DSP FINAL PROJECT

MADE BY:

1) AHMAD SAID NOUH (7086)

2) HUSSEIN MOURAD ABDELHAMID (6729)

3) ABDELRAHMAN ELSAYED (3953)

Contents

1) Code:

We divided our work into four files to make the work more readable, clean and organized: gui1.m, main.m, custom_filter.m, and custom_plot.m. The first file is gui1.m which is the file that we run to start the whole program. This file contains the gui code which we will not display here as it is irrelevant and too long. The only part of this file that we will display will be the callback function of our run button which validates all the gui inputs and displays the appropriate message if something is missing or calls the main function from the main.m file if all inputs are correct. The second file is main.m which contains the main function of our code. In this file, we call the custom_plot function from custom_plot.m file to plot the figures in a specific way and call the custom_filter function from custom_filter.m file to create specific filters.

1-a) gui.m

```
% --- Executes on button press in runBtn.
function runBtn_Callback(hObject, eventdata, handles)
isValidInputs = validateInputs(handles);
if isValidInputs
    chkValue = get(handles.fsCheckBox,'Value');
    fs = 0;
    default = 1;
    if chkValue == 1
       fs = str2double(get(handles.fs, 'String'));
       default = 0;
    end
    g1 = str2double(get(handles.g1,'String'));
    g2 = str2double(get(handles.g2,'String'));
    g3 = str2double(get(handles.g3,'String'));
    g4 = str2double(get(handles.g4,'String'));
    g5 = str2double(get(handles.g5,'String'));
    g6 = str2double(get(handles.g6,'String'));
    g7 = str2double(get(handles.g7,'String'));
    g8 = str2double(get(handles.g8,'String'));
    g9 = str2double(get(handles.g9,'String'));

    file_name = get(handles.fileName, 'String');

    filter_type = get(handles.filterType, 'Value');

    gains = [g1, g2, g3, g4, g5, g6, g7, g8, g9];

    disp('valid')

    main(file_name, default, fs, filter_type, gains);


end

function isValid = validateInputs(handles)
fileName = get(handles.fileName, 'String');
```

```
if isempty(fileName)
    errordlg('Must enter file name!!');
    isValid = 0;
    return
end

selectedIndex = get(handles.filterType, 'Value');
if selectedIndex == 1
    errordlg('Must choose filter type!!');
    isValid = 0;
    return
end

chkValue = get(handles.fsCheckBox,'Value');
if chkValue == 1
    fs = get(handles.fs, 'String');
    if isempty(fs)
        errordlg('Must enter output sampling rate!!');
        isValid = 0;
        return
    elseif isnan(str2double(fs))
        errordlg('Output sample rate must be a real number more than 32000
Hz!!');
        isValid = 0;
        return
    elseif str2double(fs) <= 32000
        errordlg('Output sample rate must be a real number more than 32000
Hz!!');
        isValid = 0;
        return
    end
end
g1 = get(handles.g1,'String');
if isempty(g1)
    errordlg('Must enter gain 1!!');
    isValid = 0;
    return
elseif isnan(str2double(g1))
    errordlg('Gain 1 must be a real number!!');
    isValid = 0;
    return
end
%
g2 = get(handles.g2,'String');
if isempty(g2)
    errordlg('Must enter gain 2!!');
    isValid = 0;
    return
elseif isnan(str2double(g2))
    errordlg('Gain 2 must be a real number!!');
    isValid = 0;
    return
end
%
g3 = get(handles.g3,'String');
if isempty(g3)
```

```
    errordlg('Must enter gain 3!!');
    isValid = 0;
    return
elseif isnan(str2double(g3))
    errordlg('Gain 3 must be a real number!!');
    isValid = 0;
    return
end
%
g4 = get(handles.g4,'String');
if isempty(g4)
    errordlg('Must enter gain 4!!');
    isValid = 0;
    return
elseif isnan(str2double(g4))
    errordlg('Gain 4 must be a real number!!');
    isValid = 0;
    return
end
%
g5 = get(handles.g5,'String');
if isempty(g5)
    errordlg('Must enter gain 5!!');
    isValid = 0;
    return
elseif isnan(str2double(g5))
    errordlg('Gain 5 must be a real number!!');
    isValid = 0;
    return
end
%
g6 = get(handles.g6,'String');
if isempty(g6)
    errordlg('Must enter gain 6!!');
    isValid = 0;
    return
elseif isnan(str2double(g6))
    errordlg('Gain 6 must be a real number!!');
    isValid = 0;
    return
end
%
g7 = get(handles.g7,'String');
if isempty(g7)
    errordlg('Must enter gain 7!!');
    isValid = 0;
    return
elseif isnan(str2double(g7))
    errordlg('Gain 7 must be a real number!!');
    isValid = 0;
    return
end
%
g8 = get(handles.g8,'String');
if isempty(g8)
    errordlg('Must enter gain 8!!');
    isValid = 0;
```

```
    return
elseif isnan(str2double(g8))
    errordlg('Gain 8 must be a real number!!');
    isValid = 0;
    return
end
%
g9 = get(handles.g9,'String');
if isempty(g9)
    errordlg('Must enter gain 9!!');
    isValid = 0;
    return
elseif isnan(str2double(g9))
    errordlg('Gain 9 must be a real number!!');
    isValid = 0;
    return
end
isValid = 1;
```

1-b) main.m

```
function main(file_name,default_fs, fs_user, filter_type, gains)
try
    [x,fs_default] = audioread(file_name);
    if default_fs == 1
        fs = fs_default;
    else
        fs = fs_user;
    end
catch ex
    errordlg('No file with such name exists!!');
    return
end
Ns = length(x);
t = 0:1/fs:(Ns-1)/fs;
T = linspace(-fs/2,fs/2,Ns);


wn = 2*170/fs;
[x1_filtered, x1_amp_filtered] = custom_filter(x, gains(1), wn, fs, Ns,
'low', 'First', 1, filter_type);


wn = [340/fs, 620/fs];
[x2_filtered, x2_amp_filtered] = custom_filter(x, gains(2), wn, fs, Ns,
'bandpass', 'Second', 4, filter_type);


wn = [620/fs, 1200/fs];
[x3_filtered, x3_amp_filtered] = custom_filter(x, gains(3), wn, fs, Ns,
'bandpass', 'Third', 7, filter_type);


wn = [1200/fs, 2000/fs];
```

```
[x4_filtered, x4_amp_filtered] = custom_filter(x, gains(4), wn, fs, Ns,
'bandpass', 'Fourth', 10, filter_type);


wn = [2000/fs, 6000/fs];
[x5_filtered, x5_amp_filtered] = custom_filter(x, gains(5), wn, fs, Ns,
'bandpass', 'Fifth', 13, filter_type);


wn = [6000/fs, 12000/fs];
[x6_filtered, x6_amp_filtered] = custom_filter(x, gains(6), wn, fs, Ns,
'bandpass', 'Sixth', 16, filter_type);


wn = [12000/fs, 24000/fs];
[x7_filtered, x7_amp_filtered] = custom_filter(x, gains(7), wn, fs, Ns,
'bandpass', 'Seventh', 19, filter_type);


wn = [24000/fs, 28000/fs];
[x8_filtered, x8_amp_filtered] = custom_filter(x, gains(8), wn, fs, Ns,
'bandpass', 'Eighth', 22, filter_type);


wn = [28000/fs, 32000/fs];
[x9_filtered, x9_amp_filtered] = custom_filter(x, gains(9), wn, fs, Ns,
'bandpass', 'Ninth', 25, filter_type);


custom_plot(x1_filtered, x1_amp_filtered, t, T, fs, ' after first filter', '
after first filter')
custom_plot(x2_filtered, x2_amp_filtered, t, T, fs, ' after second filter', '
after second filter')
custom_plot(x3_filtered, x3_amp_filtered, t, T, fs, ' after third filter', '
after third filter')
custom_plot(x4_filtered, x4_amp_filtered, t, T, fs, ' after fourth filter', '
after fourth filter')
custom_plot(x5_filtered, x5_amp_filtered, t, T, fs, ' after fifth filter', '
after fifth filter')
custom_plot(x6_filtered, x6_amp_filtered, t, T, fs, ' after sixth filter', '
after sixth filter')
custom_plot(x7_filtered, x7_amp_filtered, t, T, fs, ' after seventh filter',
' after seventh filter')
custom_plot(x8_filtered, x8_amp_filtered, t, T, fs, ' after eighth filter', '
after eighth filter')
custom_plot(x9_filtered, x9_amp_filtered, t, T, fs, ' after ninth filter', '
after ninth filter')

xt_amp_filtered = x1_amp_filtered + x2_amp_filtered + x3_amp_filtered +
x4_amp_filtered + x5_amp_filtered + x6_amp_filtered + x7_amp_filtered +
x8_amp_filtered + x9_amp_filtered;

custom_plot(x, xt_amp_filtered, t, T, fs, ' original', ' filtered')
```

```
audiowrite('new.wav',xt_amp_filtered,fs)

end
```

1-c) custom_filter.m

```
function [x_filtered, x_amp_filtered] = custom_filter(x,gain, wn, fs, Ns, type, message, i, filter_type)
   gain_watt = 10.^(gain/20);
   if filter_type == 2
      num = fir1(50,wn,type);
      den = 1;
      [num,den] = eqtflength(num, den);
      [z,p,k] = tf2zp(num,den);
      [num_amp,den_amp] = zp2tf(z,p,gain_watt*k);
      disp('fir')
   else
      [num,den] = butter(3, wn, type);
      [z,p,k] = butter(3, wn, type);
      [num_amp,den_amp] = zp2tf(z,p,gain_watt*k);
      disp('iir')
   end


   figure(i)
   subplot(2,1,1)
   freqz(num_amp,den_amp)
   figure(i+1)
   sys = tf(num_amp,den_amp);
   subplot(2,1,1)
   step(sys)
   subplot(2,1,2)
   impulse(sys)
```

```
    figure(i+2)
    zplane(z,p)
    title(strcat(message,' filter zeros and poles'))

    x_filtered = filter(num,den,x);
    x_amp_filtered = filter(num_amp,den_amp,x);
end
```

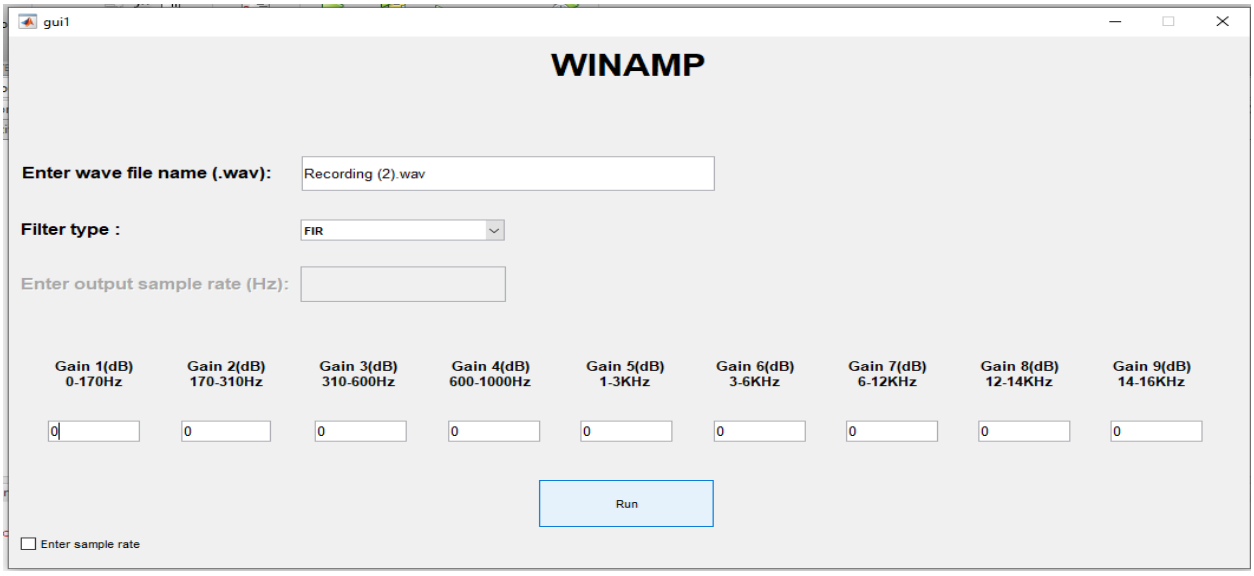
1-d) custom_plot.m

```
function custom_plot(x, x_amp, t, T, fs, message1, message2)
    X_mags = abs(fftshift(fft(x)))/fs;
    X_amp_mags = abs(fftshift(fft(x_amp)))/fs;

    figure
    subplot(2,2,1)
    plot(t,x)
    title(strcat(strcat('Signal',message1),' time domain no gain'));
    xlabel('time(sec)');
    ylabel('Amplitude')

    subplot(2,2,3)
    plot(t,x_amp)
    title(strcat(strcat('Signal',message2),' time domain with gain'));
    xlabel('time(sec)');
    ylabel('Amplitude')


    subplot(2,2,2)
    plot(T,X_mags)
    title(strcat(strcat('Signal',message1),' frequency domain no gain'))
```

