Lab 1

MINI PROJECT: Digital Filters and Their Applications

EECS3451

Name: Chethana Wickramasinghe

Student Number: 214866511

Professor: Peter Lian

13sth of April 2021

1. Introduction:

This Mini Project requires to design filters using MATLAB commands, plot frequency responses of filter and last design an appropriate filter that could remove noise from a given music signal.

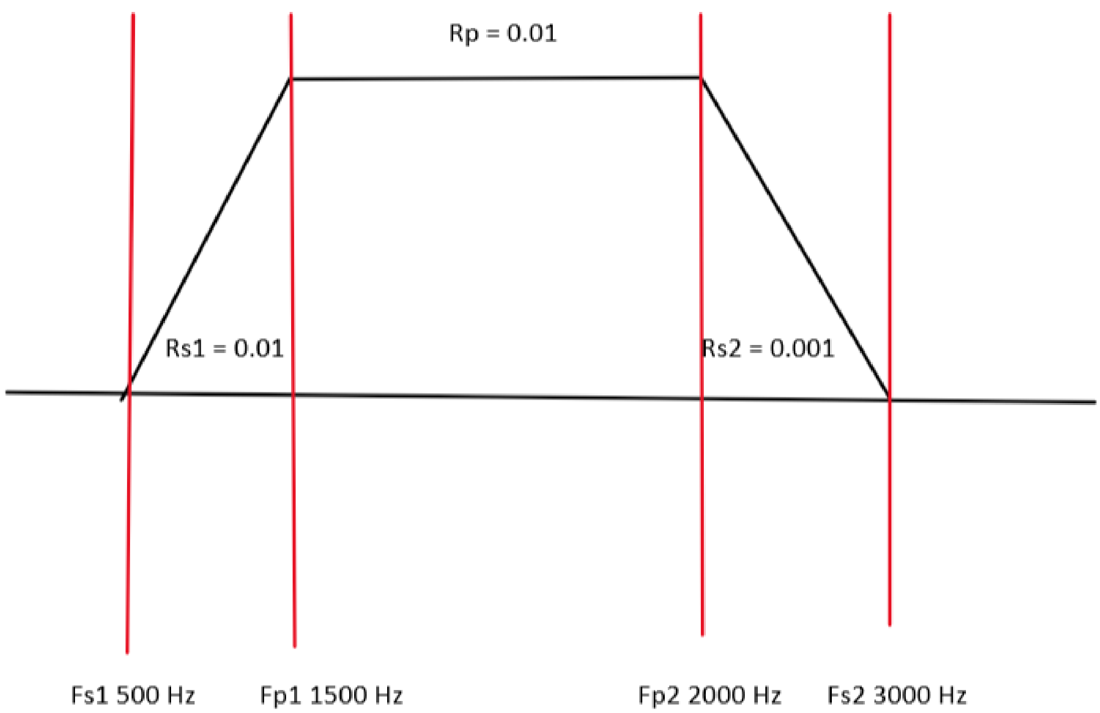
2. Equipment: MATLAB

3. Results and discussion:

Part 1: FIR Filter Design

Q1: Design a bandpass FIR filter meets the following specifications:

Stopband edge 1: 500Hz
Passband edge 1: 1500 Hz
Passband edge 2: 2000 Hz
Stopband edge 2: 3000 Hz
Sampling frequency: 8000 Hz
Pass-band ripple <= 0.01
Stopband ripple from 0 to 500Hz <= 0.01
Stopband ripple from 3000 to 4000Hz <= 0.001



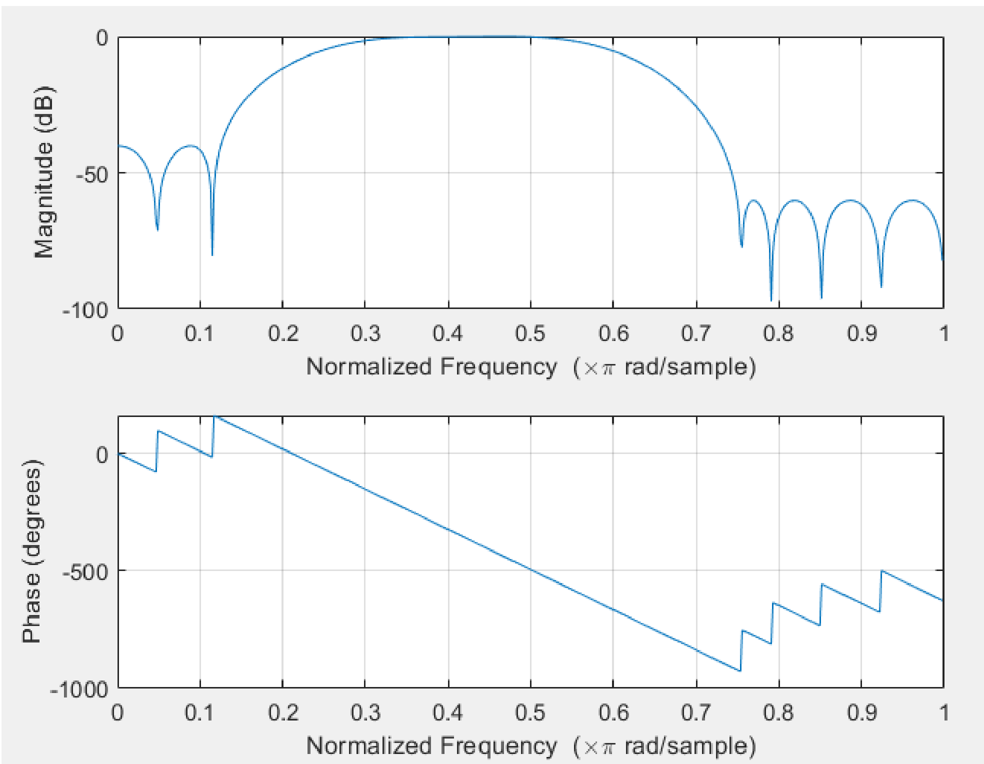
Bandpass FIR requirements

Name: Chethana Wickramasinghe Student #: 214866511

CODE

```
F=[500 ,1500, 2000, 3000]; % band pass edge frequencies
A=[0, 1, 0]; % a stop band(0), pass band(1) and a stop band(0)
Dev = [0.01, 0.01, 0.001]; % Ripples in bands
[N,Fi,Ai,W]=firpmord(F,A,Dev,8000);
% filter length / approximate order: N, normalized frequency band
edges: Fi, frequency band amplitudes: Ai, and weights W that meet
input specifications
B1=firpm(N,Fi,Ai,W);
% B1 gives the filter coefficients
freqz(B1,1); % displaying the magnitude and phase responses of the
filter
```

Magnitude and Phase responses of the filter



Filter length

$$Fs = 8000\ Hz$$

$$\Delta f = \frac{1500-500}{8000} = \frac{3000-2000}{8000} = \frac{1}{8} = 0.125$$

Low pass = $\frac{-20\log(\sqrt{\delta p \delta s})-13}{14.6\Delta f} + 1 = \frac{-20\log(\sqrt{0.01*0.001})-13}{14.6\Delta f} + 1 = 15.795$

High pass = $\frac{-20\log(\sqrt{\delta p \delta s})-13}{14.6\Delta f} + 1 = \frac{-20\log(\sqrt{0.01*0.01})-13}{14.6\Delta f} + 1 = 21.274$

Average of Low and High pass length = $\frac{15.795+21.274}{2} = 18.5344 \sim 19$

