Digital Signal Processing: Experiment 3

Design of Low Pass Filters

Rajeswari Mahapatra (15EC10044)
Group : 40

Aim of the experiment:

- Design FIR filters of various orders and cut-offs
- Check whether pass band and stop band frequencies are attenuated by the designed filters.
- Check the designed filter response corresponding to noise contaminated input.
- Perform 2-D Filtering on an image:

  (a)Laplacian Filtering

  (b)Mean Filtering

  (c)Using Windowing

Theory:

Low pass filters actually have an infinite (duration) impulse response which is not practical to implement in real life, hence FIR filters provide a way to practically implement various filters. They are non-recursive digital filters as they do not have a feedback.

Window method is most commonly used method for designing FIR filters. The simplicity of design process makes this method very popular. A window is a finite array consisting of coefficients selected to satisfy the desirable requirements.

Digital FIR Filters are specified by:

- The Windowing function
- The filter order

These two requirements are interrelated and there exists a compromise between

- The sharpness of the filter and selectivity
- Stop band attenuation

The impulse response of a Low-Pass filter is given by:

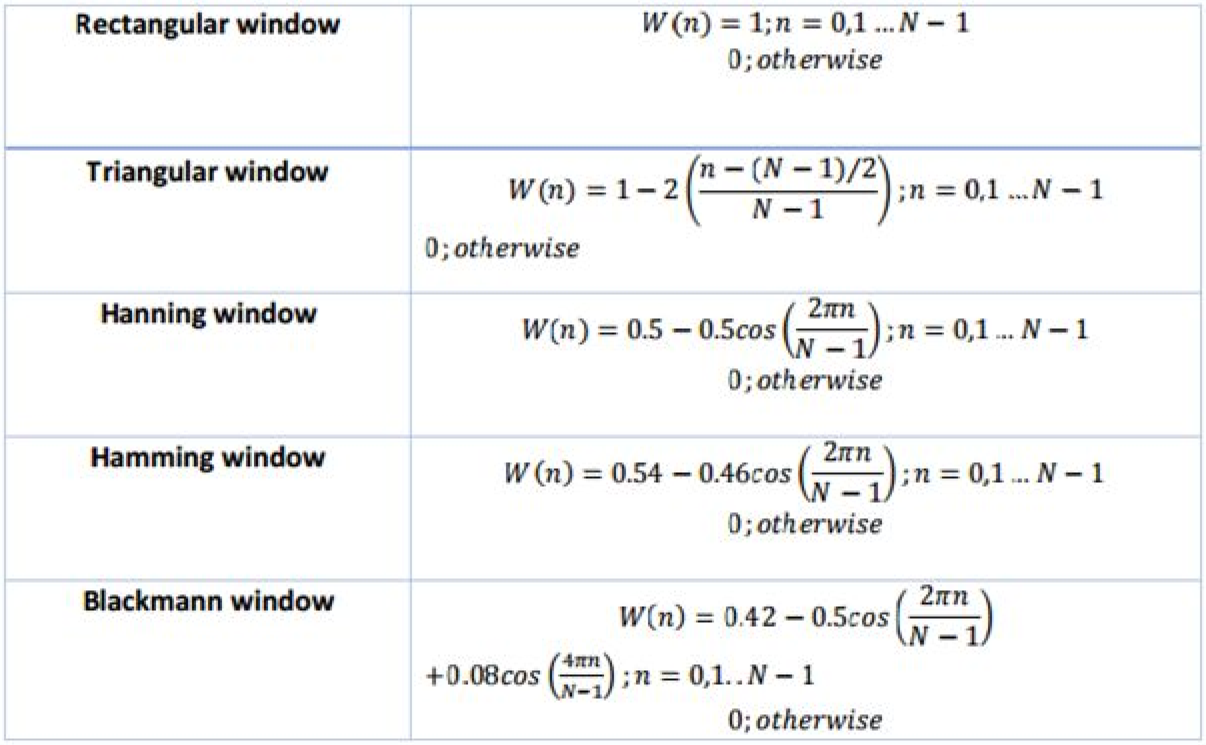
$$W(n) = \frac{sin(wc(n-k))}{\pi(n-k)} \qquad for\ n \neq k$$

$$= \frac{wc}{\pi} \qquad\qquad for\ n = k$$

N = Number of Samples

K = $\frac{N-1}{2}$

The various windowing functions in time domain are:



Rectangular window	$W(n) = 1; n = 0,1 \ldots N-1$ $0; otherwise$
Triangular window	$W(n) = 1 - 2\left(\frac{n-(N-1)/2}{N-1}\right); n = 0,1 \ldots N-1$ $0; otherwise$
Hanning window	$W(n) = 0.5 - 0.5cos\left(\frac{2\pi n}{N-1}\right); n = 0,1 \ldots N-1$ $0; otherwise$
Hamming window	$W(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right); n = 0,1 \ldots N-1$ $0; otherwise$
Blackmann window	$W(n) = 0.42 - 0.5cos\left(\frac{2\pi n}{N-1}\right)$ $+0.08cos\left(\frac{4\pi n}{N-1}\right); n = 0,1..N-1$ $0; otherwise$

Now one of these windowing functions are multiplied to the impulse response of a low pass filter to make an FIR filter.

Once the desired time domain characteristics of the FIR Filter are established, the freqz function is used to find and plot the frequency response of the FIR filter.

Design of filters for different orders

Code Snippets:

```
wc = 0.3*pi;

%%

hd = LPFilt(wc,N);

% plot(hd);

%%

w = rectWindow(N);

B1 = hd .* w;

figure;

freqz(B1,1,-0.9*pi:0.005:pi);

title(['Rectanguar Window (N=' num2str(N) ')']);

%%

w = triWindow(N);

B2 = hd .* w;

figure;

freqz(B2,1,-pi:0.005:pi);

title(['Trianguar Window (N=' num2str(N) ')']);

%%

w = hannWindow(N);

B3 = hd .* w;

figure;
```

```
freqz(B3,1,-pi:0.005:pi);

title(['Hanning Window (N=' num2str(N) ')']);

%%

w = hammWindow(N);

B4 = hd .* w;

figure;

freqz(B4,1,-pi:0.005:pi);

title(['Hamming Window (N=' num2str(N) ')']);

%%

w = blackWindow(N);

B5 = hd .* w;

figure;

freqz(B5,1,-pi:0.005:pi);

title(['Blackman Window (N=' num2str(N) ')']);
```

Results:

The results for different orders of specific filters have been tabulated:

N = 9

Window	Transition Width(π)	Peak of First Lobe (dB)	Max Stop Band attenuation (dB)
Rectangle	0.17	-18.75	66
Triangle	0.274	-18.63	23
Hamming	0.5	-48.56	100
Hanning	0.42	-44.55	93
Blackmann	0.65	NA	76

N = 65

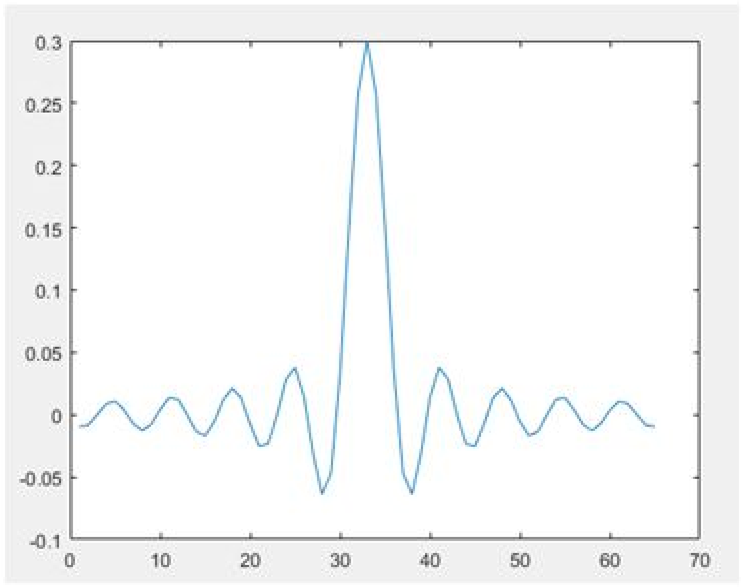
Window	Transition Width(π)	Peak of First Lobe (dB)	Max Stop Band attenuation (dB)
Rectangle	0.021	-20.94	100
Triangle	0.03	-21.26	42
Hamming	0.06	-53.54	107

Hanning	0.062	-44.43	140
Blackmann	0.1	-75.43	149

N = 257

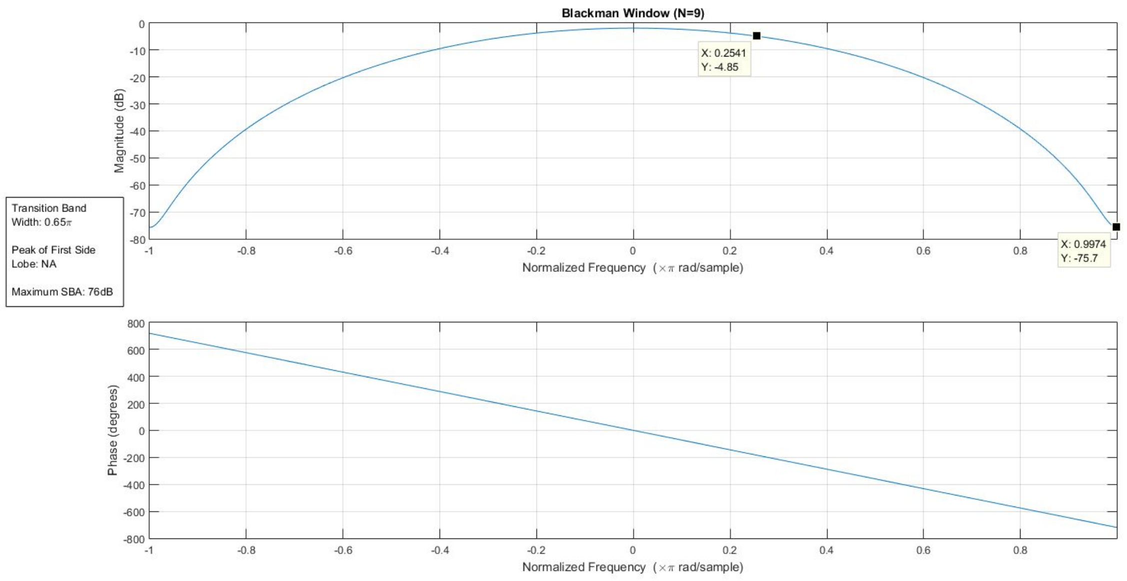
Window	Transition Width(π)	Peak of First Lobe (dB)	Max Stop Band attenuation (dB)
Rectangle	0.009	-20.86	66
Triangle	0.046	NA	51.19
Hamming	0.02	-57.24	124
Hanning	0.016	-44.24	186
Blackmann	0.02	-57.24	194