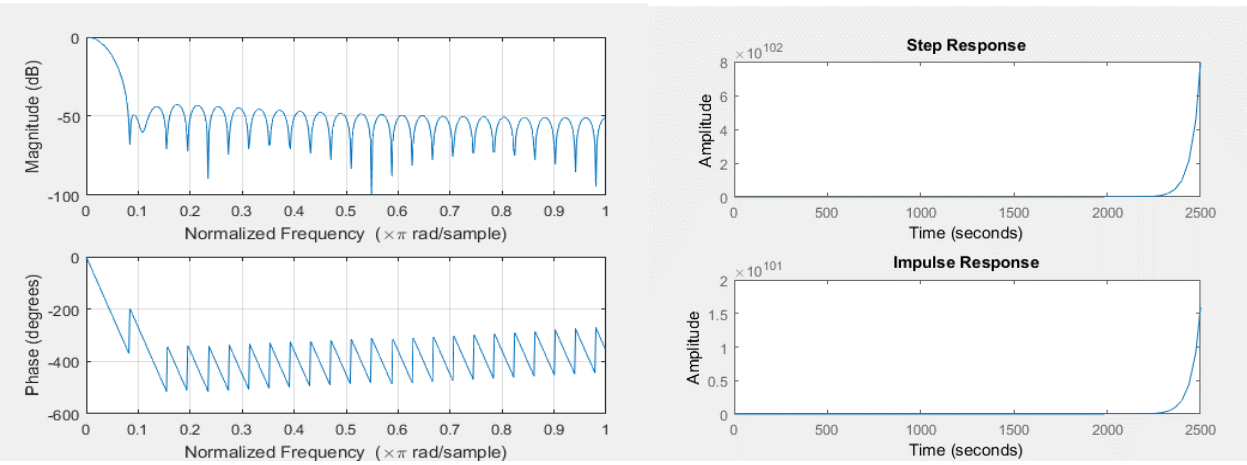
```
    xlabel('freq (Hz)')
    ylabel('Magnitude')

    subplot(2,2,4)
    plot(T,X_amp_mags)
    title(strcat(strcat('Signal',message2),' frequency domain with gain'))
    xlabel('freq (Hz)')
    ylabel('Magnitude')

end
```

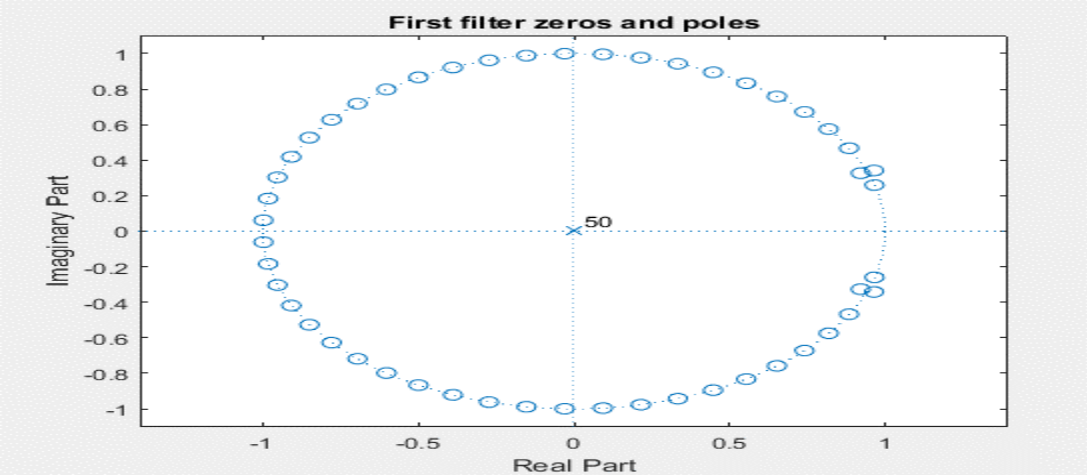
2) Sample runs:

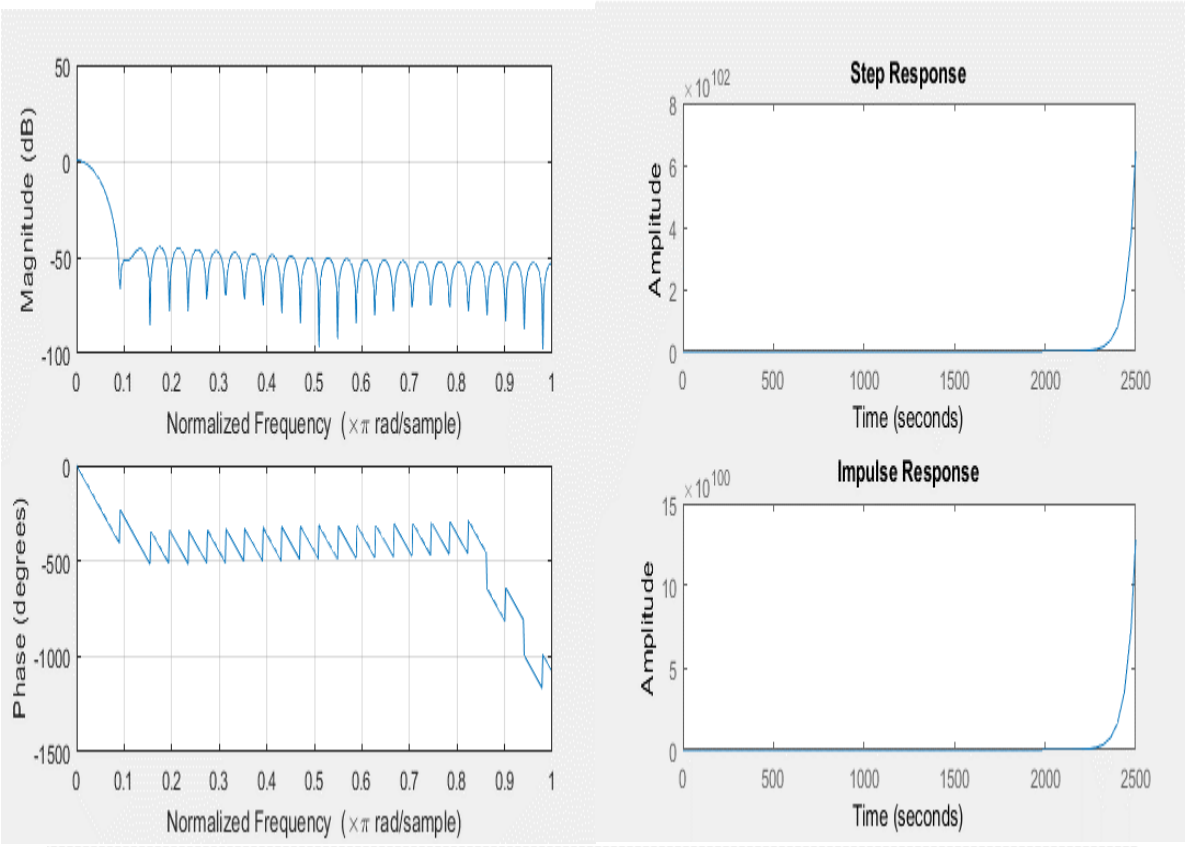
2-a) FIR with file default sample rate:

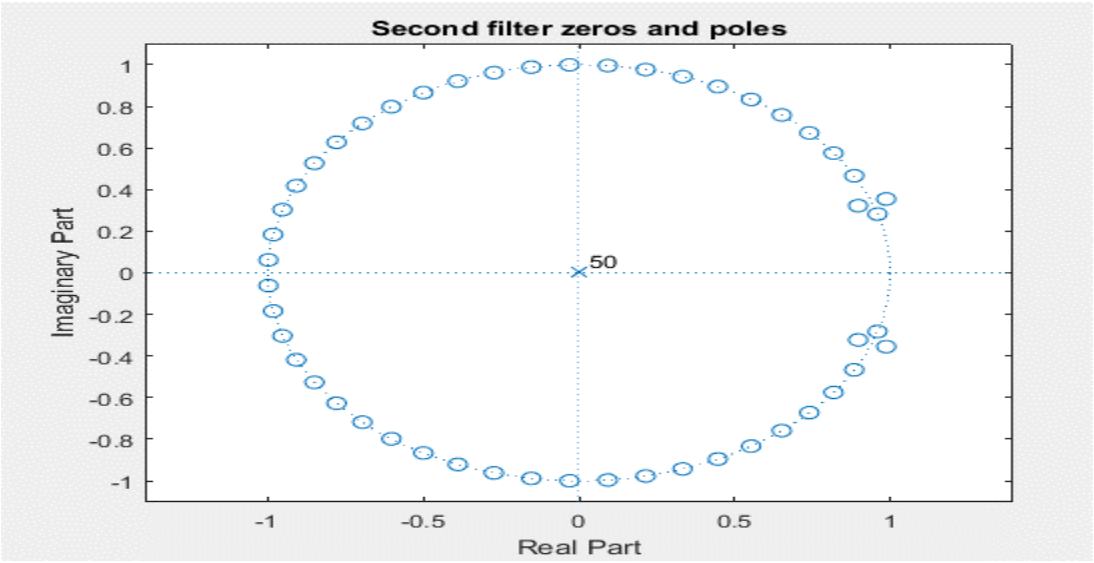


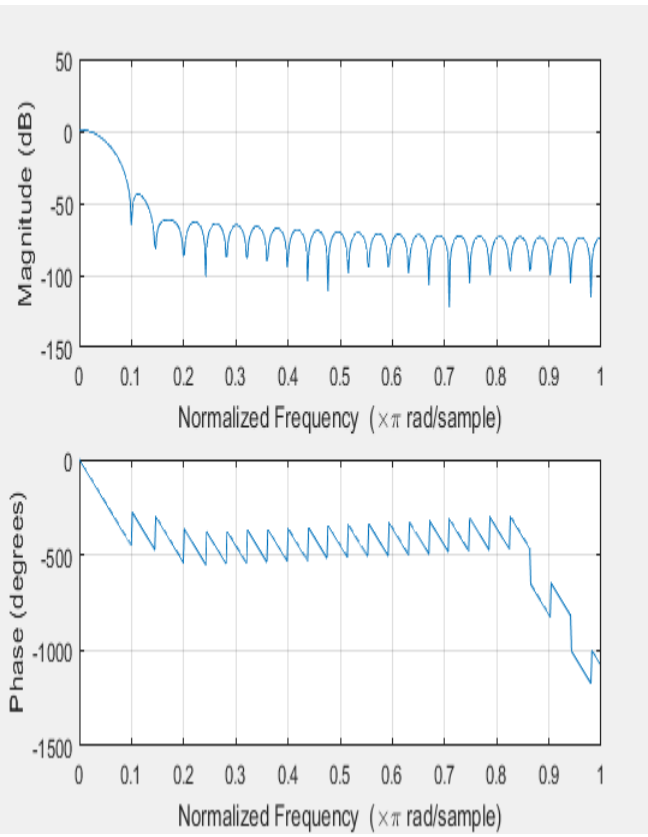
2-a-i) Analysis of the nine filters:

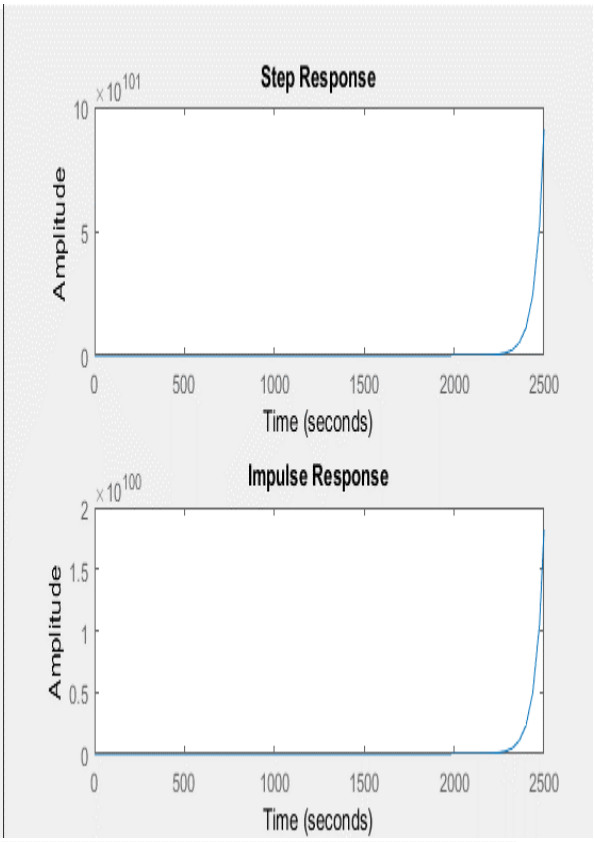


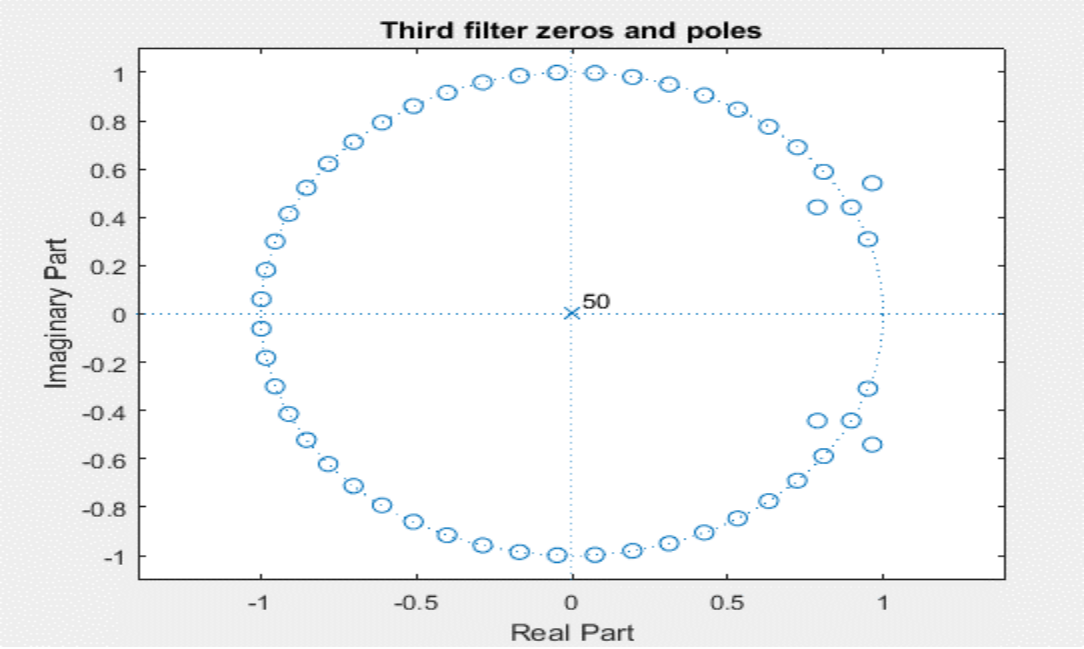


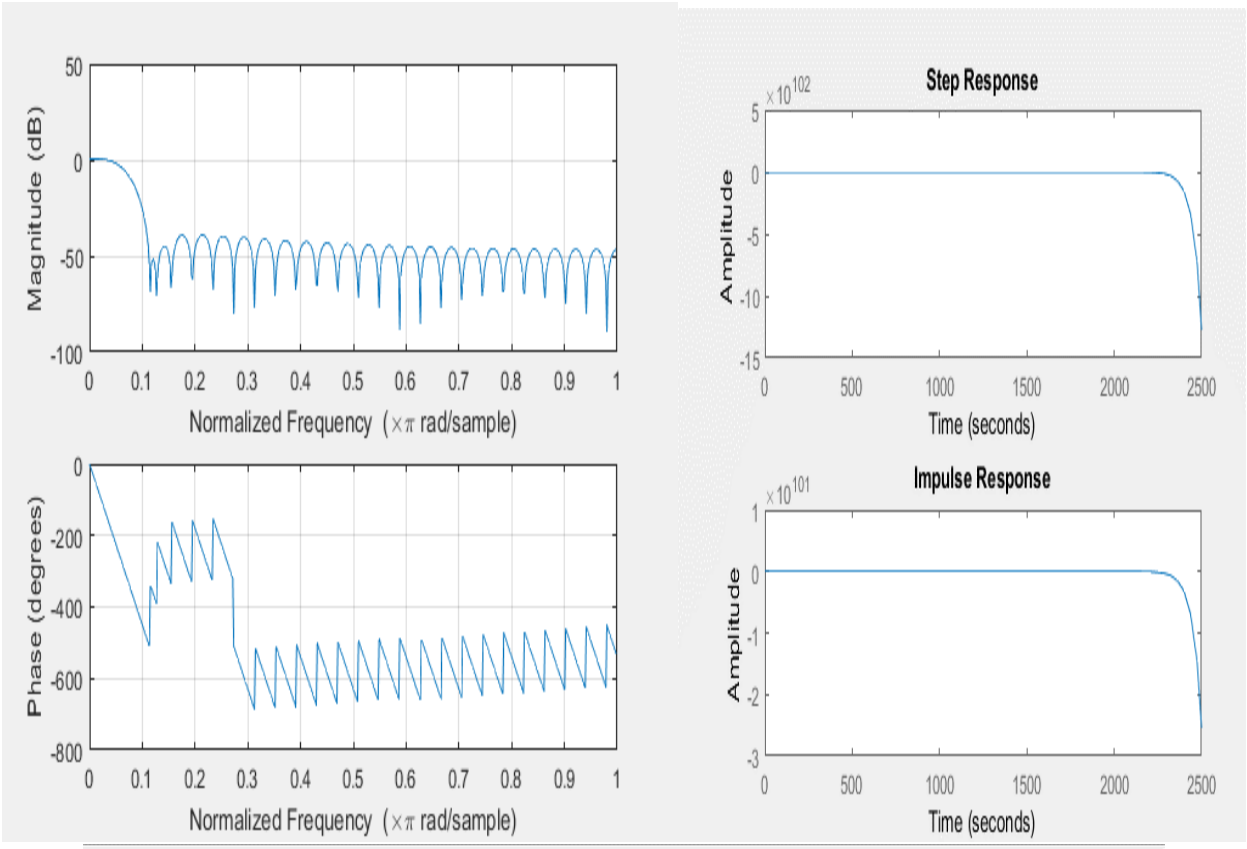

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^102
Amplitude
Time (seconds)
Impulse Response
×10^100
Amplitude
Time (seconds)

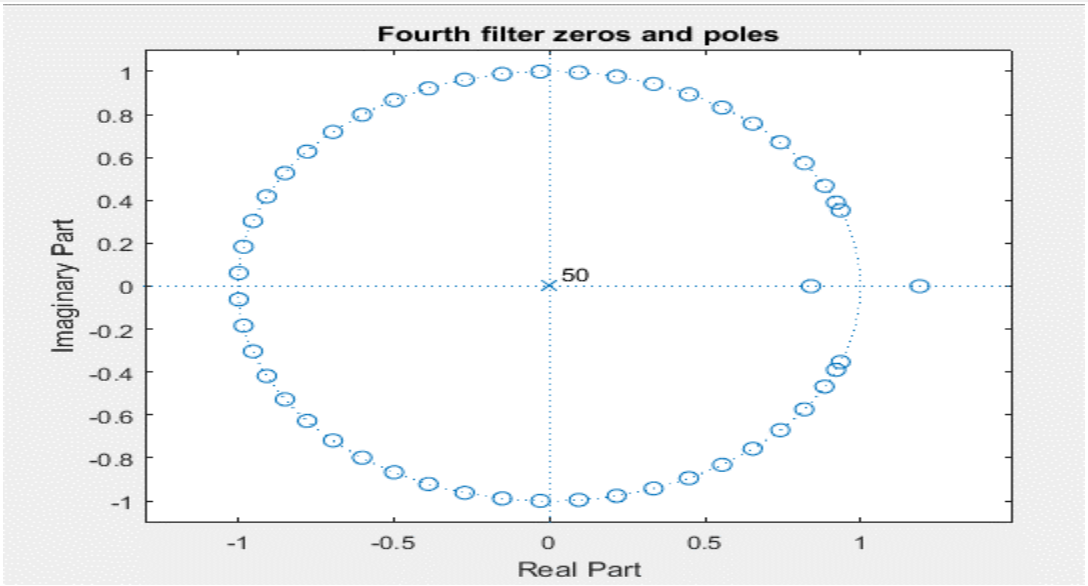

Second filter zeros and poles
Imaginary Part
Real Part
50

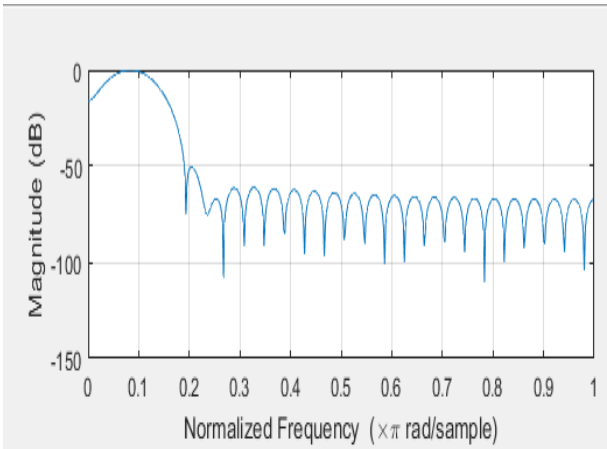

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)

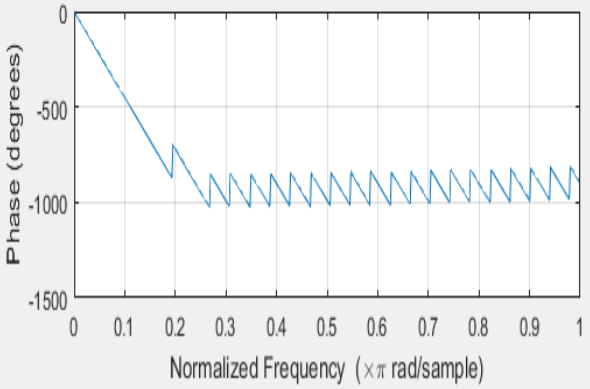

Step Response
Amplitude
Time (seconds)
Impulse Response
Amplitude
Time (seconds)

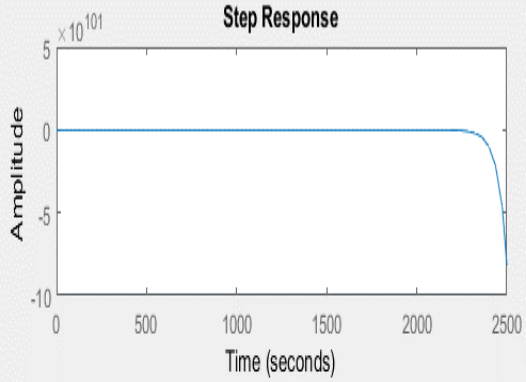

Third filter zeros and poles
Imaginary Part
Real Part
50

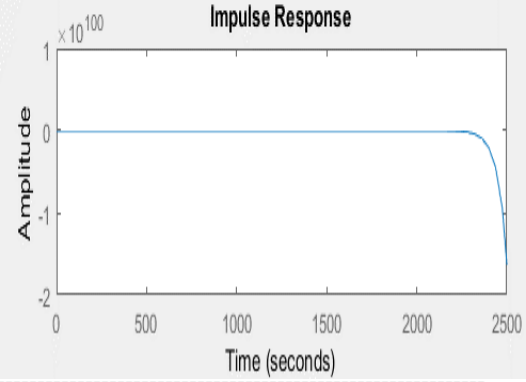

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^102
Amplitude
Time (seconds)
Impulse Response
×10^101
Amplitude
Time (seconds)

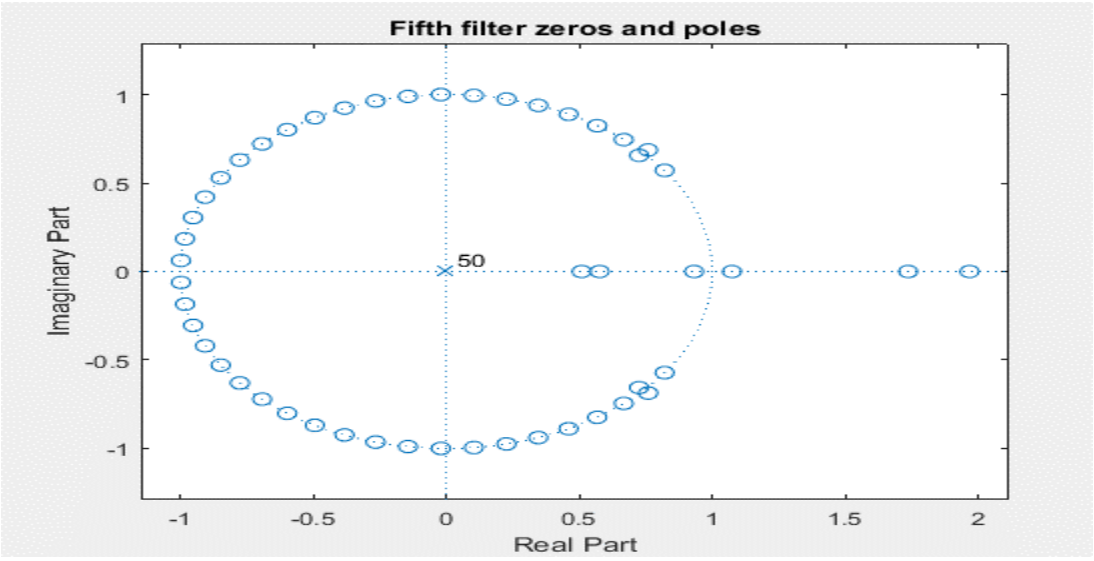

Fourth filter zeros and poles
Imaginary Part
Real Part
50

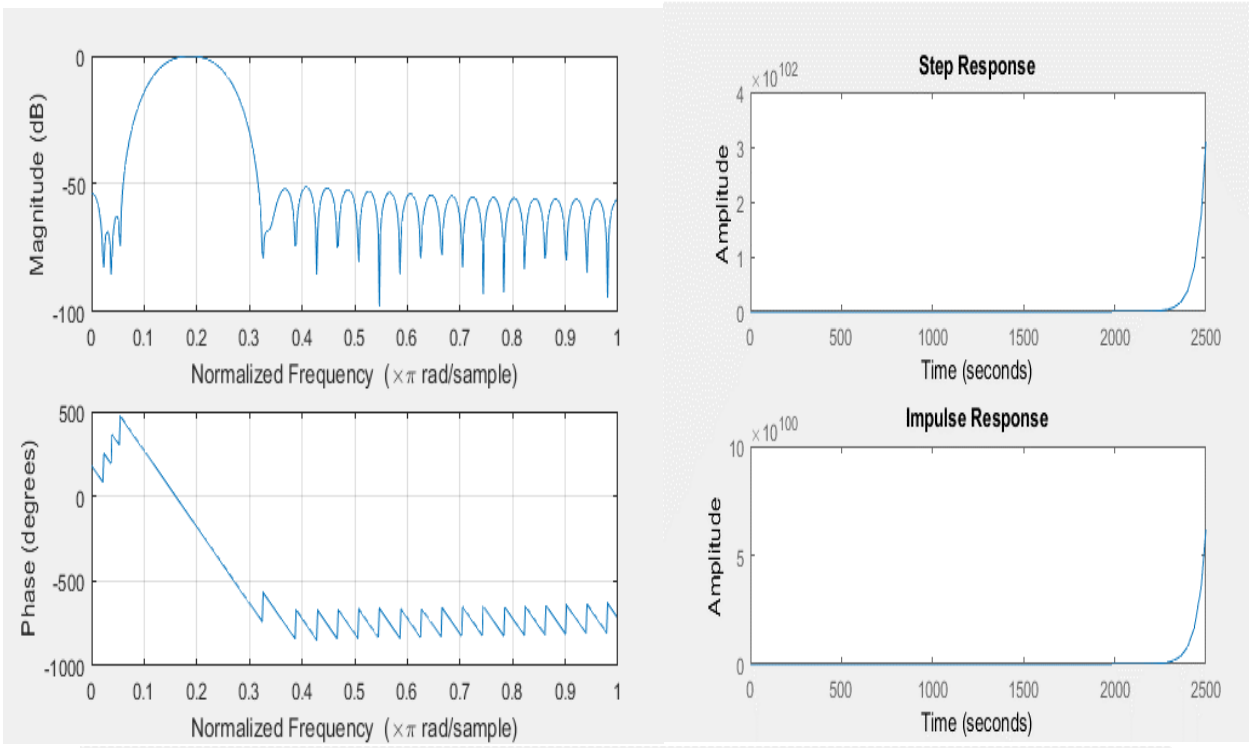

Magnitude (dB)
0
-50
-100
-150
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)

