ELEC 342 Lab 5: Introduction to Simulink and Filter Design using MATLAB.

Name :	JUNPENG GAI
ID number:	40009896
Course number :	ELEC342-X Lab 2224
Date performed:	Monday, 3 April 2023
Due Date:	Monday, 20 April 2023
Lab Instructor Name:	Salameh, Ahmed

"I certify that this submission is my original work and meets the Faculty's Expectations of Originality"

盖俊朋

Contents

1.Objectives

The objectives of this lab are to get a deeper understanding of the signal process in the MATLAB and get to know how to use the FDAT in the MATLAB environment. We have 2 parts in this lab section, one we learn how to use the FDAT in the MATLAB and then we learn how to use the Simulink to design the filter and export the data in the MATLAB workspace.

2.Theory

2.1 Filters:

In the MATLAB, we can design following filters: Lowpass Filter, Highpass filters, Bandpass Filter, Band stop Filer, Notch Filter. There are 2 ways designing the filter.

2.1.1 SPTool:

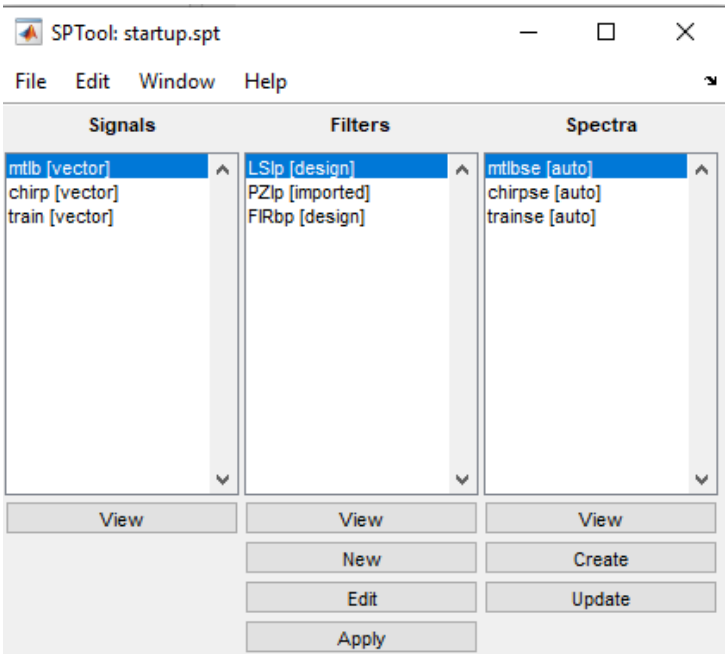


Figure 1 SPTool plane.

By clicking the New, we can design our own filter in the following menu.

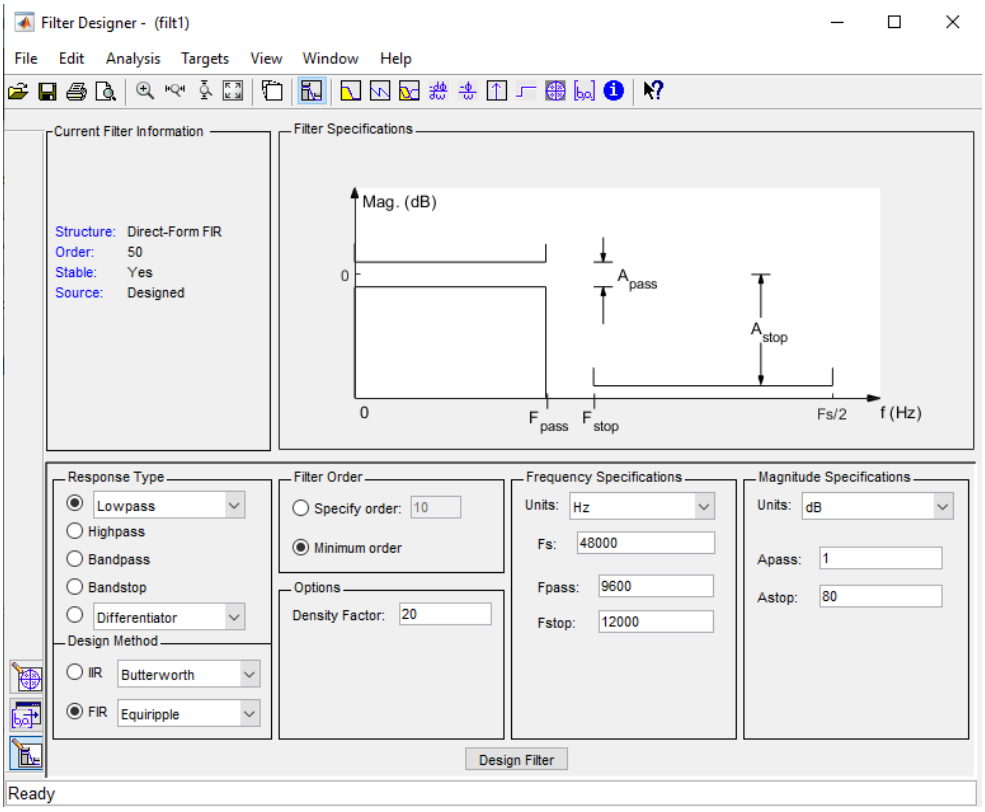


Figure 2 Filter design plane

In this plane we can specify the option for type of filters, cutoff frequency, stop frequency, passband and stop band magnitude and what order we want.

2.1.2 FDATool

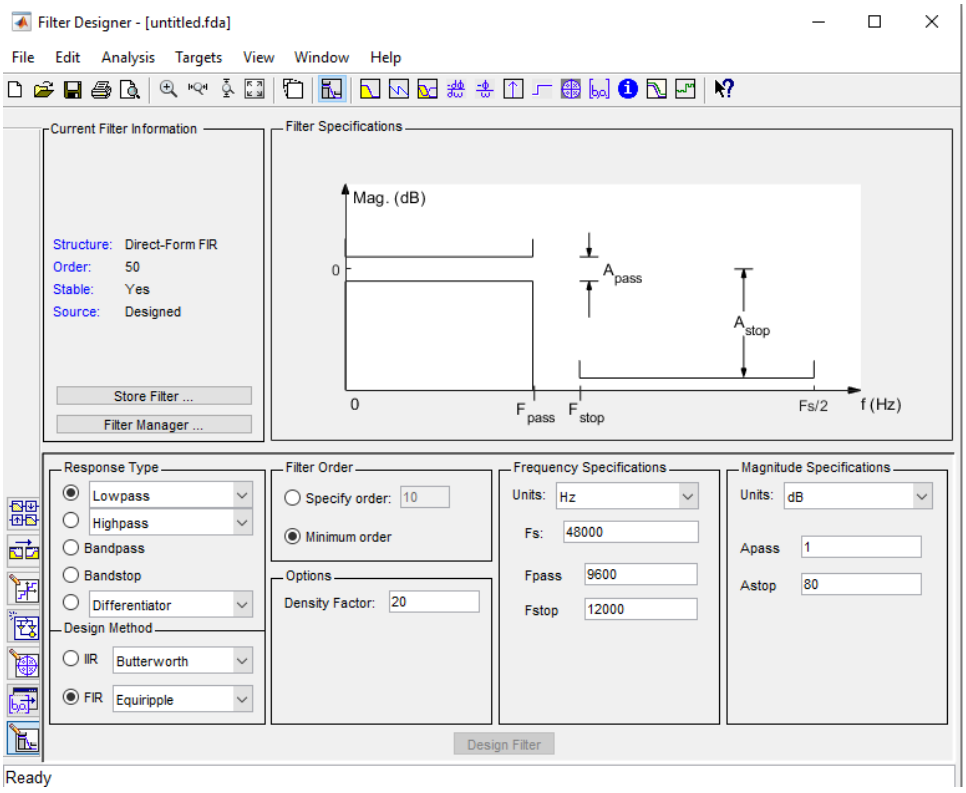


Figure 3 FDATool menu

Here is another option of designing a filter in MATLAB, by typing FDATool in the command window we will see the above figure, which is familiar to the SPTool menu. We can specify our specification in this menu and the filter will be generated.

2.2 How to design a filter:

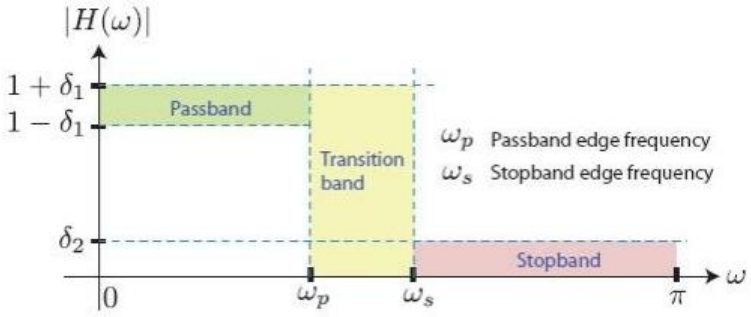


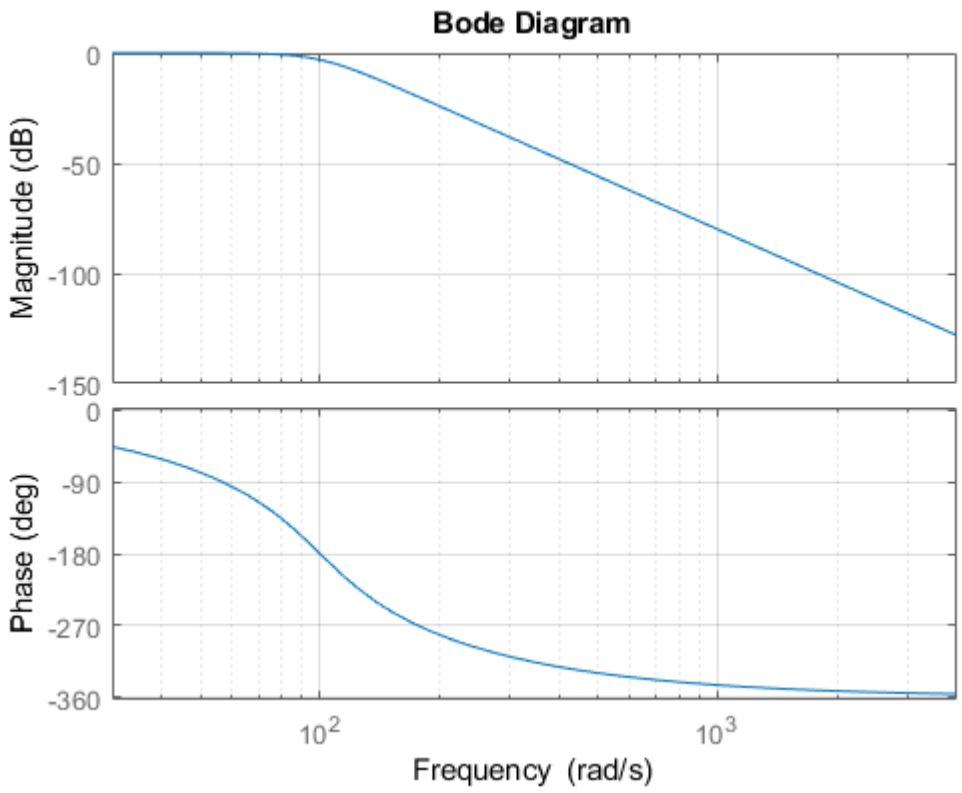
Figure 4 Specifications in the filter.

We can also design a filter by using the MATLAB directly to calculate the coefficients of the Fourier Transform.

Here is an example:

```
%junpeng gai
%40009896
[b, a] = butter (4, 100,'s');
g = tf(b, a)
bode(g,{30,4000});
grid;
```

```
g =

                 1e08

  ----------------------------------------------------

  s^4 + 261.3 s^3 + 3.414e04 s^2 + 2.613e06 s + 1e08

Continuous-time transfer function.
```



Published with MATLAB® R2020a

Which means we can directly design a filter and plot its magnitude and phase diagram.

2.3 Simulink:

We will use different step interval and frequency to sample the signal in the lab section. We can see the different results when we applied different sample frequencies.

In the library there is a block called HighPass Filter showing below, which we can modify the specification before we run the simulation.

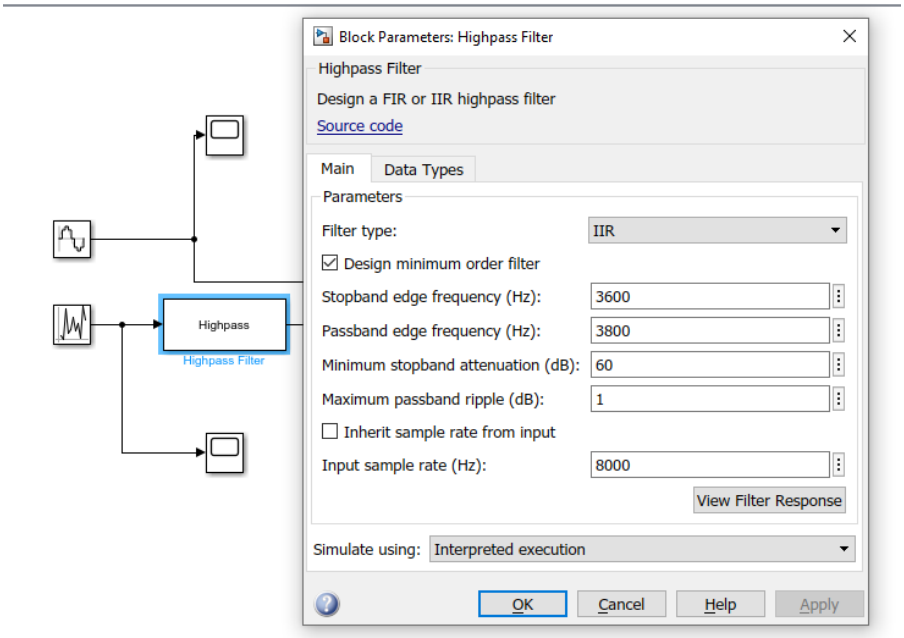


Figure 5 Simulink Filter specifications

2.4 Importing Audio into MATLAB Workspace

In the MATLAB there are some useful commands:

2.4.1 sound(y,fs)

In this command we can play the audio with fiven sampling frequency Fs.

2.4.2 [mySound Fs] =audioread('name.wav')

We can use this command to save our audio file to the work space.

2.4.3 audiowrite(mysound,Fs,'filename')

We can save our vector to a .wav file with this command.

I will use all above command in the prelab question, so more details will be covered in the following section.

2 Prelab questions

2.1 Prelab question 1

sends the one-dimensional array or vector sampledSignal to the audio card in your computer. The second argument specifies the sampling frequency in Hertz. The values in sampledSignal are assumed to be real numbers in the range [−1, 1]. Values outside this range are clipped to −1 or 1.

Download the audio file "lab_5_Audio_1.wav" from the course directory /groups/e/elec364_1 (see Laboratory Guidelines). Use wavread command to read the audio file "lab_5_Audio_1.wav" >> [y,fs]=wavread('lab_5_Audio_1.wav');

After reading, listen to

a) >> sound(y,fs)

*b) >> sound(0.25*y,fs) and >> sound(4*y,fs)*

Explain in what way these are different from what you heard in the part (a).

*c) >> sound(y, fs/2) and >> sound(y, fs*2) Explain how these are different from what you heard in the part (a).*

```
%JUNPENG GAI
%40009896
[y,fs]=audioread('lab_5_Audio_1.wav');
sound(y,fs)
%sound(0.25*y,fs)
%sound(4*y,fs)
%sound(y, fs/2)
%sound(y, fs*2)
```

Published with MATLAB® R2020a

2.1.1 a

It's the normal speed.

2.1.2 b

Sound (0.25*y,fs) will be smaller and sound (4*y,fs)'s sound will be louder.

2.1.3 c

Sound (y, fs/2) will be slower and sound (y, fs*2)'s sound will be faster.

2.2 Prelab question 2

Question 2. Designing the Filters Using FDATOOL, perform the following tasks, assuming a sampling rate of 8 kHz: I. Design a minimum order, stable, lowpass Butterworth filter with a passband frequency of 1 kHz and a stopband frequency of 1.4 kHz. Make the attenuation 1 dB at the passband frequency and 80 dB at the stopband frequency. II. Design a minimum order, stable, lowpass Chebyshev Type I filter with the same specifications as the Butterworth filter. III. Design a lowpass FIR filter using the Blackman Window with a cutoff frequency of 1 kHz. Specify the order of the filter such that the first minimum in the stopband (preceding the first lobe) is as close to 1.4 kHz as possible without exceeding it.

