of sequence with low pass filter
c2=conv(x,(W2)); %conv of sequence with high pass filter
c1_f=c1(2:2:end); % down sampling
c2_f=c2(2:2:end); % down sampling
C=[c1_f c2_f] % apending the down sampled values. c and C are
same

% Wavelet Reconstrcution
x1=waverec(c,s,'haar'); %reconstructing the wave using waverec
[R1, R2]=wfilters('haar','r'); % R1,R2 are low pass and high
pass filter resp.
r1 = upsample(c1_f,2); %upsampling the down sampled values(which
came from low pass) making the sequence big again
r2 = upsample(c2_f,2); %upsampling the down sampled values(which
came from high pass) making the sequence big again
r1_f=conv(r1,R1); % convolution between upsampled vector and low
pass filter
r2_f=conv(r2,R2); % convolution between upsampled vector and
high pass filter
X=r1_f+r2_f % adding to get the original value
```

```
x % X == x So sequence was reconstructed properly
```

Wavelet decompostion
Fourier can do one level decomposition only whereas wavelet can do more levels of decomposition so features extraction will be more. The maximum level of decomposition in wavelet depends on the length of the sequence. N = 2^K where N is the length of sequence and K is the level of decomposition.
Steps:

1. Convolution of the sequence with low pass and high pass filters separatively. Size of so formed vector is m+n-1 where m is the size of the sequence and n is the size of filter
2. Down sample the vectors got from 1.(take only even values)
3. Appending the down sampled vectors to get the coefficients

For filters matlab command wfilters can be used
To verify the coefficients wavedec command is used as c will give the coefficients in wavedec command.

Wavelet Reconstruction

Steps:

1. upsample the vectors
2. Convolution of the upsampled sequences with low pass and high pass filters.
3. Adding the so formed vectors
4. Or use waverec command in the matlab

OUTPUT :

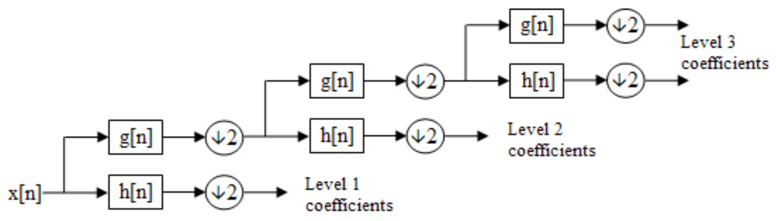
```
c =

 Columns 1 through 5

   4.2426    2.8284    7.0711    2.8284    1.4142

 Columns 6 through 8

   5.6569   -1.4142         0

C =

 Columns 1 through 5
```

```
   4.2426    2.8284    7.0711    2.8284    1.4142

  Columns 6 through 8

   5.6569   -1.4142         0

X =

  Columns 1 through 5

   4.0000    2.0000    6.0000   -2.0000    4.0000

  Columns 6 through 9

   6.0000    2.0000    2.0000         0

x =

    4    2    6   -2    4    6    2    2
```

2.

AIM:

To write a function file to decompose the signal to ‘k’ levels using the two band wavelet decomposition filters called ‘haar’.



CODE:

```
function [sol ,l] = wavdecomposition(x,Len)
if(floor((log2(length(x))))==(log2(length(x))))
    c=[];
   %haar wavelets decomposition and reconstruction filters
   [ld ,hd ,lr, hr] =wfilters('haar');
   s1=[length(x)]; % preserving the sequence length
   ca=[]; % approximation coefficients
 cd=[];% decomposition coefficients
 for L=1:Len
    %sequence is 1:convolve sequence is summed to get ca and cd.
   if(length(ca) == 1)
       ca =sum( conv(x,ld));
```

```
        cd = sum(conv(x,hd));
    else
    %if not convolve with low pass filter and then downsampled to
get cd
    %and convolve with high pass filter and then downsampled to
get cv
         ca = downsample(conv(x,ld),2,1);
         cd = downsample(conv(x,hd),2,1);
    end
    %lengths are preserved
     l=[length(ca) length(cd) s1];
     s1=l(2:end);
     c=[cd c];
     sol=[ca c];
     x=ca; % approximation coefficients go to further
decomposition
  end
  else
     disp('invalid input')
  end