According to the code: $N\ generated\ using$ 'firpmord' $function = 19$ as well

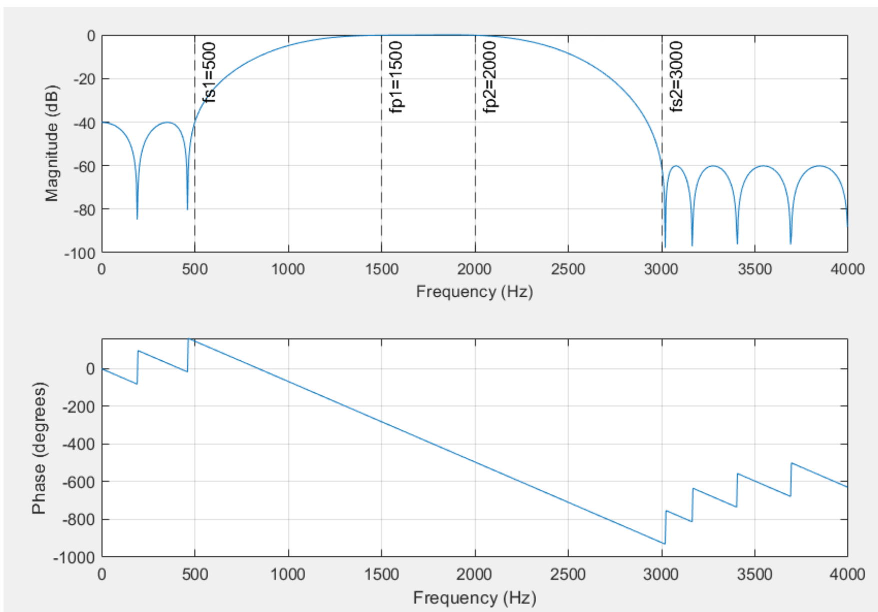
Filter coefficients

Following are the coefficients of the filter generated using the ‘firpmord’ function,

```
B1 =

  Columns 1 through 15

   -0.0089   -0.0039    0.0275    0.0187   -0.0025    0.0512   -0.0068   -0.2256   -0.1293    0.2845    0.2845   -0.1293   -0.2256   -0.0068    0.0512

  Columns 16 through 20

   -0.0025    0.0187    0.0275   -0.0039   -0.0089
```

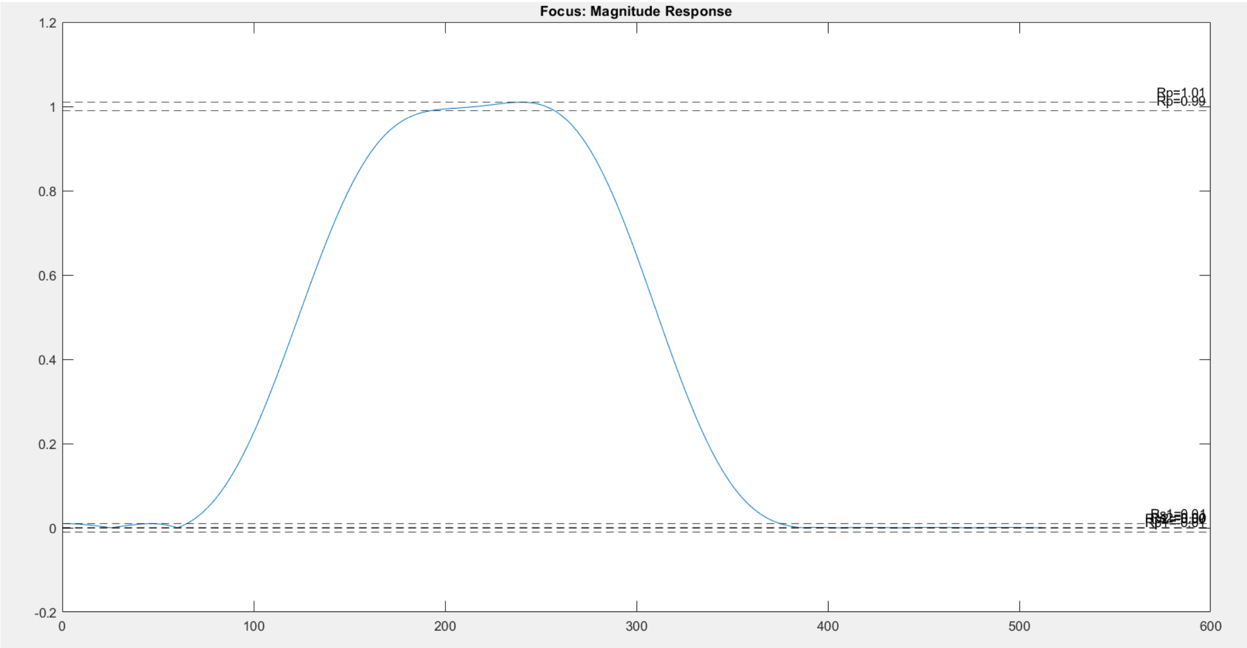
Verify

```
freqz(B1,1,1024,8000);
xline(500,'--k',sprintf('fs1=%0.0f', 500))
xline(1500,'--k',sprintf('fp1=%0.0f', 1500))
xline(2000,'--k',sprintf('fp2=%0.0f', 2000))
xline(3000,'--k',sprintf('fs2=%0.0f', 3000))
```