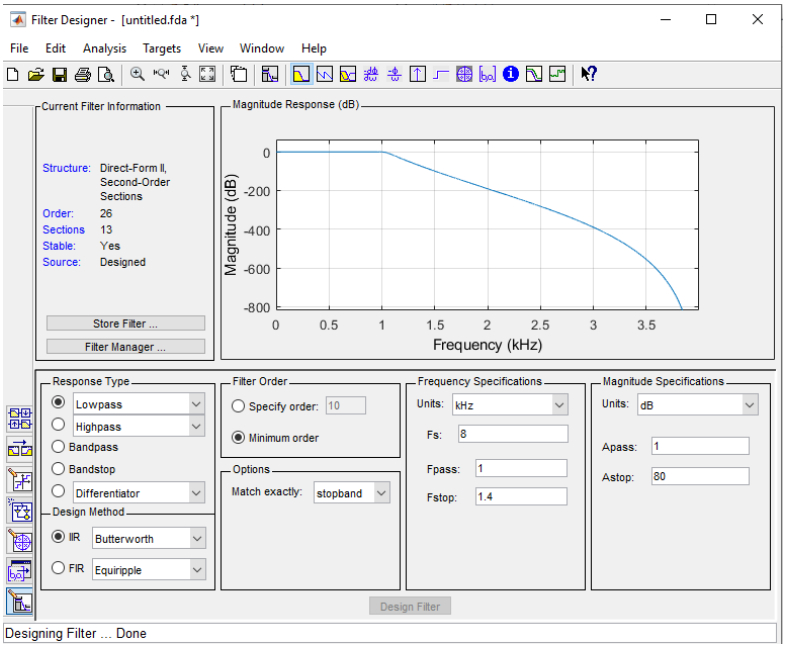


Figure 6 lowpass Butterworth filter

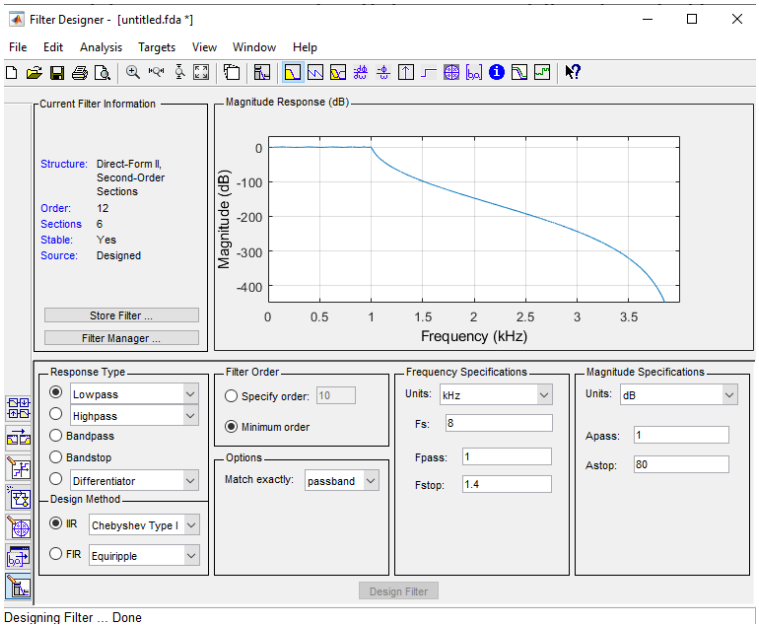


Figure 7 Chebyshev Type I

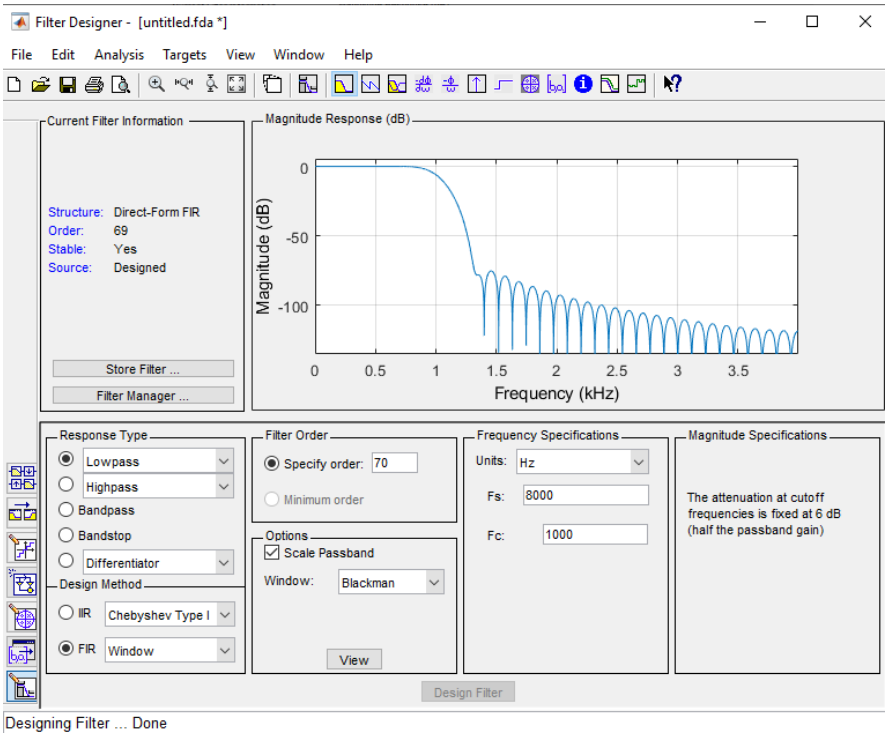


Figure 8 Blackman Window

2.2.1 A

a) What is the order of the lowpass Butterworth filter you designed?

Based on the figure 6. We can see that the order of the Butterworth filter is 26.

2.2.2 B

b) What is the order of the lowpass Chebyshev Type I filter you designed?

Based on the figure 7. We can see that the order of the Butterworth filter is 12.

2.2.3 C

c) Compare the implementation cast of each of the 3 filters. Do you see how inefficient the windowing technique is? How much more expensive in terms of memory is the windowing technique from the best IIR filter?

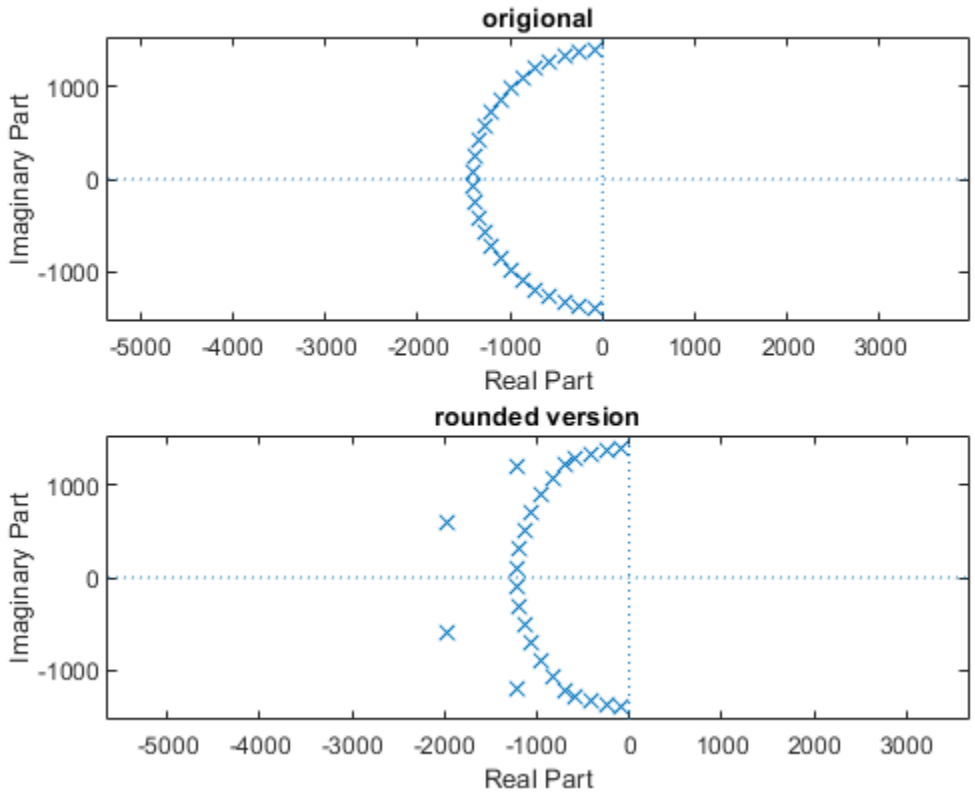
Yes, the window method is less efficient which requires an order of 69. So, a FIR filter with an order of 69 would require 68 multipliers and 4,692 adders for implementation. Which is far more expensive than 12 orders or 26 orders IIR.

2.2.4 D

d) For the filter you designed in Part I, round the filter coefficients (b,a) to the nearest integer value. Use the MATLAB round command:

```
%junpeng gai
%40009896
[b, a] = butter (26, 1400,'s');
b1=round(b);
a1=round(a);


subplot(2,1,1)
```

```
hold on
title('origional')
zplane(b,a)
hold off

subplot(2,1,2)
hold on
title('rounded version')
zplane(b1,a1)
hold off
```



Published with MATLAB® R2020a

2.3 Prelab question 3

Question 3. Creating a Noisy Signal using Simulink You will create a sinusoid single corrupted by high frequency noise. Both the sine wave and the noise will be discrete; thus, your whole Simulink model will be discrete (this is purposely done to simplify things). Here's how to create such a signal:

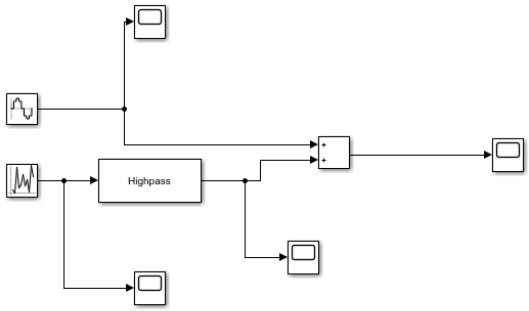


Figure 9 Simulink for question 3

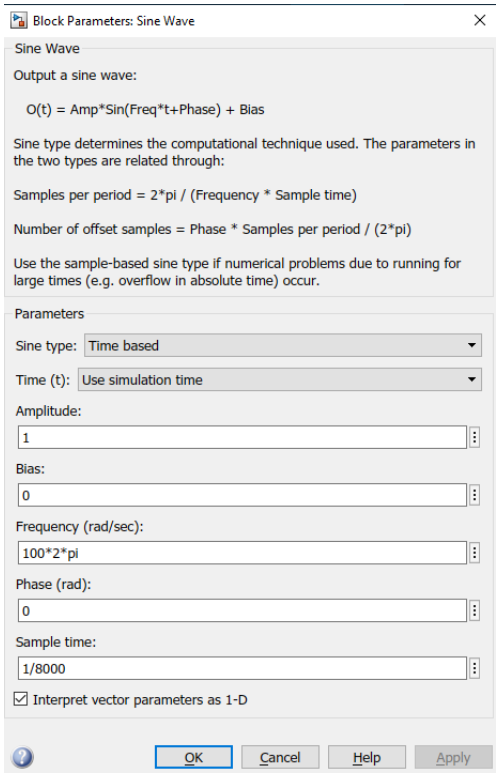


Figure 10 Original sin wave parameter

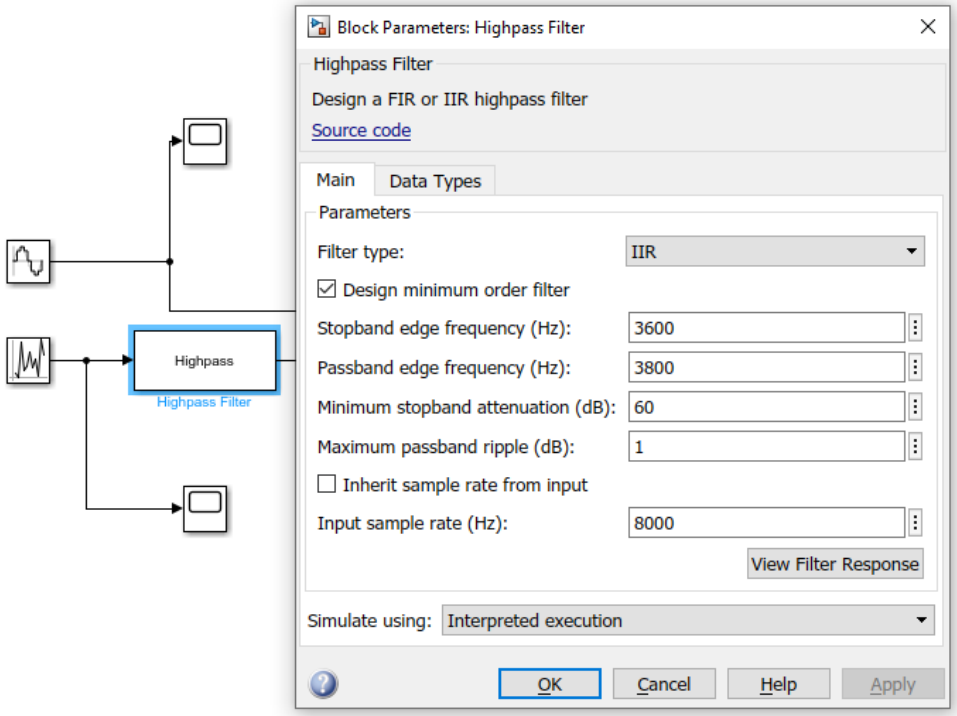


Figure 11 Filter parameter

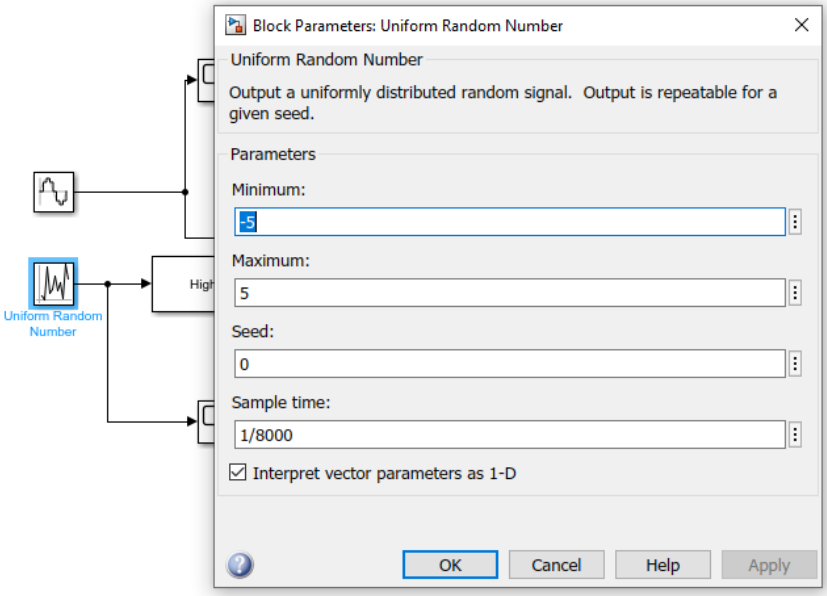


Figure 12 Uniform random number block

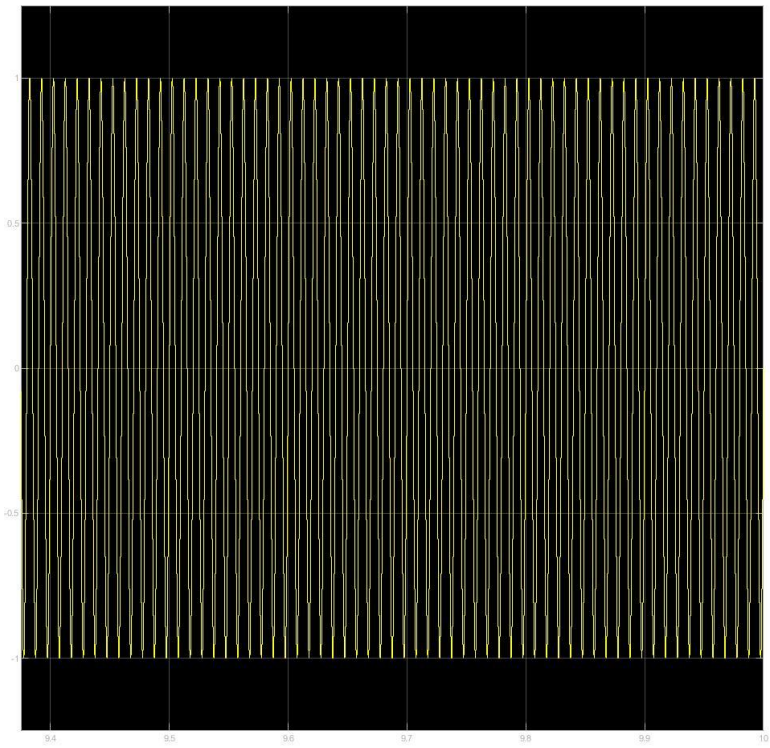


Figure 13 Original signal

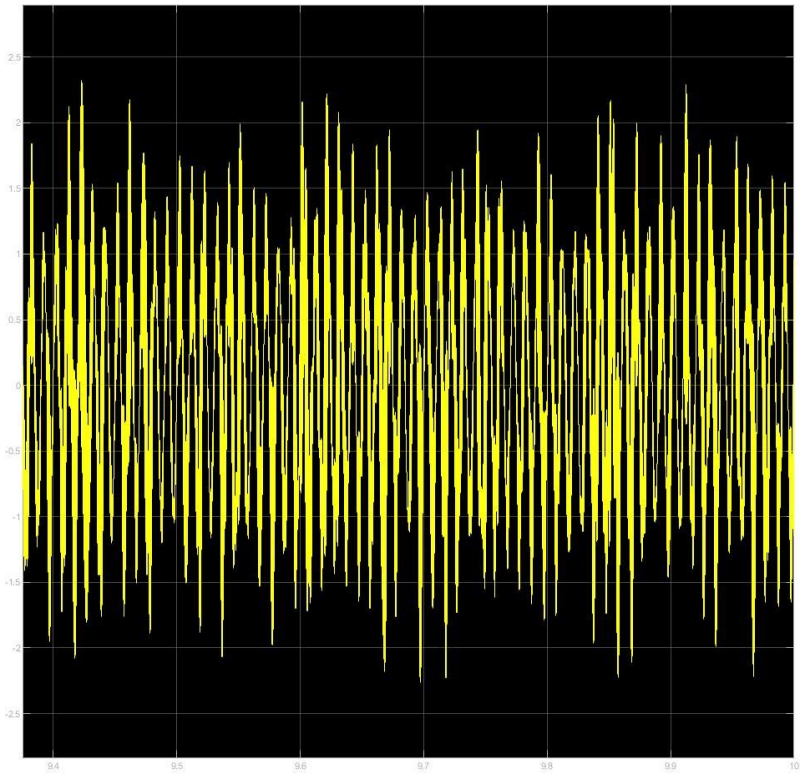


Figure 14 Noisy signal

Also, I saved the data in the work space as required.