x=1:8
[s,l] = wavdecomposition(x,2)
[s,l]=wavedec(x,2,'haar')
```

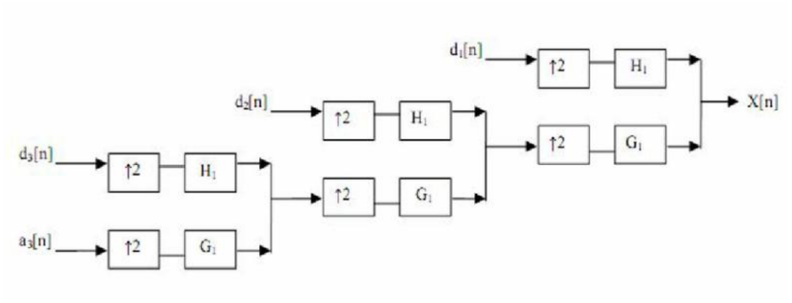
OUTPUT:

```
x =

 Columns 1 through 8

   1   2   3   4   5   6   7   8

s =

 Columns 1 through 4

   5.0000   13.0000   -2.0000   -2.0000

 Columns 5 through 8

  -0.7071  -0.7071  -0.7071  -0.7071

l =
   2    2    4    8

s =

 Columns 1 through 4

   5.0000   13.0000   -2.0000   -2.0000

 Columns 5 through 8

  -0.7071   -0.7071   -0.7071   -0.7071

l =

   2    2    4    8
```

3.

AIM:

To write a function file to reconstruct the decomposed signal using the two band wavelet reconstruction filters called ‘haar’.



CODE:

```
function dec = wavrecon(s,l)
%haar wavelets decomposition and reconstruction filters
[ld ,hd ,lr, hr] =wfilters('haar');
dec=s;
le = l(1:length(l)-1); %removing the length of input signal
for k = 1: length(l)-2
    %ca:approxiation coefficients
    %cd:decomposition coefficients
    ca = dec(1:le(1)); % the approximation coefficients(Low
frequency) of decomposed signal are taken
    cd = dec(le(1)+1 : (le(1)+1)+le(2)-1);% the decomposition
coefficients(high frequency)of decomposed signal are taken
    %upsample the ca.
    %convolve with the respective low pass and high pass
reconstruction filters
    a = conv(upsample(ca,2),lr);
    b = conv(upsample(cd ,2) ,hr);
    %addition of coefficients to get the previous level
approximation coefficients
    ca=a+b;
    %removing the zeros
    ca=ca(1:find(ca,1,'last'));
    if(ca(1)==ca(2)) %if the coefficients are equal retain the
coefficients.
        ca = ca(1);
    end
    cd=dec((le(1)+1)+le(2):end);
    dec=[ca cd];% appending to get the previous level cd and cv
    le = [length(ca) le(3:end)]; %previous level lengths
end

x
xrec=wavrecon(s,l)
```

OUTPUT:

```
x =

   1   2   3   4   5   6   7   8

xrec =

  Columns 1 through 5

    1.0000    2.0000    3.0000    4.0000    5.0000

  Columns 6 through 8

    6.0000    7.0000    8.0000
```

4.

AIM:

To implement the code in Matlab (Signal Representation using Fourier and wavelets) and give our analysis on the results.

A. Load the piece-regular signal of length 256.
B. Compute the fft coefficients of the signal using the built-in matlab command.
C. Make the fft coefficients from location 81 to 256 as ‘zero’. (i.e., we are using only the 80 fft coefficients. In order to make the length same, just make all other coefficients to be ‘zero’).
D. Reconstruct the signal using the fft coefficients, in which the values of the location from 81 to 256 are zero.
E. Compute the wavelet coefficient’s using ‘haar’ filter subjected to 4 levels of decomposition.
F. Retain only the 80 wavelet coefficients and make rest of all the other coefficients as ‘zero’.
G. Reconstruct the signal using the coefficients obtained in the previous step.
H. Compute the PSNR of the reconstructed signal using Fourier coefficients with reference to the original signal (use the built-in command ‘psnr’).
I. Compute the PSNR of the reconstructed signal using wavelet coefficients with reference to the original signal.
J. Plot the original signal, Fourier and wavelet reconstructed signal in a single figure window.

CODE:

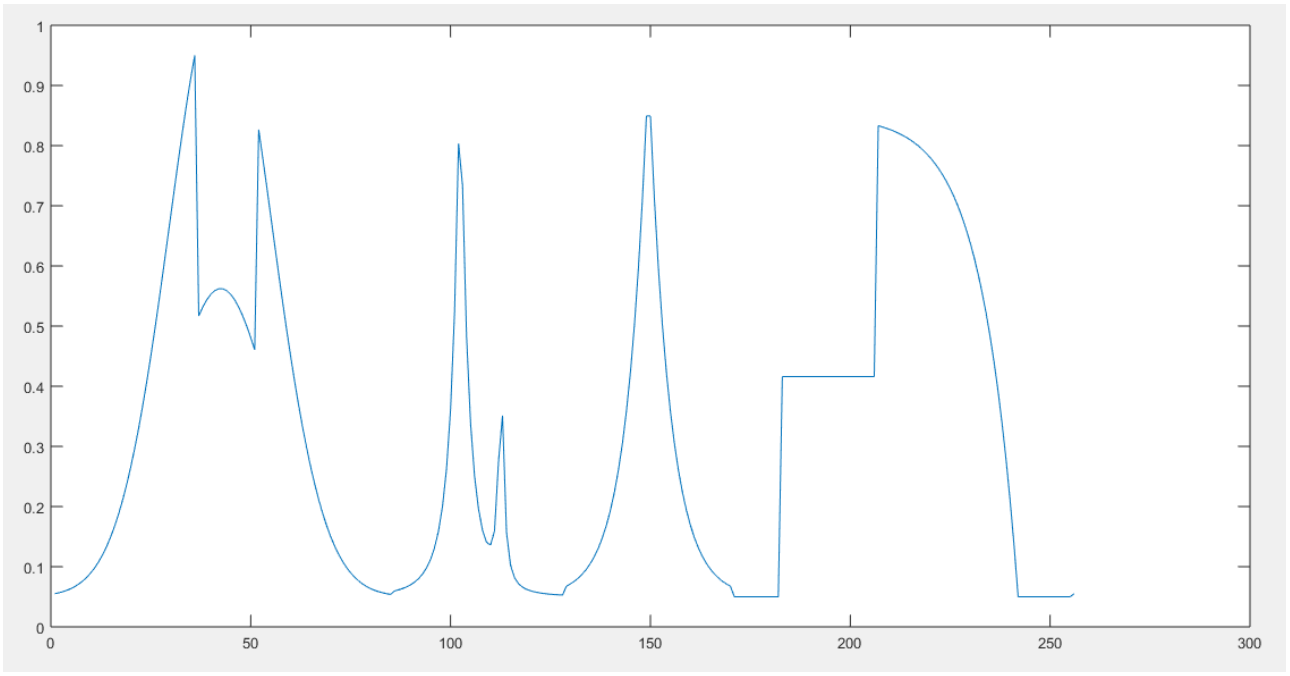
```
clc;
clear all
close all

% A.
n=256; % size is 256
axis([1 n 0 1]); % makes axis x from 0 to 256, y from 0 to 1
x=MakeSignal('Piece-Regular',n); % making signal of size 256
x=rescale(x,0.05,0.95); %rescaling  x
plot(x) % plotting x

% B.
x1=fft(x) % computing fft of original signal
figure
stem(x1) % plotting the coefficients

% C.
x1(81:256)=0  % making coefficients between 81 to 256 as 0.
figure
stem(x1) %plotting changed coefficients

% D.
xf=ifft(x1); % reconstructing the signal
figure
plot(real(xf)) % plotting the reconstructed signal

% E.
[c s]=wavedec(x,4,'haar'); % c is wavelet coefficients(length
will be same as sequence length
```

```
figure
stem(c) % plotting the coefficients

% F.
c(81:256)=0 % making coefficients between 81 and 256 as 0
figure
stem(c) %plotting changed coefficients

% G.
xw=waverec(c,s,'haar'); %reconstructing the signal using waverec
figure
plot(xw) % plotting the reconstructed signal

% H
psnr_fourier_original=psnr(xf,x); % calculating psnr between
fourier and orginal signal
psnr_fourier_original

%I
psnr_wavelet_original=psnr(xw,x); % calculating psnr between
wavelet and orginal signal
psnr_wavelet_original