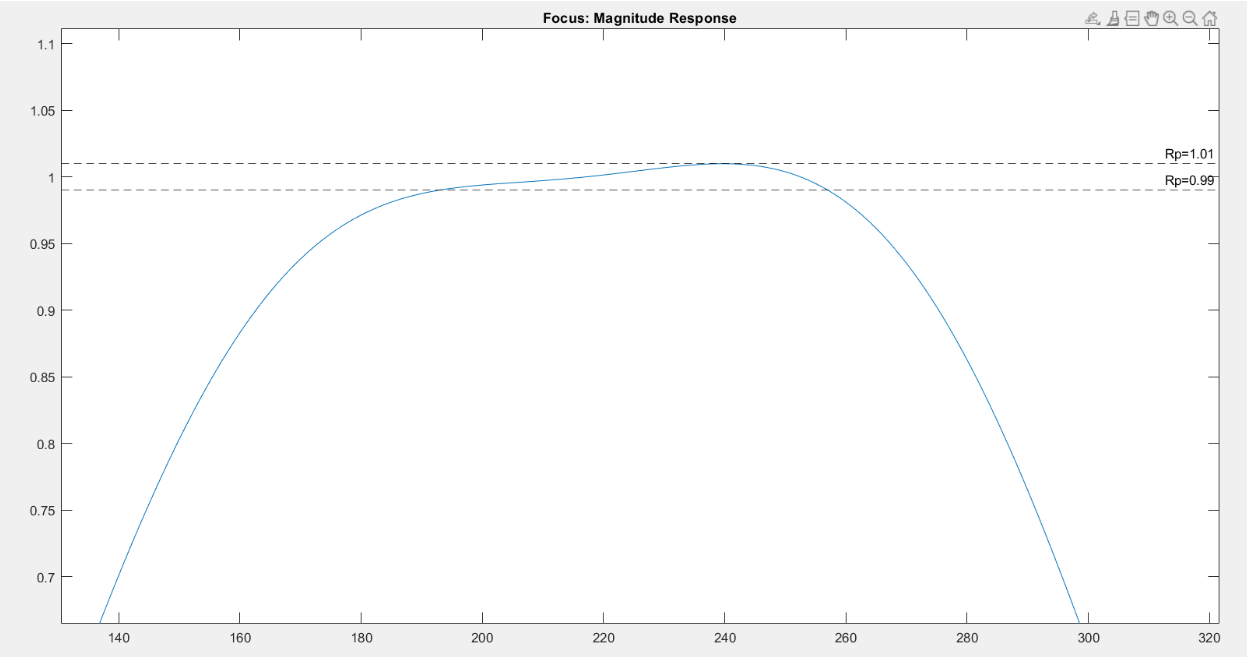


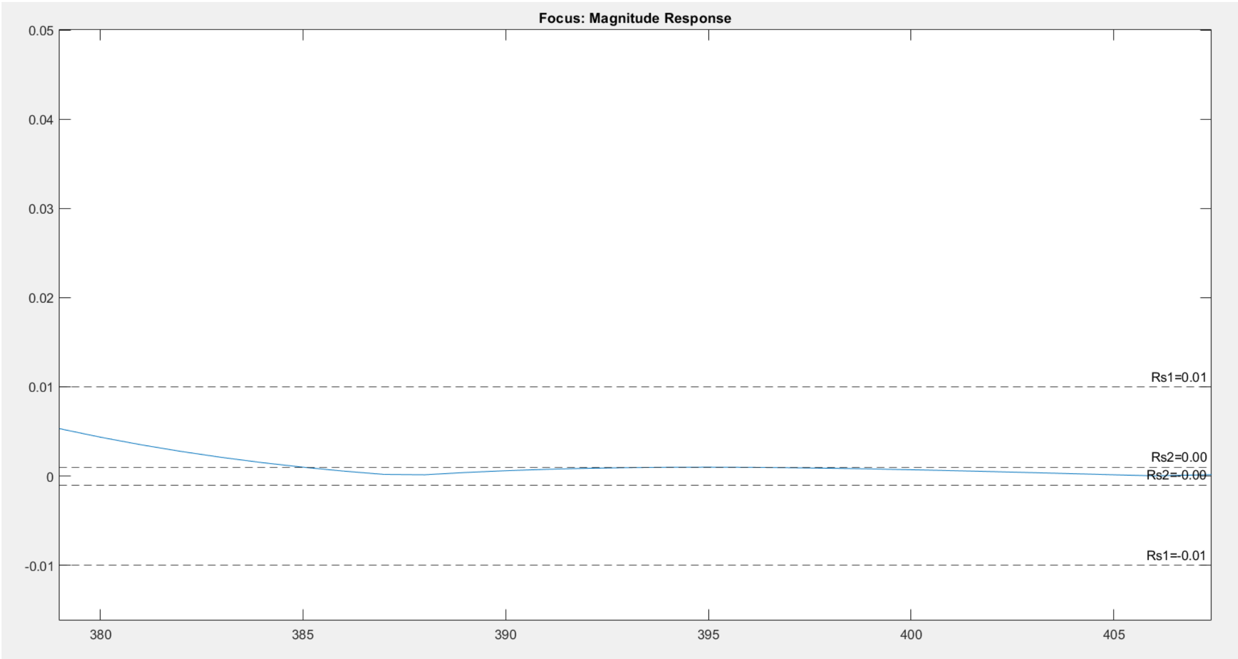
Verifying ripples

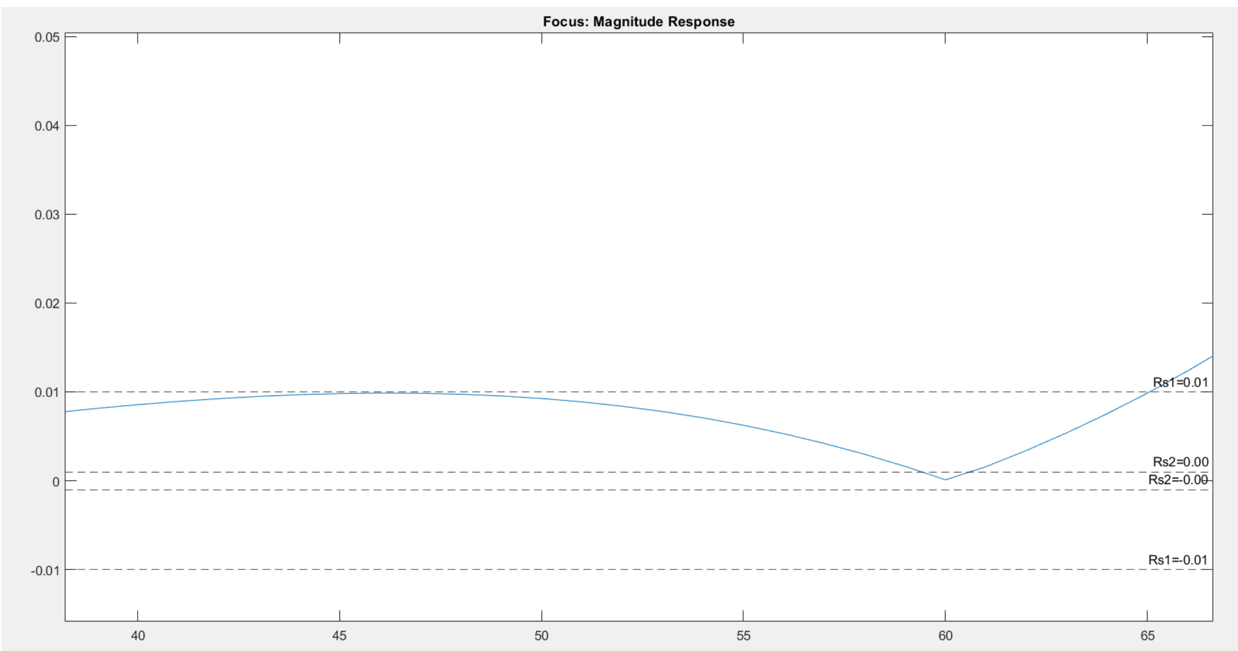
```
plot(abs(freqz(B1,1)))
title('Focus: Magnitude Response');
yline(0+0.01,'--k',sprintf('Rs1=%0.2f', 0+0.01))
yline(0-0.01,'--k',sprintf('Rp1=%0.2f', 0-0.01))
yline(0+0.001,'--k',sprintf('Rs2=%0.2f', 0+0.001))
yline(0-0.001,'--k',sprintf('Rp2=%0.2f', 0-0.001))
yline(1+0.01,'--k',sprintf('Rp=%0.2f', 1+0.01))
yline(1-0.01,'--k',sprintf('Rp=%0.2f', 1-0.01))
```


Focus: Magnitude Response
Rp=1.01
Rp=0.99
Rs1=0.01
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
100
200
300
400
500
600

Zoomed in


Focus: Magnitude Response
Rp=1.01
Rp=0.99
1.1
1.05
1
0.95
0.9
0.85
0.8
0.75
0.7
140
160
180
200
220
240
260
280
300
320


Focus: Magnitude Response
0.05
0.04
0.03
0.02
0.01
0
-0.01
Rs1=0.01
Rs2=0.00
Rs2=-0.00
Rs1=-0.01
380
385
390
395
400
405


Focus: Magnitude Response
0.05
0.04
0.03
0.02
0.01
0
-0.01
Rs1=0.01
Rs2=0.00
Rs2=-0.00
Rs1=-0.01
40
45
50
55
60
65