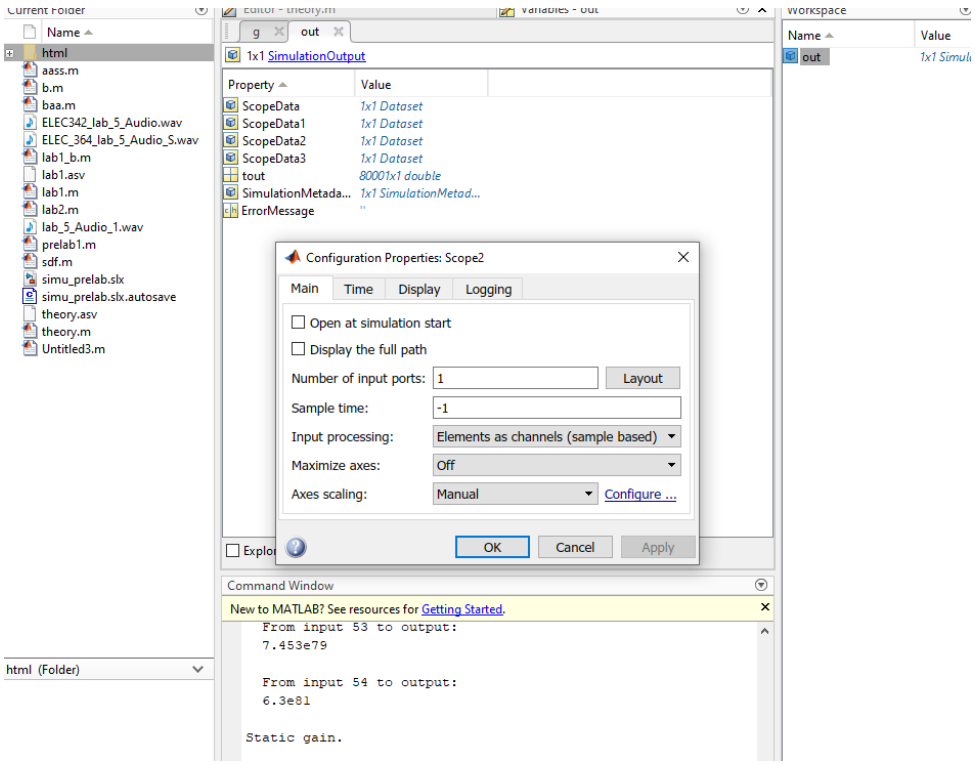


Figure 15 Save data to work space

3 Tasks/Results/Discussion

3.1 Problem 1

Problem 1. Consider the input signal $x(t) = \sin(100t) + \sin(2000t) + \sin(6000t)$ We need to pass the term $\sin(2000t)$ and to attenuate the other terms. a) Design a digital fourth and eighth-order IIR Butterworth filters. Plot the input signal along with the magnitude response of the filter and its output signal for each filter. b) Design a digital fourth and eighth-order IIR Chebyshev type I filters. Plot the input signal along with the magnitude response of the filter and its output signal for each filter. (consider $Rp = 1\ dB$). c) Design a digital fourth and eighth-order IIR Elliptic filters. Plot the input

signal along with the magnitude response of the filter and its output signal for each filter. (consider Rp = 1 dB, Rs = 60 dB) d) For parts a), b), and c) plot the discrete Fourier transform of the input signal and the output signal for eighth order filters, you can use FFT command.

For parts a), b), and c) plot the discrete Fourier transform of the input signal and the output signal for eighth order filters, you can use FFT command.

I included (d) in the answer of (a), (b) and (c).

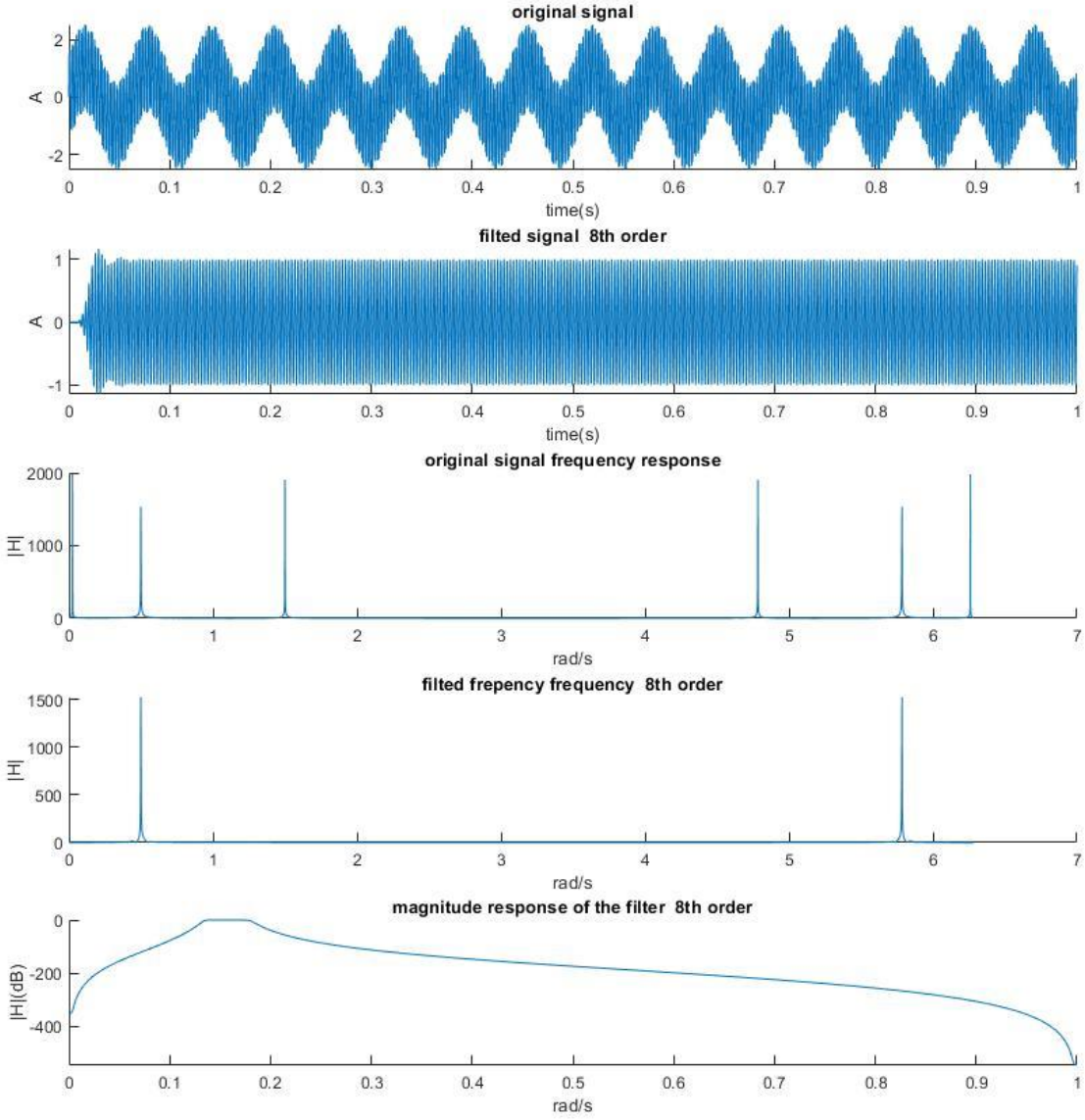
a) Design a digital fourth and eighth-order IIR Butterworth filters. Plot the input signal along with the magnitude response of the filter and its output signal for each filter.

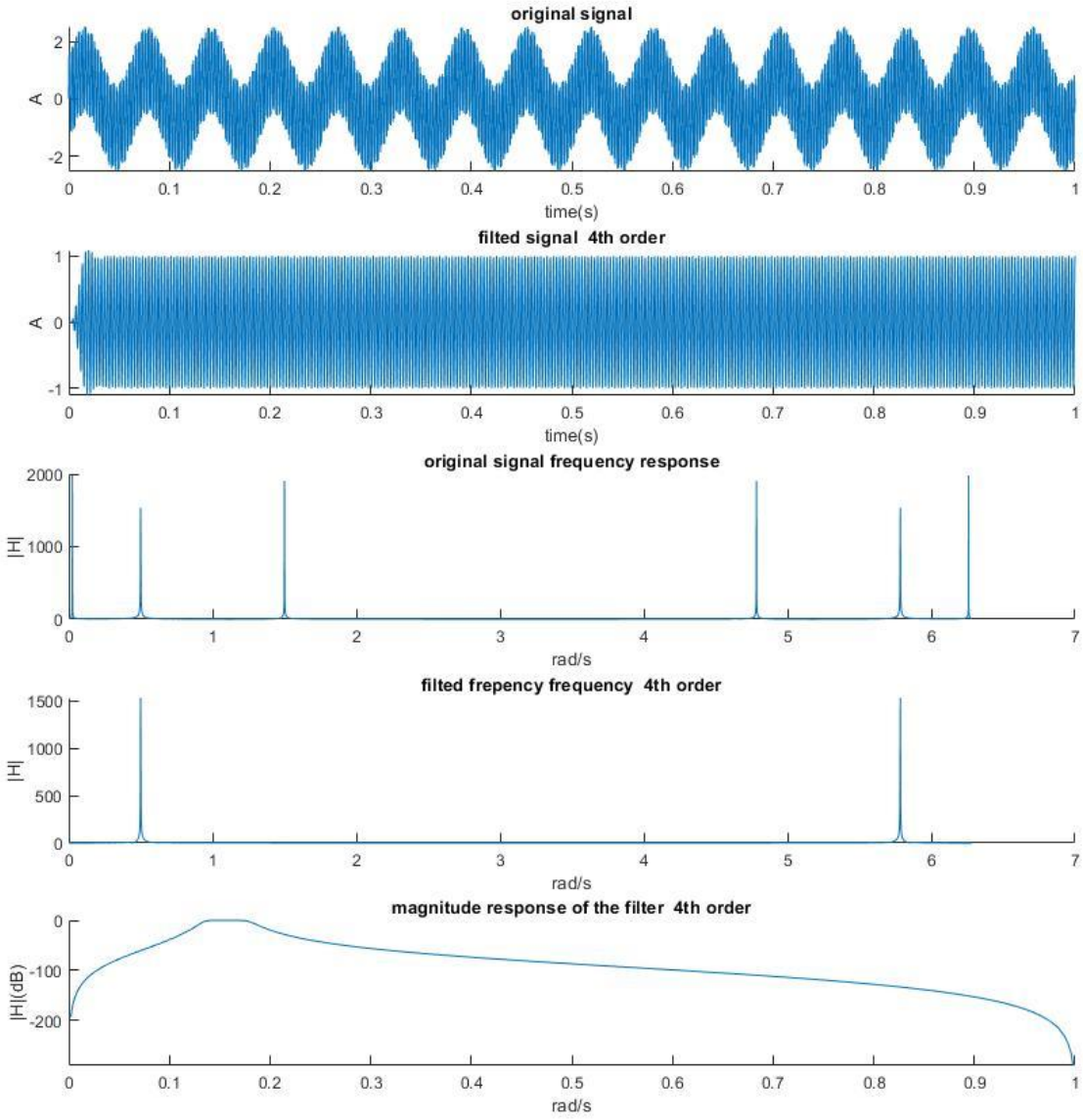
```
%JUNPENG GAI
%40009896
fs=4000;
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = butter(4, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
```

```
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
```

```
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = butter(8, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
```

```
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```


original signal
A
time(s)
filted signal 8th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 8th order
|H|
rad/s
magnitude response of the filter 8th order
|H|(dB)
rad/s



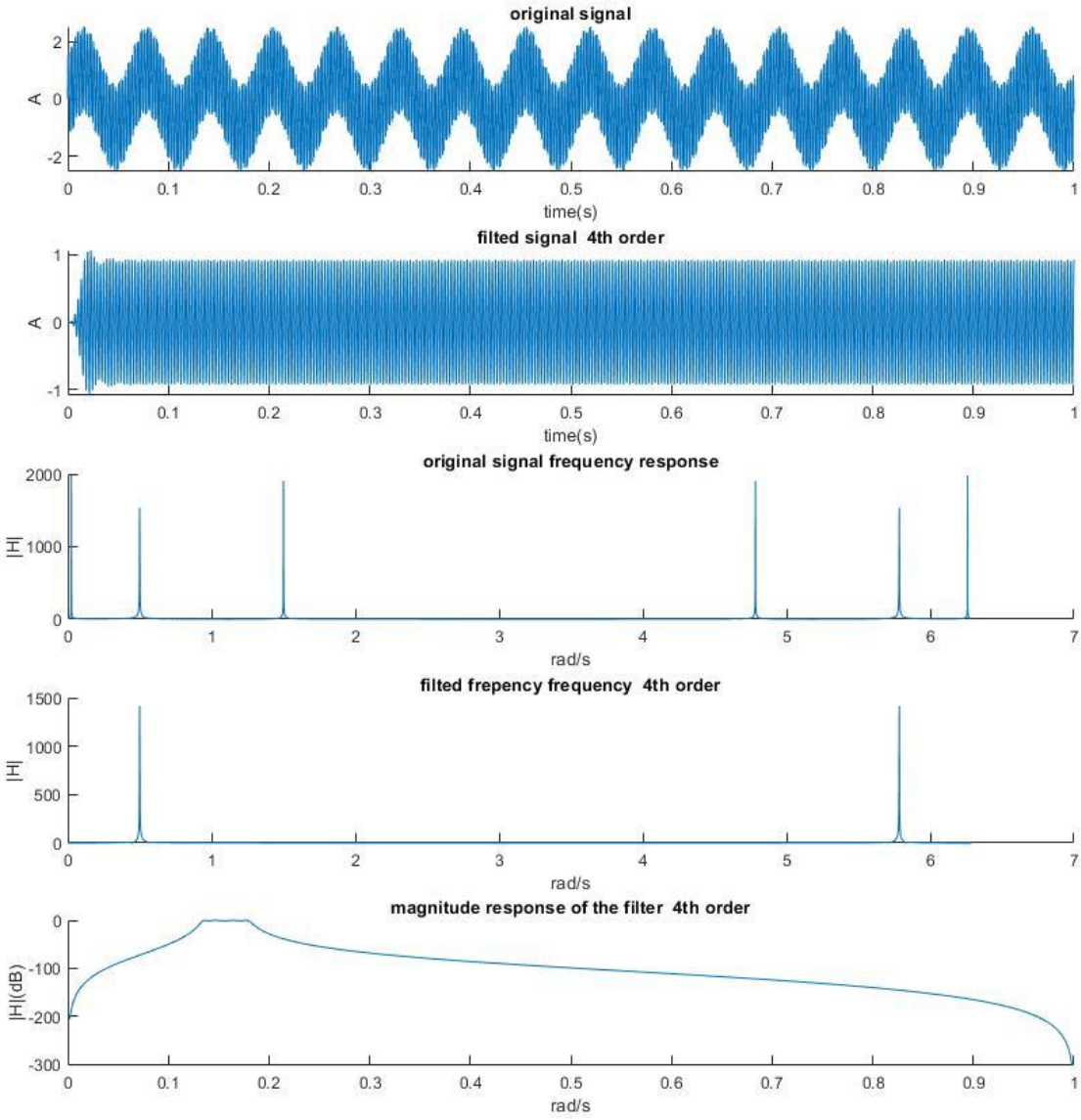
Published with MATLAB® R2020a

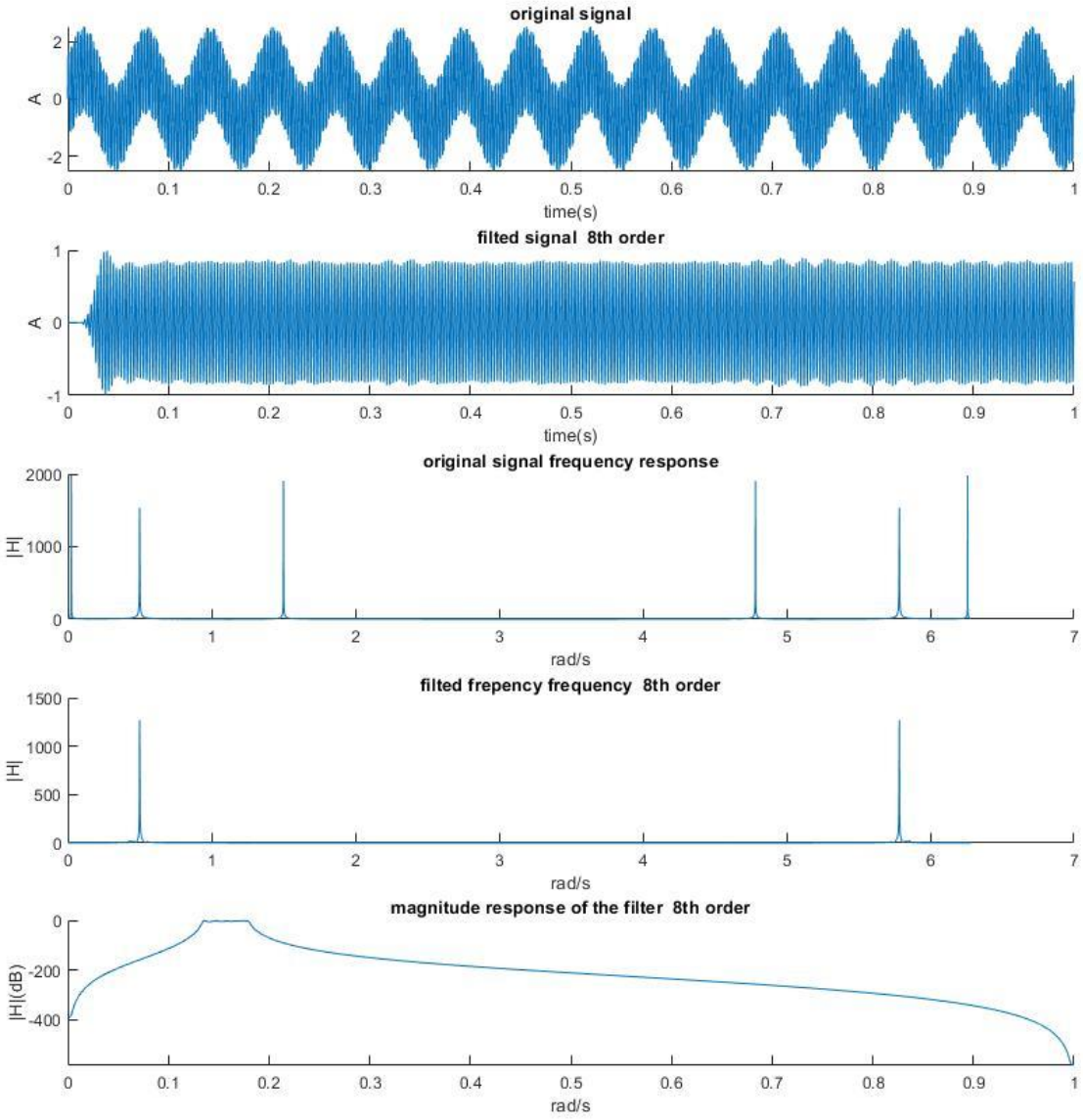
b) Design a digital fourth and eighth-order IIR Chebyshev type I filters. Plot the input signal along with the magnitude response of the filter and its output signal for each filter. (consider $Rp = 1\ dB$).