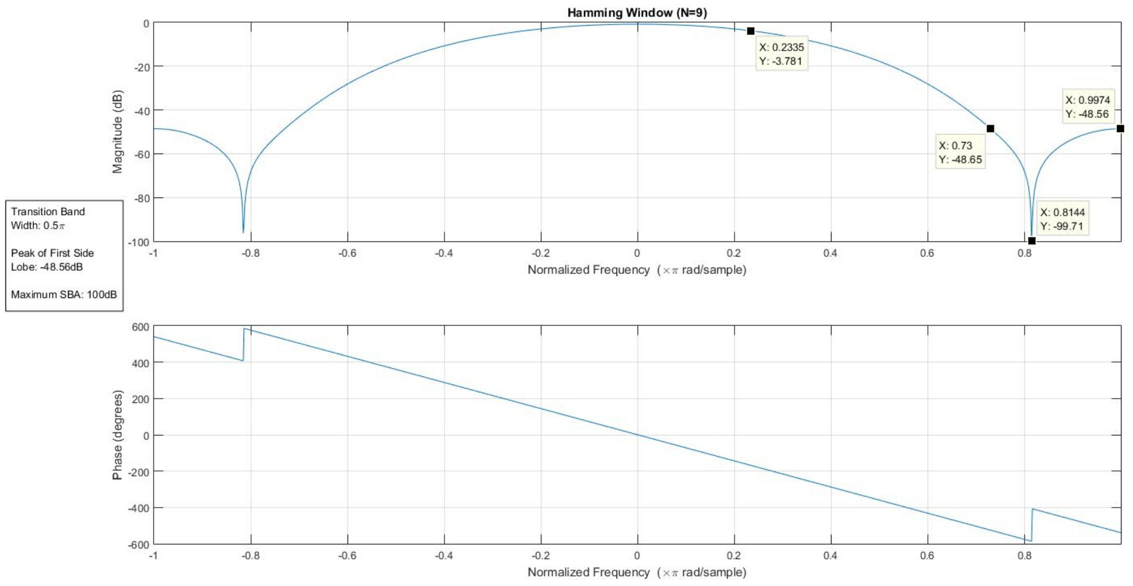
Plots of Frequency Responses:

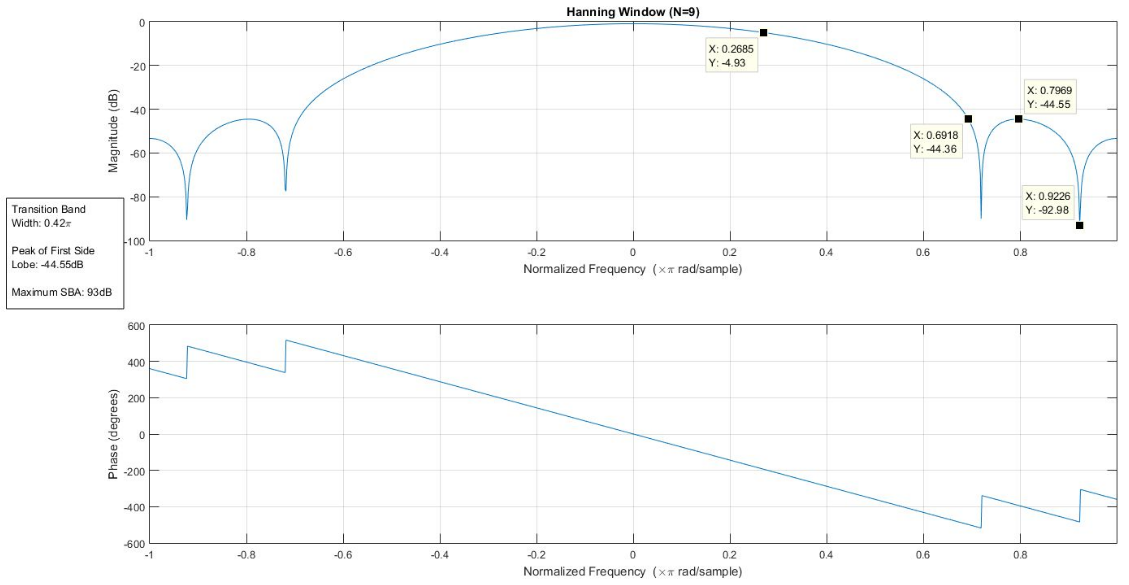


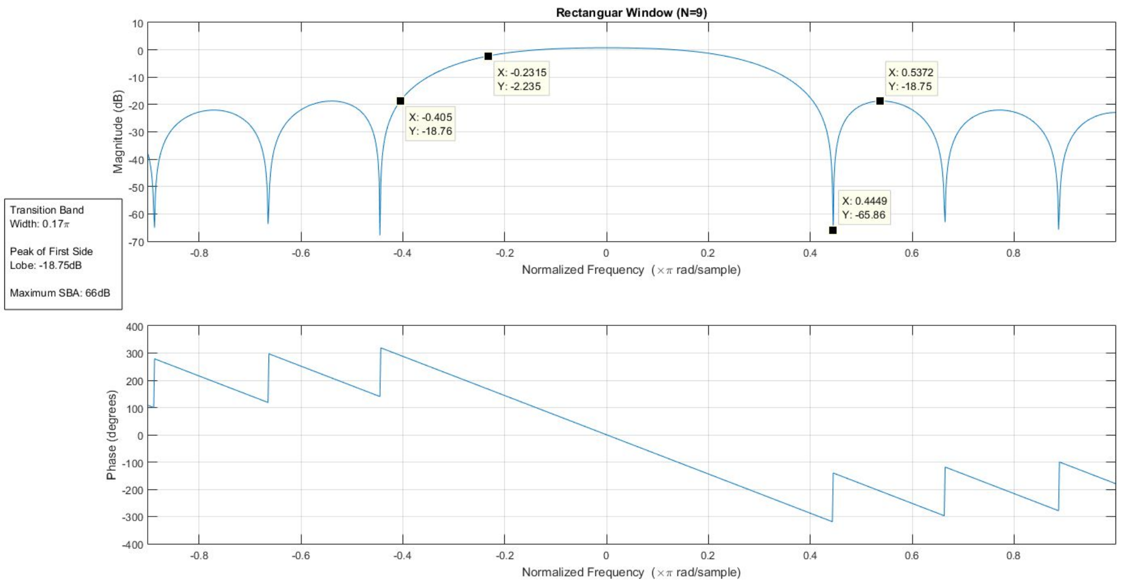
Impulse response of an ideal low-pass filter

For N=9,




Hamming Window (N=9)
Magnitude (dB)
Phase (degrees)
Normalized Frequency (×π rad/sample)
X: 0.2335
Y: -3.781
X: 0.73
Y: -48.65
X: 0.9974
Y: -48.56
X: 0.8144
Y: -99.71
Transition Band Width: 0.5π
Peak of First Side Lobe: -48.56dB
Maximum SBA: 100dB

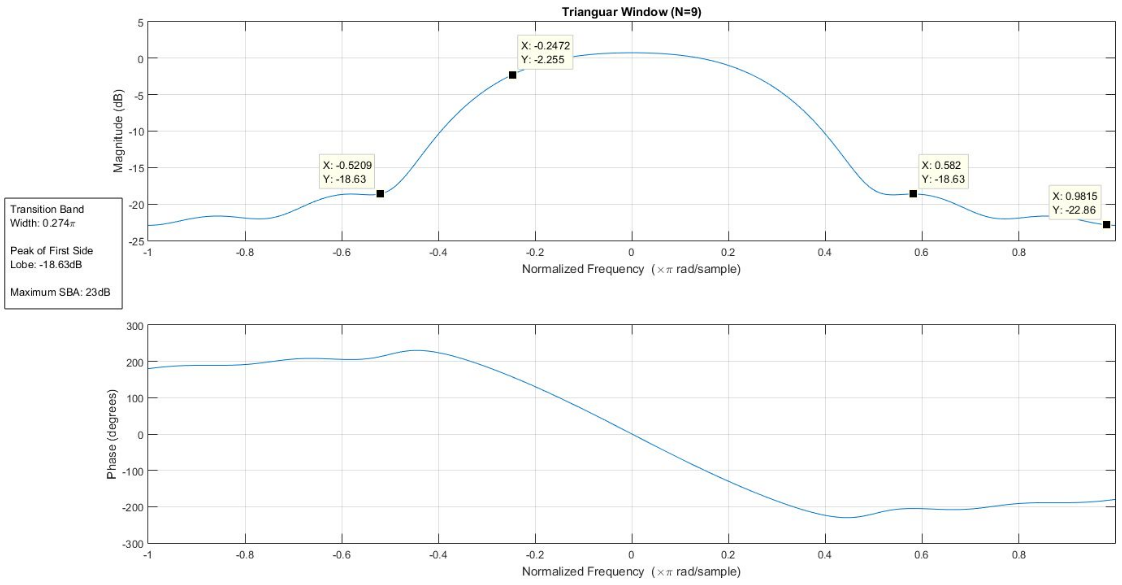

Hanning Window (N=9)
Magnitude (dB)
Phase (degrees)
Normalized Frequency (×π rad/sample)
X: 0.2685
Y: -4.93
X: 0.6918
Y: -44.36
X: 0.7969
Y: -44.55
X: 0.9226
Y: -92.98
Transition Band Width: 0.42π
Peak of First Side Lobe: -44.55dB
Maximum SBA: 93dB