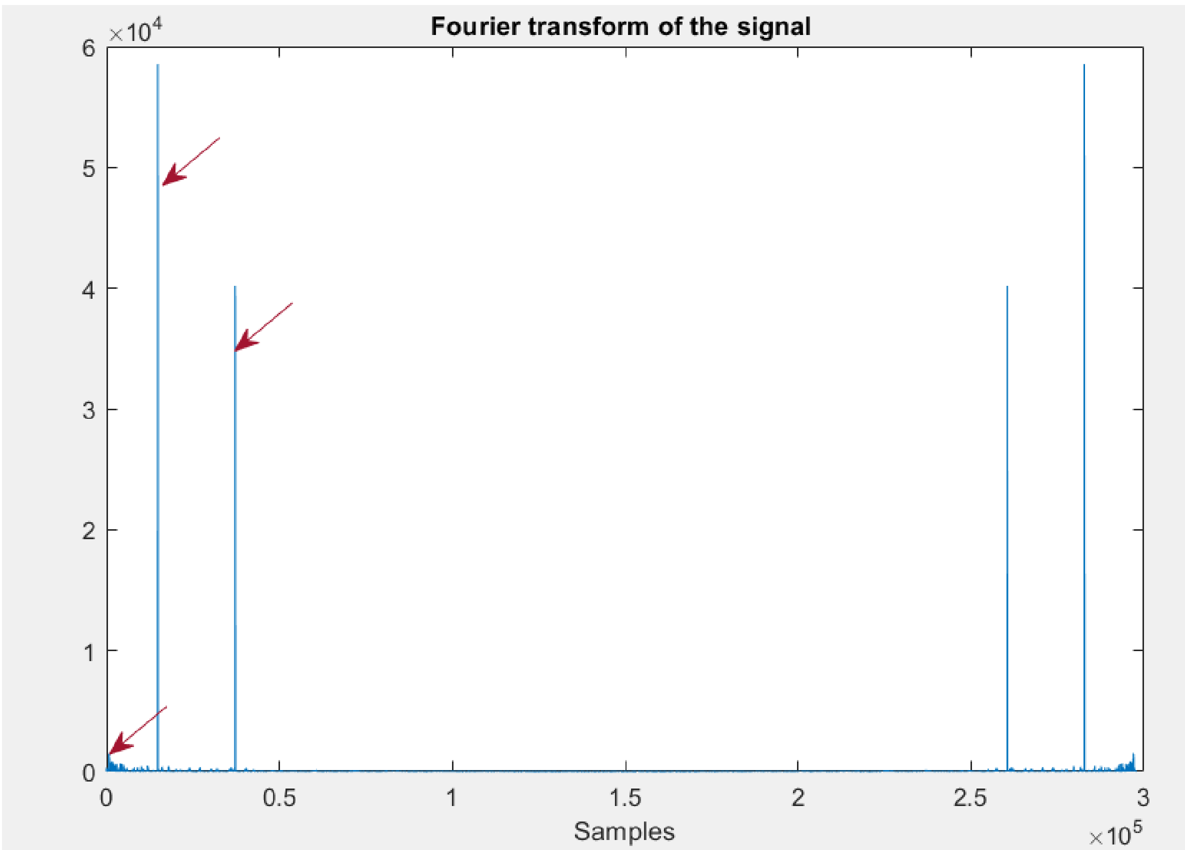
Part 2: Removing Noises in the Music Signal

Q2: Apply the digital filter(s) to recover original music from a noisy signal.

Following are the steps taken in the process of removing the noise from the provided "music_noisy.wav" file that can be found in the course website.

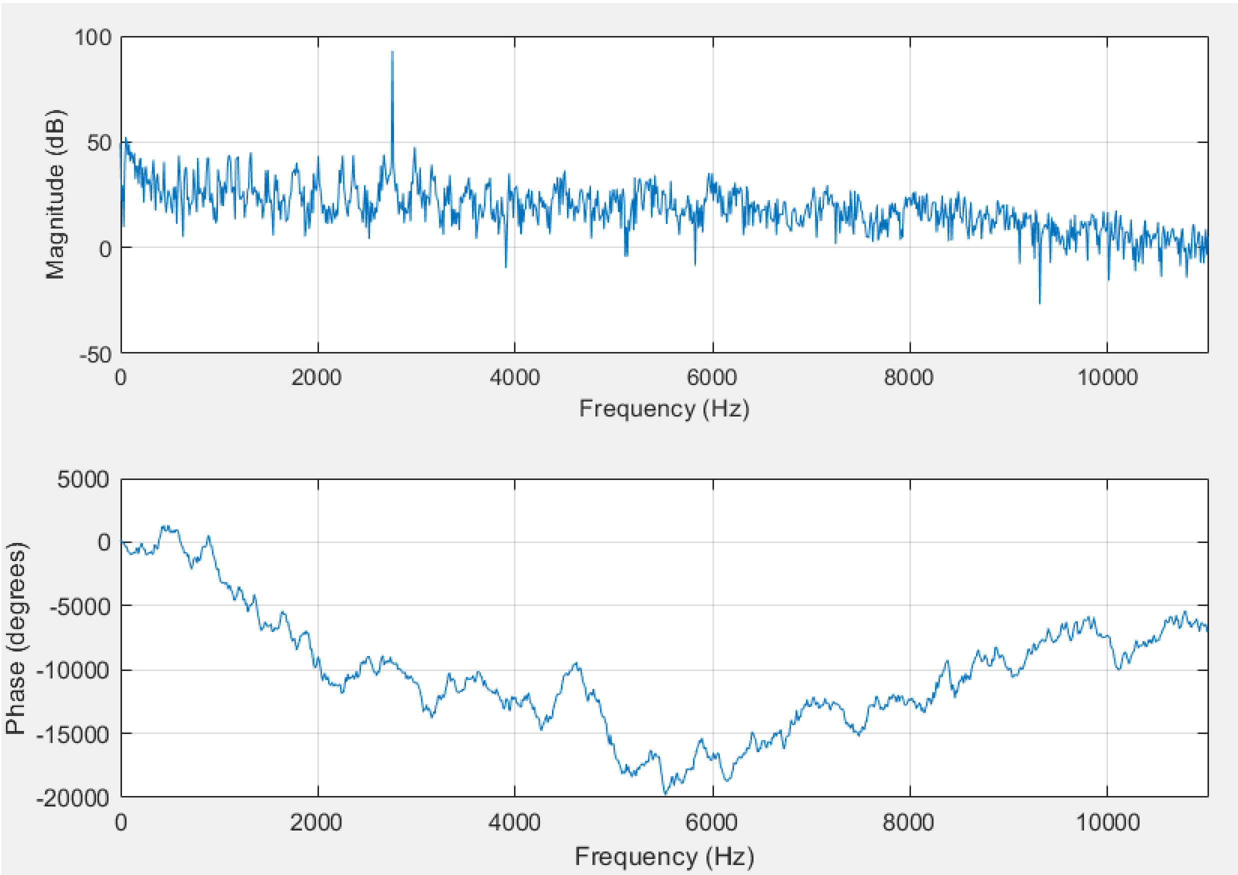
Identifying the noises in the noisy signal in the .wav file

```
[y,fs]=audioread('music_noisy.wav'); % reading the music with noise
from the .wav file
sound(y,fs); %listening to the corrupted music file
, y: input signal, fs: sample frequency of y
% fs = 22050 Hz
plot(abs(fft(y))); %Fourier transform of the noisy music
xlabel('Samples');
title('Fourier transform of the signal');
```



Red arrows are pointing at the unusual peaks/noise that are in this music file.