```
%JUNPENG GAI
%40009896
fs=4000;
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = cheby1(4,1, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
```

```
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = cheby1(8, 1,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
```

```
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```


original signal
A
time(s)
filted signal 4th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 4th order
|H|
rad/s
magnitude response of the filter 4th order
|H|(dB)
rad/s



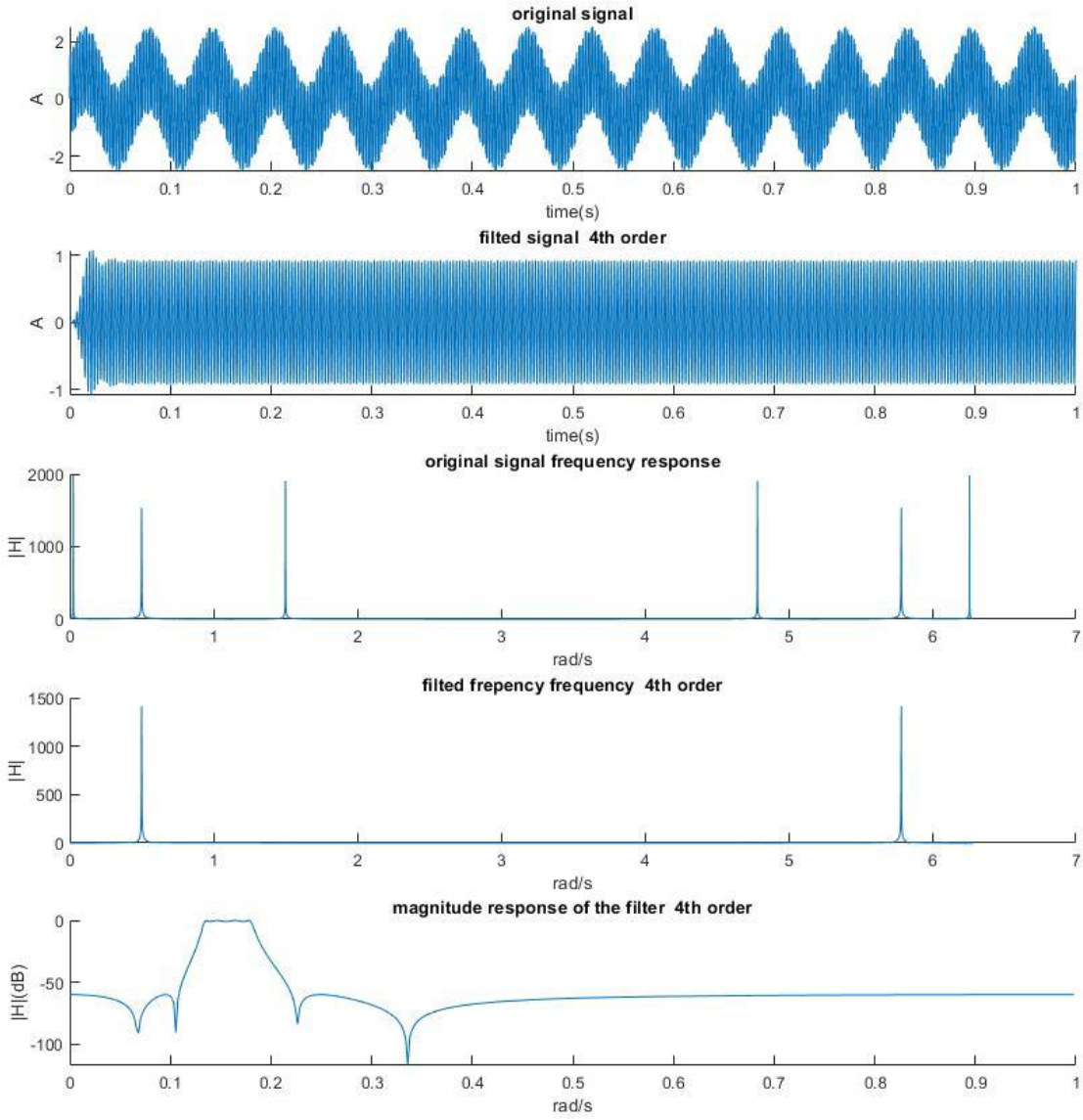
c) Design a digital fourth and eighth-order IIR Elliptic filters. Plot the input signal along with the magnitude response of the filter and its output signal for each filter. (consider $Rp = 1\ dB$, $Rs = 60\ dB$

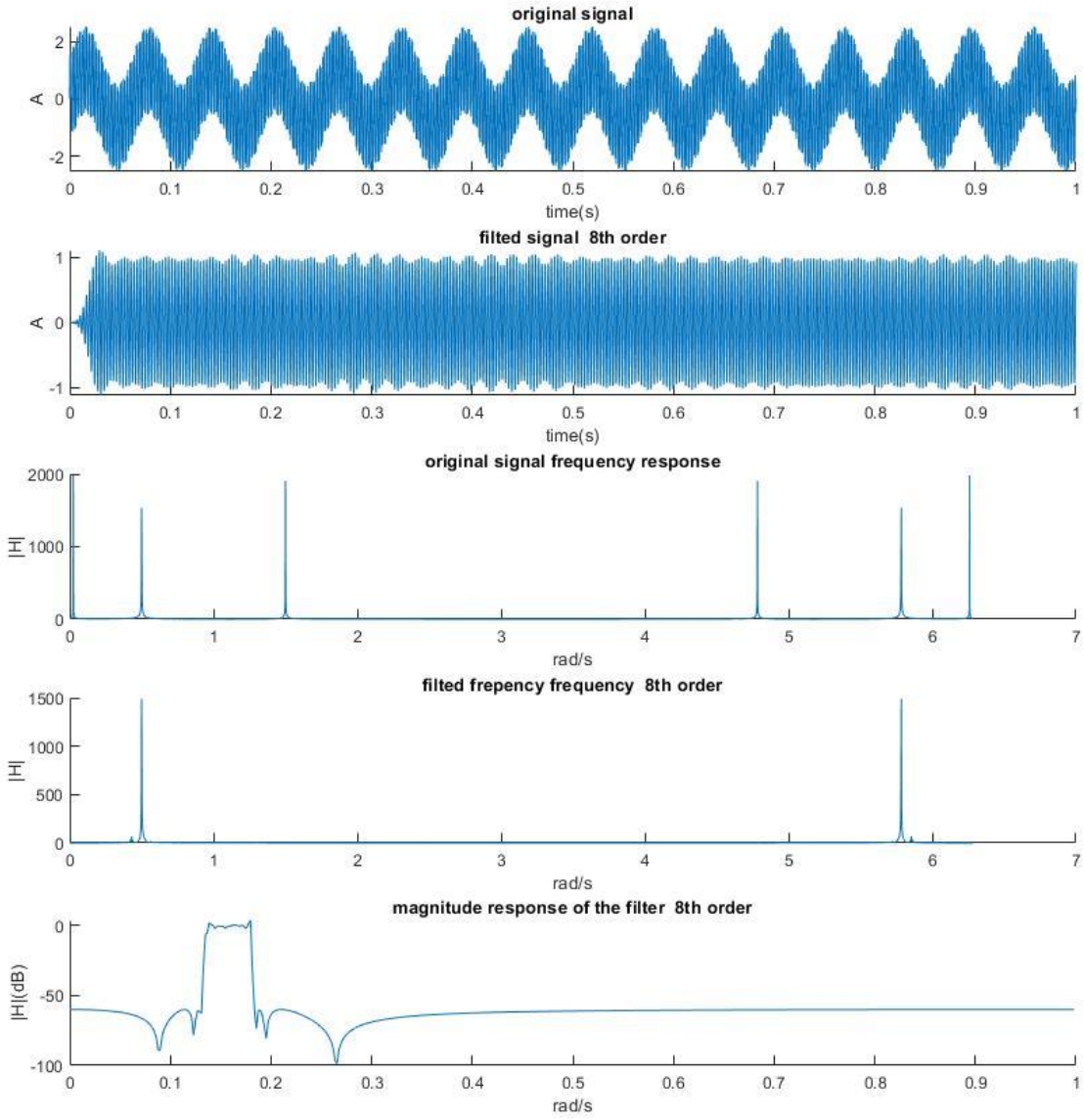
```
%JUNPENG GAI
```

```
%40009896
fs=4000;
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = ellip(4,1, 60,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
```

```
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = ellip(8, 1,60,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
```

```
hold off
subplot(5,1,2)
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```


original signal
A
time(s)
filted signal 4th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 4th order
|H|
rad/s
magnitude response of the filter 4th order
|H|(dB)
rad/s



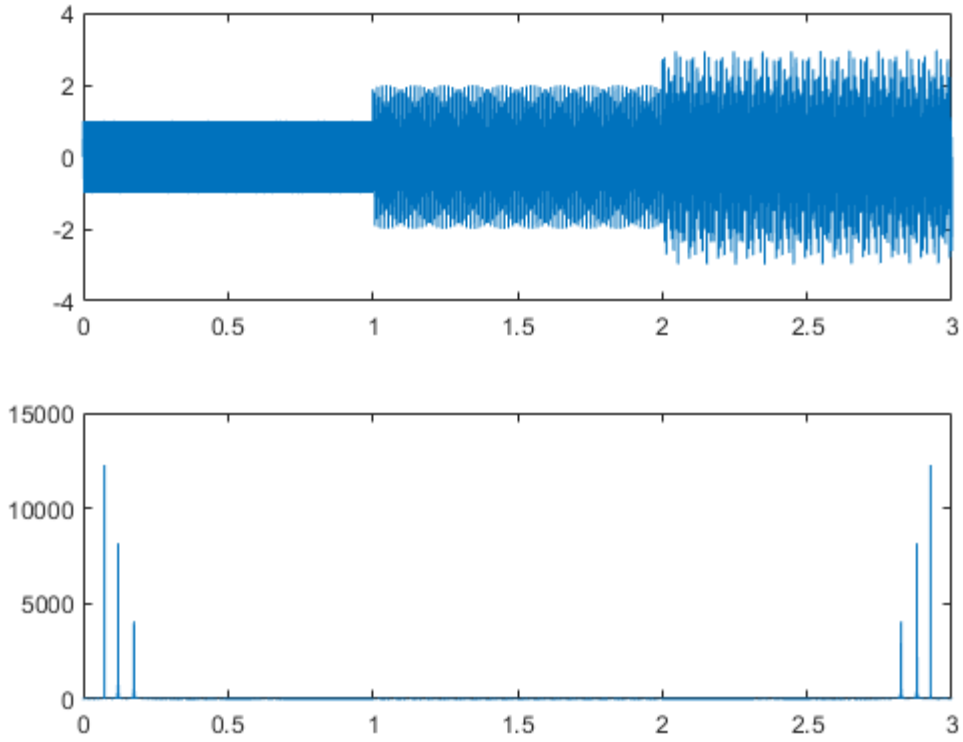
3.2 Problem 2

Problem 2. Part A: Create a signal with three seconds duration. The signal contains three tones of equal amplitude, as following a) Contain a 200 Hz tone for 0 < t < 3. b) Contain a 330 Hz tone for 1

< t < 3. c) Contain a 480 Hz tone for 2 < t < 3. Note: The signal should have a sampling frequency equal to 8192 Hz. View the signal before proceeding (in time domain and in frequency domain), and listen to the signal using soundsc command.

3.2.1 Part a

```
%junpeng gai
%40009896
s=3;
fs=8192;
t1=linspace(0,3,8192*3);
t2=horzcat(zeros(1,8192),linspace(1,3,8192*2));
t3=horzcat(zeros(1,8192*2),linspace(2,3,8192));



x1=sin(400*pi*t1);
x2=sin(660*pi*t2);
x3=sin(960*pi*t3);
x=x1+x2+x3;