Transition Band Width: 0.17π

Peak of First Side Lobe: -18.75dB

Maximum SBA: 66dB

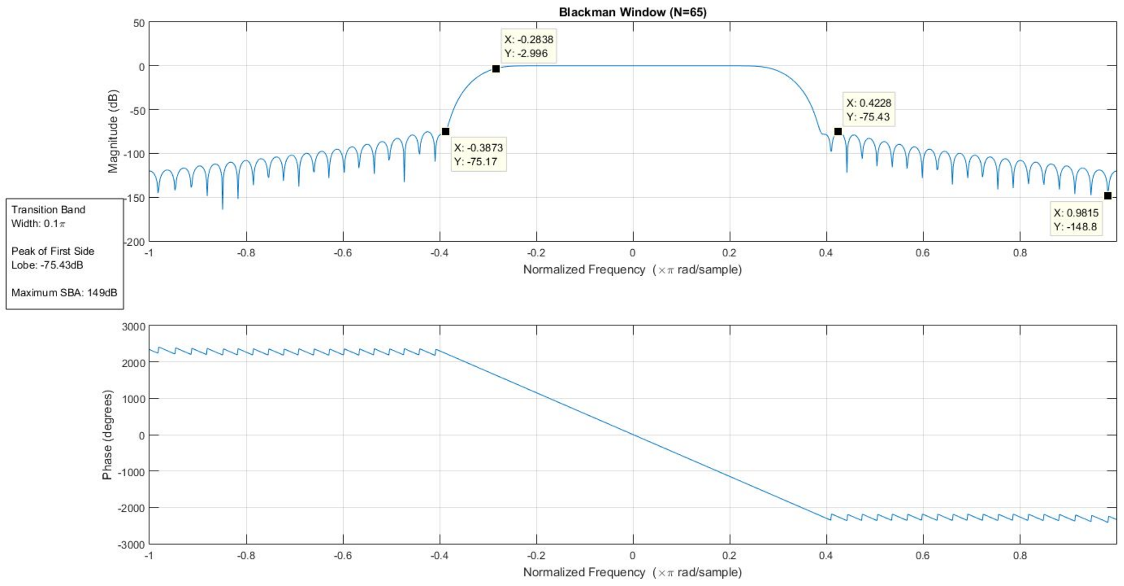


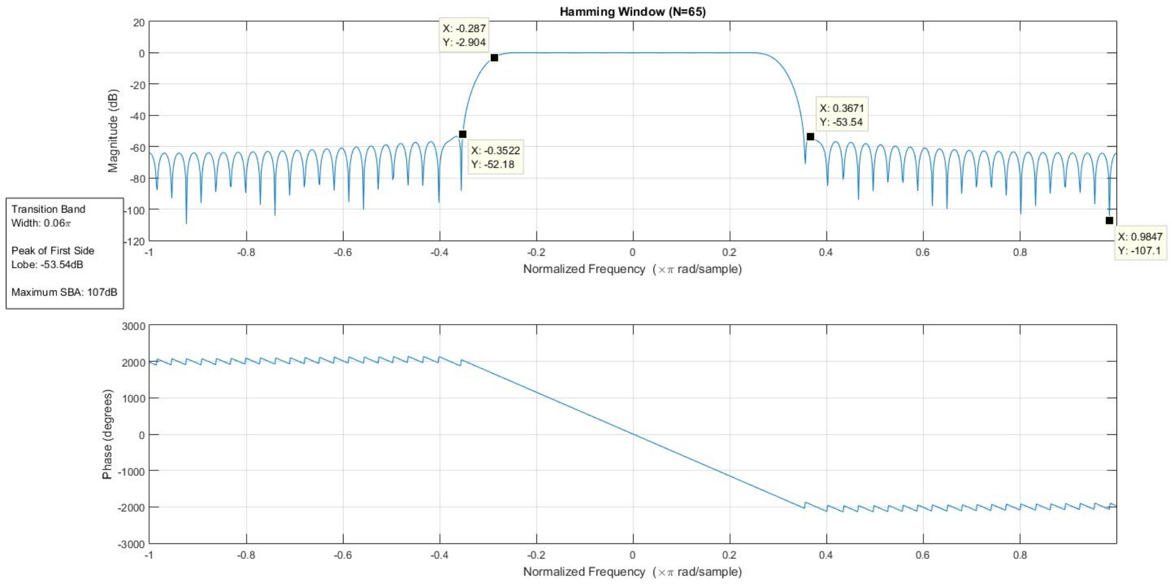
Transition Band Width: 0.274π

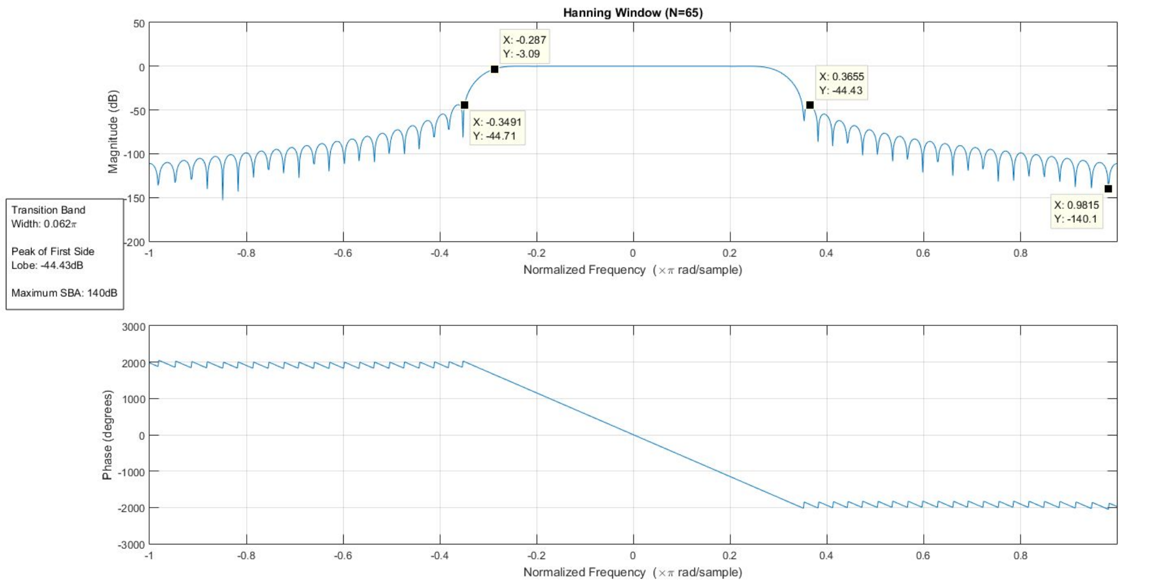
Peak of First Side Lobe: -18.63dB

Maximum SBA: 23dB

For N=65,


Blackman Window (N=65)
X: -0.2838
Y: -2.996
X: -0.3873
Y: -75.17
X: 0.4228
Y: -75.43
X: 0.9815
Y: -148.8
Transition Band Width: 0.1π
Peak of First Side Lobe: -75.43dB
Maximum SBA: 149dB
Magnitude (dB)
Phase (degrees)
Normalized Frequency (×π rad/sample)

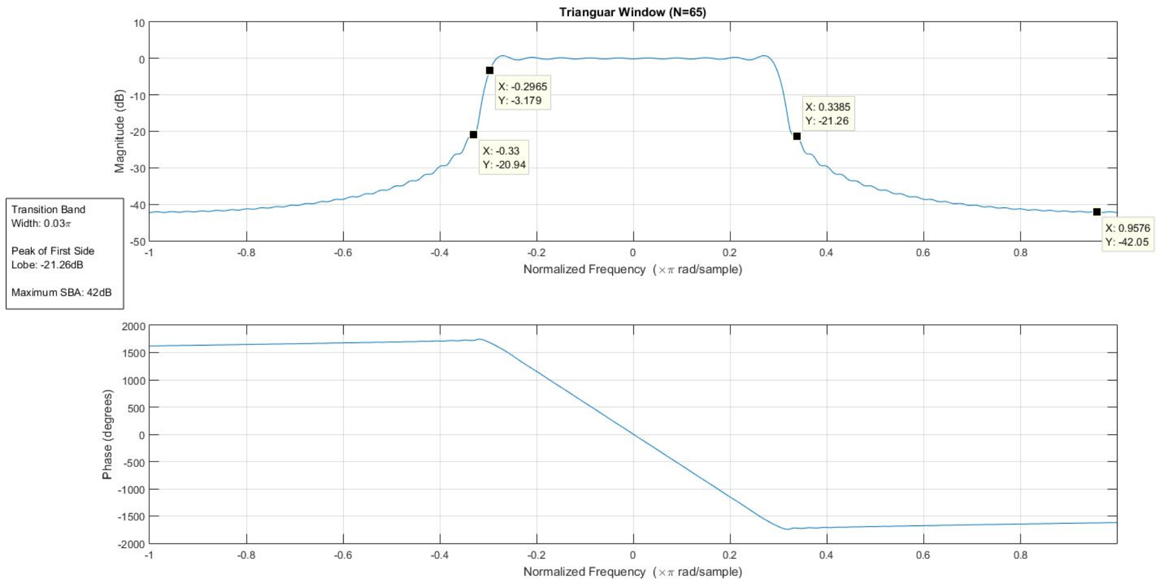

Hamming Window (N=65)
X: -0.287
Y: -2.904
X: -0.3522
Y: -52.18
X: 0.3671
Y: -53.54
X: 0.9847
Y: -107.1
Transition Band Width: 0.06π
Peak of First Side Lobe: -53.54dB
Maximum SBA: 107dB
Magnitude (dB)
Phase (degrees)
Normalized Frequency (×π rad/sample)