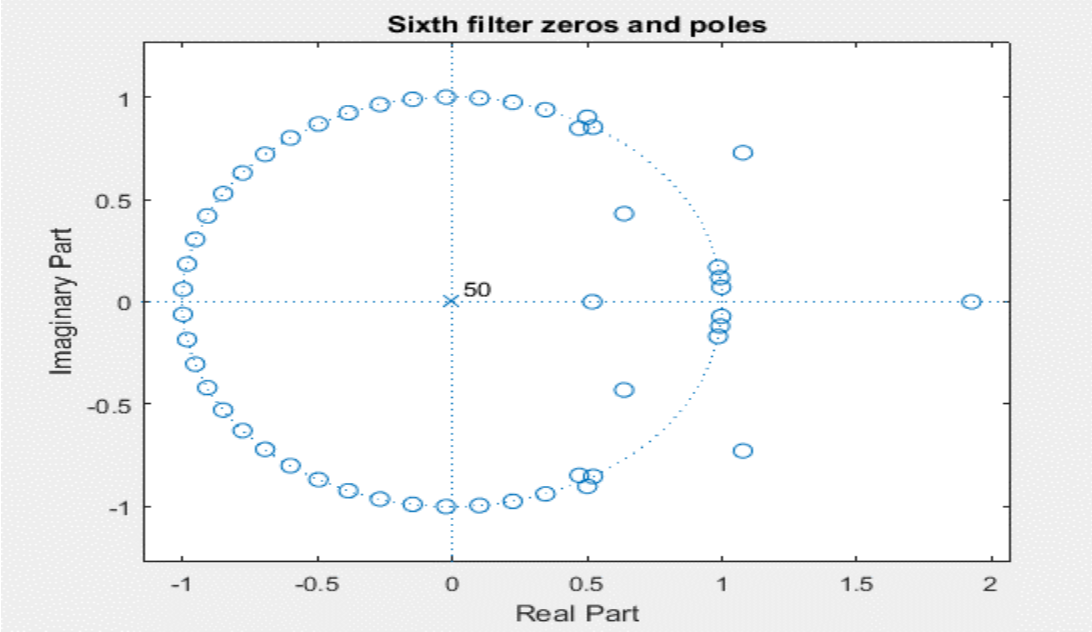

Phase (degrees)
0
-500
-1000
-1500
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)

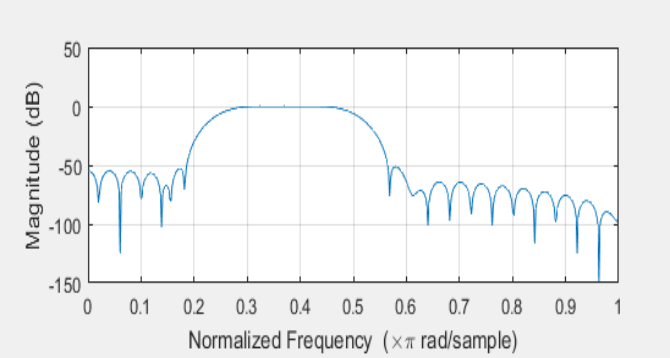

Step Response
×10^101
5
0
-5
-10
Amplitude
0 500 1000 1500 2000 2500
Time (seconds)

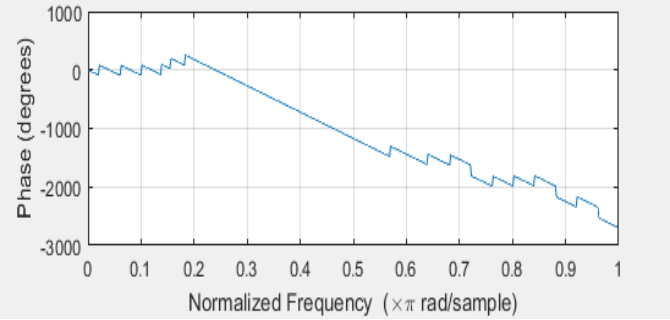

Impulse Response
×10^100
1
0
-1
-2
Amplitude
0 500 1000 1500 2000 2500
Time (seconds)

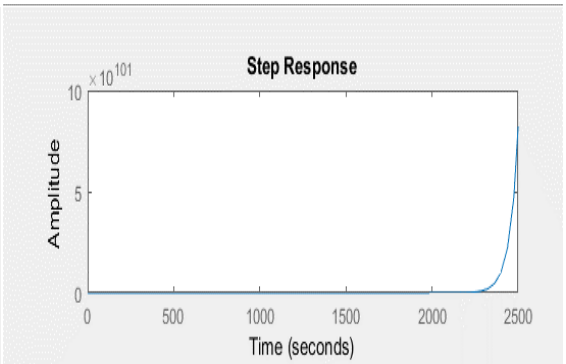

Fifth filter zeros and poles
Imaginary Part
1
0.5
0
-0.5
-1
50
-1 -0.5 0 0.5 1 1.5 2
Real Part

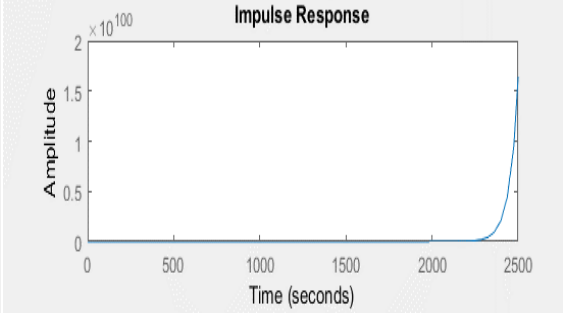

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
Amplitude
Time (seconds)
Impulse Response
Amplitude
Time (seconds)

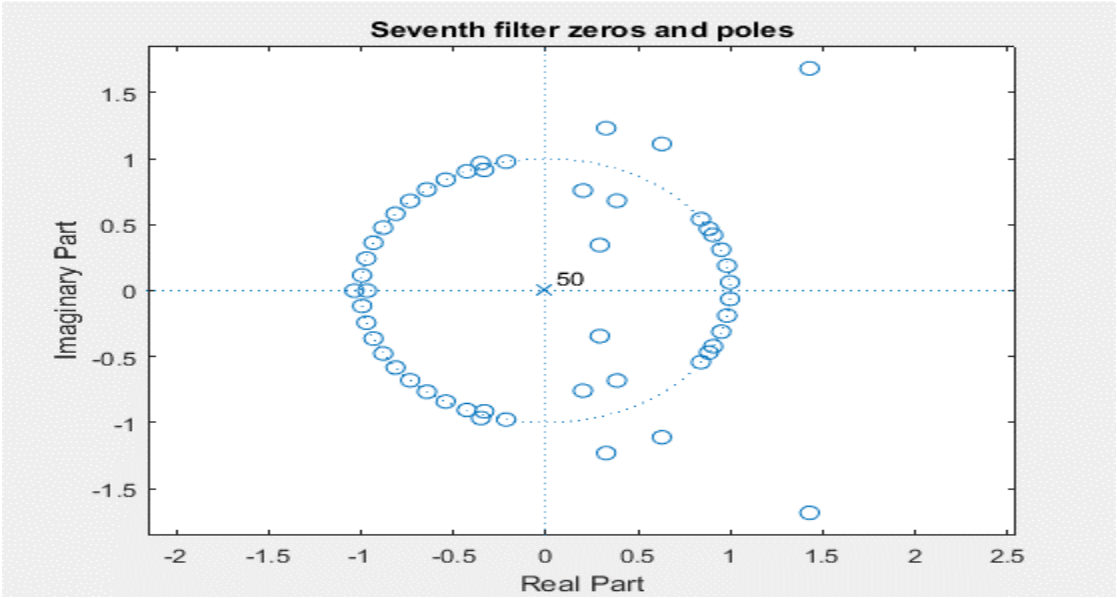

Sixth filter zeros and poles
Imaginary Part
Real Part
50

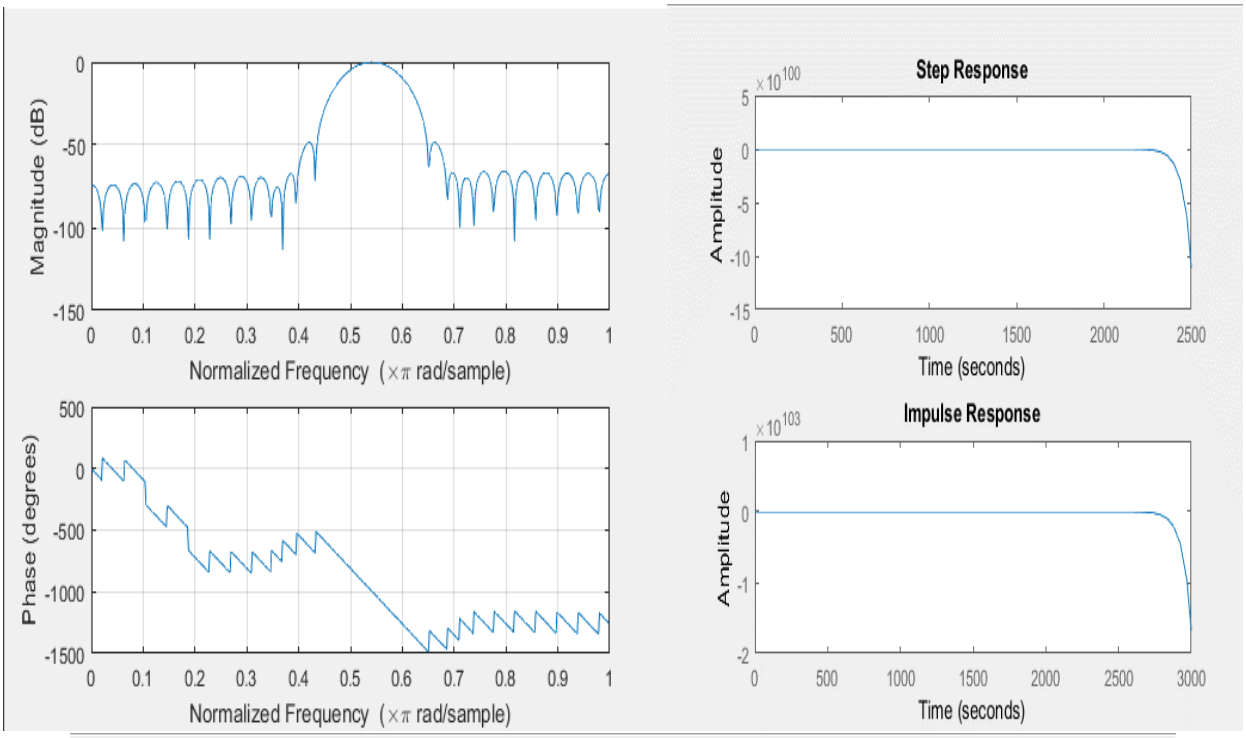

Magnitude (dB)
Normalized Frequency (×π rad/sample)

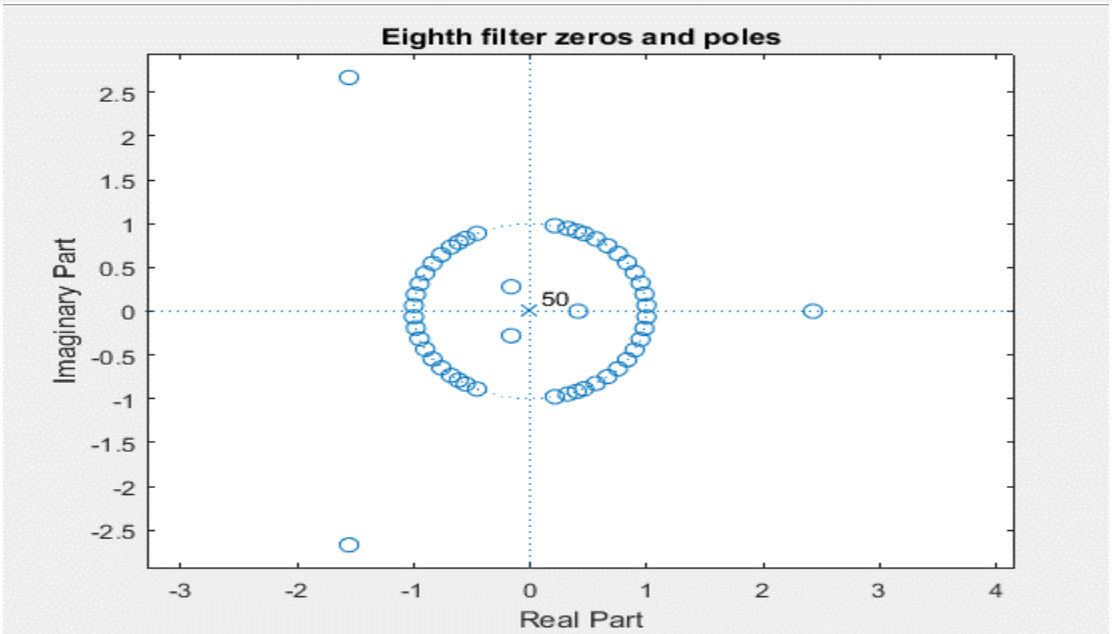

Phase (degrees)
Normalized Frequency (×π rad/sample)