X=abs(fft(x));
subplot(2,1,1)
plot(t1,x);
subplot(2,1,2)
plot(t1,X);
```



Published with MATLAB® R2020a

Part B: Design a filter to remove the 330 Hz component from the main signal.

a) *Plot the designed response (magnitude and phase) of the filter.*

It's shown in the following figure

b) *Filter the main signal with the designed filter.*

It's shown in the following figure

c) *Compare signals and spectra (in time domain and in frequency domain) before and after filtering, and listen to the filtered signal using soundsc command.*

It's shown in the following figure.

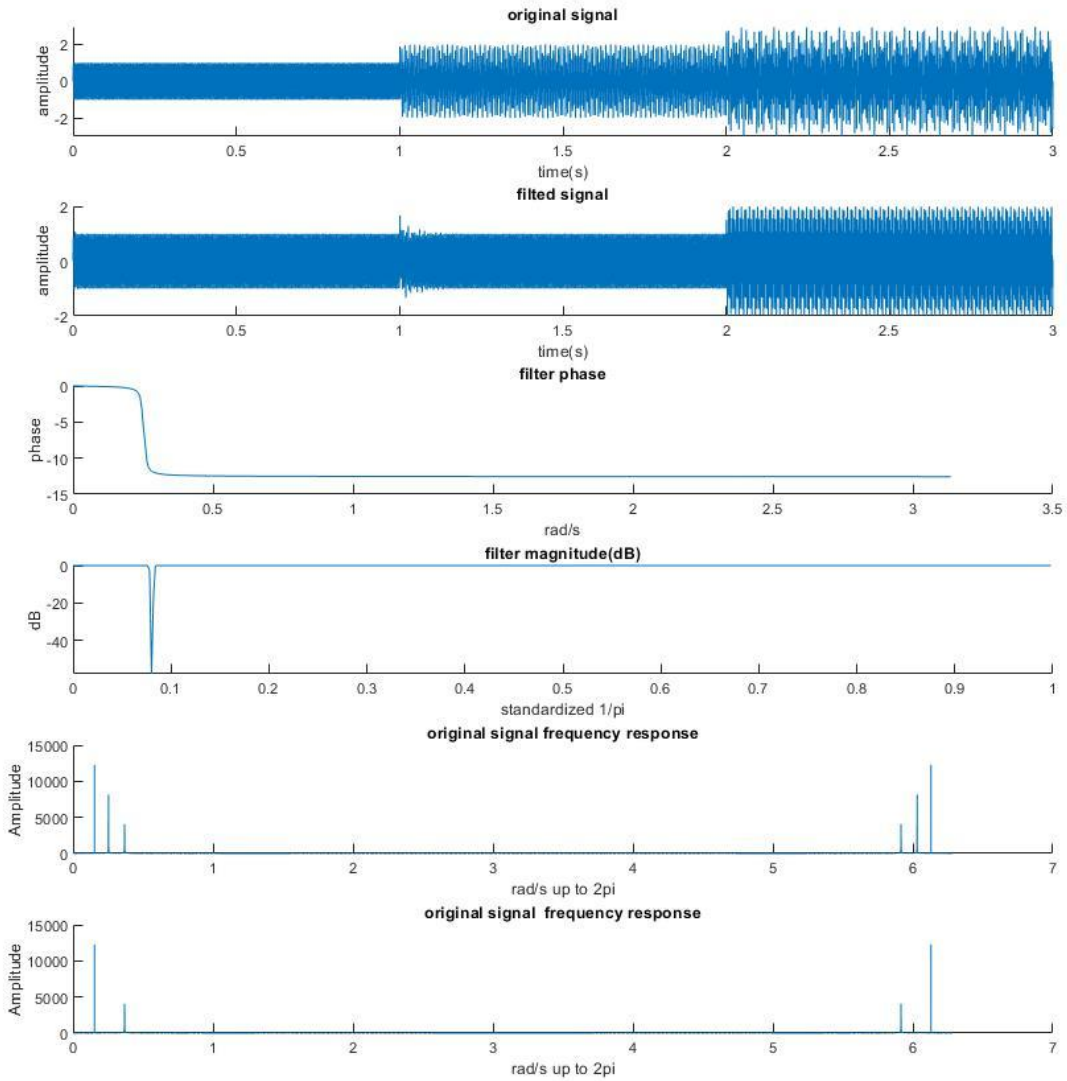
We listen to it and the quality has been improved.

```
%junpeng gai
%40009896
s=3;
fs=8192;
t1=linspace(0,3,8192*3);
t2=horzcat(zeros(1,8192),linspace(1,3,8192*2));
t3=horzcat(zeros(1,8192*2),linspace(2,3,8192));



x1=sin(400*pi*t1);
x2=sin(660*pi*t2);
x3=sin(960*pi*t3);
x=x1+x2+x3;
X=abs(fft(x));
[b, a] = butter(4, [320, 340]/(8192/2), 'stop');
[h,w]=freqz(b,a);
s1= filter(b, a, x);
S1=abs(fft(s1));
subplot(6,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('amplitude')

plot(t1,x);
subplot(6,1,2)
hold on
title('filted signal')
```

```
xlabel('time(s)')
ylabel('amplitude')

plot(t1,s1);
hold off

subplot(6,1,3)
hold on
title('filter phase')
xlabel('rad/s')
ylabel('phase')

plot(w, unwrap(angle(h)));
hold off

subplot(6,1,4)
hold on
title('filter magnitude(dB)')
xlabel('standardized 1/pi')
ylabel('dB')

plot(w/pi, 20*log10(abs(h)));
hold off

subplot(6,1,5)
hold on
title('original signal frequency response')
xlabel('rad/s up to 2pi')
ylabel('Amplitude')

plot(t1/3*2*pi,X);
hold off
```

```
subplot(6,1,6)
hold on
title('original signal  frequency response')
xlabel('rad/s up to 2pi')
ylabel('Amplitude')

plot(t1/3*2*pi,S1);
hold off
```

.



3.2 Problem 3

Problem 3. Building Simulink model to read an audio file and manipulate the sound by adding upsamping or downsampling block. a) Download the audio file "ELEC_364_lab_5_Audio_S.wav" from course directory /groups/e/elec364_1 (see Laboratory Guidelines). b) Open new Simulink model window, and import the audio file "ELEC_364_lab_5_Audio_S.wav". Set Audio output data type to double. c) Add To Audio Device block and connect the output of the audio block to the input of the To Audio Device block. Click run button and listen

After the down sample by 8, I noticed that the sound quality is lower than the original sound.

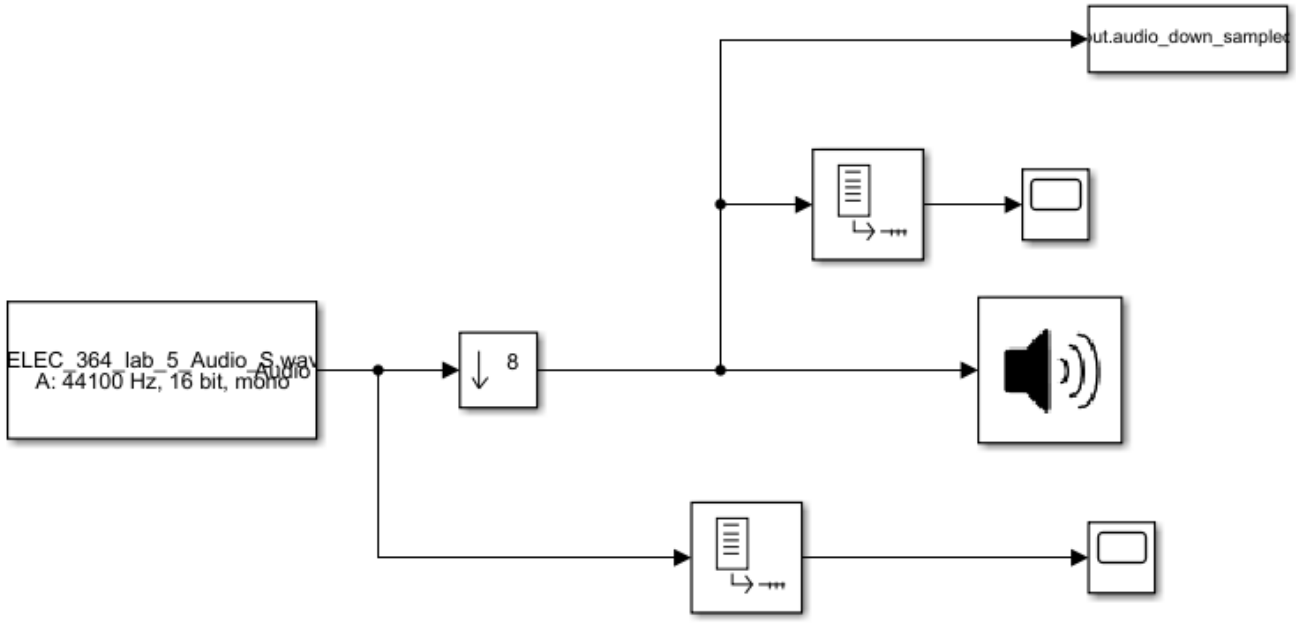
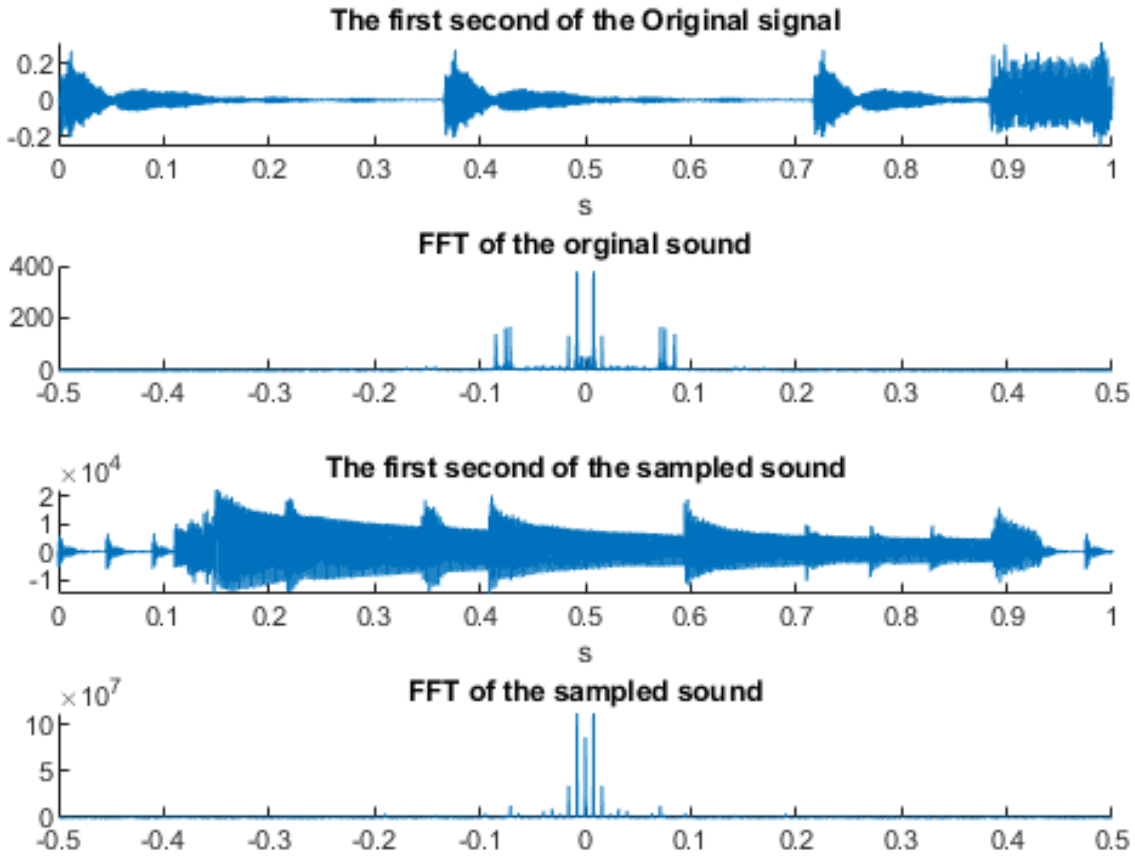


Figure 16 down sample simulation

3.2.1.1 Down sample 8

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
%[y,Fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len*N);
y1=y(1:len);
x1=x(1:len);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);

X1=fft(x1);
X1S=X1;
X1S(1:len/2)=X1(len/2+1:len);
X1S(len/2+1:len)=X1(1:len/2);

subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
```

```
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
plot(t,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t-0.5,abs(X1S))
hold off
```

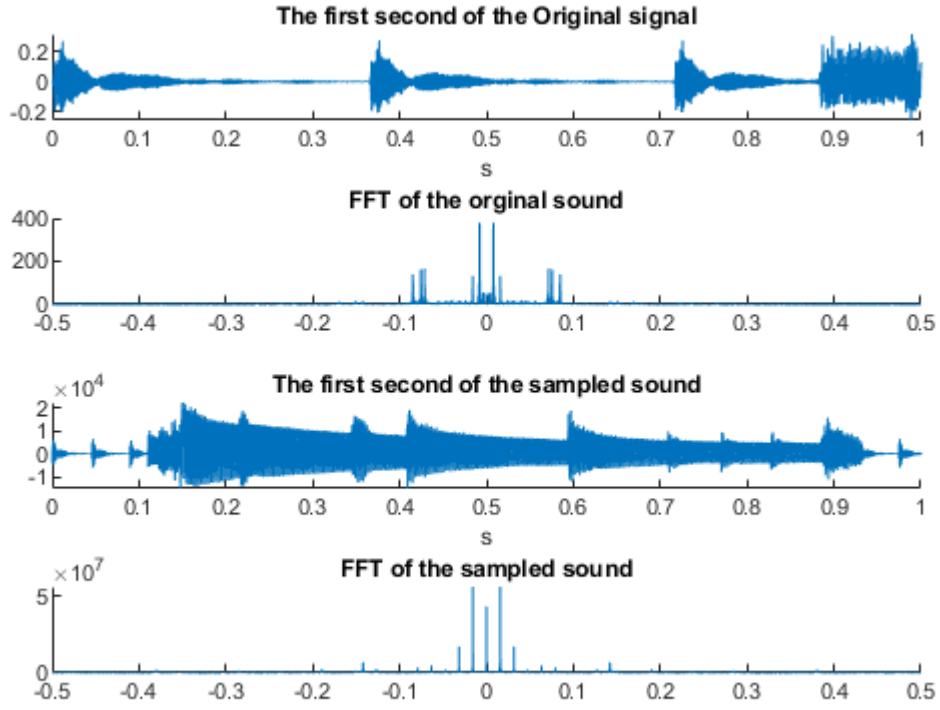


Published with MATLAB® R2020a

3.2.1.2 Down sample 16

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
len1=len/2;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
t1=linspace(0,N,len1);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);

X1=fft(x1);
X1S=X1;
X1S(1:len1/2)=X1(len1/2+1:len1);
X1S(len1/2+1:len1)=X1(1:len1/2);

subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
```

```
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
plot(t1,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t1-0.5,abs(X1S))
hold off
```



Published with MATLAB® R2020a

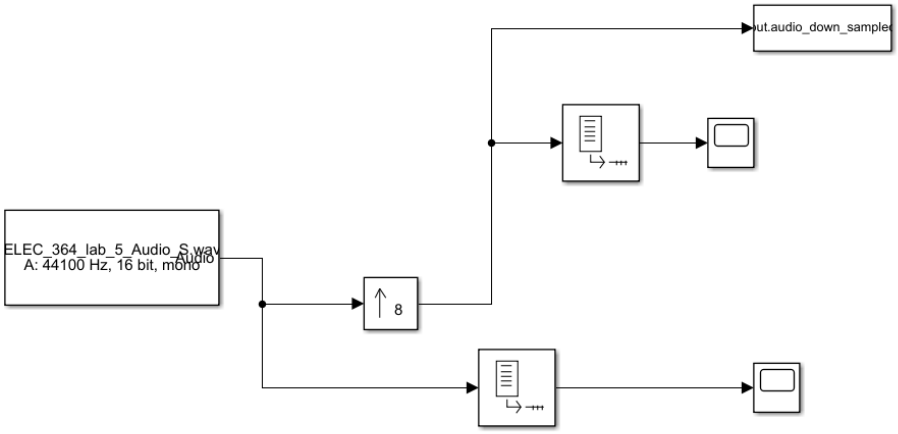
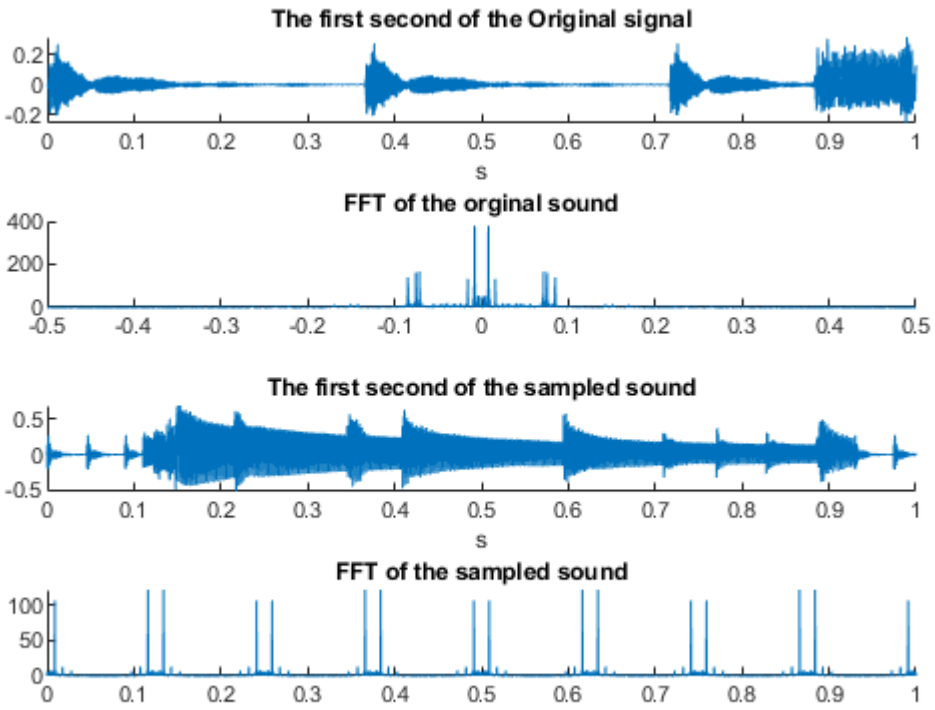


Figure 17 up sample simulation

3.2.2.1 Up sample 8

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
len1=44100*8*352800/44100;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1,44100);


t1=linspace(0,N,length(x1));
```

```
subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
plot(t1,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t,abs(X1))
hold off
```