Transition Band Width: 0.062π

Peak of First Side Lobe: -44.43dB

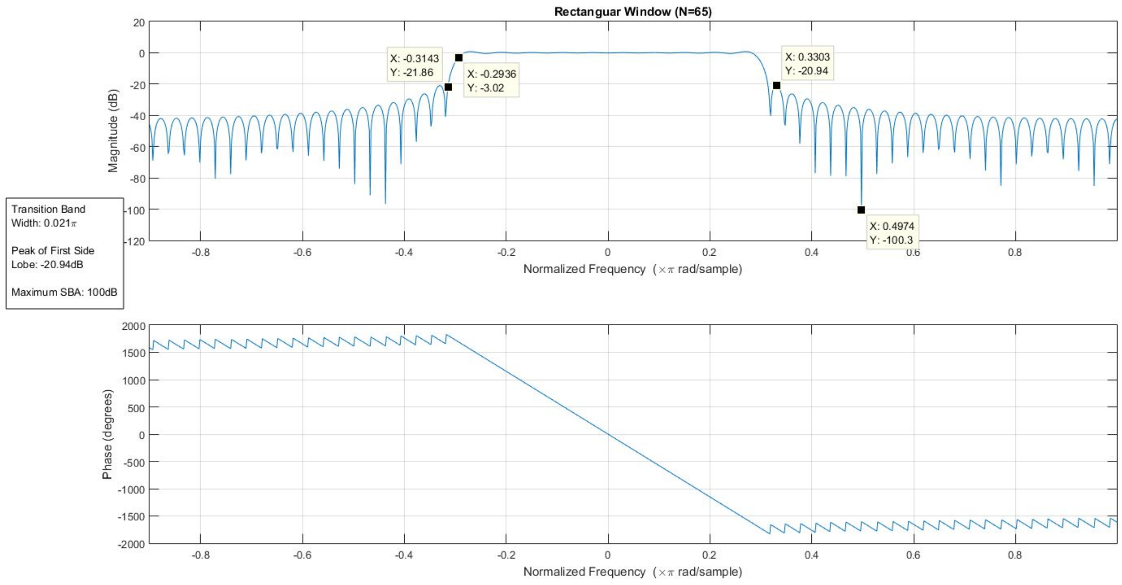
Maximum SBA: 140dB



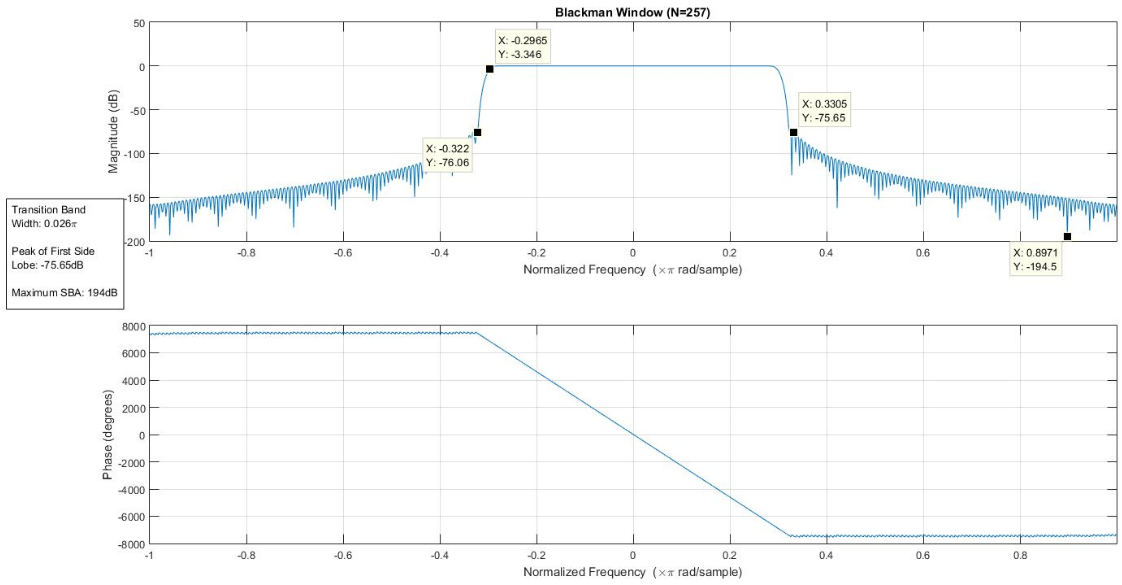
Transition Band Width: 0.03π

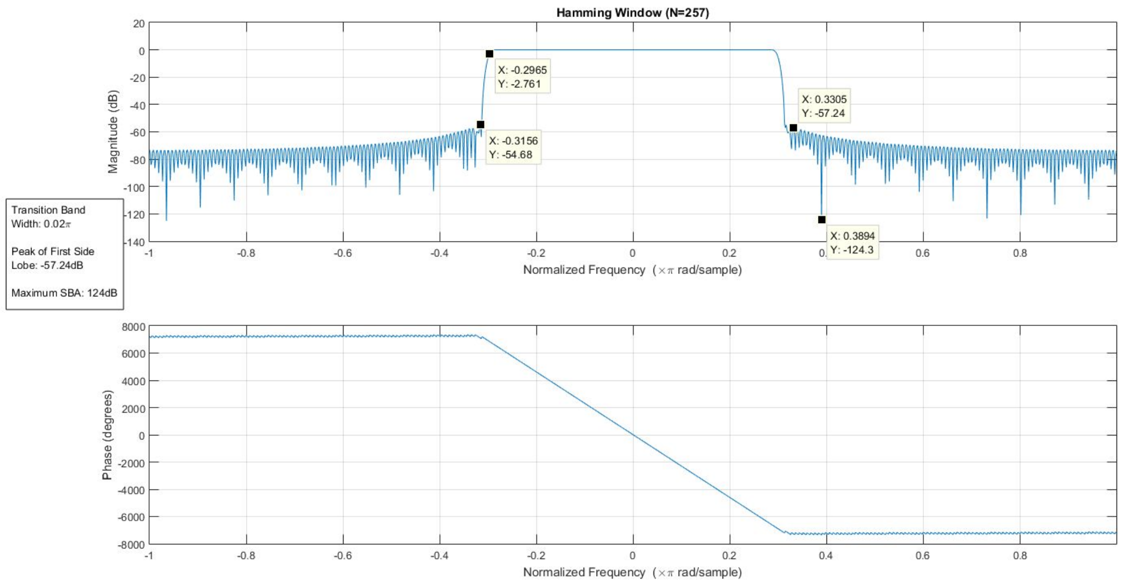
Peak of First Side Lobe: -21.26dB

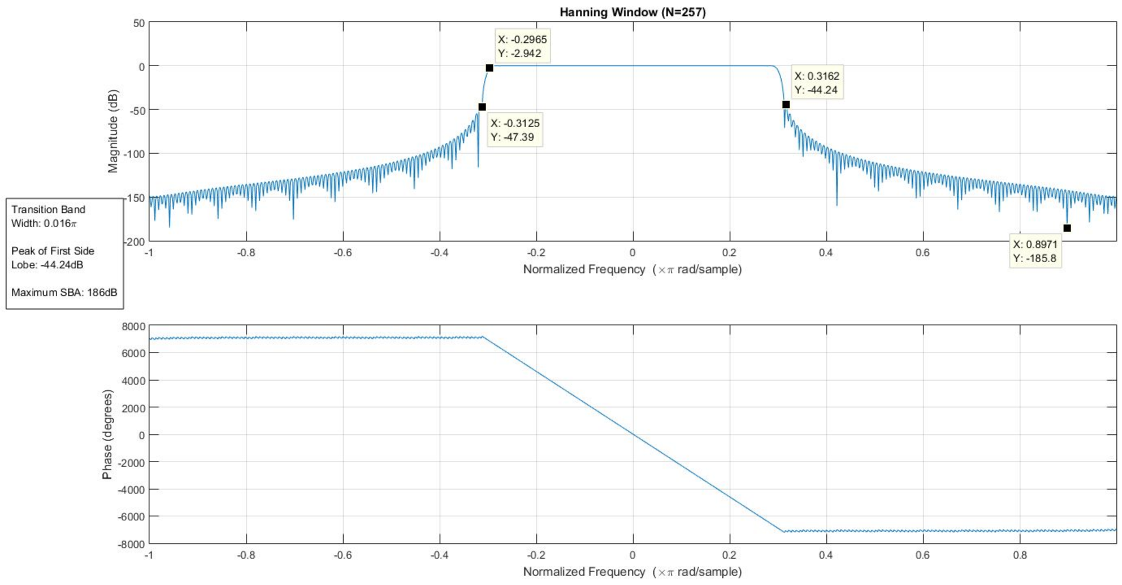
Maximum SBA: 42dB

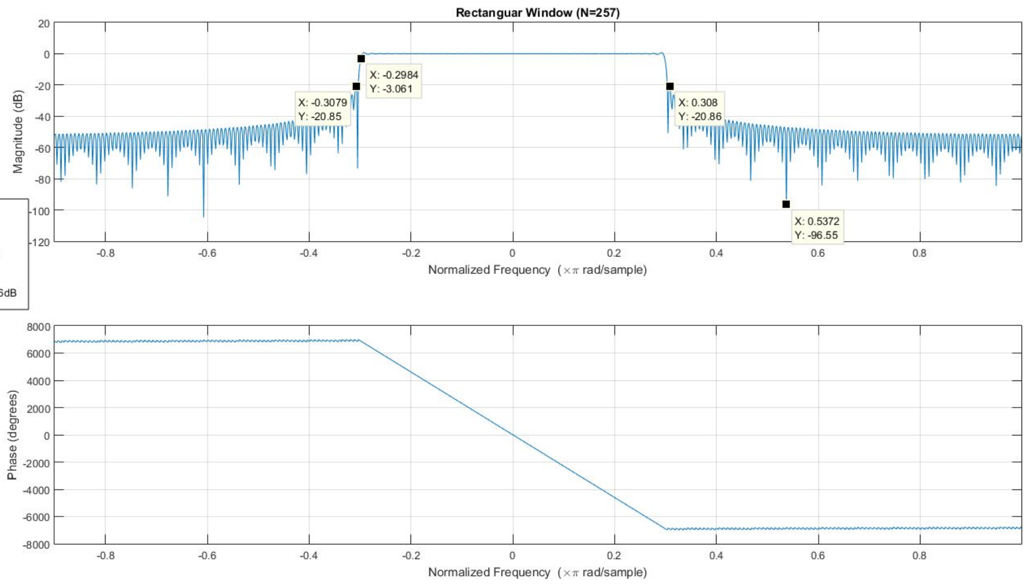

Rectanguar Window (N=65)
Magnitude (dB)
X: -0.3143
Y: -21.86
X: -0.2936
Y: -3.02
X: 0.3303
Y: -20.94
X: 0.4974
Y: -100.3
Transition Band Width: 0.021π
Peak of First Side Lobe: -20.94dB
Maximum SBA: 100dB
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)

For N=257,


Blackman Window (N=257)
Magnitude (dB)
X: -0.2965
Y: -3.346
X: 0.3305
Y: -75.65
X: -0.322
Y: -76.06
X: 0.8971
Y: -194.5
Transition Band Width: 0.026π
Peak of First Side Lobe: -75.65dB
Maximum SBA: 194dB
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)


Hamming Window (N=257)
Magnitude (dB)
X: -0.2965
Y: -2.761
X: -0.3156
Y: -54.68
X: 0.3305
Y: -57.24
X: 0.3894
Y: -124.3
Transition Band Width: 0.02π
Peak of First Side Lobe: -57.24dB
Maximum SBA: 124dB
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)

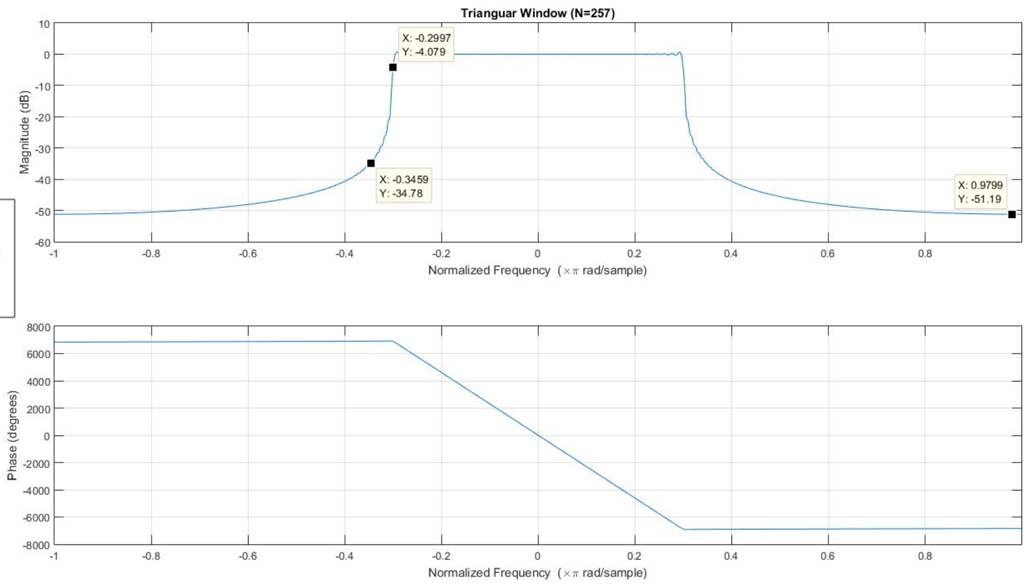

Hanning Window (N=257)
Magnitude (dB)
X: -0.2965
Y: -2.942
X: -0.3125
Y: -47.39
X: 0.3162
Y: -44.24
X: 0.8971
Y: -185.8
Transition Band Width: 0.016π
Peak of First Side Lobe: -44.24dB
Maximum SBA: 186dB
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)



Transition Band Width: 0.009π

Peak of First Side Lobe: -20.86dB

Maximum SBA: 66dB



Transition Band Width: 0.046π

Peak of First Side Lobe: NA

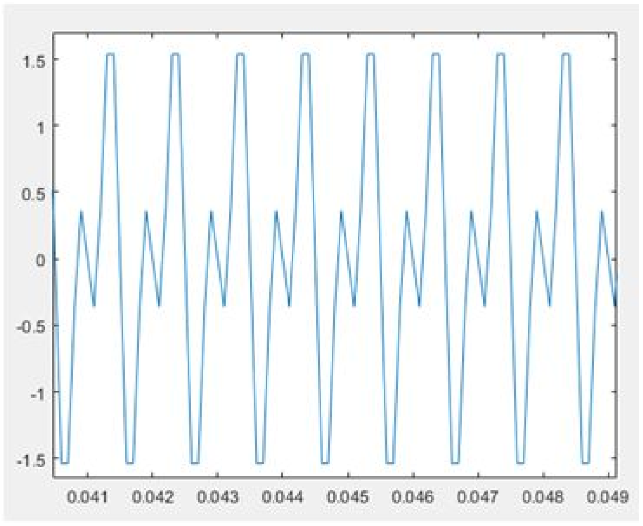
Maximum SBA: 51.19dB

Filtered response for a given input:

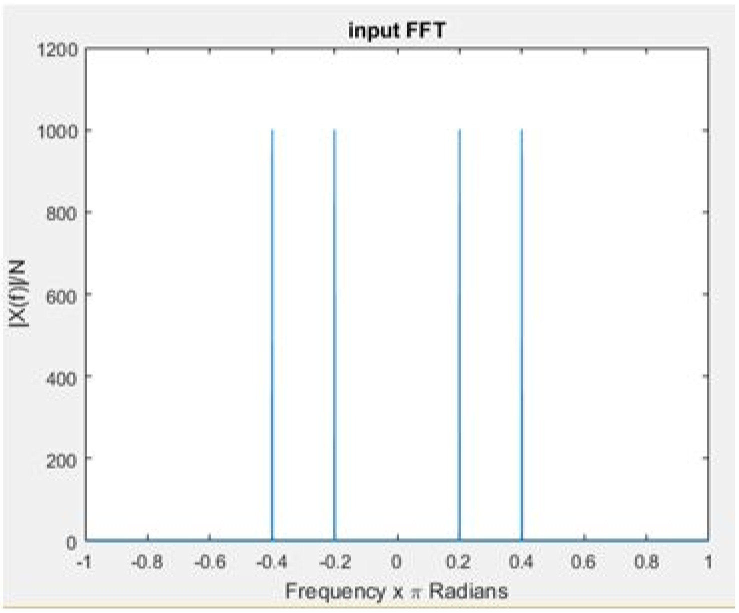
A sinusoid signal is generated such that one component is in the passband and another in the stop band.

```
x = sin(2*pi*0.1*Fs*t)+sin(2*pi*0.8*Fs*t)
```