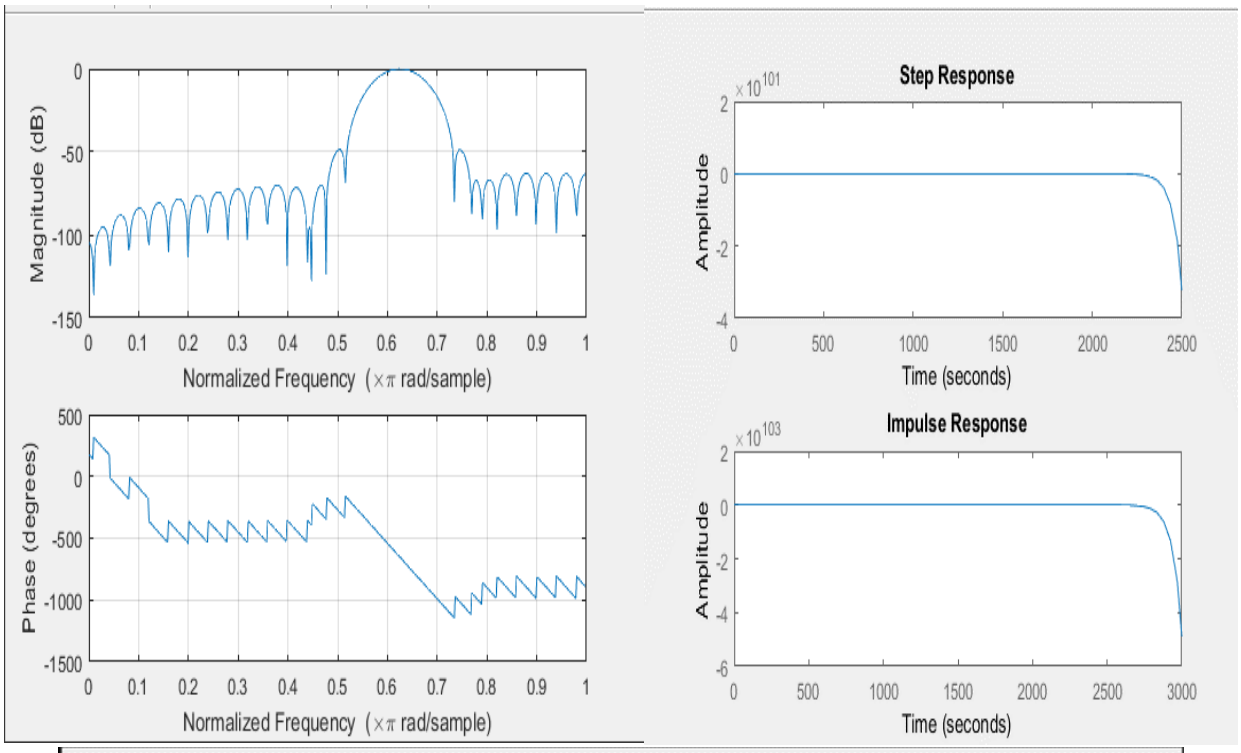

Step Response
Amplitude
Time (seconds)

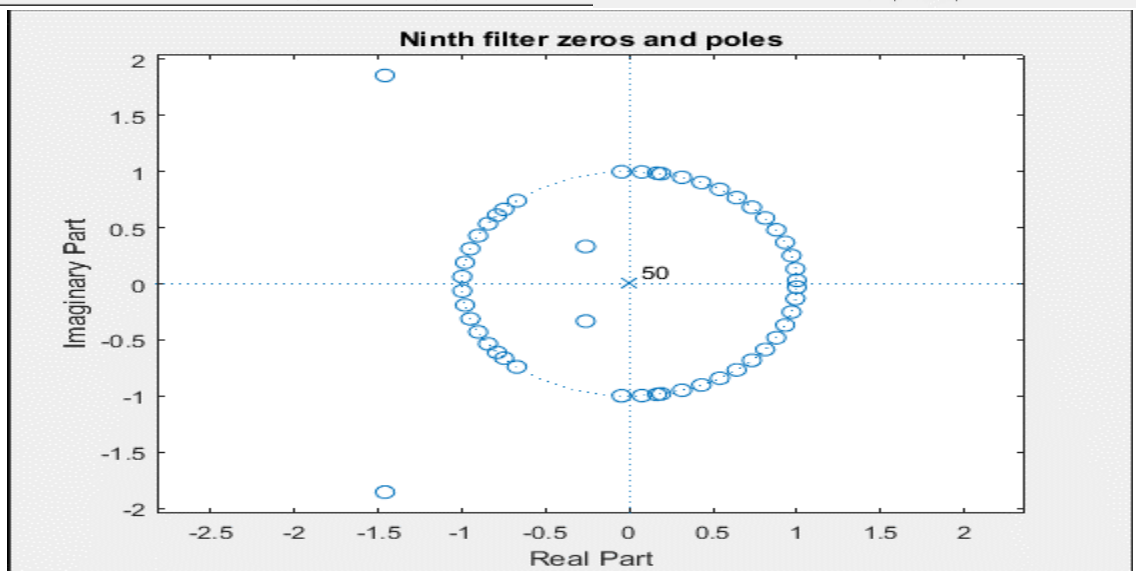

Impulse Response
Amplitude
Time (seconds)


Seventh filter zeros and poles
Imaginary Part
Real Part
50

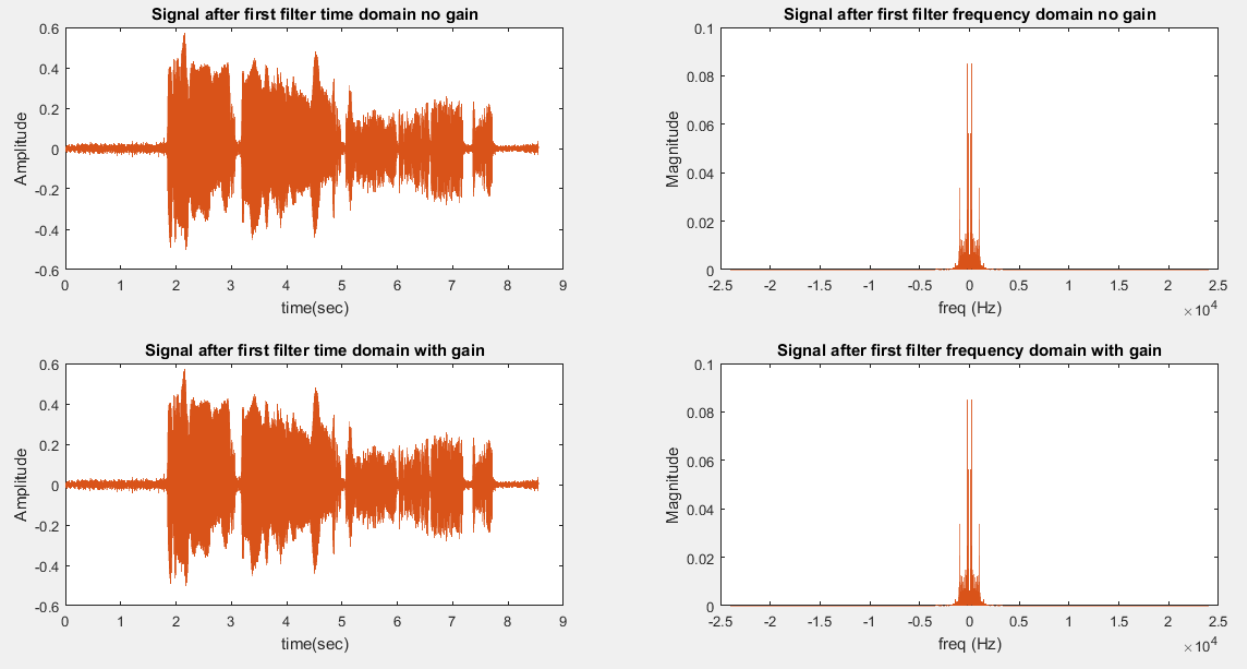

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^100
Amplitude
Time (seconds)
Impulse Response
×10^103
Amplitude
Time (seconds)

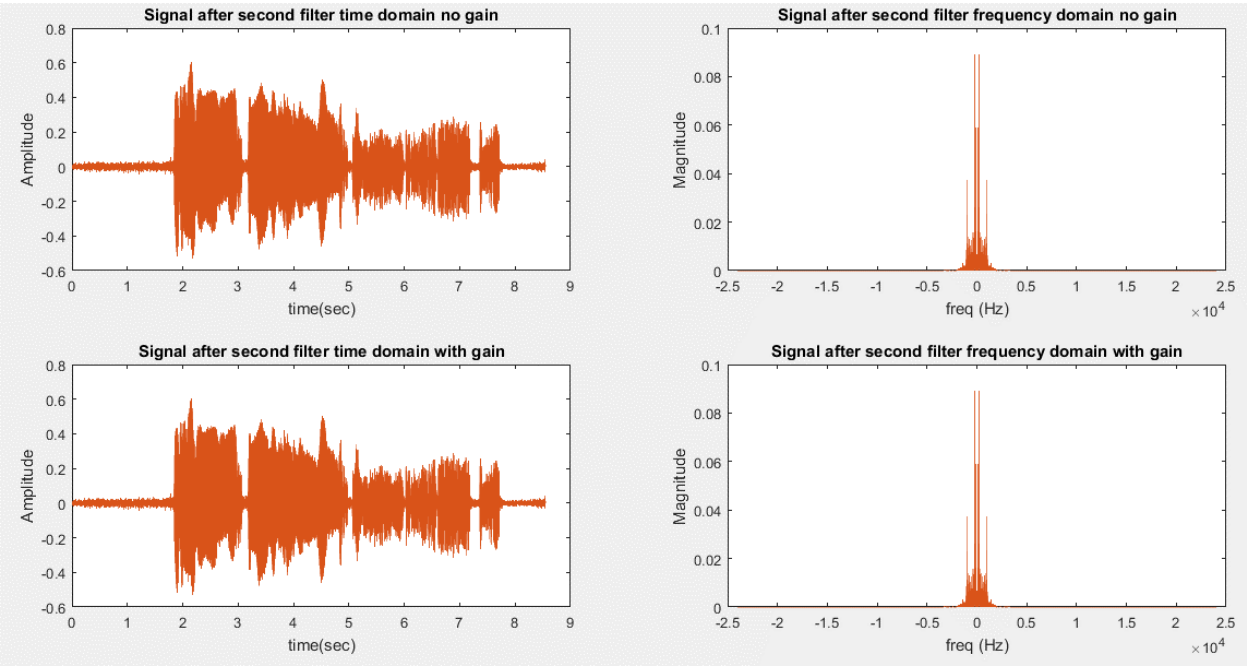

Eighth filter zeros and poles
Imaginary Part
Real Part
50

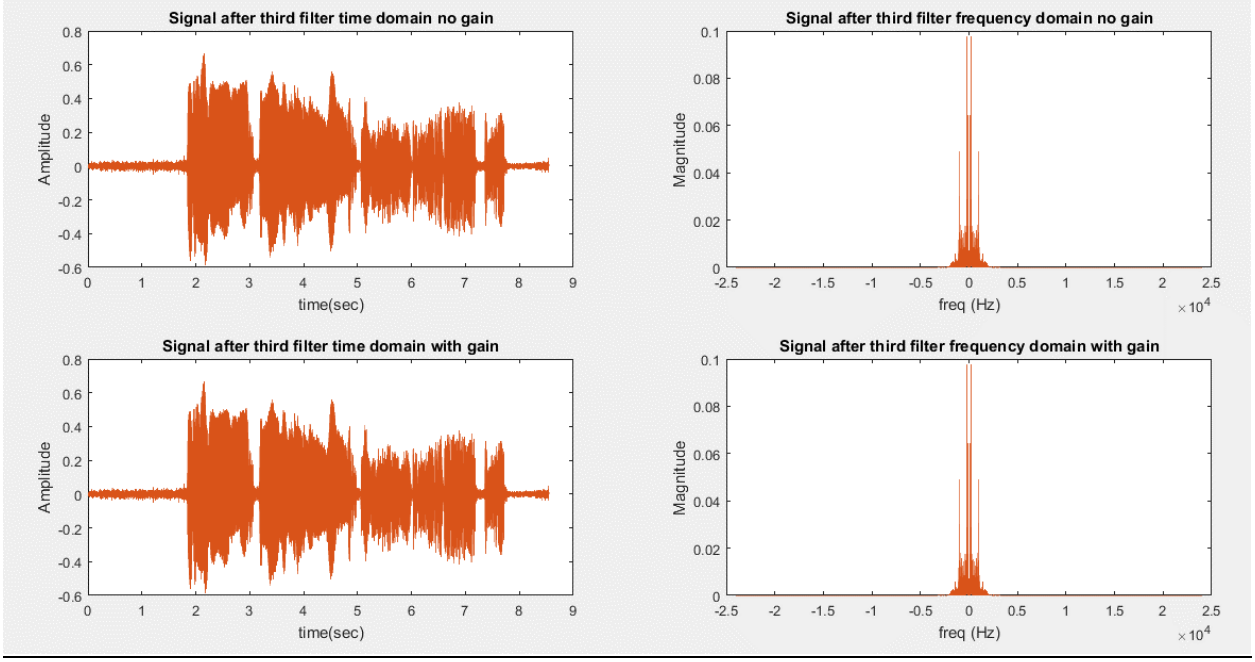

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
Amplitude
Time (seconds)
Impulse Response
Amplitude
Time (seconds)