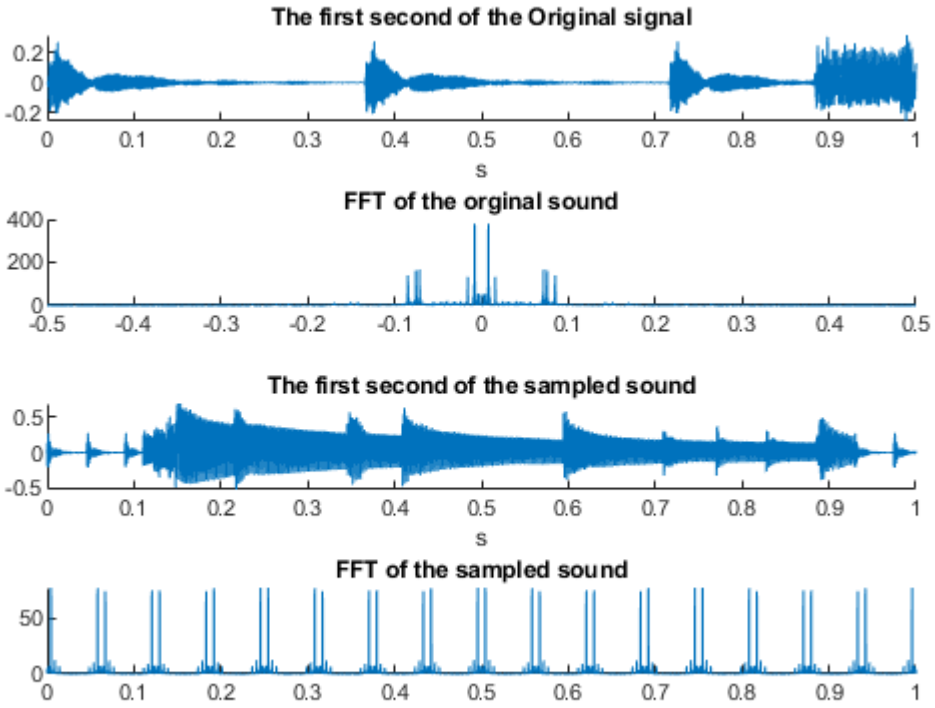


Published with MATLAB® R2020a

3.2.2.2 Up sample 16

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
len1=44100*16*352800/44100;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
```

```
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1,44100);


t1=linspace(0,N,length(x1));


subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
plot(t1,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t,abs(X1))
hold off
```



Published with MATLAB® R2020a

4 Conclusion

By doing the lab section, we get a deeper understanding of the signal process in the MATLAB and we now have a better understanding of signal process and how to use Simulink to do the signal process simulation.

5 Appendix

5.1 Prelab question 1

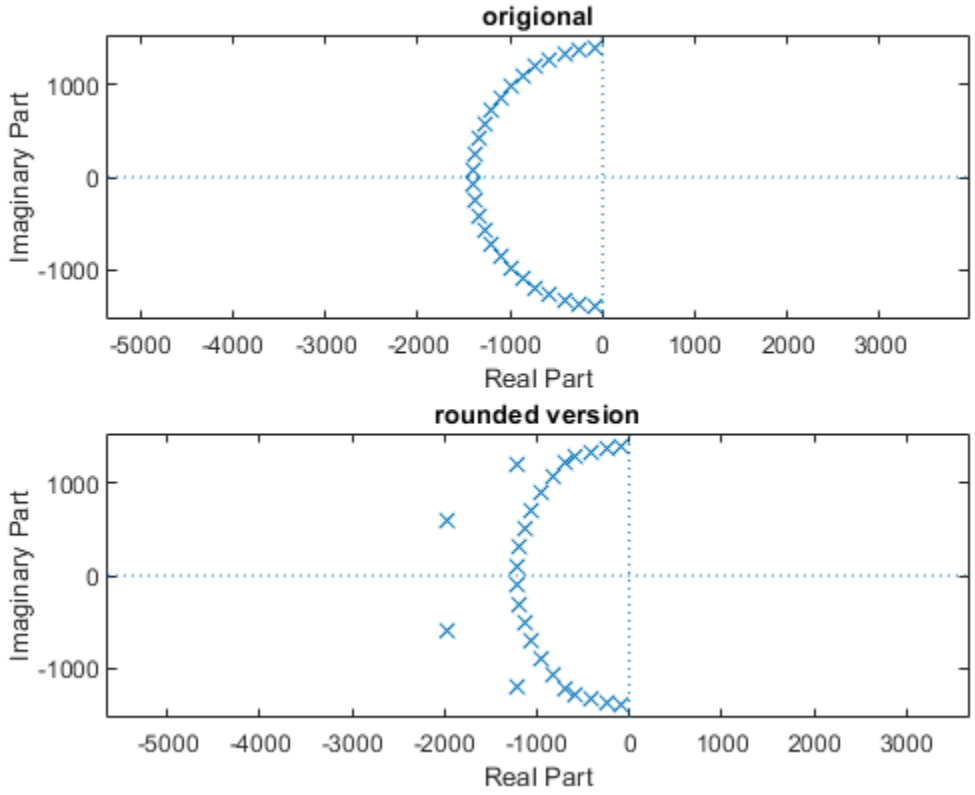
```
%JUNPENG GAI
%40009896
[y,fs]=audioread('lab_5_Audio_1.wav');
sound(y,fs)
%sound(0.25*y,fs)
%sound(4*y,fs)
%sound(y, fs/2)
%sound(y, fs*2)
```

Published with MATLAB® R2020a

5.2 Prelab question 2

```
%junpeng gai
%40009896
[b, a] = butter (26, 1400,'s');
b1=round(b);
a1=round(a);


subplot(2,1,1)
hold on
title('origional')
zplane(b,a)
hold off


subplot(2,1,2)
```

```
hold on
title('rounded version')
zplane(b1,a1)
hold off
```



Published with MATLAB® R2020a

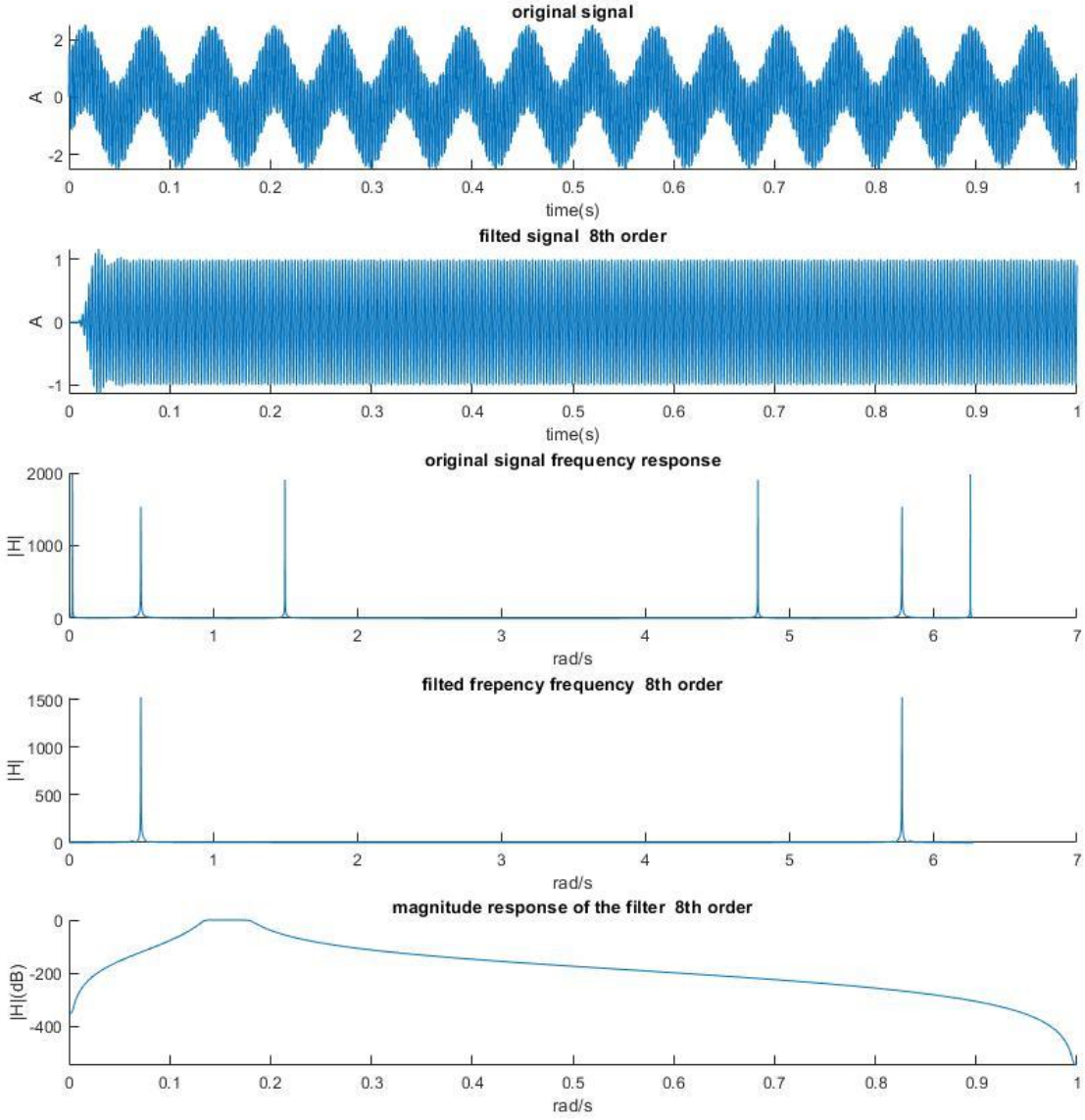
5.3 Prelab question 3

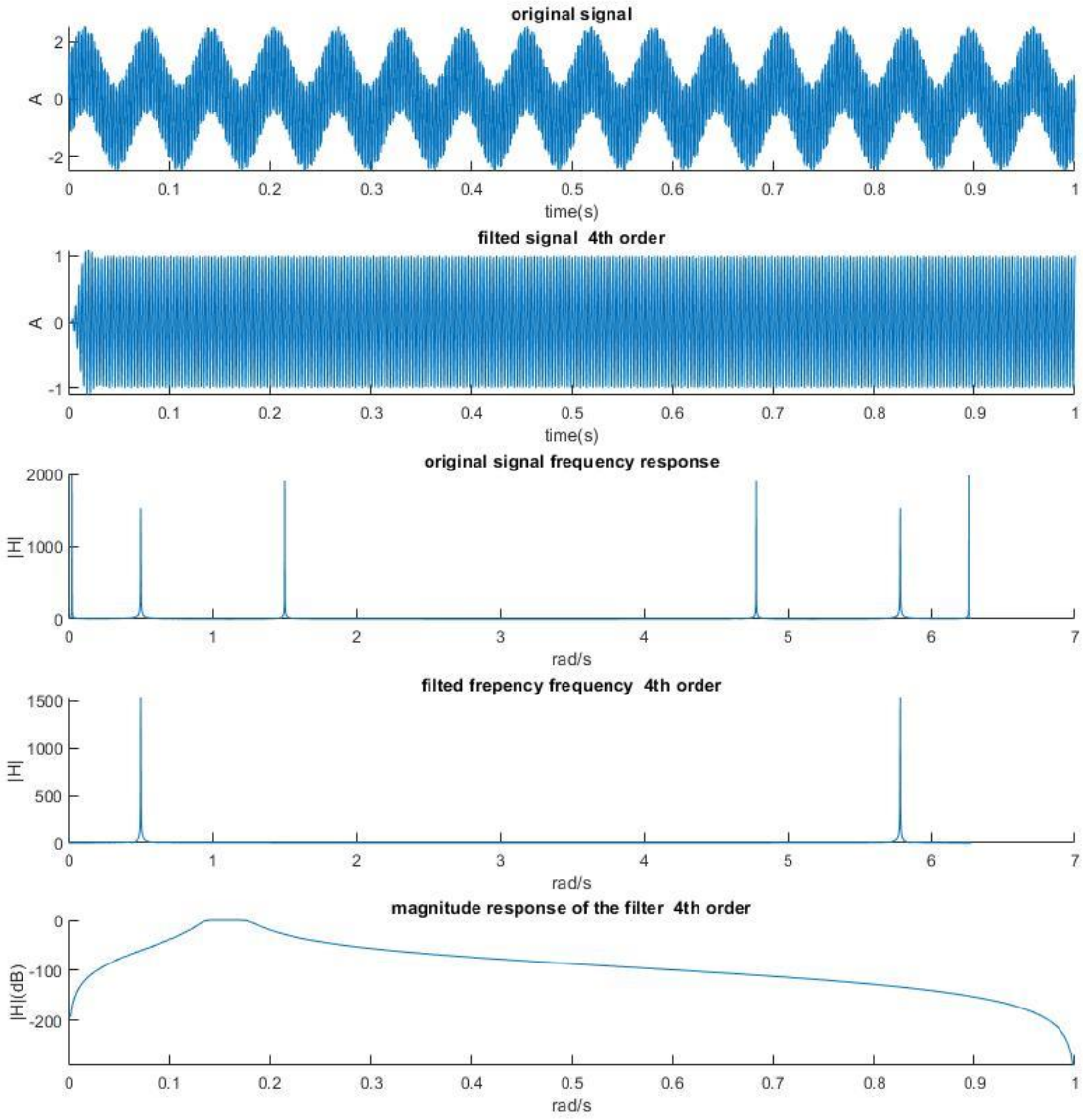
5.4 Problem 1

```
%JUNPENG GAI
%40009896
fs=4000;
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = butter(4, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
```

```
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = butter(8, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
```

```
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```


original signal
A
time(s)
filted signal 8th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 8th order
|H|
rad/s
magnitude response of the filter 8th order
|H|(dB)
rad/s