Generated Signal:

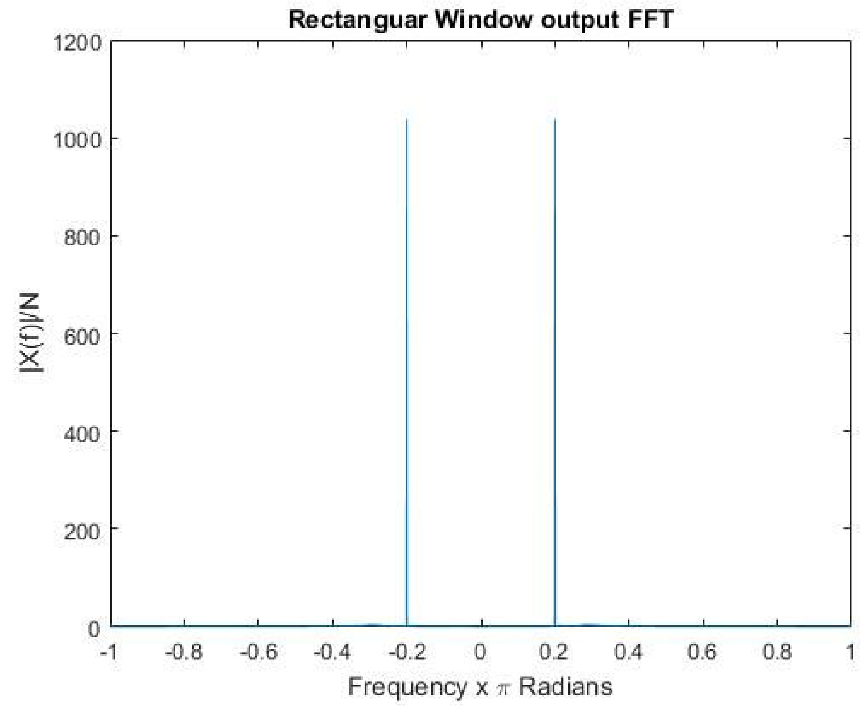


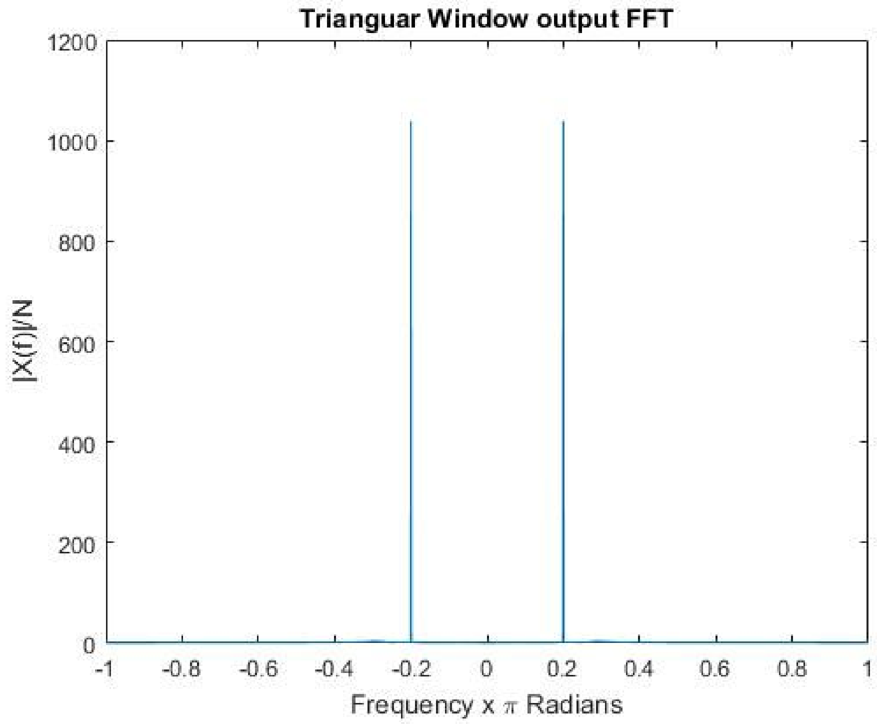
Input Signal DFT:

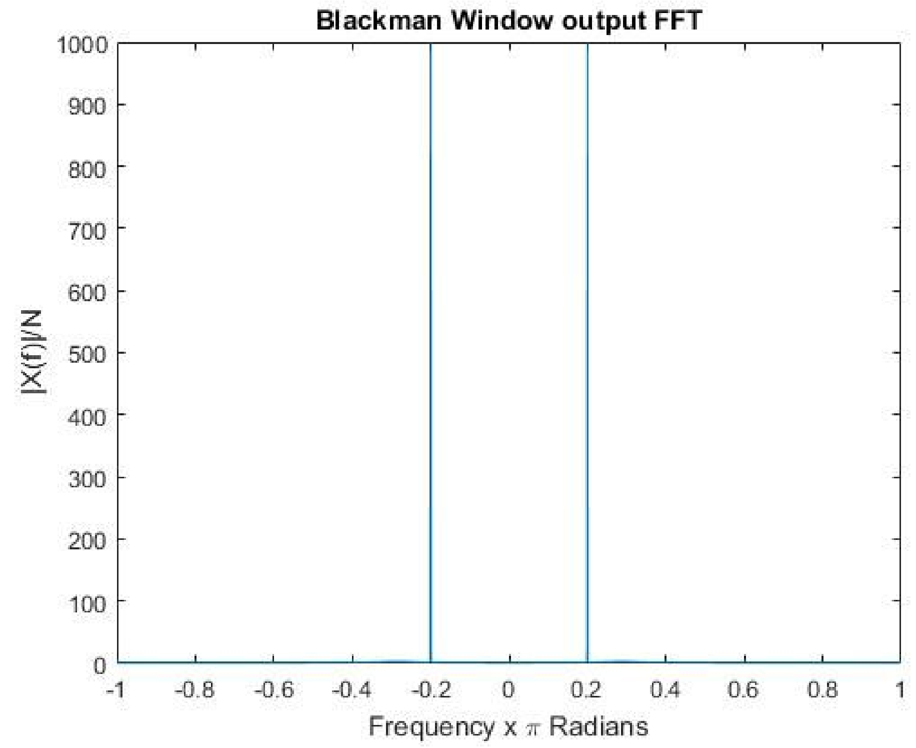


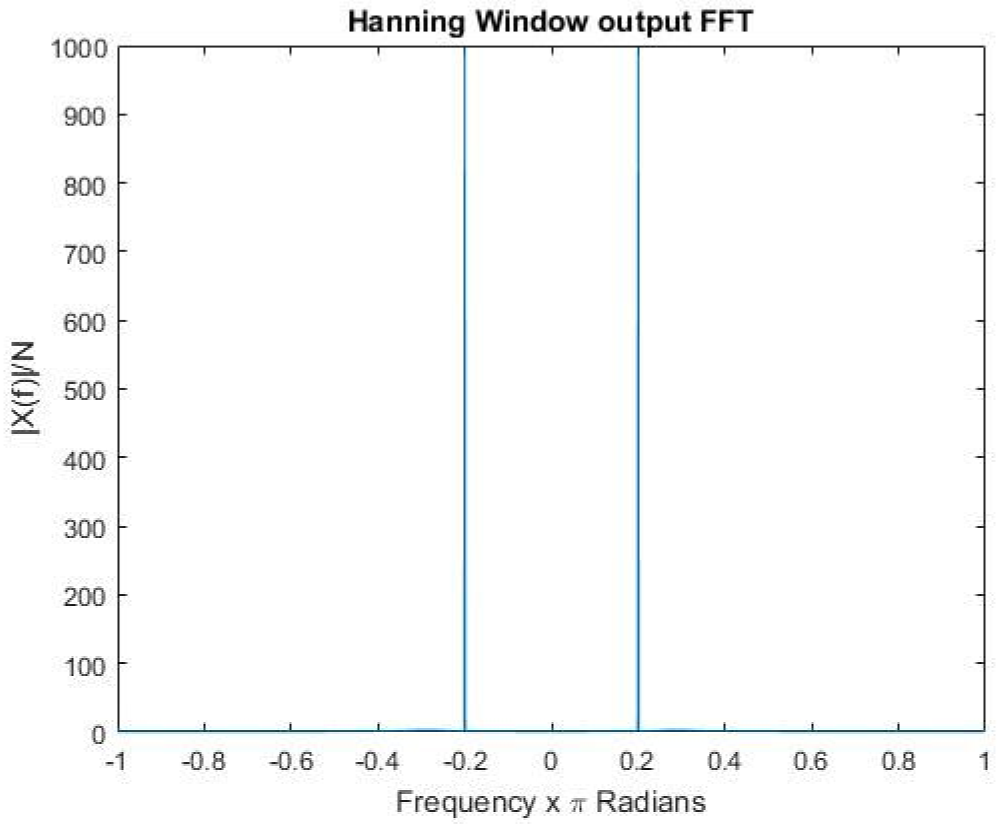
One of the components has a digital frequency of 0.2 while the other is at 0.4. Since the cut-off is at 0.3. One is in the passband while the other is in the stopband.

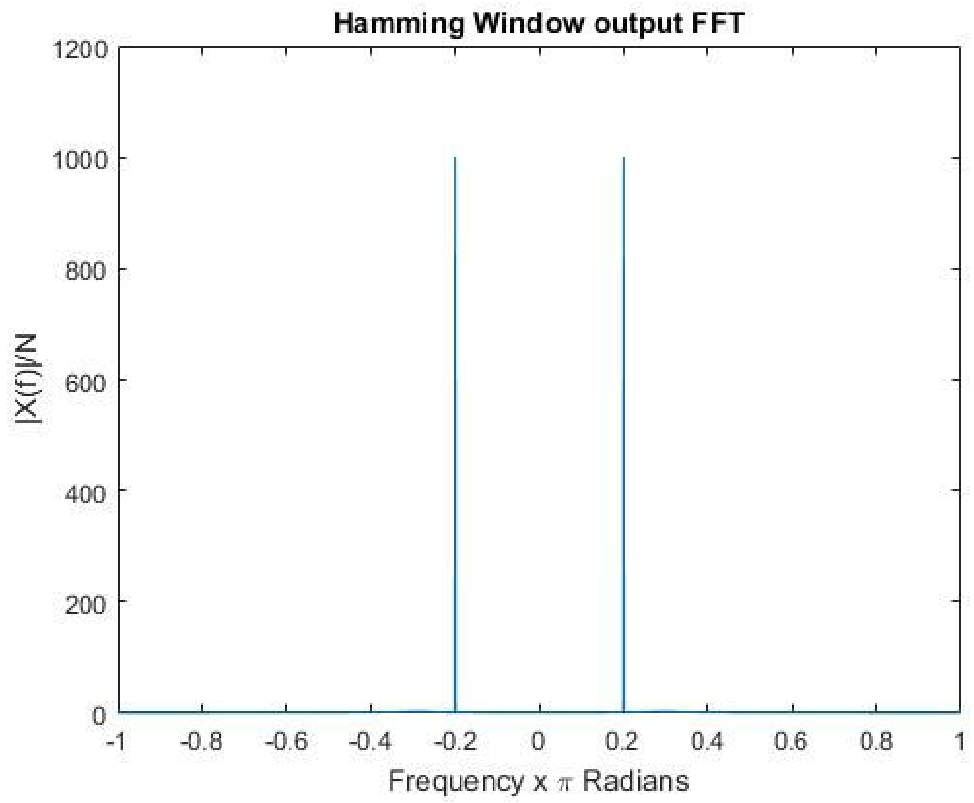
Filtered Outputs:





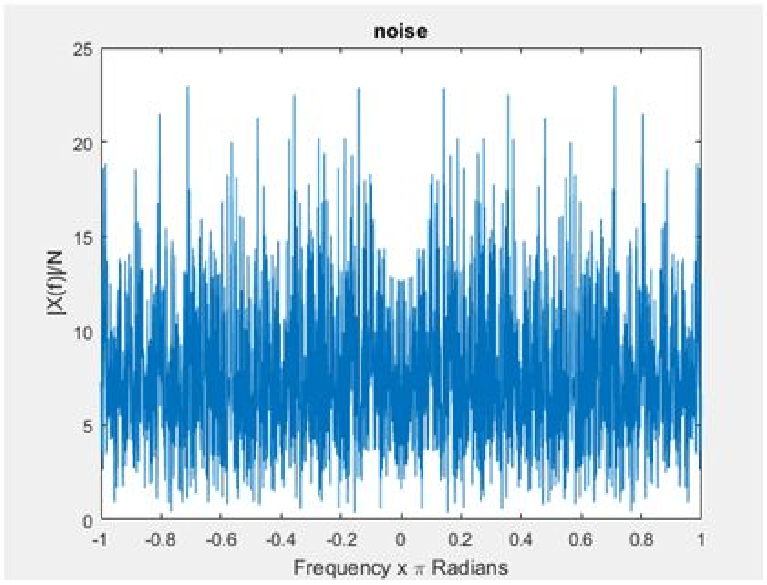

Blackman Window output FFT
|X(f)|/N
Frequency x π Radians
1000
900
800
700
600
500
400
300
200
100
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1


Hanning Window output FFT
|X(f)|/N
Frequency x π Radians
1000
900
800
700
600
500
400
300
200
100
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

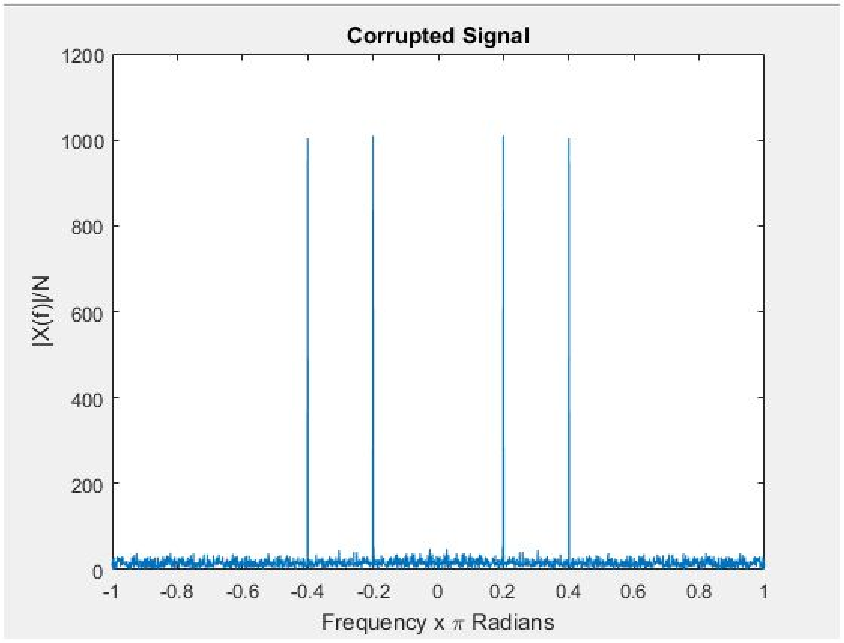

Hamming Window output FFT
|X(f)|/N
Frequency x π Radians
1200
1000
800
600
400
200
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Addition of noise:

AWGN noise of a specific variance is now added to this signal.



Power spectral density of noise



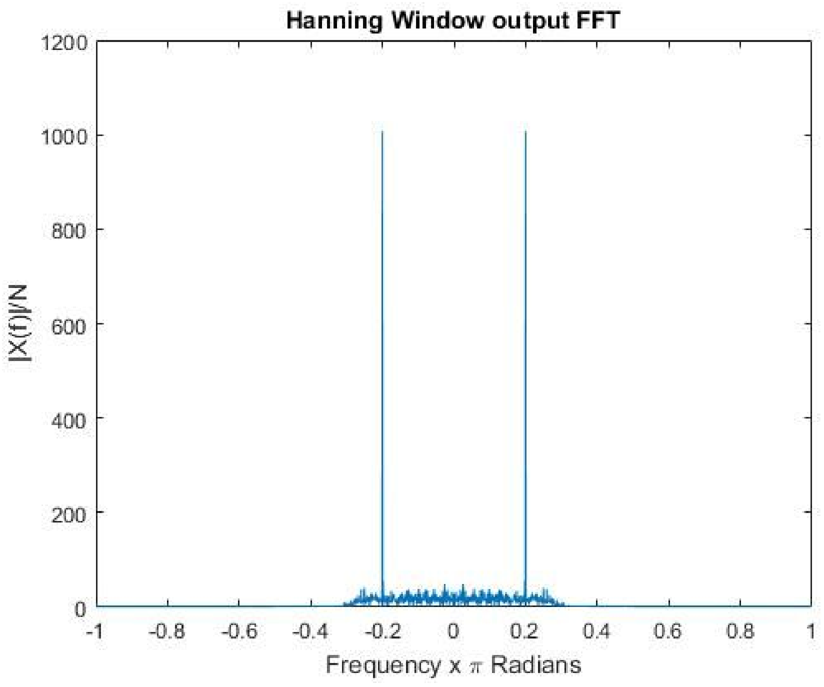
Noise corrupted signal

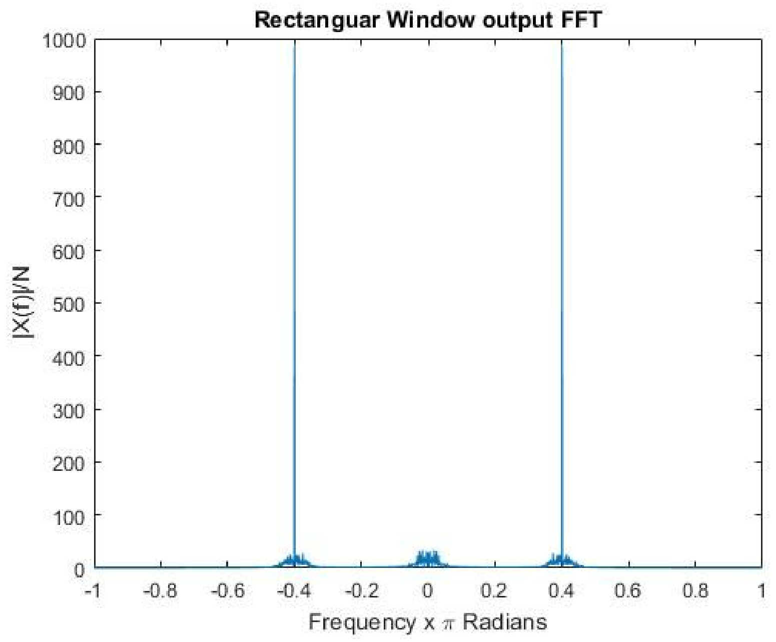
Code Snippets:

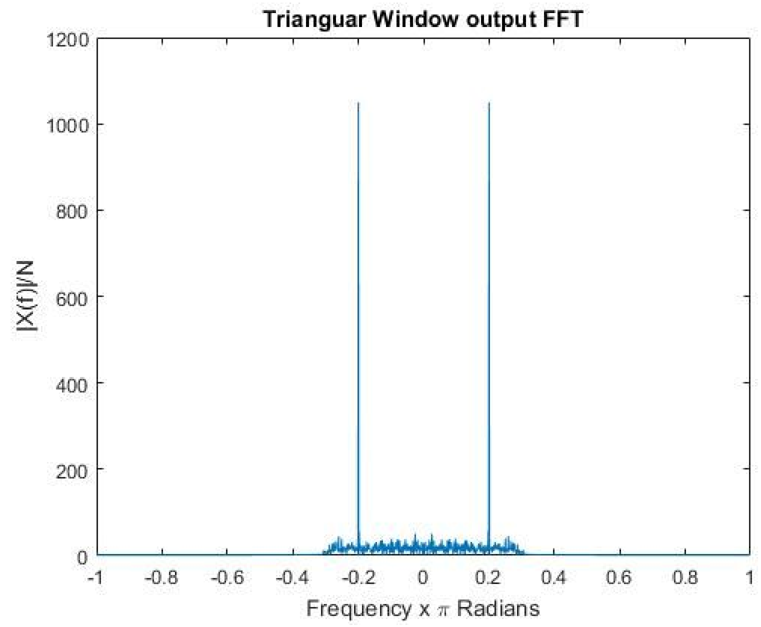
```
%% Generating Input Signal
Fs = 10 * 1e3;            % Sampling frequency
T = 1/Fs;             % Sampling period
L = 2000;             % Length of signal
t = (0:L-1)*T;        % Time vector
x = sin(2*pi*0.1*Fs*t)+sin(2*pi*0.8*Fs*t);
%plot(t,x);
r = rms(x);
plotdft(x,Fs,'input FFT')
%% Filtering using FIR Fiters
y = filtfilt(B1,1,x);
plotdft(y,Fs,'Rectanguar Window output FFT')

y = filtfilt(B2,1,x);
plotdft(y,Fs,'Trianguar Window output FFT')

y = filtfilt(B3,1,x);
plotdft(y,Fs,'Hanning Window output FFT')

y = filtfilt(B4,1,x);
plotdft(y,Fs,'Hamming Window output FFT')

y = filtfilt(B5,1,x);
```

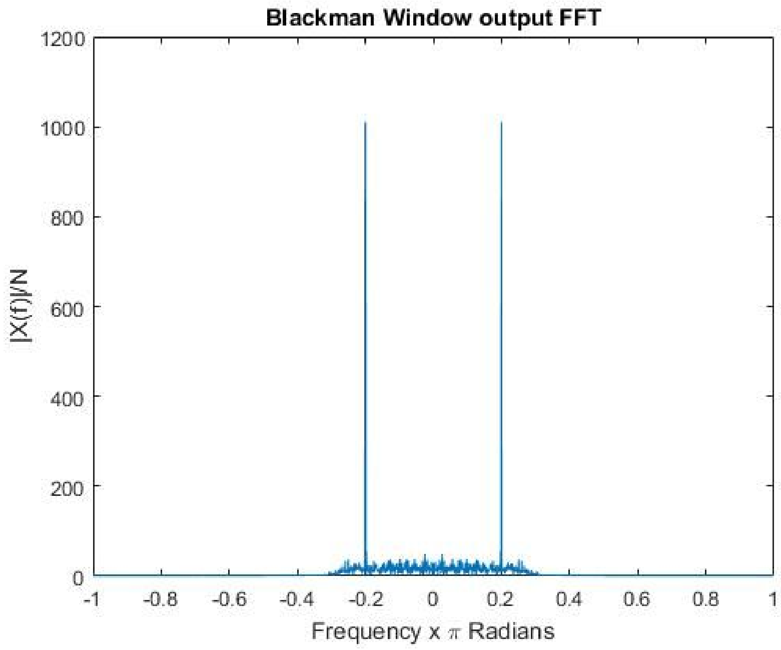
```
plotdft(y,Fs,'Blackman Window output FFT')

%% Generation of noise

n = 0.4*r*randn(1,L);

plotdft(n,Fs,'noise')

%%

x1 = x+n;

plotdft(x1,Fs,'Corrupted Signal')

%% Filtering using FIR Fiters

y1 = filtfilt(B1,1,x1);

plotdft(y1,Fs,'Rectanguar Window output FFT')

snr(y1)

y1 = filtfilt(B2,1,x1);

plotdft(y1,Fs,'Trianguar Window output FFT')

snr(y1)

y1 = filtfilt(B3,1,x1);

plotdft(y1,Fs,'Hanning Window output FFT')

snr(y1)

y1 = filtfilt(B4,1,x1);

plotdft(y1,Fs,'Hamming Window output FFT')

snr(y1)

y1 = filtfilt(B5,1,x1);

plotdft(y1,Fs,'Blackman Window output FFT')

snr(y1)
```