%J
figure
plot(1:n,x); % plotting original signal
hold on
plot(1:n,xf,'g.');  % plotting fourier reconstructed signal
hold on
plot(1:n,xw,'b*');  % plotting wavelet reconstructed signal
hold on
```

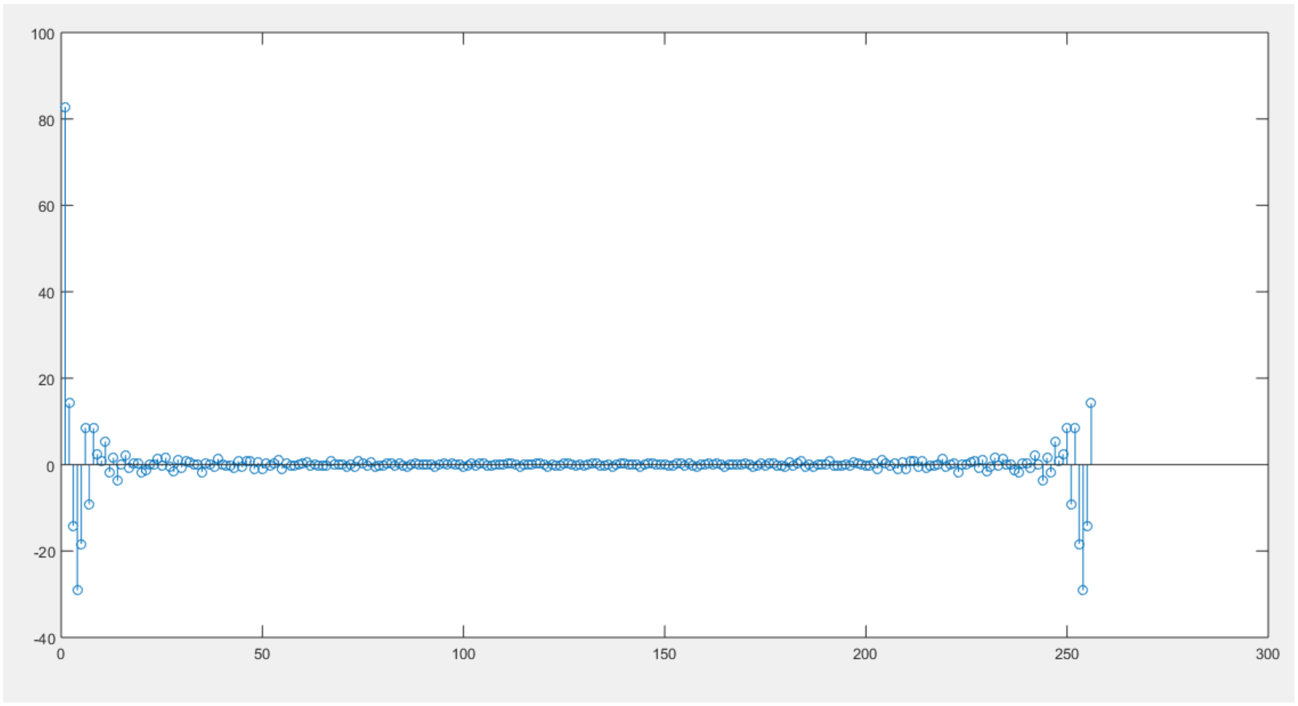