Published with MATLAB® R2020a

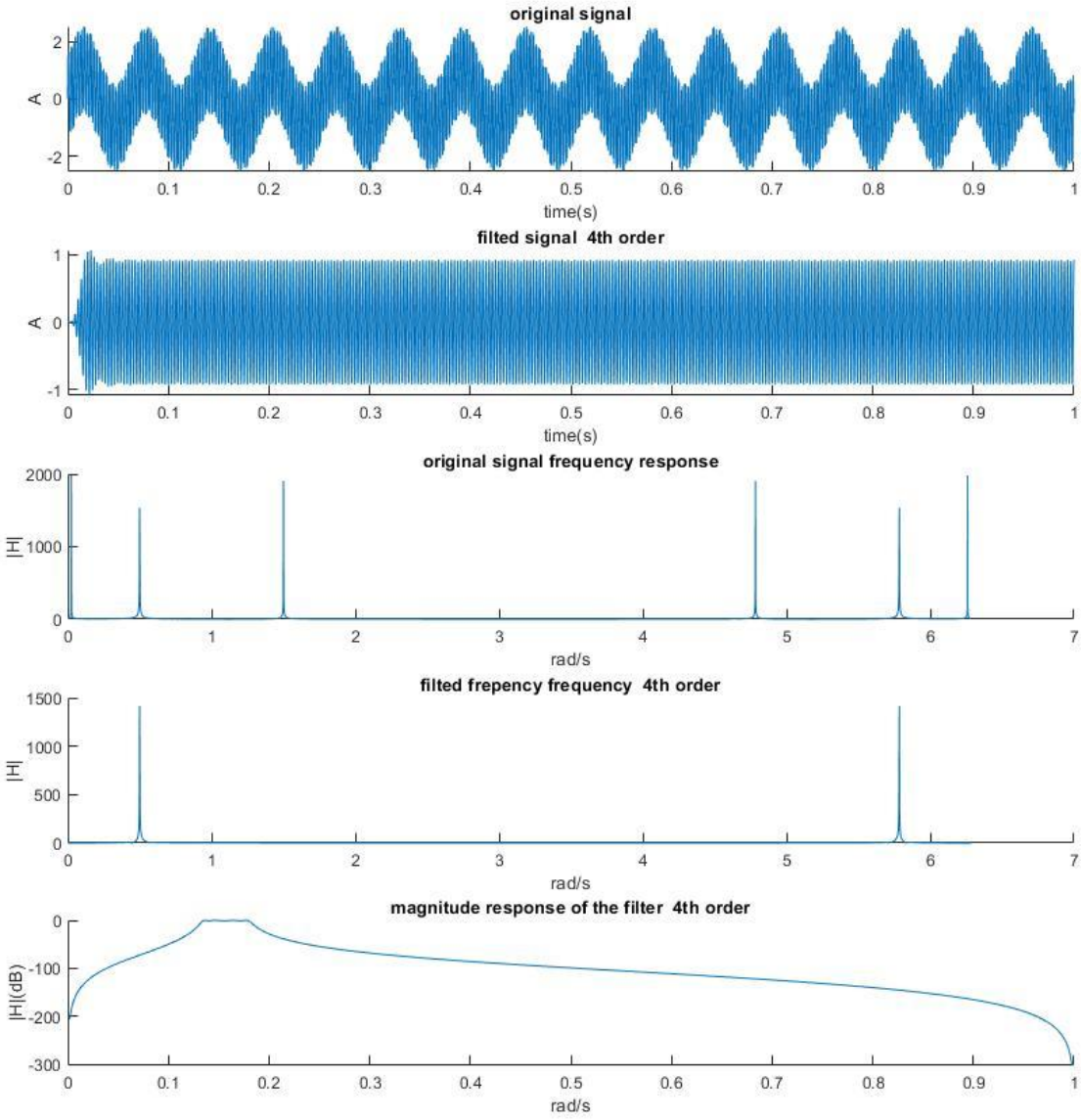
5.5 Problem 1

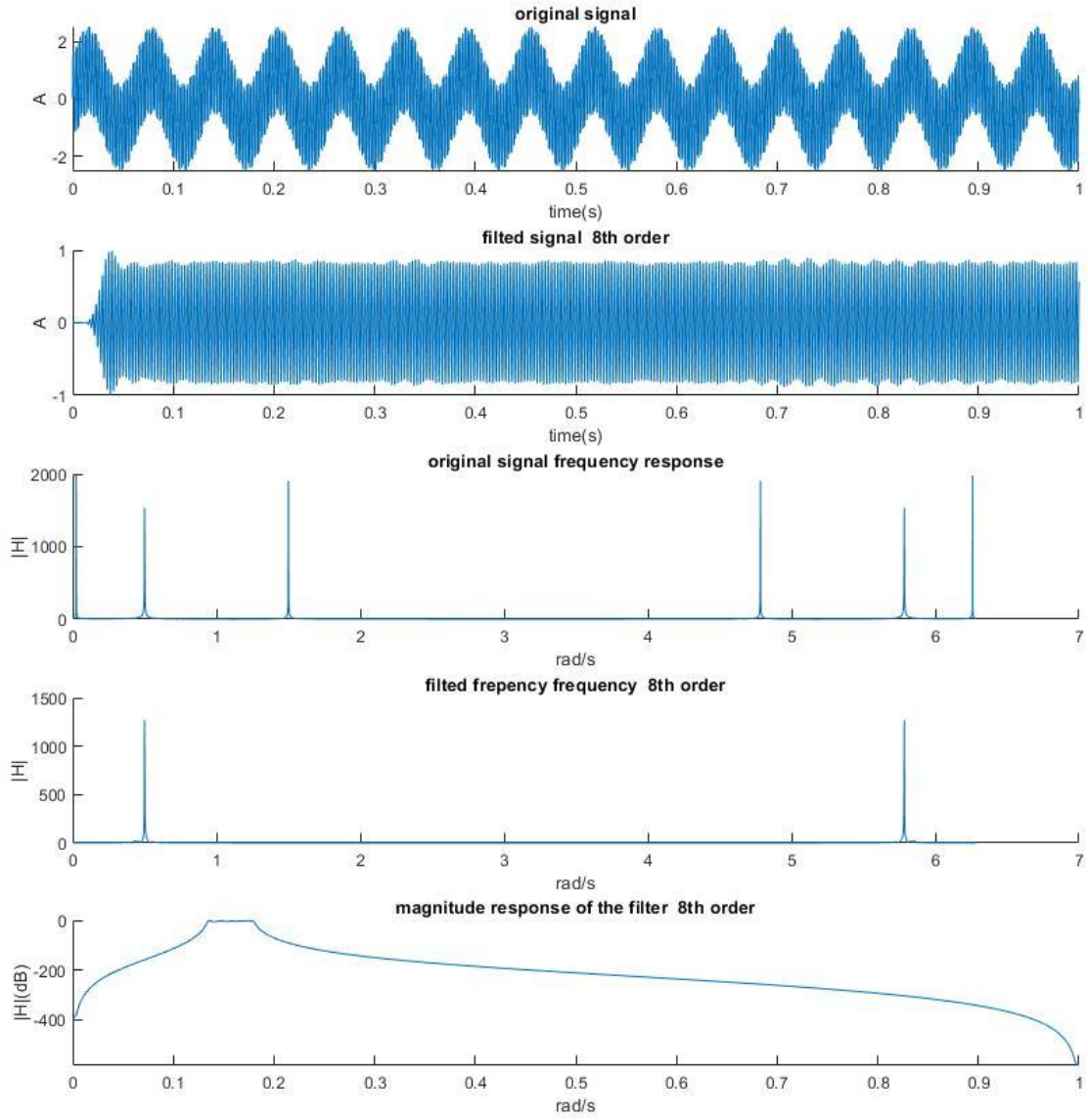
```
%JUNPENG GAI

%40009896
```

```
fs=4000;
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = cheby1(4,1, [0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
```

```
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = cheby1(8, 1,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
```

```
subplot(5,1,2)
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```


original signal
A
time(s)
filted signal 4th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 4th order
|H|
rad/s
magnitude response of the filter 4th order
|H|(dB)
rad/s



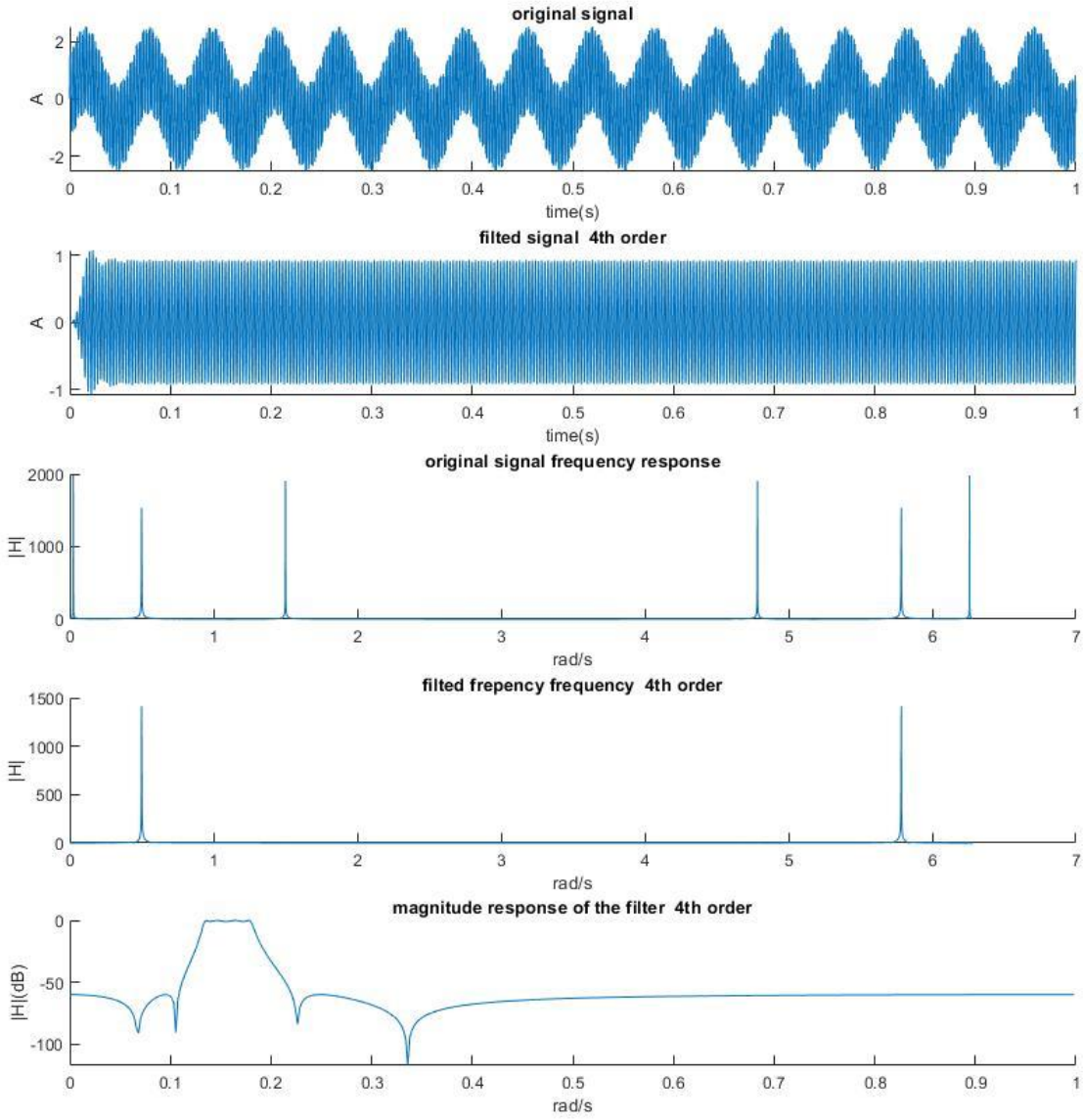
5.6 Problem 1

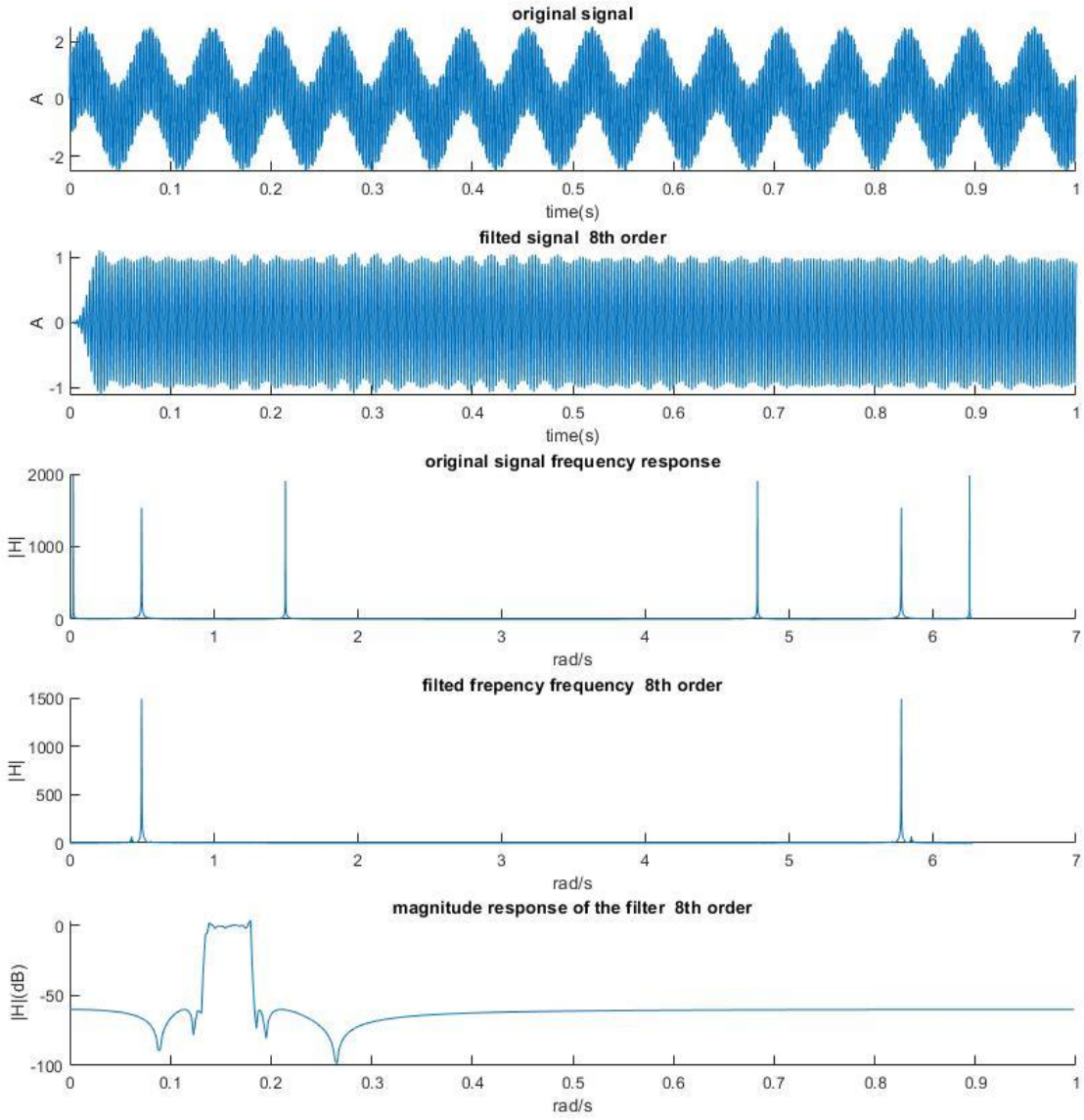
```
%JUNPENG GAI

%40009896

fs=4000;
```

```
%6000/2pi=900+,
%so the sampling frequenncy should be >900*2 hz;
%6000    954Hz
%2000    318Hz
%100     16Hz
%so 4000hz is 1;
%passband is between (1800hz,2200hz) where 4000hz is
%1 after normalization.
t =linspace(0,1,4000*1);
x=sin(100*t)+sin(2000*t)+sin(6000*t);
[b, a] = ellip(4,1, 60,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
hold on
title('filted signal  4th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
```

```
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  4th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  4th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off

% 8th order
[b, a] = ellip(8, 1,60,[0.067 0.09]/0.5, 'bandpass');
[h,w]=freqz(b,a);
X=abs(fft(x));
x1= filter(b, a, x);
X1=abs(fft(x1));
figure
subplot(5,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('A')
plot(t,x);
hold off
subplot(5,1,2)
```

```
hold on
title('filted signal  8th order')
xlabel('time(s)')
ylabel('A')
plot(t,x1);
hold off
subplot(5,1,3)
hold on
title('original signal frequency response')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X);
hold off
subplot(5,1,4)
hold on
title('filted frepency frequency  8th order')
xlabel('rad/s')
ylabel('|H|')
plot(t*2*pi,X1);
hold off
subplot(5,1,5)
hold on
title('magnitude response of the filter  8th order')
xlabel('rad/s')
ylabel('|H|(dB)')
plot(w/pi,20*log10(abs(h)));
hold off
```

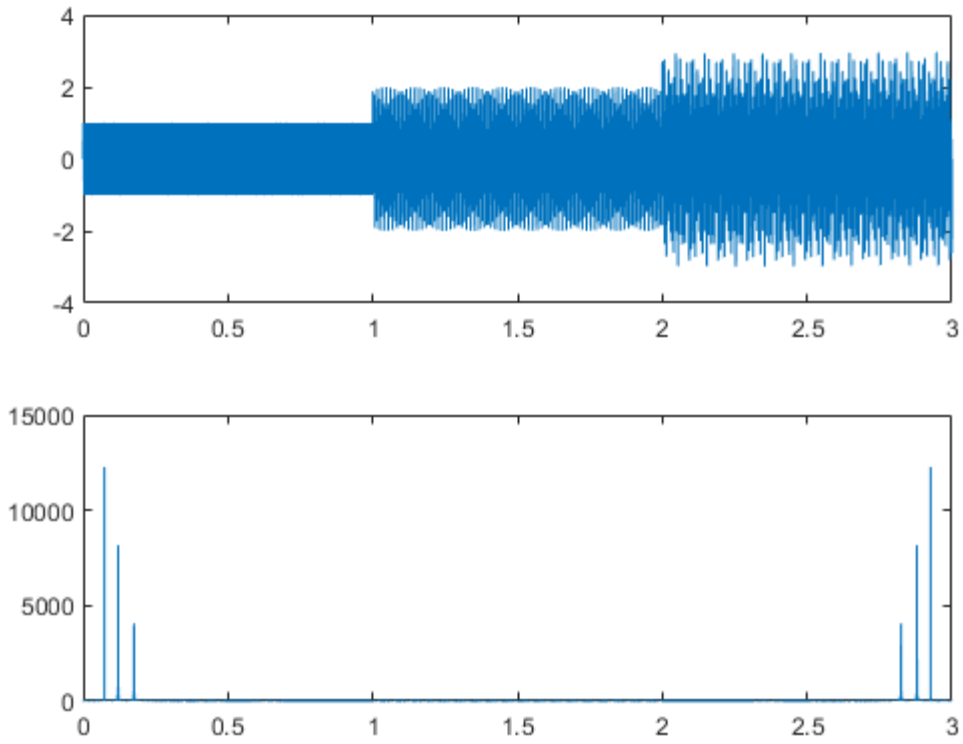

original signal
A
time(s)
filted signal 4th order
A
time(s)
original signal frequency response
|H|
rad/s
filted frepency frequency 4th order
|H|
rad/s
magnitude response of the filter 4th order
|H|(dB)
rad/s



5.7 Problem 2.a