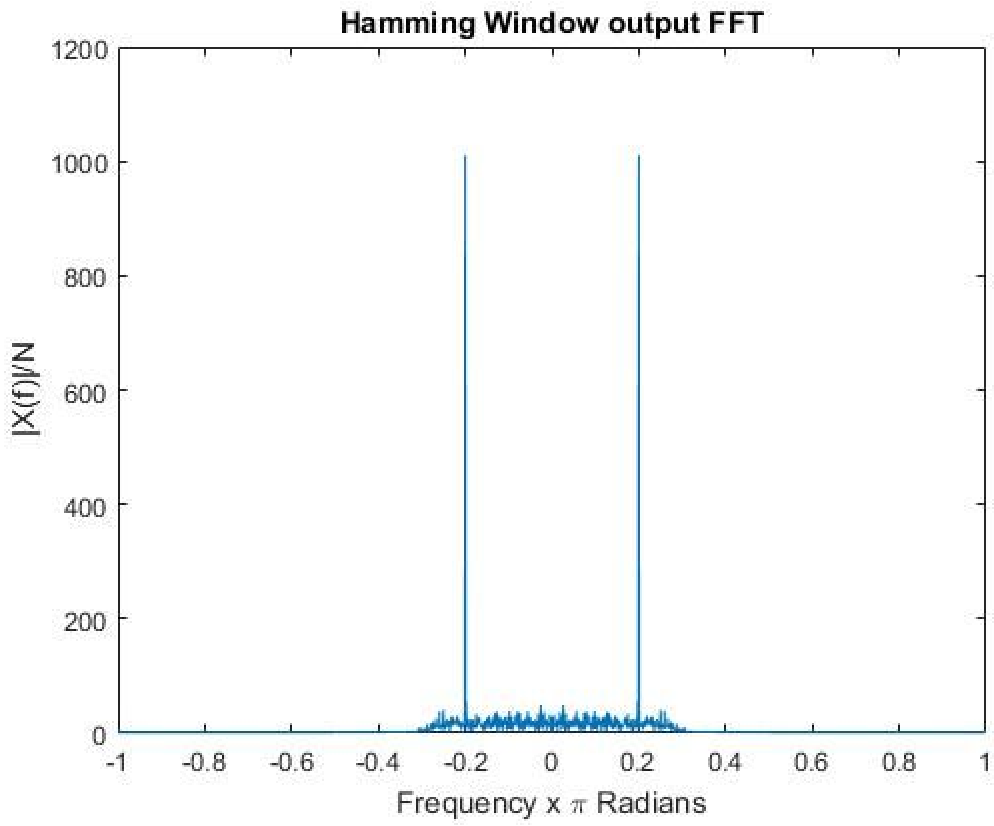
Filtered Output:





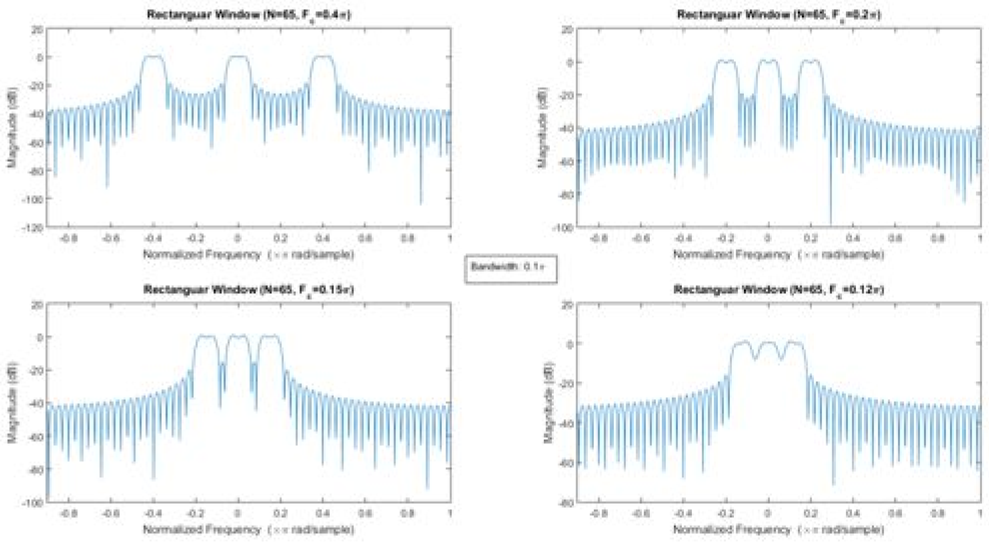

Trianguar Window output FFT
|X(f)|/N
Frequency x π Radians
1200
1000
800
600
400
200
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

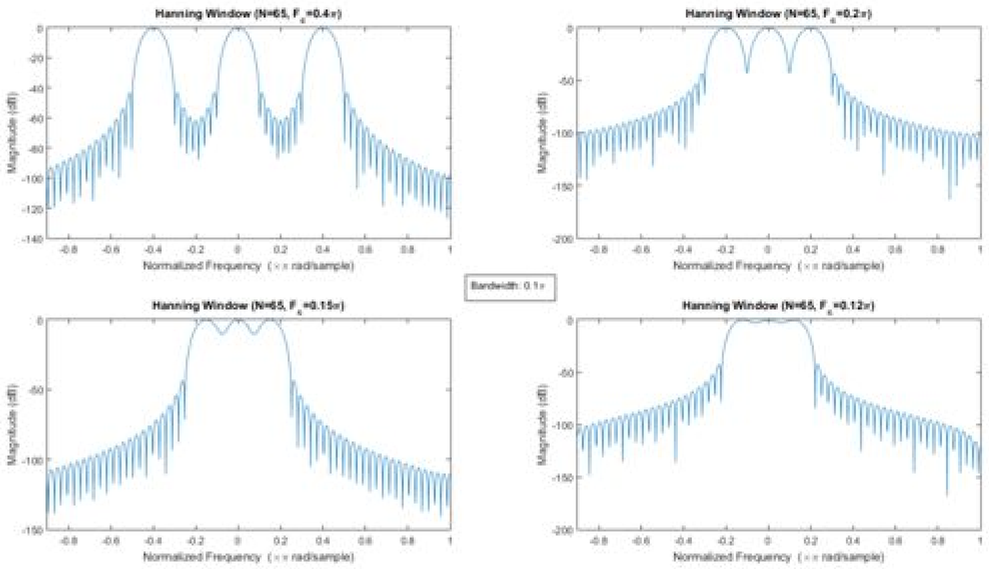

Blackman Window output FFT
|X(f)|/N
Frequency x π Radians
1200
1000
800
600
400
200
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1


Hamming Window output FFT
|X(f)|/N
Frequency x π Radians
1200
1000
800
600
400
200
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Limitations of Windowing Method:

The windowing method for designing FIR filters is very effective, but is usually used to design filters with only a single passband. For multiple passbands, we have constructive interference in stop bands and degradation of filter sharpness as well as stop band attenuation. This can be seen in the following two screenshots, for the rectangular and hanning windows:





As we can see, in rectangular windowing technique, as the passbands are brought closer, we have an added passband ripple (>1.5dB, which is the accepted standard for a good filter)

The selectivity is also hampered. This is even more evident in the Hanning windowing method. Although any passband ripple has been avoided, due to the slow roll-off of the Hanning FIR filter, we have lesser sharpness and selectivity and even at F_c=0.15π, we have sufficient interference, keeping us from differentiating the bands clearly.

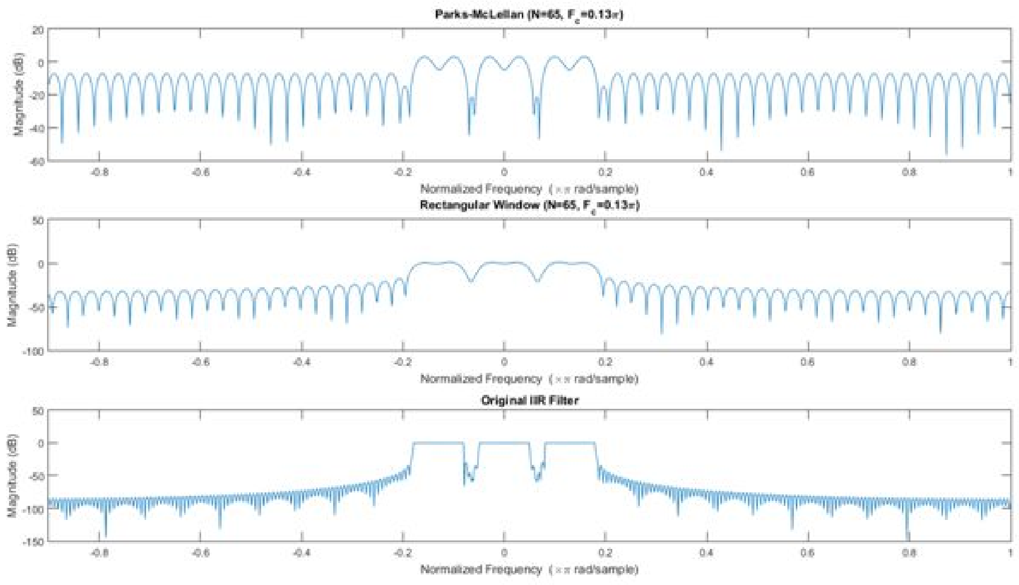
Code Snippets:

```
wc = 0.05*pi;                         %cutoff of band on either side

Fc = 0.13;                            %center frequency of pass band

N=65;

h1 = 2*BPFilt(wc,N,0.5*Fc)+LPFilt(wc,N); %compensate for division of power in
sidebands

w = rectWindow(N);

B1 = h1 .* w;

[A,~]=freqz(B1,1,-0.9*pi:0.005:pi);

%subplot(2,2,1)

plot(-0.9:1.9/size(A,2):1-1.9/size(A,2),20*log10(abs(A)));

xlim([-0.9 1]);

xlabel('Normalized Frequency  (\times\pi rad/sample)');

ylabel('Magnitude (dB)');

title(['Rectangular Window (N=' num2str(N) ', F_c=' num2str(Fc) '\pi)']);
```

Design of FIR filters using Parks-McClellan algorithm:

Parks-McClellan algorithm uses the Remez exchange algorithm and Chebyshev approximation theory to design optimal FIR filter corresponding to a given frequency response. This is optimal in the sense that the maximum error between the desired and actual response is minimized. We do see some extra ripple, which is a characteristic of the Chebyshev approximation.