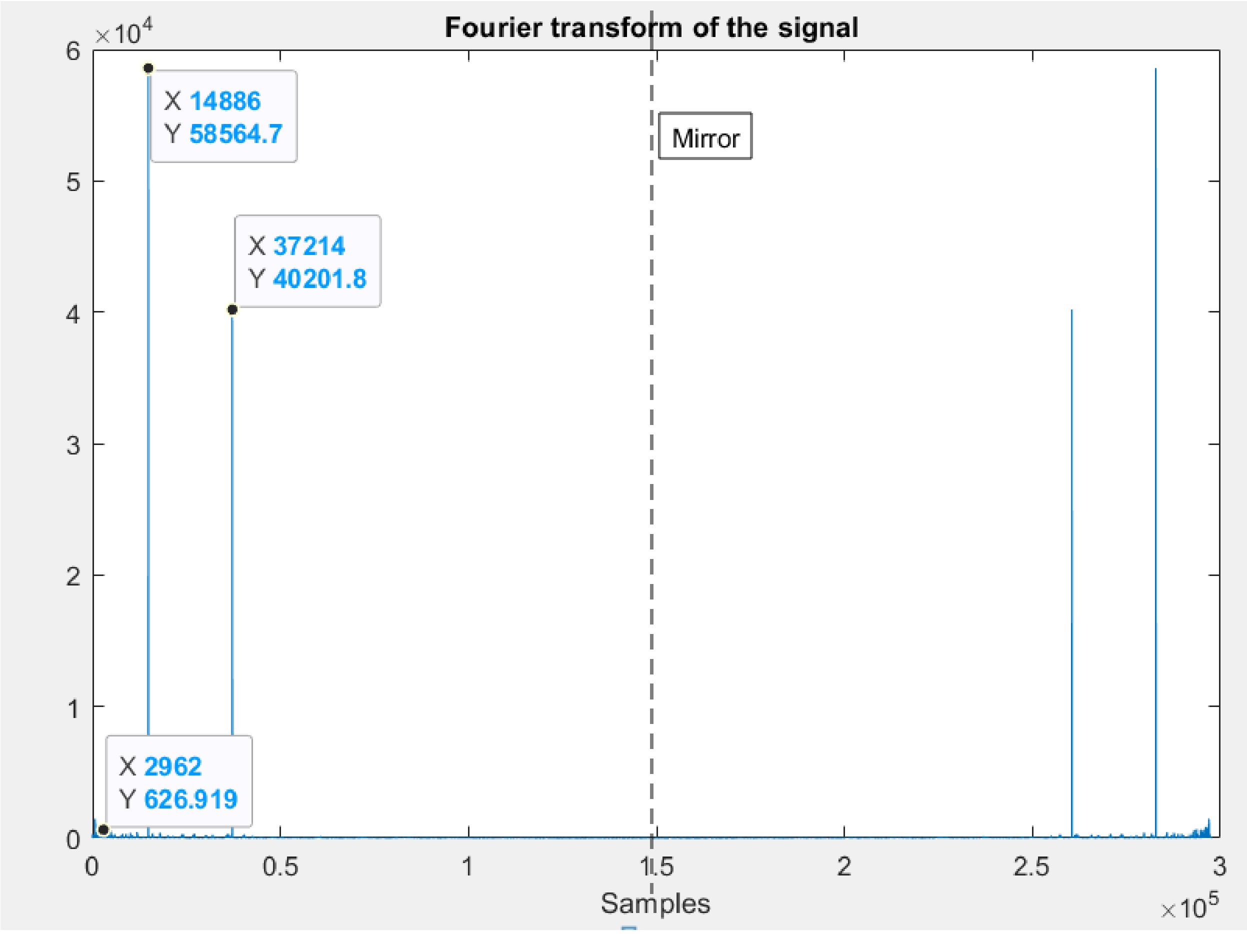
```
freqz(y,1,1024,fs); % displaying the magnitude and phase responses of
the noisy music
```



Name: Chethana Wickramasinghe Student #: 214866511

Designing a filter

From the previous observations



The observable space is about 300 000 samples and since we are looking at a sample frequency of 22050 coming from the "music_noisy.wav" file, ratio is around $\frac{300000}{22050} = 13.605$

There are multiple small peaks at the till 2962, which is about $\frac{2962}{13.6} = 217.7\ Hz$ -> can use a high pass filter to remove all the small peaks before ~ 217.7 Hz.

1) A High Pass filter with ~ 218 Hz as the cut off frequency.

There is a big peak at about 14886, which is $\frac{14886}{13.6} = 1094.558\ Hz$ -> can use a bandstop filter to remove this peak.

2) A Band Stop filter with ~ 1094 ± 300 Hz as the cut off frequency.

There is another big peak at about 37214, which is $\frac{37214}{13.6} = 2737\ Hz$ -> can use a bandstop filter to remove this peak.

3) A Band Stop filter with ~ 2727 Hz ± 300 Hz as the cut off frequency.

With the found specifications following are 2 methods used in this report to remove the noise.