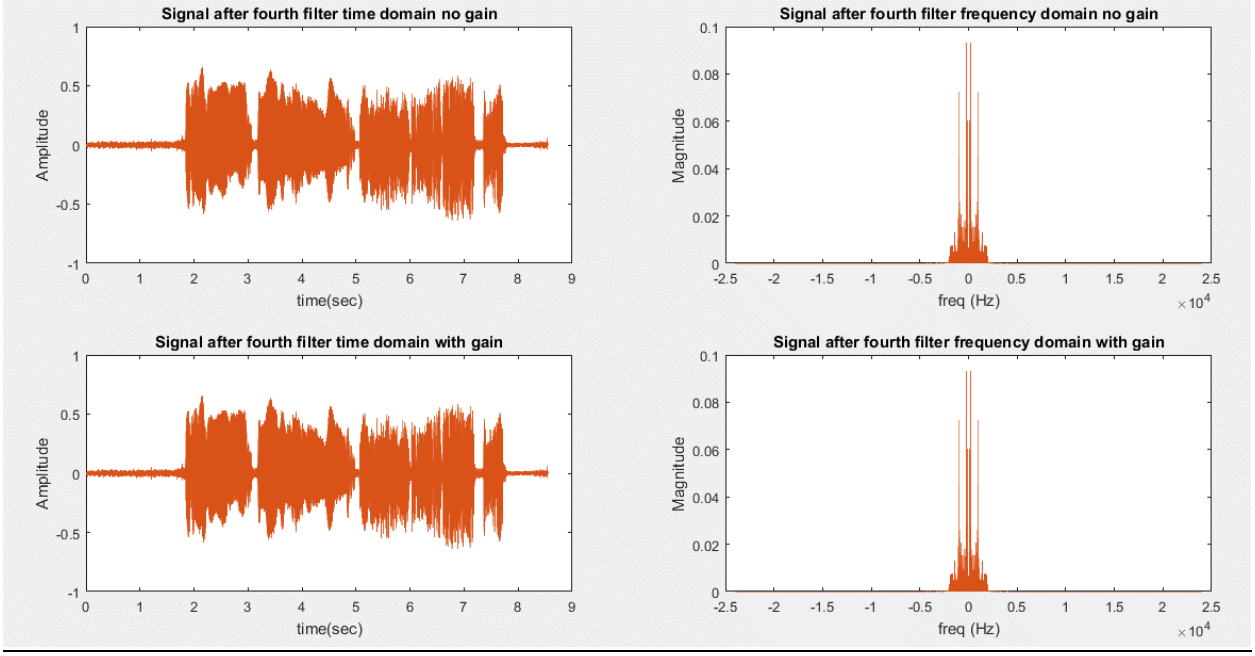

Ninth filter zeros and poles
Imaginary Part
Real Part
50

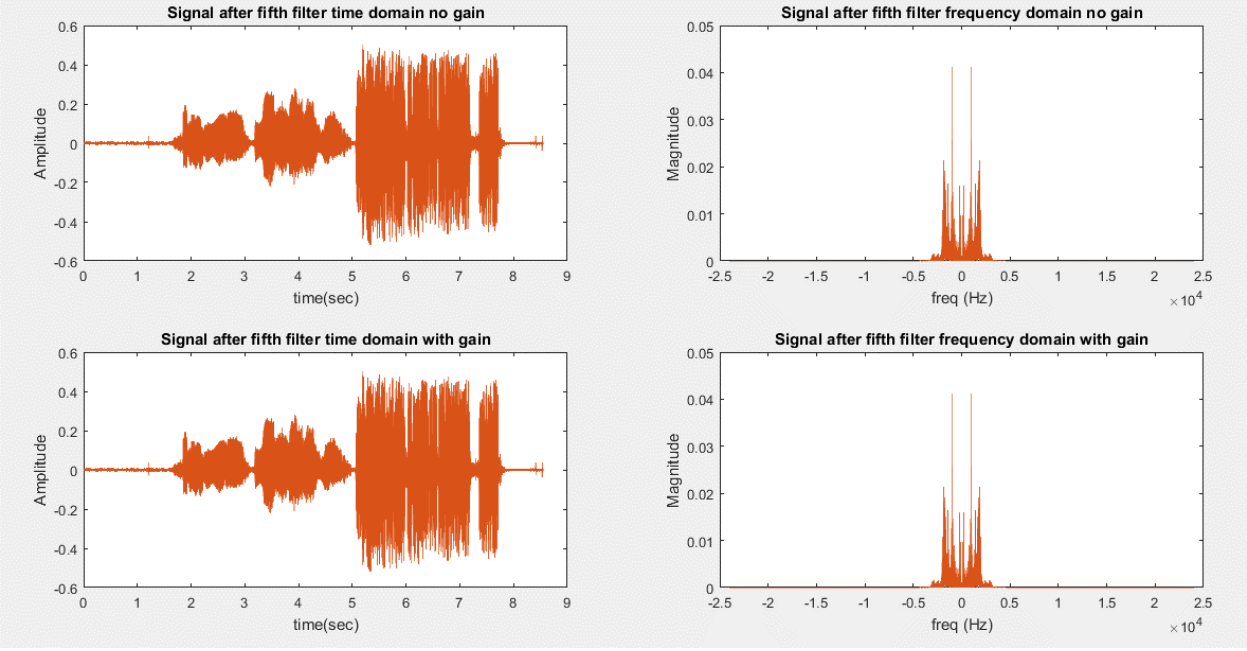
2-a-ii) All figures in time and frequency domain:

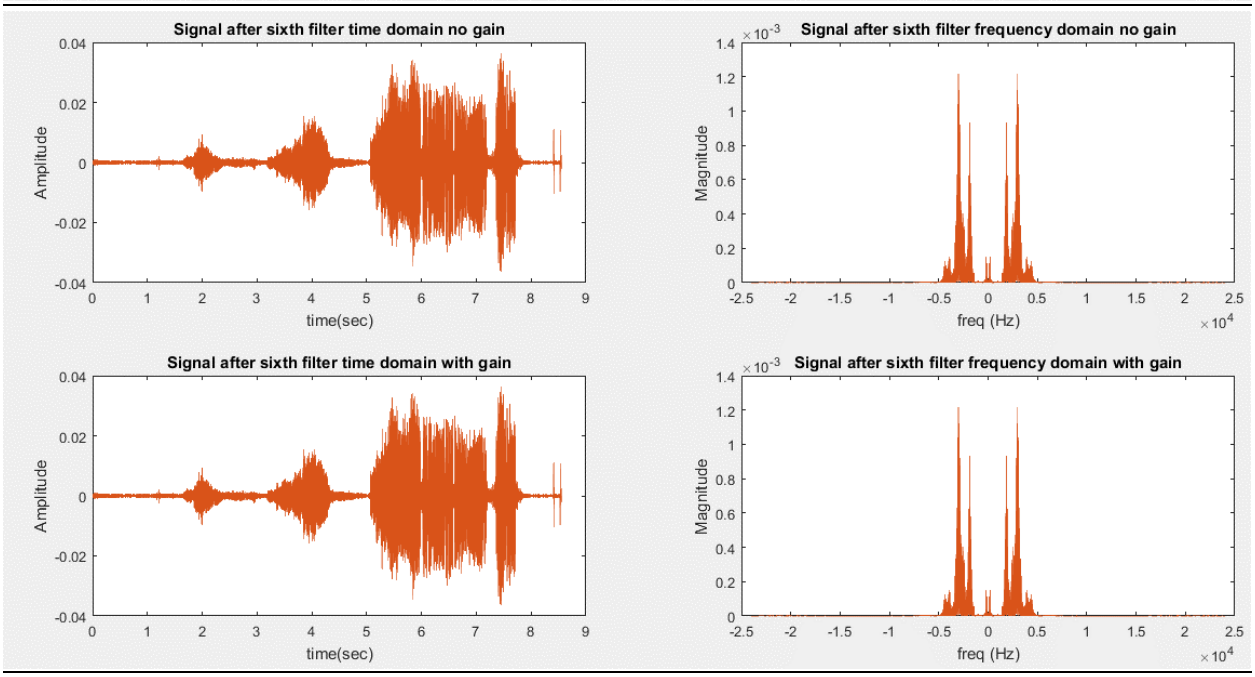


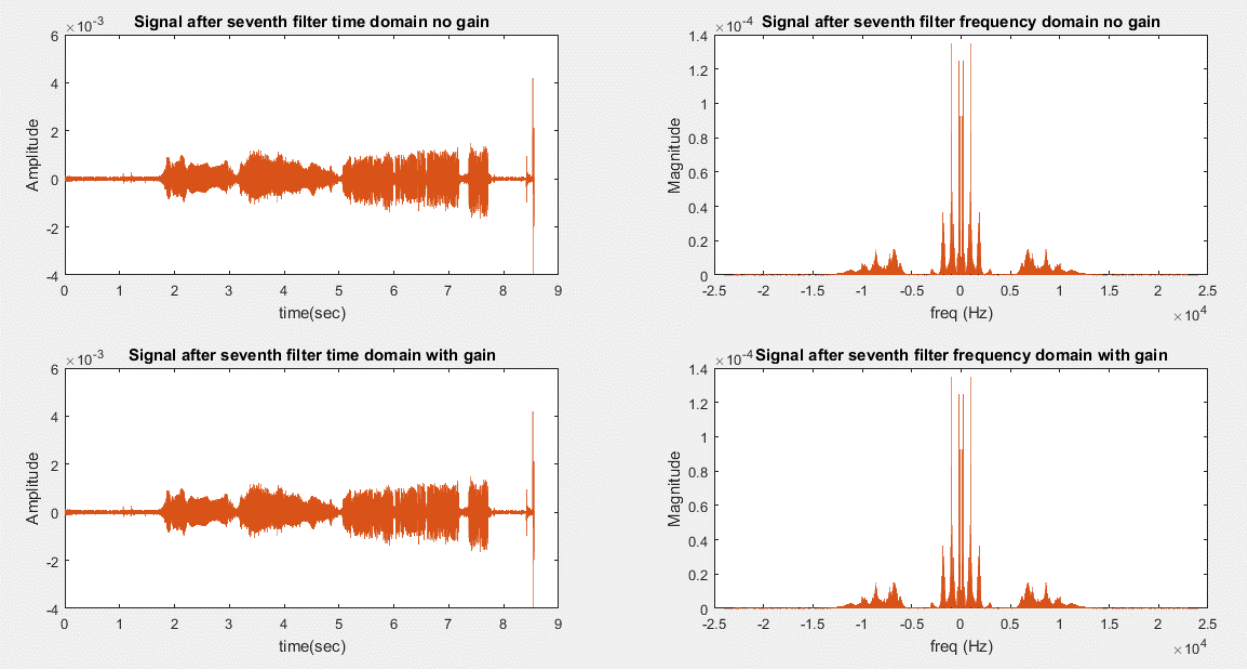


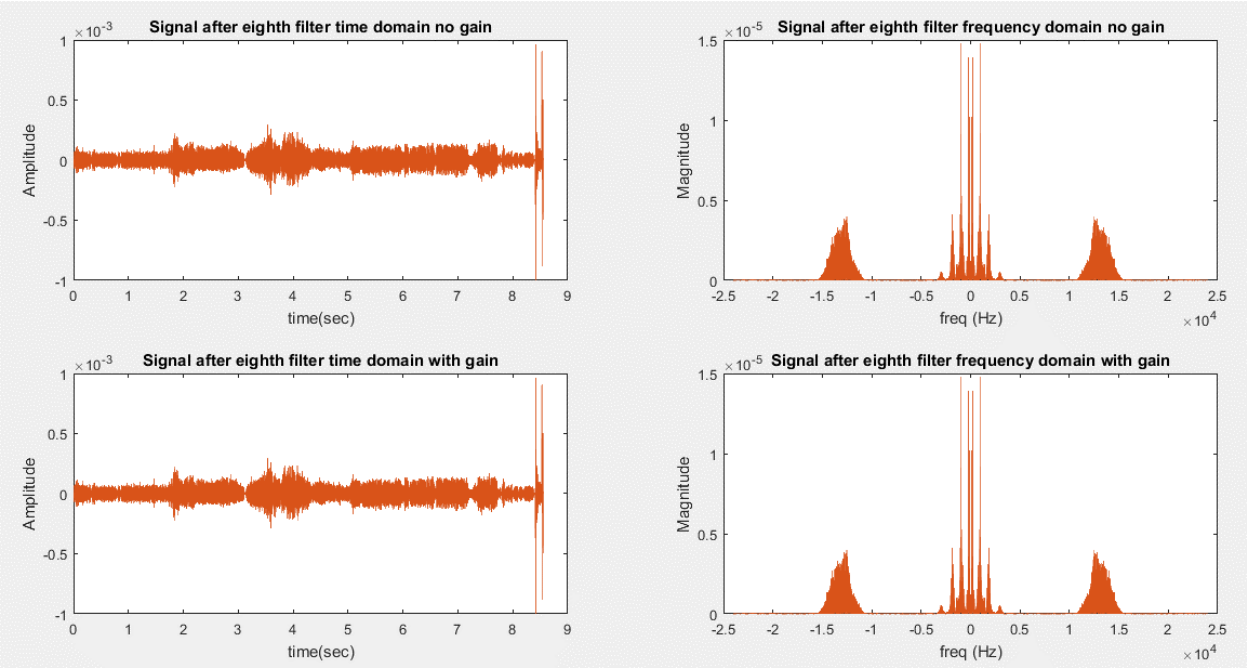

Signal after third filter time domain no gain
Amplitude
time(sec)
Signal after third filter frequency domain no gain
Magnitude
freq (Hz)
Signal after third filter time domain with gain
Amplitude
time(sec)
Signal after third filter frequency domain with gain
Magnitude
freq (Hz)


Signal after fourth filter time domain no gain
Amplitude
time(sec)
Signal after fourth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after fourth filter time domain with gain
Amplitude
time(sec)
Signal after fourth filter frequency domain with gain
Magnitude
freq (Hz)


Signal after fifth filter time domain no gain
Amplitude
time(sec)
Signal after fifth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after fifth filter time domain with gain
Amplitude
time(sec)
Signal after fifth filter frequency domain with gain
Magnitude
freq (Hz)


Signal after sixth filter time domain no gain
Amplitude
time(sec)
Signal after sixth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after sixth filter time domain with gain
Amplitude
time(sec)
Signal after sixth filter frequency domain with gain
Magnitude
freq (Hz)


Signal after seventh filter time domain no gain
Amplitude
time(sec)
Signal after seventh filter frequency domain no gain
Magnitude
freq (Hz)
Signal after seventh filter time domain with gain
Amplitude
time(sec)
Signal after seventh filter frequency domain with gain
Magnitude
freq (Hz)


Signal after eighth filter time domain no gain
Amplitude
time(sec)
Signal after eighth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after eighth filter time domain with gain
Amplitude
time(sec)
Signal after eighth filter frequency domain with gain
Magnitude
freq (Hz)