Piece-regular signal of size 256 is loaded using MakeSignal and rescale.
Reconstructing the signal using wavelet gives a better result than fft because wavelet does more levels of decomposition where as fft does only 1 level of decomposition

OUTPUT:

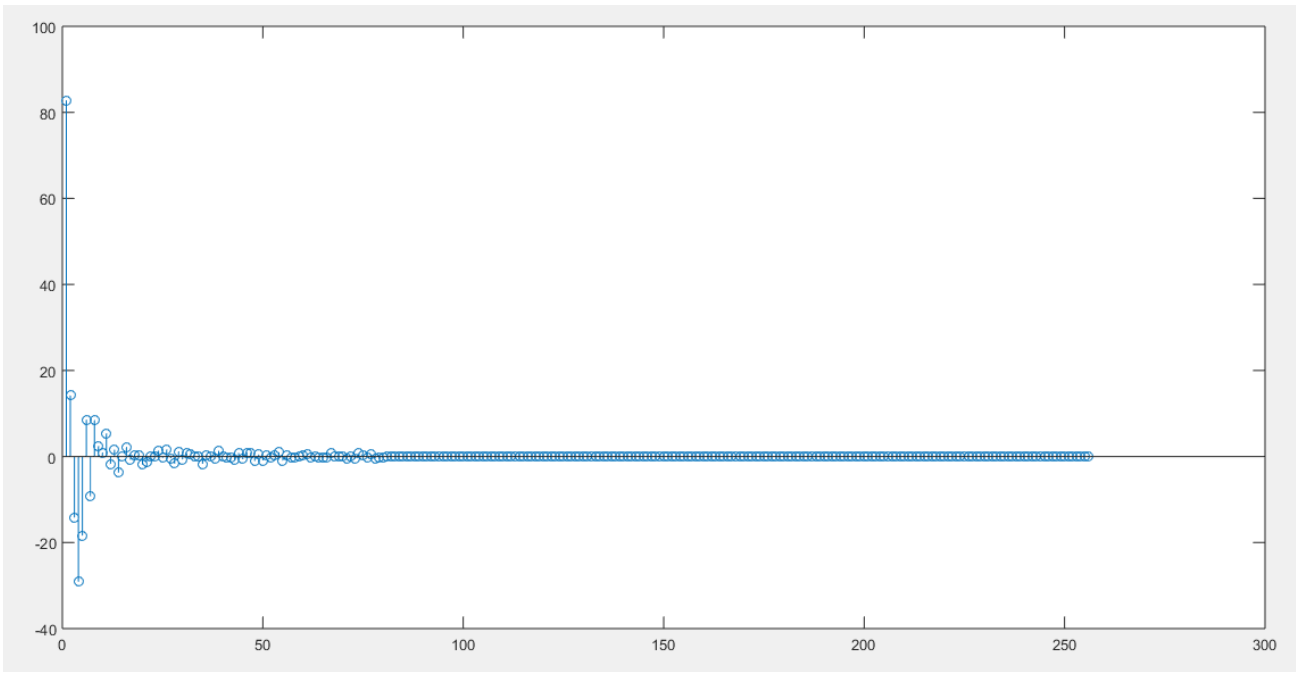
A.