1)Time Domain Method

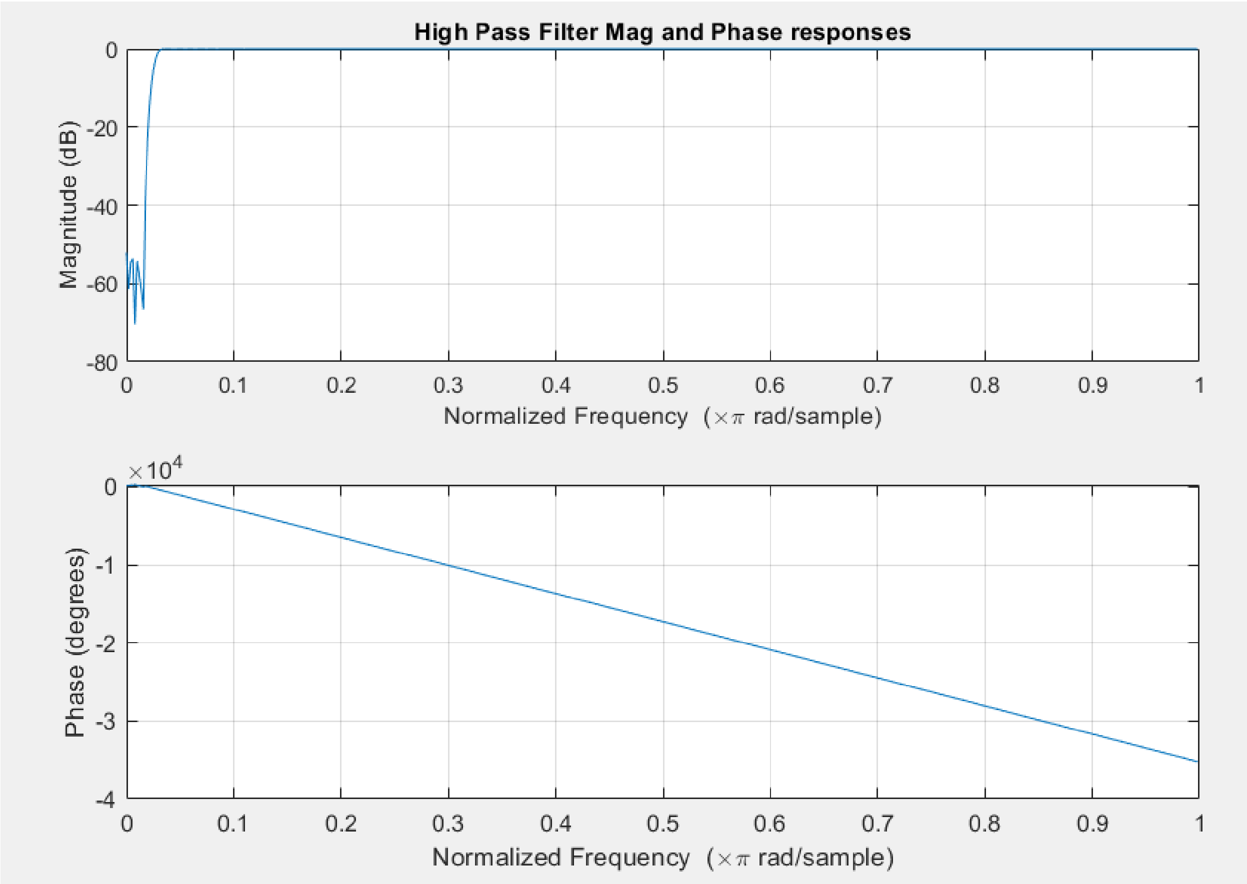
2)Frequency Domain Method

Functions used: '[fir1](fir1)'

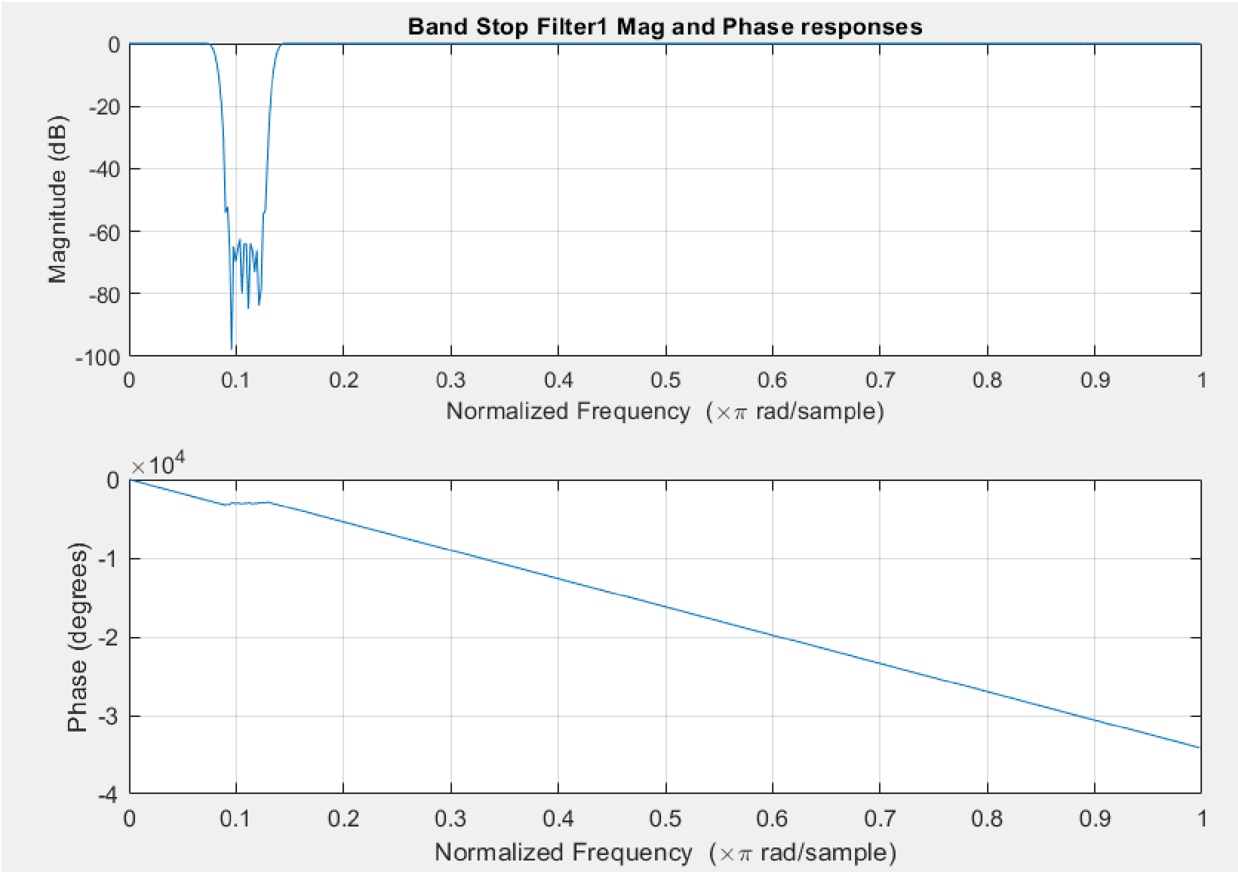
Note: fir1(n,Wn) -> decided to take a higher nth order (about 400) value since higher the number the more better results. Wn are the cut-off frequencies but will have slightly different values than decided before because following used in the code are adjusted so that a music is recovered in the best way possible.

CODE OF THE FILTER DESIGN

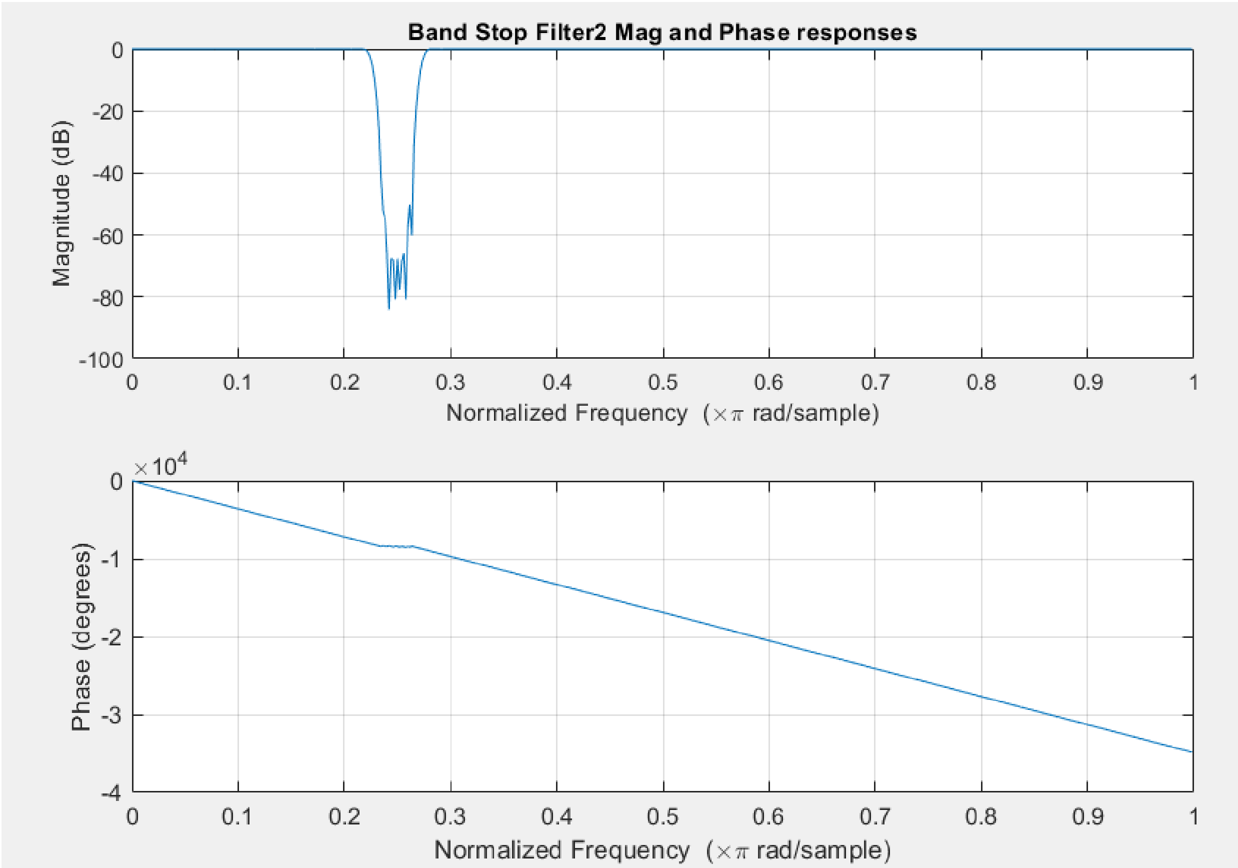
```
fhp1 = fir1(400, 274/(fs/2), 'high'); % high pass filter
freqz(fhp1,1);
title('High Pass Filter Mag and Phase responses');
```




```
fbs2 = fir1(400, [900, 1500]/(fs/2), 'stop'); % band stop filter 1
freqz(fbs2,1);
title('Band Stop Filter1 Mag and Phase responses');
```




```
fbs3 = fir1(400, [2500, 3000]/(fs/2), 'stop'); % band stop filter 2
freqz(fbs3,1);
title('Band Stop Filter2 Mag and Phase responses');
```