Here are the results, showing a comparison between the filter response for Parks-McClellan method, Windowing method and IIR filter (desired frequency response):



Code Snippets:

```
h2=2*BPFilt(wc,1001,0.5*Fc)+LPFilt(wc,1001);

[a,~]=freqz(h2,1,-0.9*pi:0.005:pi);

a(1:566)=[];

f=0:1/size(a,2):1-1/size(a,2);
```

```
b=firpm(N,f,abs(a));


subplot(3,1,1)

[A,~]=freqz(b,1,-0.9*pi:0.005:pi);

plot(-0.9:1.9/size(A,2):1-1.9/size(A,2),20*log10(abs(A)));

xlim([-0.9 1]);

xlabel('Normalized Frequency  (\times\pi rad/sample)');

ylabel('Magnitude (dB)');

title(['Parks-McLellan (N=' num2str(N) ', F_c=' num2str(Fc) '\pi)']);


subplot(3,1,2)

[A,~]=freqz(B1,1,-0.9*pi:0.005:pi);

plot(-0.9:1.9/size(A,2):1-1.9/size(A,2),20*log10(abs(A)));

xlim([-0.9 1]);

xlabel('Normalized Frequency  (\times\pi rad/sample)');

ylabel('Magnitude (dB)');

title(['Rectangular Window (N=' num2str(N) ', F_c=' num2str(Fc) '\pi)']);


subplot(3,1,3)

[A,~]=freqz(h2,1,-0.9*pi:0.005:pi);

plot(-0.9:1.9/size(A,2):1-1.9/size(A,2),20*log10(abs(A)));

xlim([-0.9 1]);

xlabel('Normalized Frequency  (\times\pi rad/sample)');

ylabel('Magnitude (dB)');

title('Original IIR Filter');
```

2D Filtering:

Theory:

The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection (see zero crossing edge detectors). The Laplacian is often applied to an image that has first been smoothed with something approximating a Gaussian smoothing filter in order to reduce its sensitivity to noise, and hence the two variants will be described together here. The operator normally takes a single gray level image as input and produces another gray level image as output.

Mean filtering is a simple, intuitive and easy to implement method of smoothing images, *i.e.* reducing the amount of intensity variation between one pixel and the next. It is often used to reduce noise in images.

The idea of mean filtering is simply to replace each pixel value in an image with the mean (`average') value of its neighbors, including itself. This has the effect of eliminating pixel values which are unrepresentative of their surroundings. Mean filtering is usually thought of as a convolution filter. Like other convolutions it is based around a kernel, which represents the shape and size of the neighborhood to be sampled when calculating the mean. Often a 3×3 square kernel is used, as shown in Figure 1, although larger kernels (*e.g.* 5×5 squares) can be used for more severe smoothing. (Note that a small kernel can be applied more than once in order to produce a similar but not identical effect as a single pass with a large kernel.)

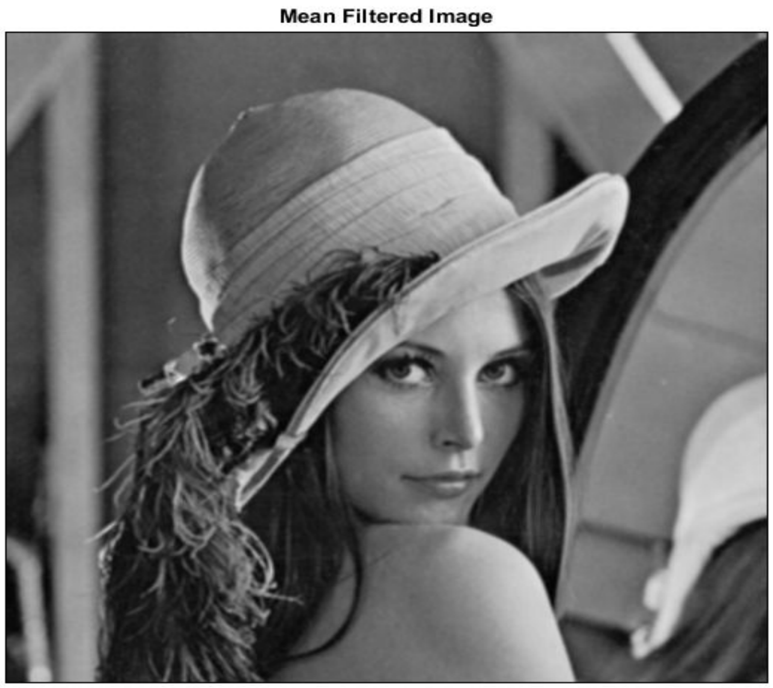
Code Snippets:

```
clear all;

img = imread('grayscale.png');

figure;

imshow(img);

title('Original Image');
```

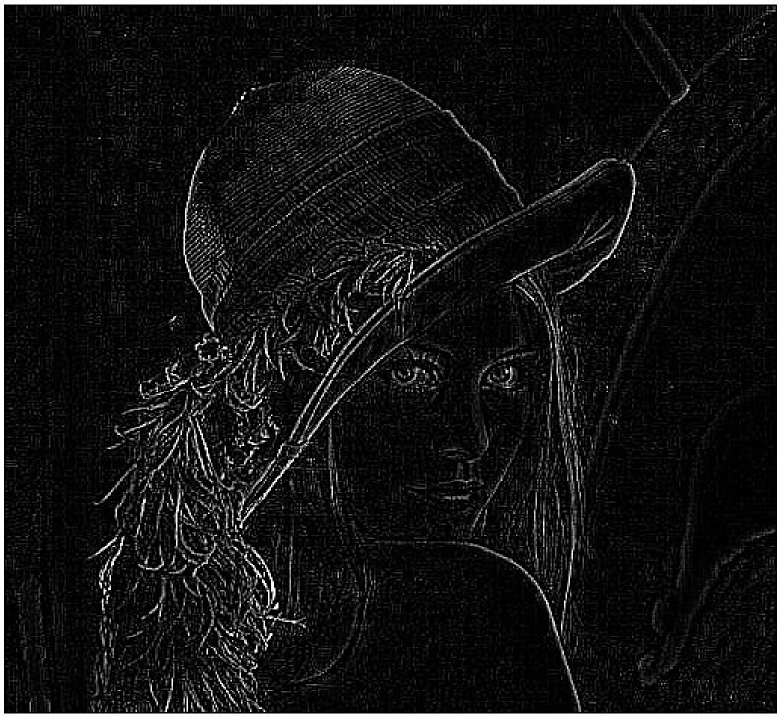
```
img = double(img);

meanker = repmat([1 1 1],3,1)/9;

lker = [-1 -1 -1;-1 8 -1; -1 -1 -1];

mimg = zeros(size(img));

limg = zeros(size(img));



for rows=2:length(img(:,1))-1

    for cols = 2:length(img(1,:))-1

        mimg(rows,cols) = sum(sum(img(rows-1:rows+1,cols-1:cols+1).*meanker));

        limg(rows,cols) = sum(sum(img(rows-1:rows+1,cols-1:cols+1).*lker));


    end

end


figure;

imshow(uint8(mimg))

title('Mean Filtered Image');

immse(mimg,img)


figure;

imshow(uint8(limg));

title('Laplacian Filtered Image');

immse(limg,img)


%LOW PASS FILTERING

b = zeros(size(img));
```

```
for rows=1:512

    for cols=1:512

        if(sqrt((rows-256)^2 + (cols-256)^2) <= 100)

            b(rows,cols) = 1;

        end

    end

end

imgfft = fftshift(fft2(img));

lpimg = ifft2(imgfft.*b);


figure;

imshow(uint8(abs(lpimg)));

title('Low pass filtered Image');
```

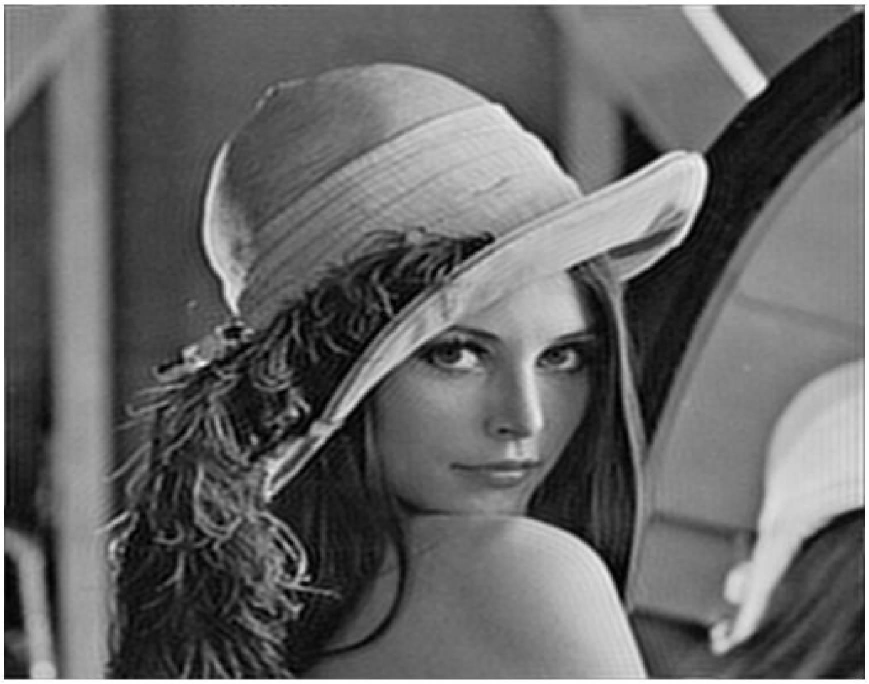
Original Image


Mean Filtered Image

Laplacian Filtered Image



Low pass filtered Image



Discussions:

- Windowing method is used to realize Low pass filters by making the impulse response finite, making it possible to be implemented using limited hardware.
- Multiple windowing schemes were used to obtain various FIR Filters and we can see that as the order of the filter increases the number of side lobes increased, at the same time the cut-off was sharper.
- The filters whose windows were narrower in time domain were stretched in the frequency domain.
- Rectangular has a better transition time at the cost of a worse stop band attenuation while Blackmann window had a better stop band attenuation at the cost of cutoff.
- While using windowing method, we have already used a truncated Sinc sequence. This is as MATLAB only takes finite sequences and since we are going to window it anyway, it doesn't matter. So, in a sense, we have already rectangular-windowed it beforehand.
- Following from the above, it is clear that we cannot provide the frequency response of the above to the Parks-McClellan algorithm as we need the desired IIR frequency response to convert to FIR, and the above filter is already FIR.

- The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred, either in error or as an natural effect of a particular method of image acquisition.

- Because these kernels are approximating a second derivative measurement on the image, they are very sensitive to noise. To counter this, the image is often Gaussian smoothed before applying the Laplacian filter. This pre-processing step reduces the high frequency noise components prior to the differentiation step.

- To achieve an efficient implementation of the optimal filter design using the Parks-McClellan algorithm, two difficulties have to be overcome:

    1. Defining a flexible exchange strategy, and

    2. Implementing a robust interpolation method.

- In some sense, the programming involved the implementation and adaptation of a known algorithm for use in FIR filter design. Two faces of the exchange strategy were taken to make the program more efficient:

1. Allocating the extremal frequencies between the pass and stop bands, and

2. Enabling movement of the extremals between the bands as the program iterated.

- It was observed that the Parks-McClellan algorithm and windowing method has slight difference when it comes to the nature in the stop band region. The plot shown below describes the difference. The red line shows the plot of the Park Mcclellan and the blue is for the windowing method.