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
50
100
150
200
250
300

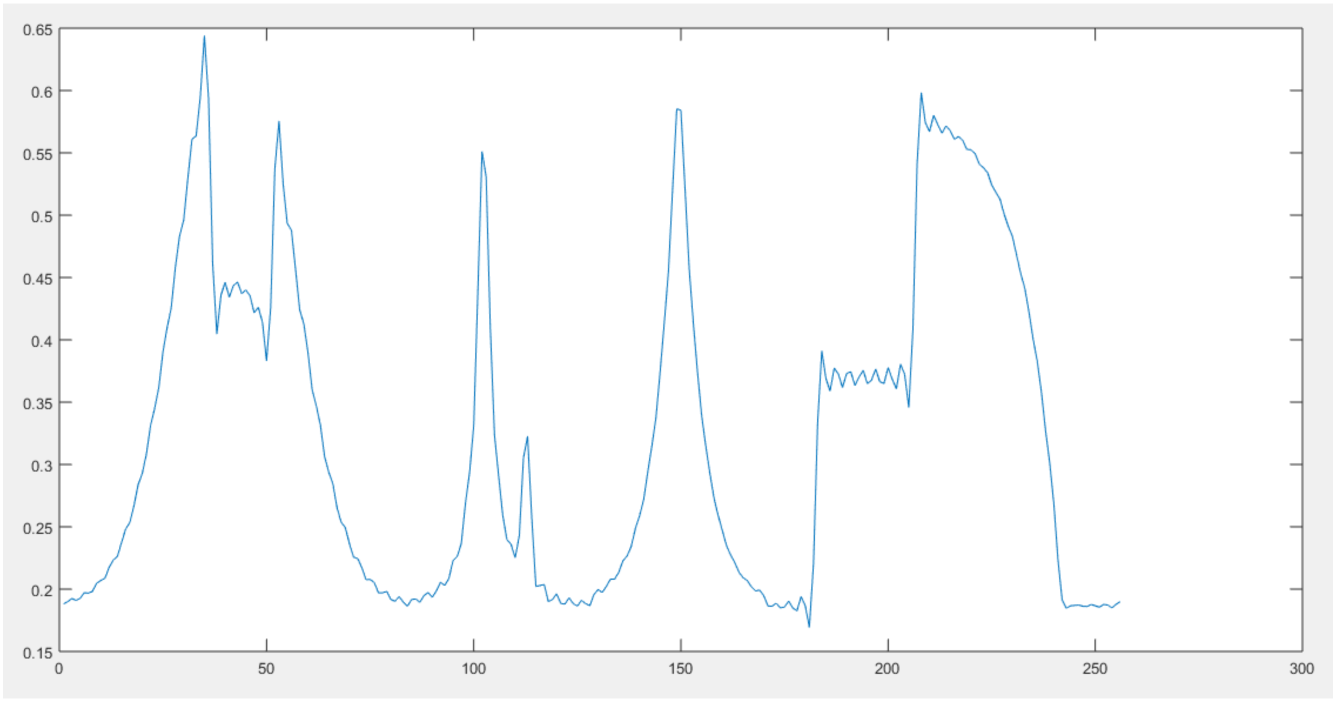
B


100
80
60
40
20
0
-20
-40
0
50
100
150
200
250
300

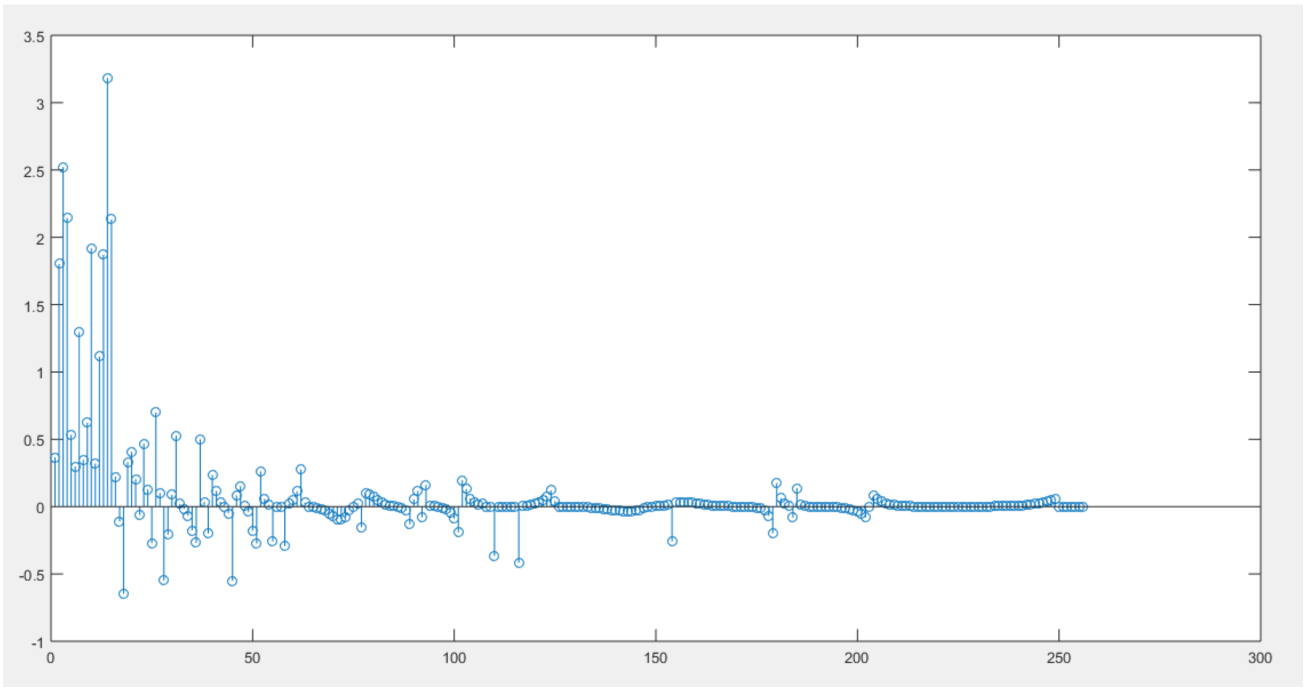
C


100
80
60
40
20
0
-20
-40
0
50
100
150
200
250
300

D