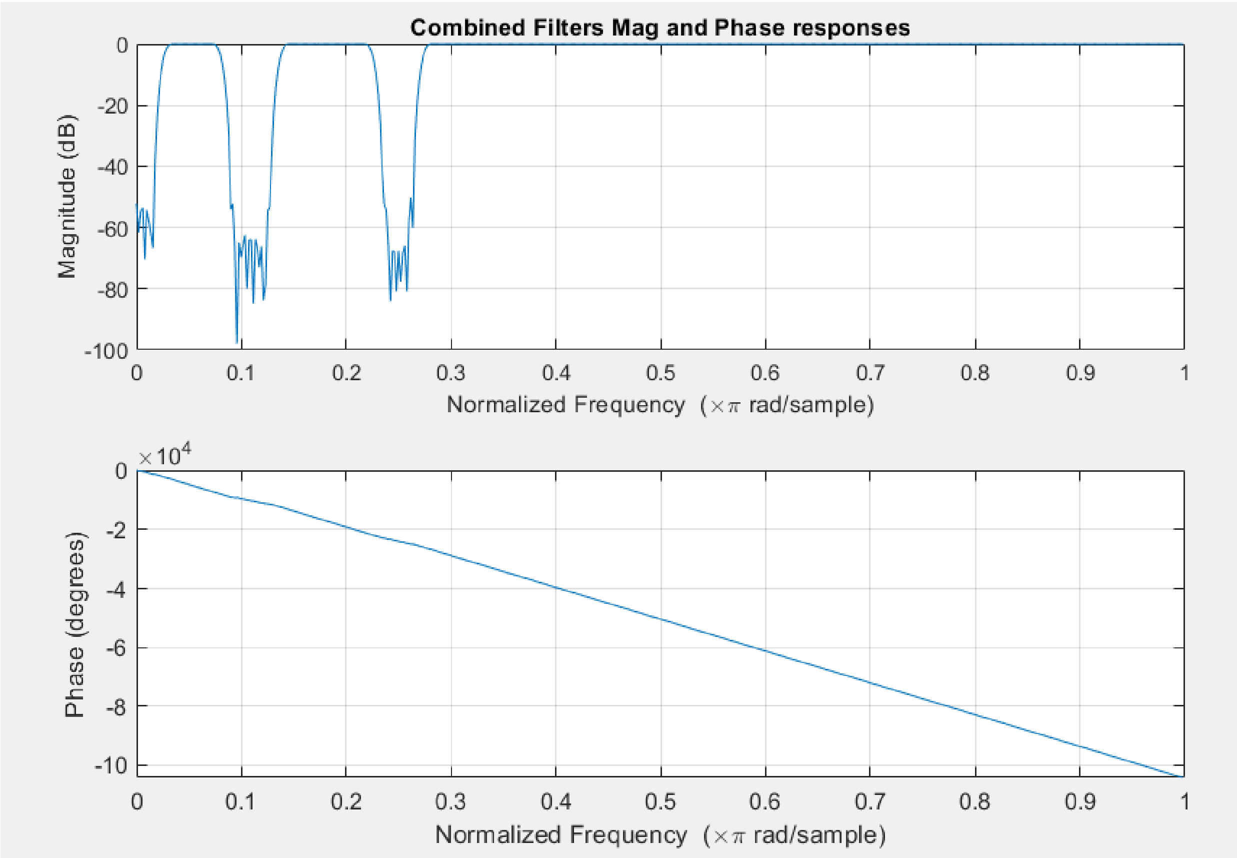


1) Time Domain Method

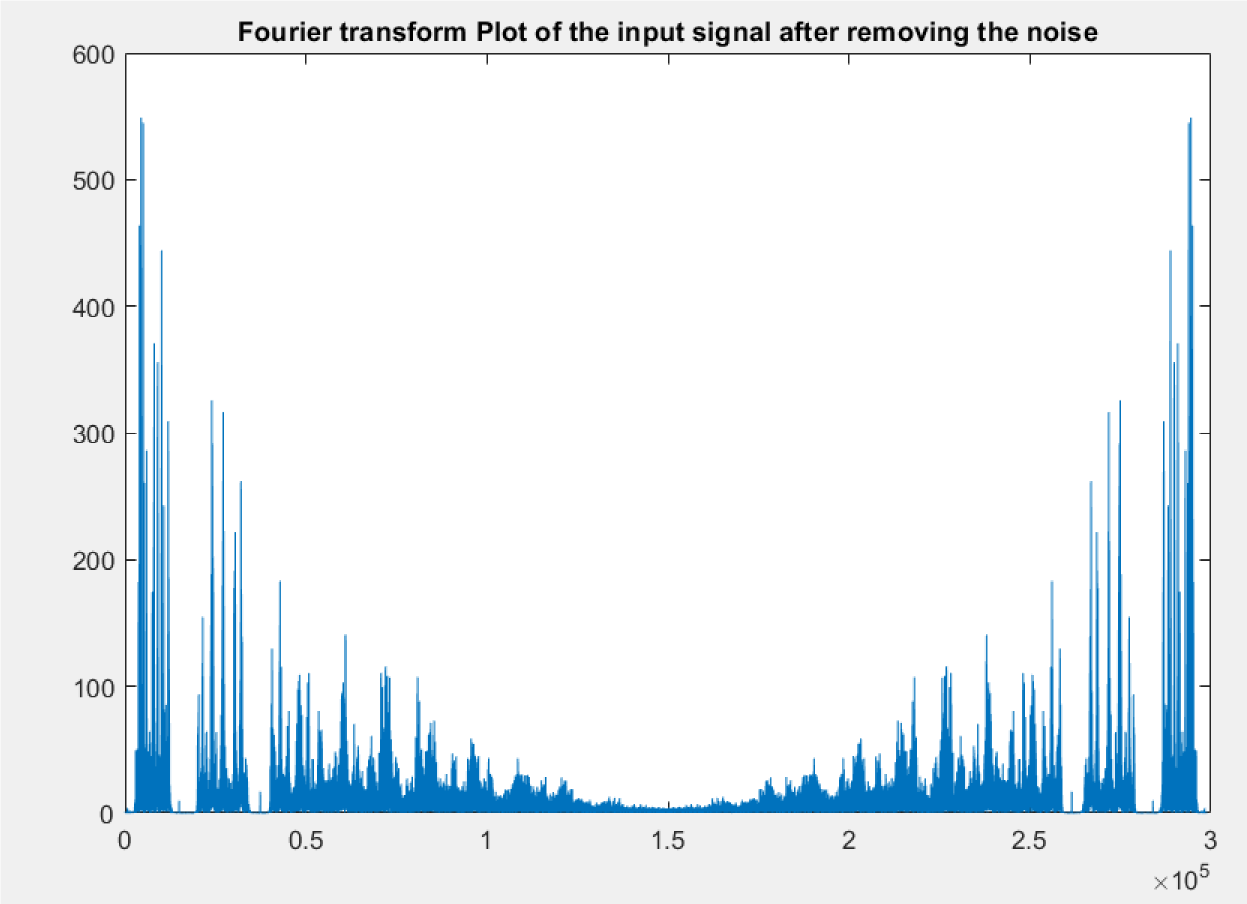
Input signal -> convolution with the filters -> Output filtered signal