```
%junpeng gai
```

```
%40009896
s=3;
fs=8192;
t1=linspace(0,3,8192*3);
t2=horzcat(zeros(1,8192),linspace(1,3,8192*2));
t3=horzcat(zeros(1,8192*2),linspace(2,3,8192));



x1=sin(400*pi*t1);
x2=sin(660*pi*t2);
x3=sin(960*pi*t3);
x=x1+x2+x3;

X=abs(fft(x));
subplot(2,1,1)
plot(t1,x);
subplot(2,1,2)
plot(t1,X);
```



Published with MATLAB® R2020a

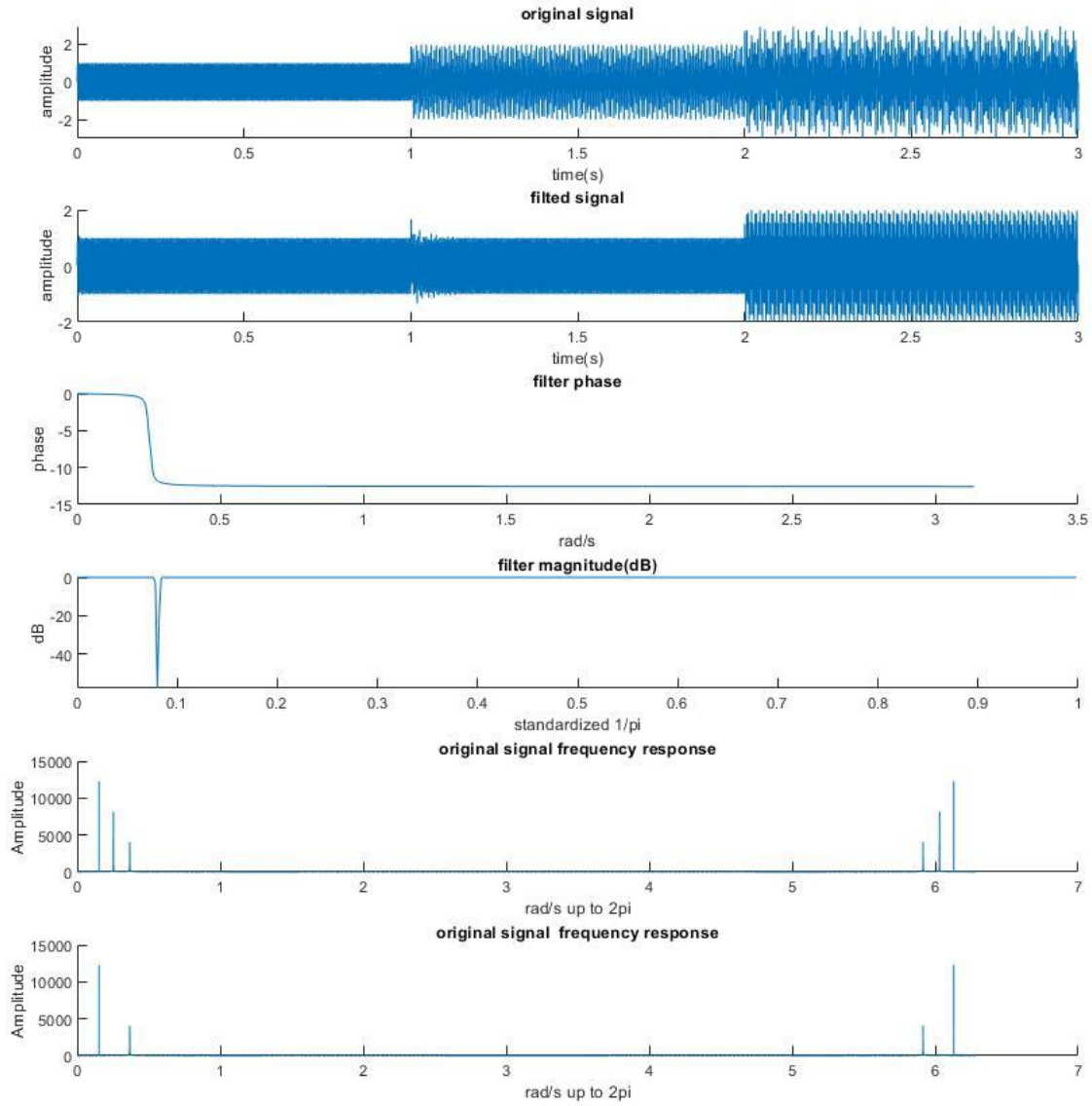
5.8 Problem 2.b

```
%junpeng gai
%40009896
s=3;
fs=8192;
```

```
t1=linspace(0,3,8192*3);
t2=horzcat(zeros(1,8192),linspace(1,3,8192*2));
t3=horzcat(zeros(1,8192*2),linspace(2,3,8192));



x1=sin(400*pi*t1);
x2=sin(660*pi*t2);
x3=sin(960*pi*t3);
x=x1+x2+x3;
X=abs(fft(x));
[b, a] = butter(4, [320, 340]/(8192/2), 'stop');
[h,w]=freqz(b,a);
s1= filter(b, a, x);
S1=abs(fft(s1));
subplot(6,1,1)
hold on
title('original signal')
xlabel('time(s)')
ylabel('amplitude')

plot(t1,x);
subplot(6,1,2)
hold on
title('filted signal')
xlabel('time(s)')
ylabel('amplitude')

plot(t1,s1);
hold off

subplot(6,1,3)
hold on
title('filter phase')
```

```
xlabel('rad/s')
ylabel('phase')

plot(w, unwrap(angle(h)));
hold off

subplot(6,1,4)
hold on
title('filter magnitude(dB)')
xlabel('standardized 1/pi')
ylabel('dB')

plot(w/pi, 20*log10(abs(h)));
hold off

subplot(6,1,5)
hold on
title('original signal frequency response')
xlabel('rad/s up to 2pi')
ylabel('Amplitude')

plot(t1/3*2*pi,X);
hold off

subplot(6,1,6)
hold on
title('original signal  frequency response')
xlabel('rad/s up to 2pi')
ylabel('Amplitude')

plot(t1/3*2*pi,S1);
hold off
```

.

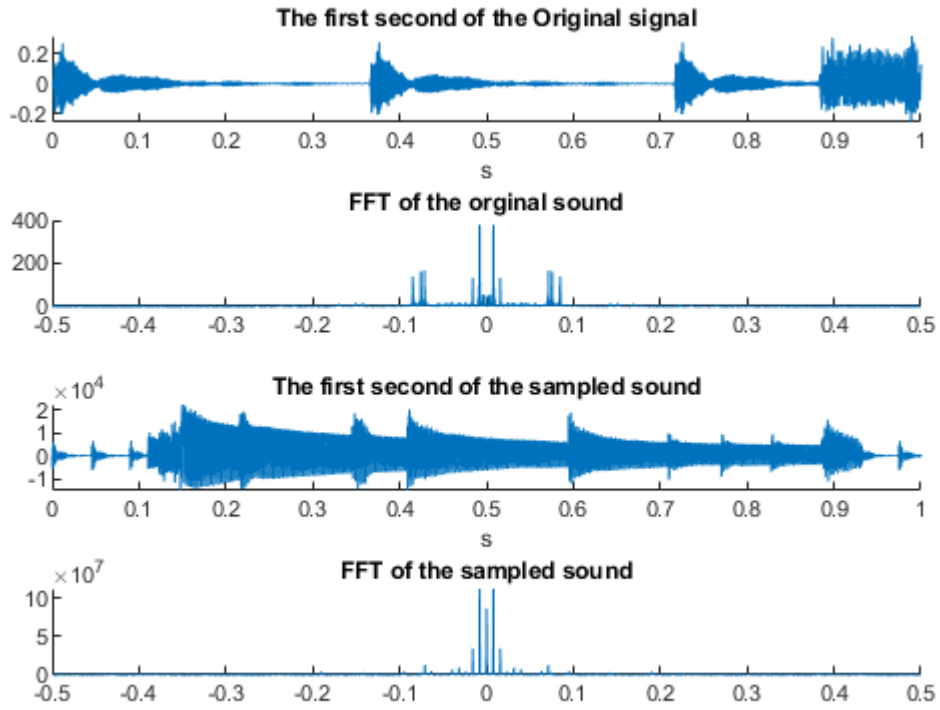


5.9 Problem 3-down sample 8

```
%Junpeng Gai
```

```
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
%[y,Fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len*N);
y1=y(1:len);
x1=x(1:len);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1);
X1S=X1;
X1S(1:len/2)=X1(len/2+1:len);
X1S(len/2+1:len)=X1(1:len/2);


subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
```

```
plot(t,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t-0.5,abs(X1S))
hold off
```



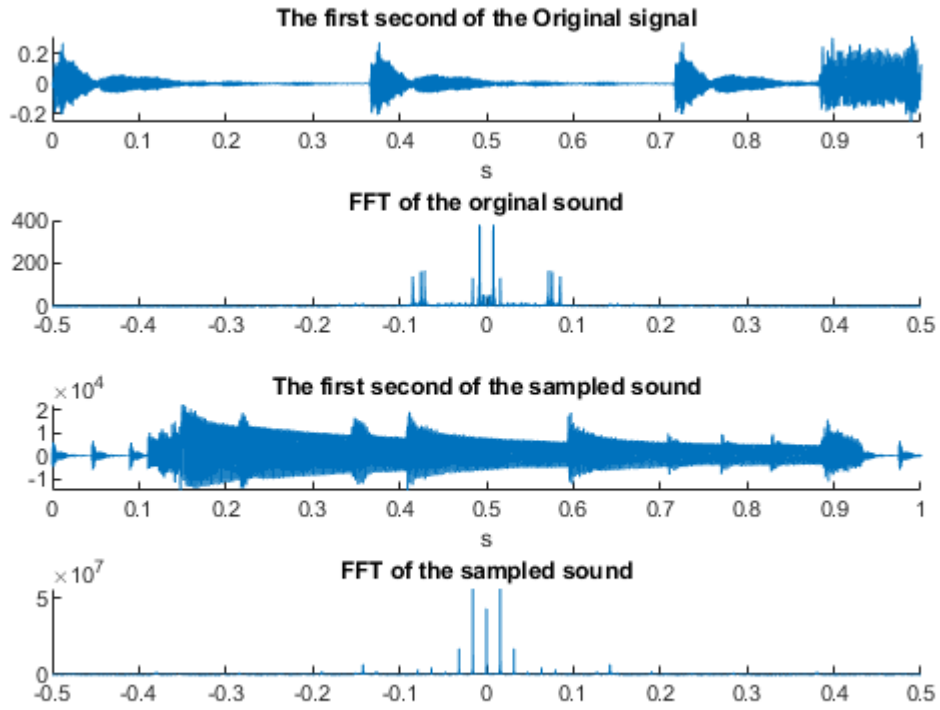
Published with MATLAB® R2020a

5.10 Problem 3-down sample 16

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
```

```
N=1;
len=44100*N;
len1=len/2;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
t1=linspace(0,N,len1);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1);
X1S=X1;
X1S(1:len1/2)=X1(len1/2+1:len1);
X1S(len1/2+1:len1)=X1(1:len1/2);


subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
```

```
plot(t1,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t1-0.5,abs(X1S))
hold off
```



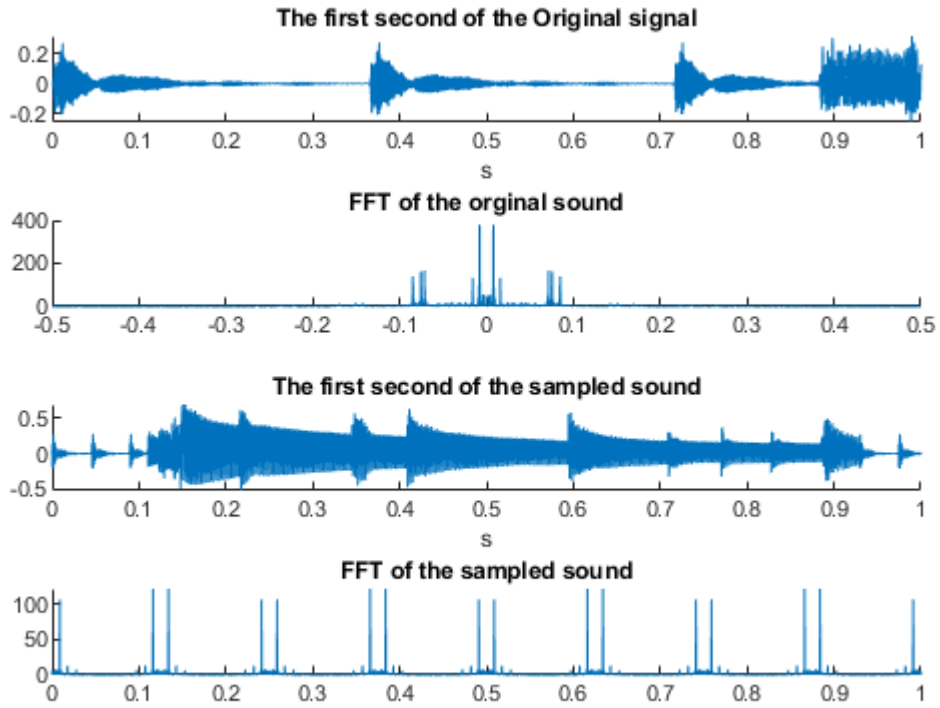
Published with MATLAB® R2020a

5.11 Problem 3-up sample 8

```
%Junpeng Gai
```

```
%40009896
x=out.audio_down_sampled';
N=1;
len=44100*N;
len1=44100*8*352800/44100;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1,44100);


t1=linspace(0,N,length(x1));


subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
```

```
plot(t1,x1)
hold off
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t,abs(X1))
hold off
```



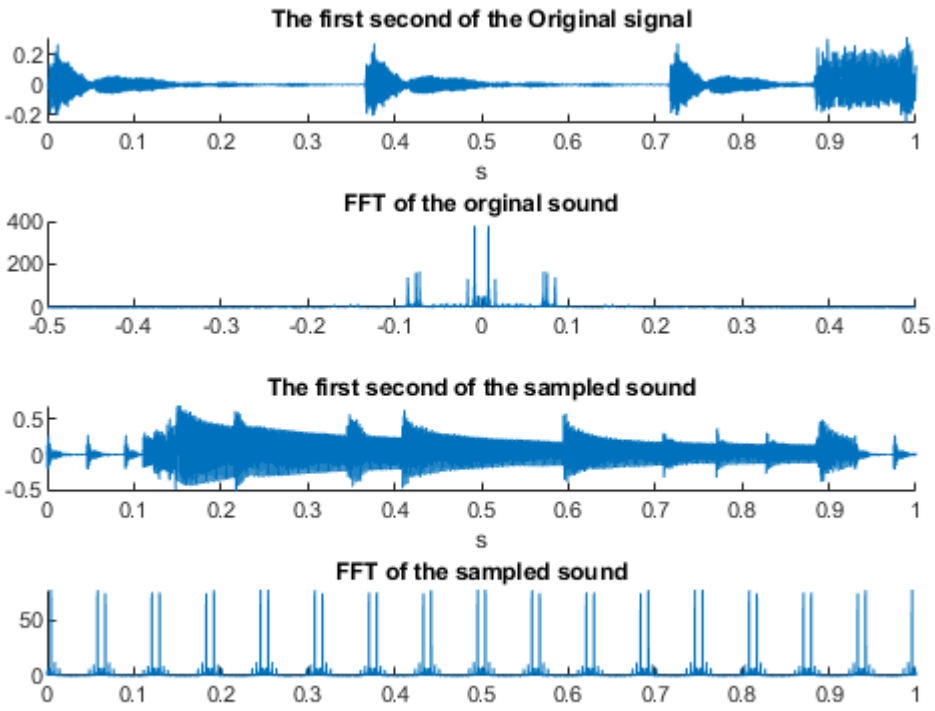
Published with MATLAB® R2020a

5.12 Problem 3-up sample 16

```
%Junpeng Gai
%40009896
x=out.audio_down_sampled';
```

```
N=1;
len=44100*N;
len1=44100*16*352800/44100;
%[y,fs]=audioread('ELEC_364_lab_5_Audio_S.wav')
t=linspace(0,N,len);
y1=y(1:len);
x1=x(1:len1);
Y1=fft(y1);
Y1S=Y1;
Y1S(1:len/2)=Y1(len/2+1:len);
Y1S(len/2+1:len)=Y1(1:len/2);


X1=fft(x1,44100);


t1=linspace(0,N,length(x1));


subplot(4,1,1)
hold on
title('The first second of the Original signal')
xlabel('s')
plot(t,y1)
hold off
subplot(4,1,2)
hold on
title('FFT of the orginal sound')
plot(t-0.5,abs(Y1S))
hold off
subplot(4,1,3)
hold on
title('The first second of the sampled sound')
xlabel('s')
plot(t1,x1)
hold off
```

```
subplot(4,1,4)
hold on
title('FFT of the sampled sound')
plot(t,abs(X1))
hold off
```



Published with MATLAB® R2020a