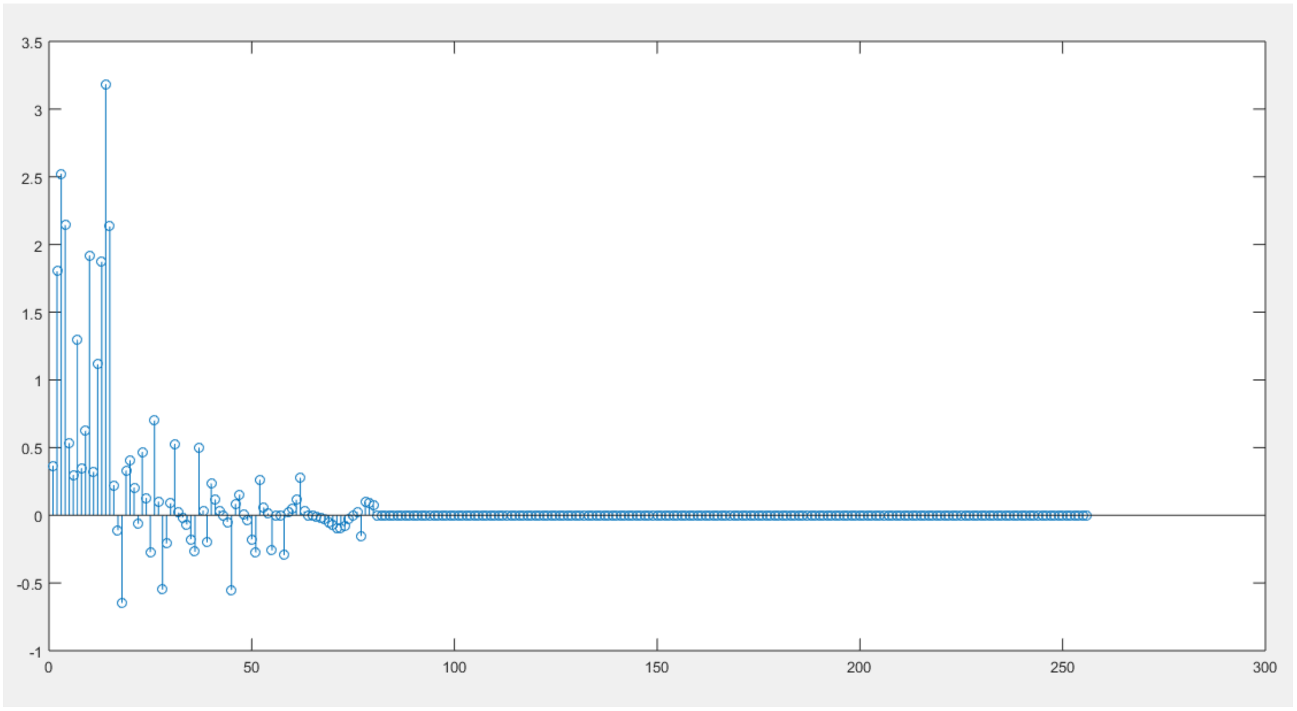

0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0
50
100
150
200
250
300

E



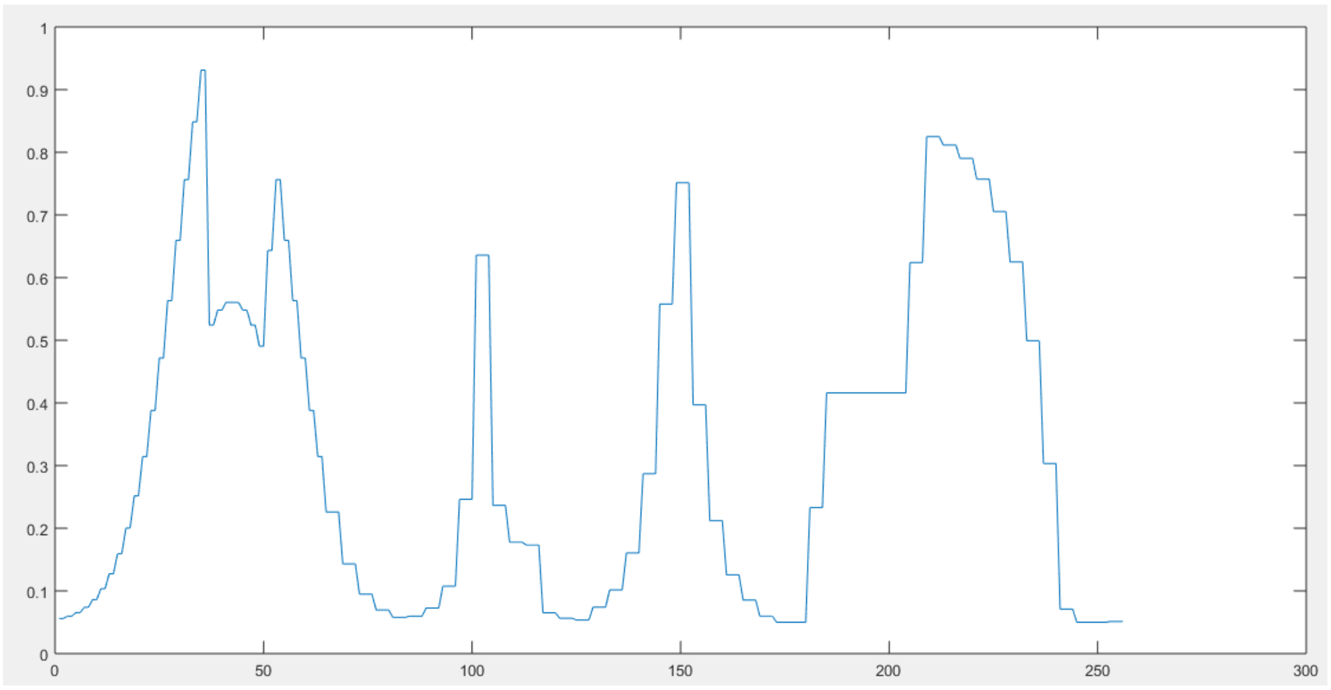
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
50
100
150
200
250
300

F



3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