CODE AND RESULTS

```
filter_Combine = conv(fhp1, fbs2); % convolution of 1st 2 filters
filter_Combine = conv(filter_Combine, fbs3); % convolution with the
remaining filters
freqz(filter_Combine,1);
title('Combined Filters Mag and Phase responses');
```




```
filtered_music = conv(filter_Combine, y); % convolution of the signal
with filters
plot(abs(fft(filtered_music))); % Fourier transform of the filtered
music
title('Fourier transform Plot of the input signal after removing the
noise');
```




```
sound(filtered_music,fs) % listen to verify
audiowrite('Filtered_music_td_method.wav',filtered_music,fs); % store
the new filtered music
```

Click on this link to listen to the filtered music.

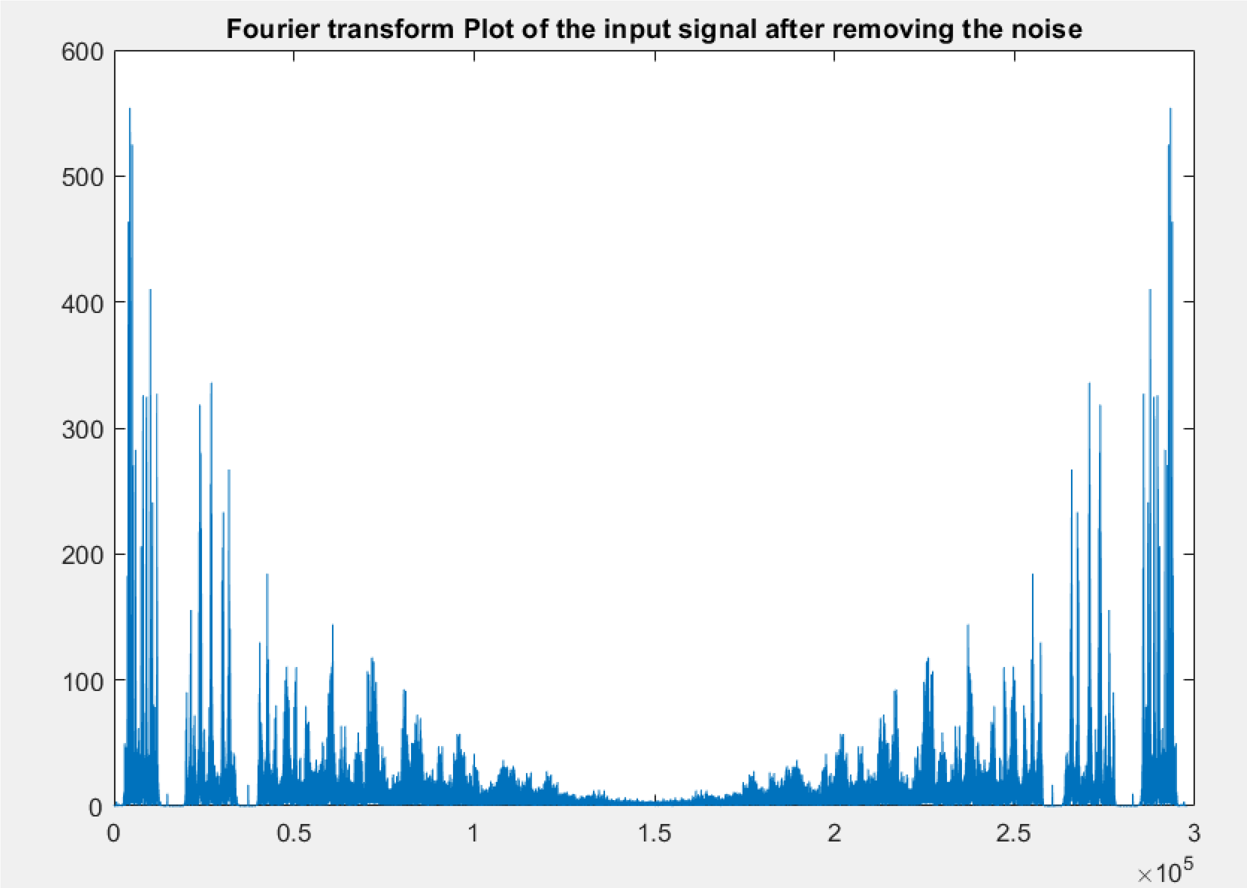
2) Frequency Domain Method

Input signal X filters -> Output filtered signal

CODE AND RESULTS

```
fft_y = fft(y); % Fourier transform of the input signal (music with
noise)
fft_fhp1 = fft(fhp1, length(fft_y))'; % Fourier transform of first
filter
fft_fbs2 = fft(fbs2, length(fft_y))'; % Fourier transform of second
filter
fft_fbs3 = fft(fbs3, length(fft_y))'; % Fourier transform of third
filter
filtered_music = fft_y.*fft_fhp1.*fft_fbs2.*fft_fbs3; % multiply input
signal with the filters in their frequency domains
```

```
plot(abs(filtered_music));
title('Fourier transform Plot of the input signal after removing the
noise');
```




```
filtered_music_time_domain = ifft(filtered_music); % inverse fourier
transform of the final filtered signal to listen it later
sound(filtered_music_time_domain,fs) % listen to verify
audiowrite('Filtered_music_fd_method.wav',filtered_music_time_domain,f
s); % store the new filtered music
```

Click on this link to listen to the filtered music.

HURRAYY!!! NO MORE NOICE

4. Conclusion: state what you learn from this lab, lab objectives you achieved, and any difficulties you met.

Project helped to learn how to design FIR filters and use then in different methods to remove noise from a music signal using MATLAB.

It was satisfying to hear the music without the noise and make me wonder how these techniques have been applied in real work application.

Thank you for having set up this interesting mini project and other previous labs. I learned the most from these labs-based assignments.