50
100
150
200
250
300

G



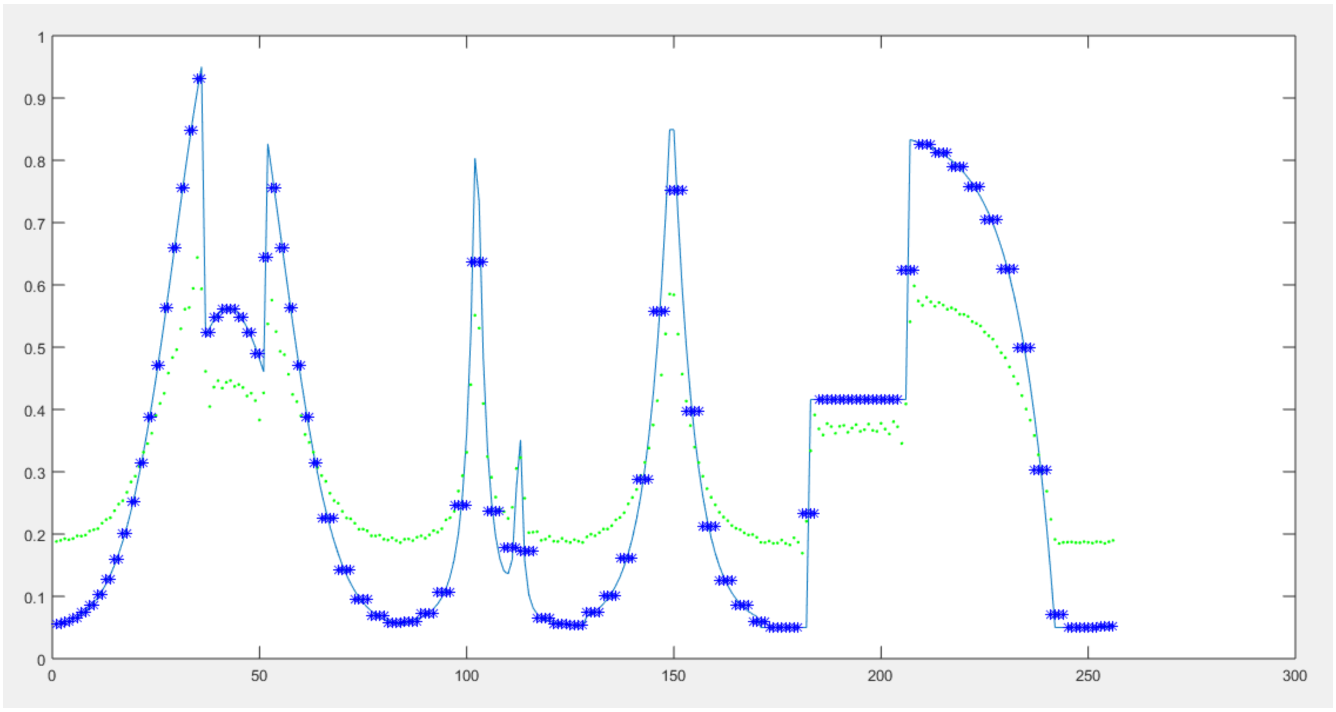
H

```
psnr_fourier_original =

   14.6087
```

I

```
psnr_wavelet_original =

   25.4106
```

J


0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
50
100
150
200
250
300