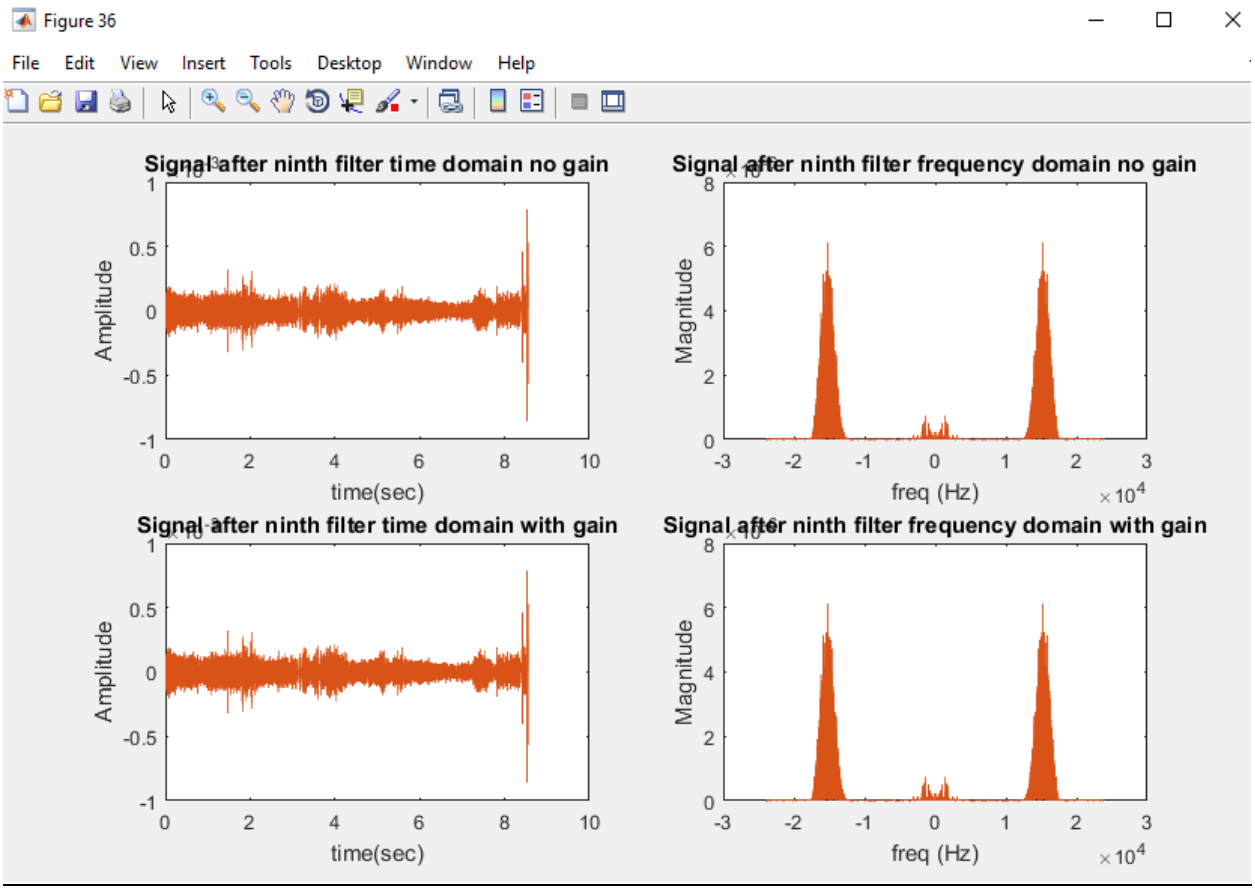


Figure 36
File Edit View Insert Tools Desktop Window Help
Signal after ninth filter time domain no gain
Amplitude
time(sec)
Signal after ninth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after ninth filter time domain with gain
Amplitude
time(sec)
Signal after ninth filter frequency domain with gain
Magnitude
freq (Hz)

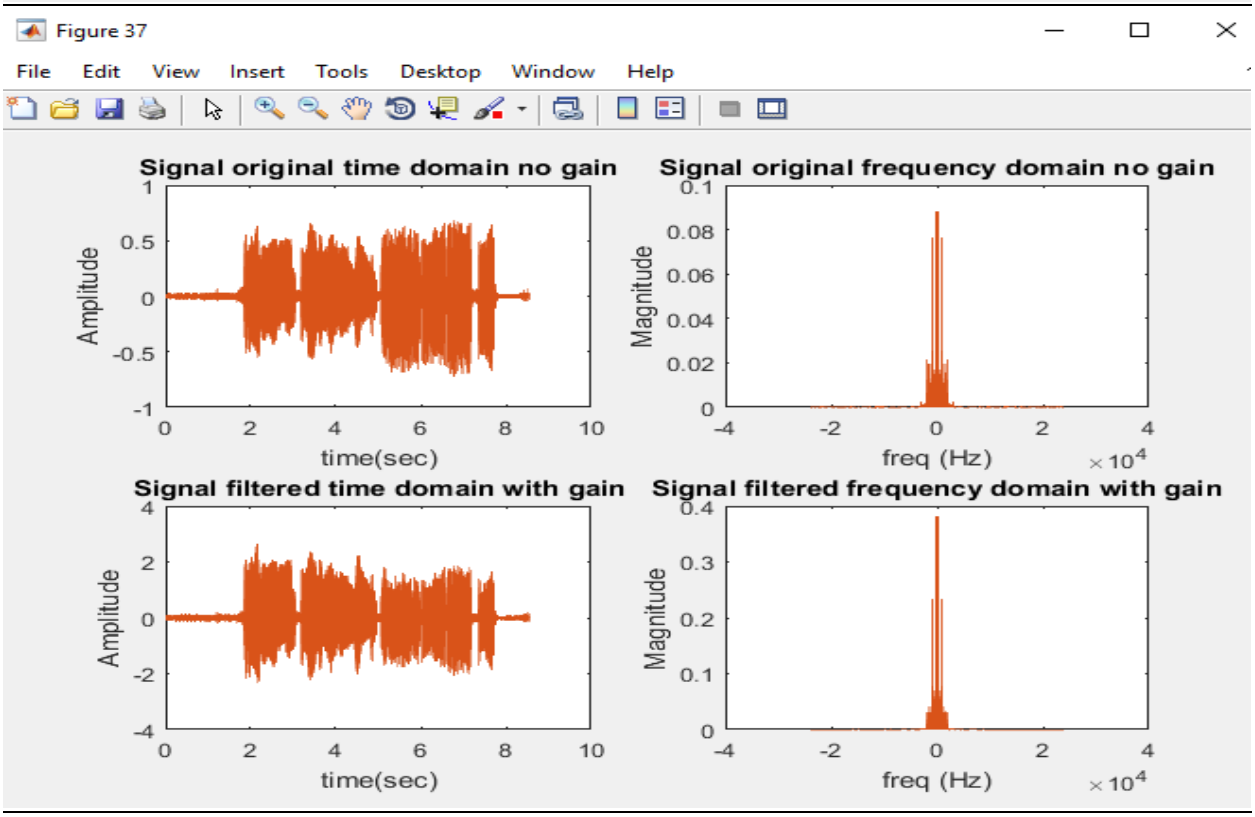
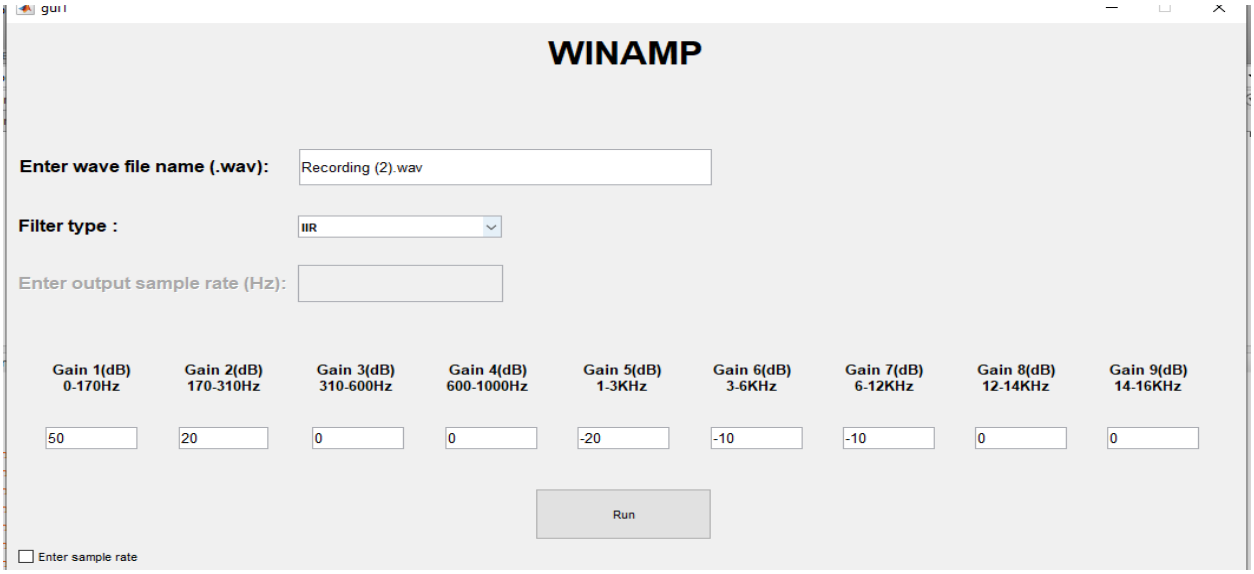
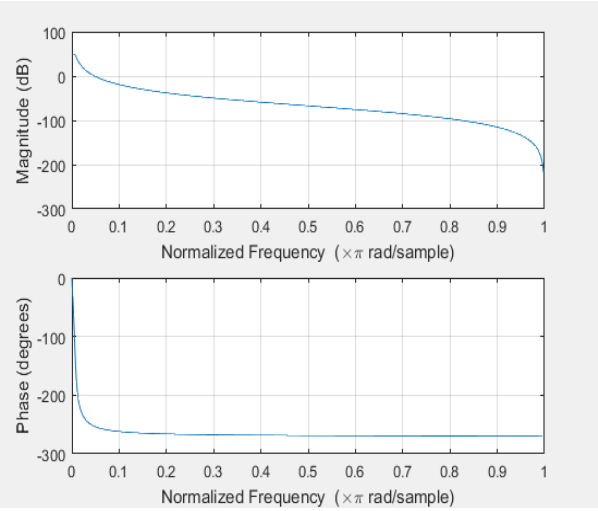


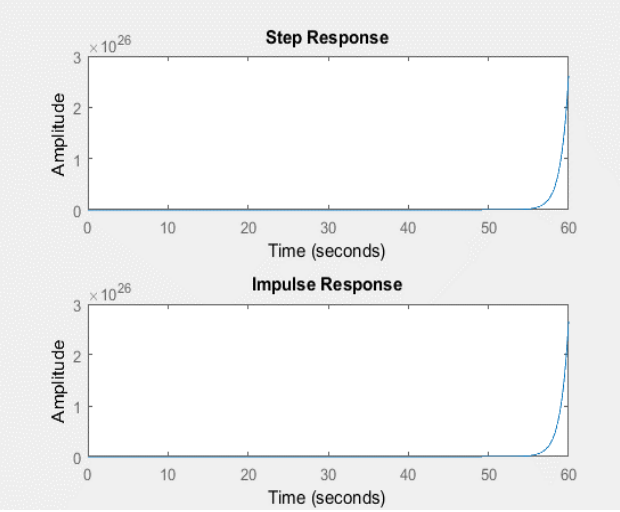
Figure 37
File Edit View Insert Tools Desktop Window Help
Signal original time domain no gain
Amplitude
time(sec)
Signal original frequency domain no gain
Magnitude
freq (Hz)
Signal filtered time domain with gain
Amplitude
time(sec)
Signal filtered frequency domain with gain
Magnitude
freq (Hz)

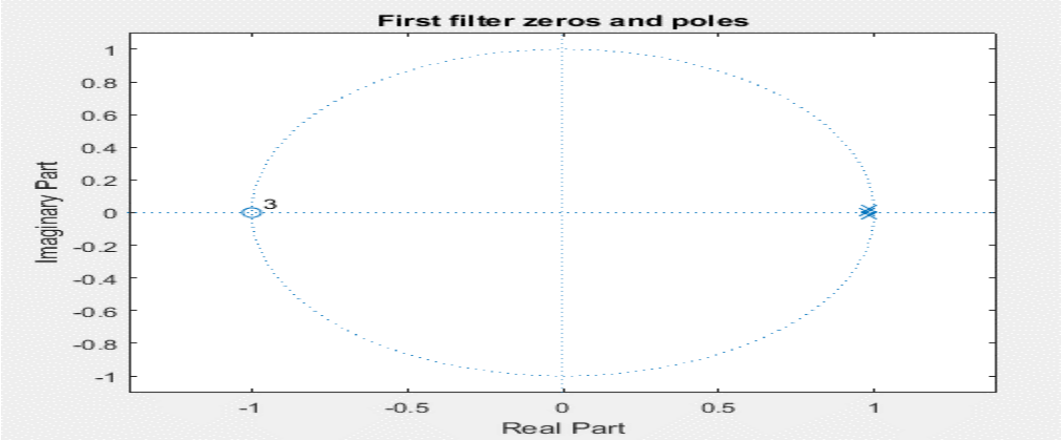
2-b) IIR with file default sample rate:

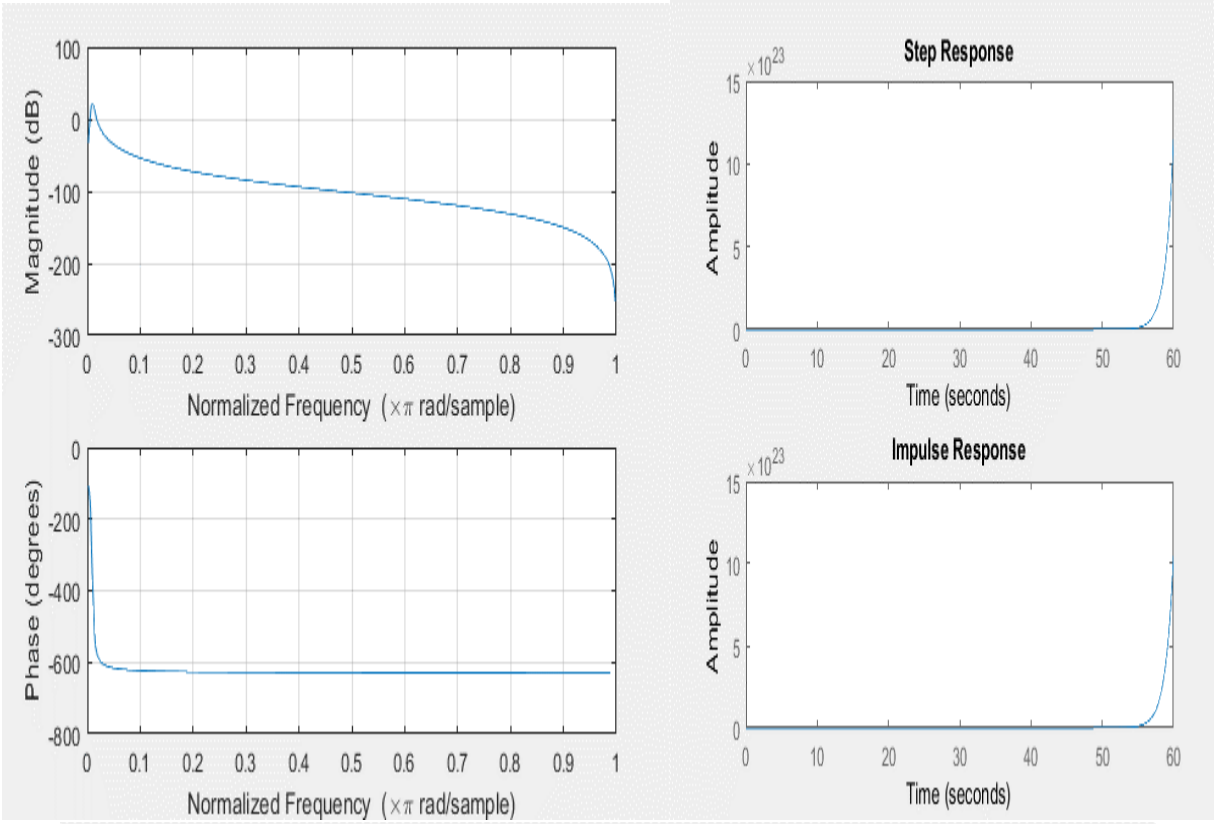


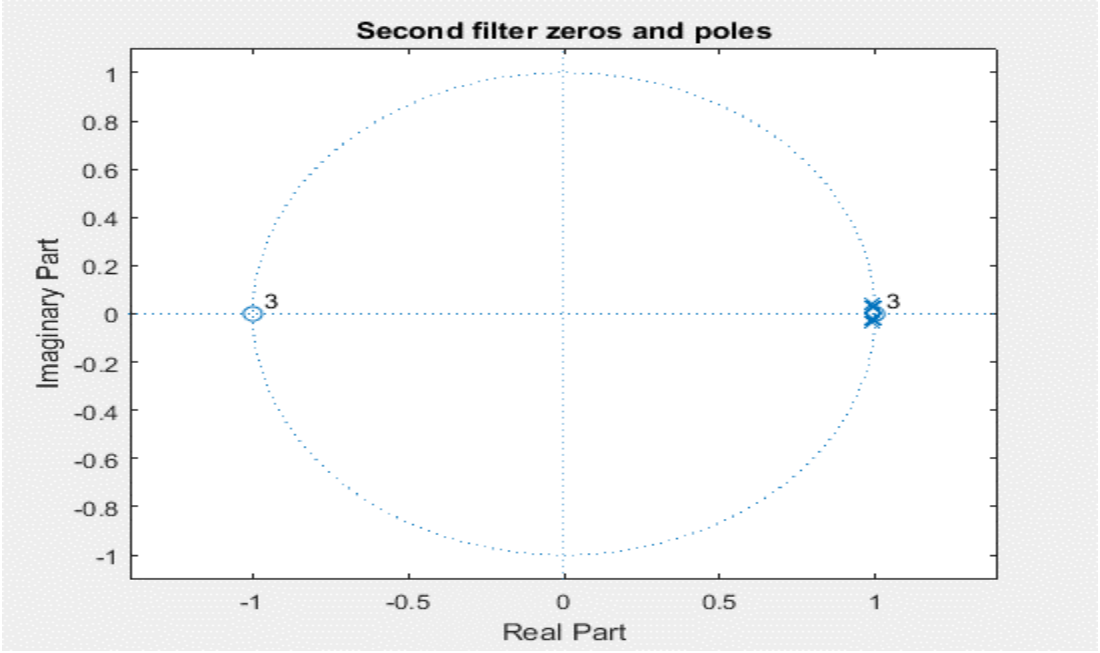
2-b-i) Analysis of the nine filters:

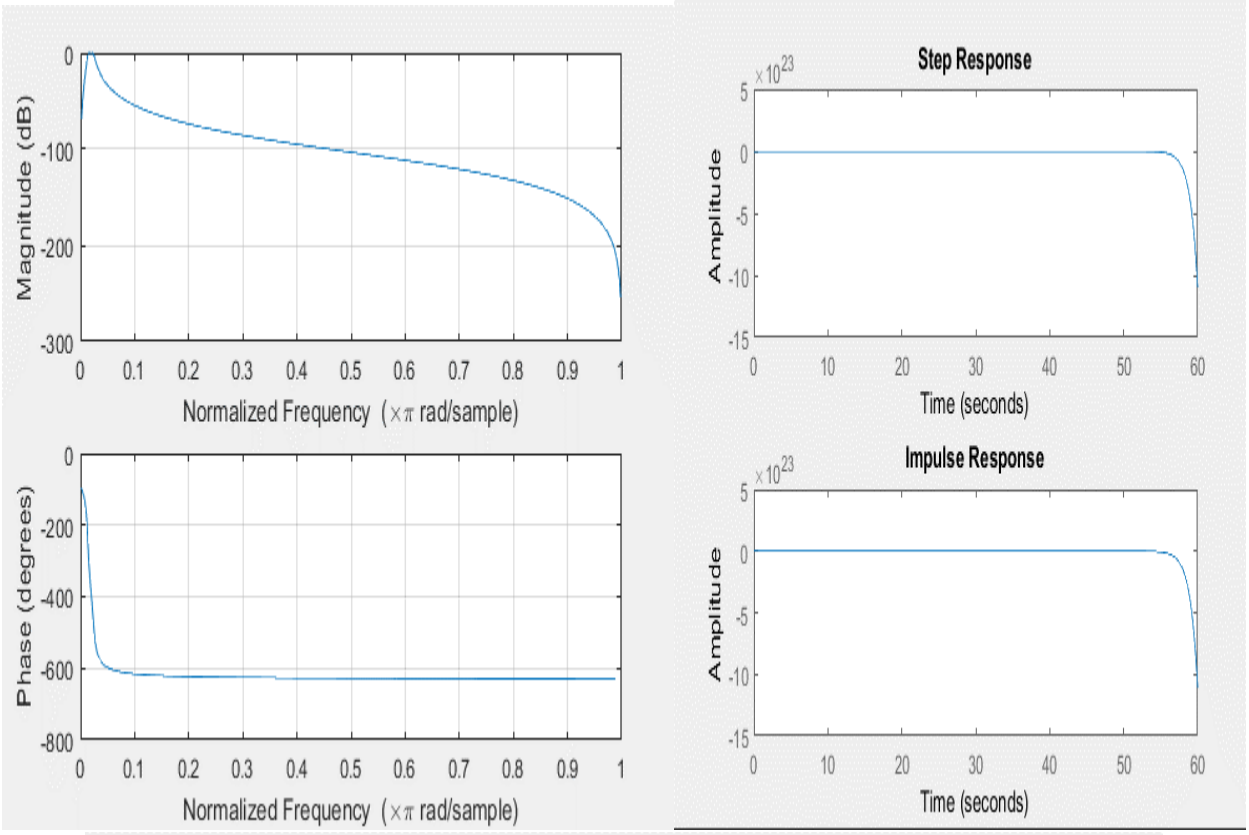


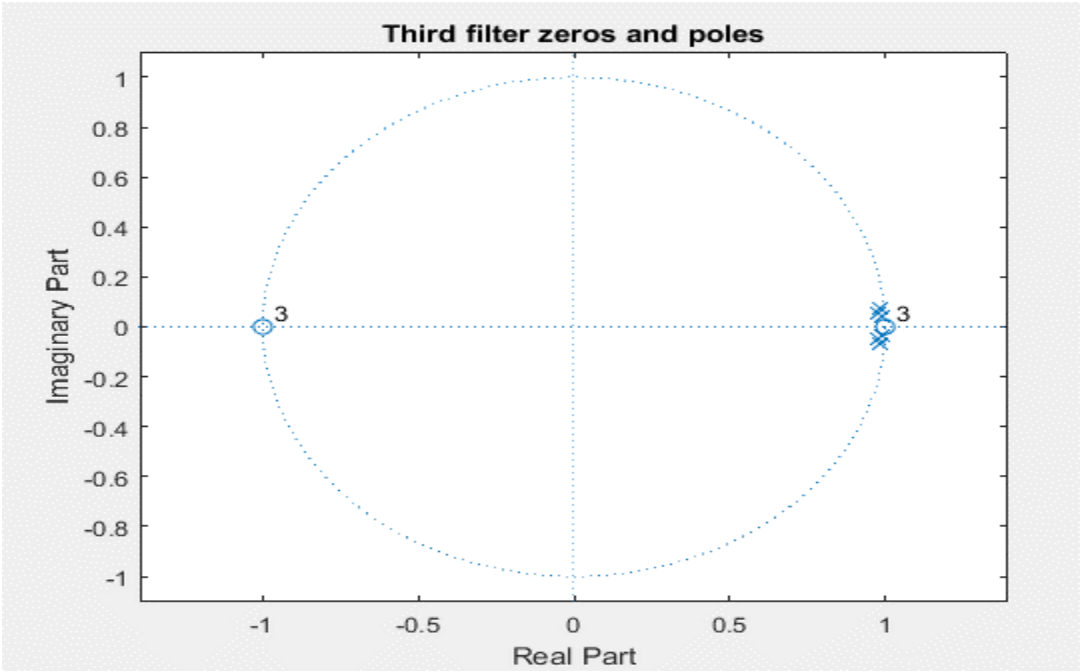


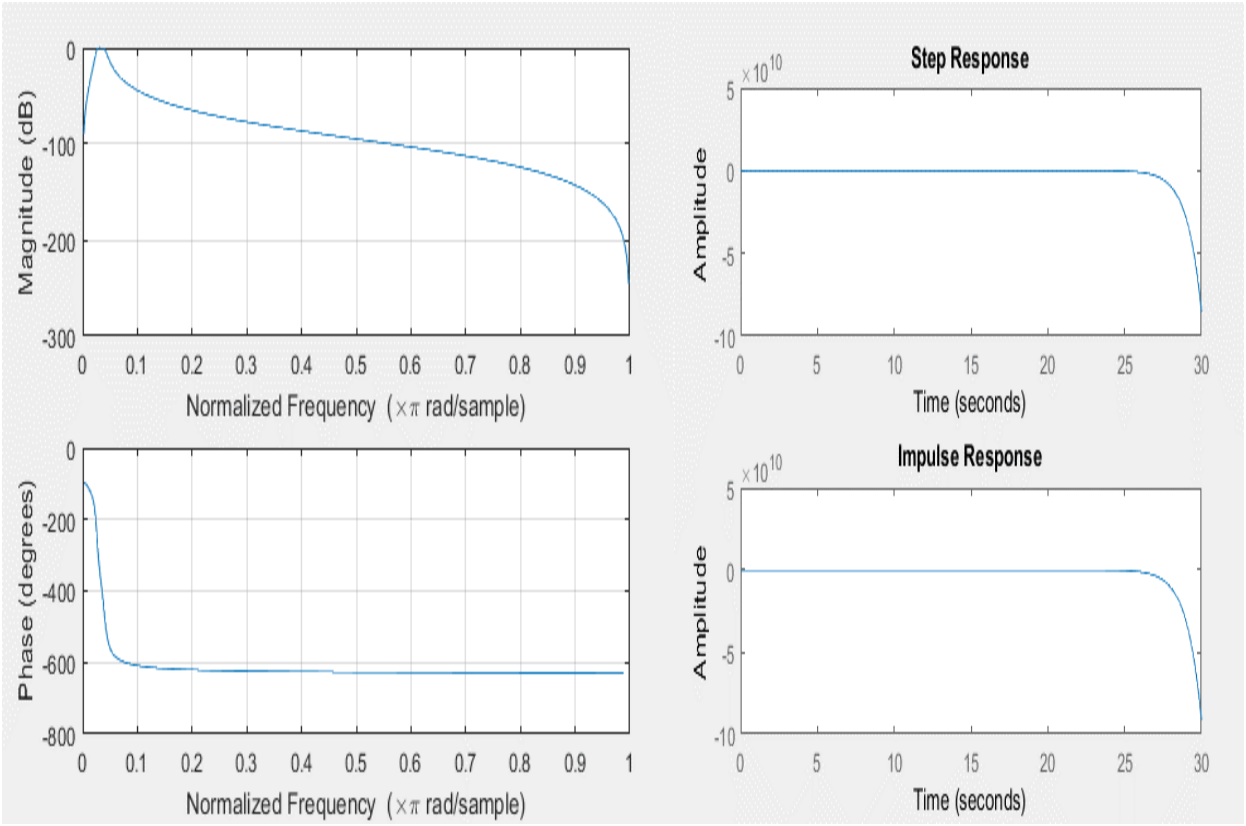


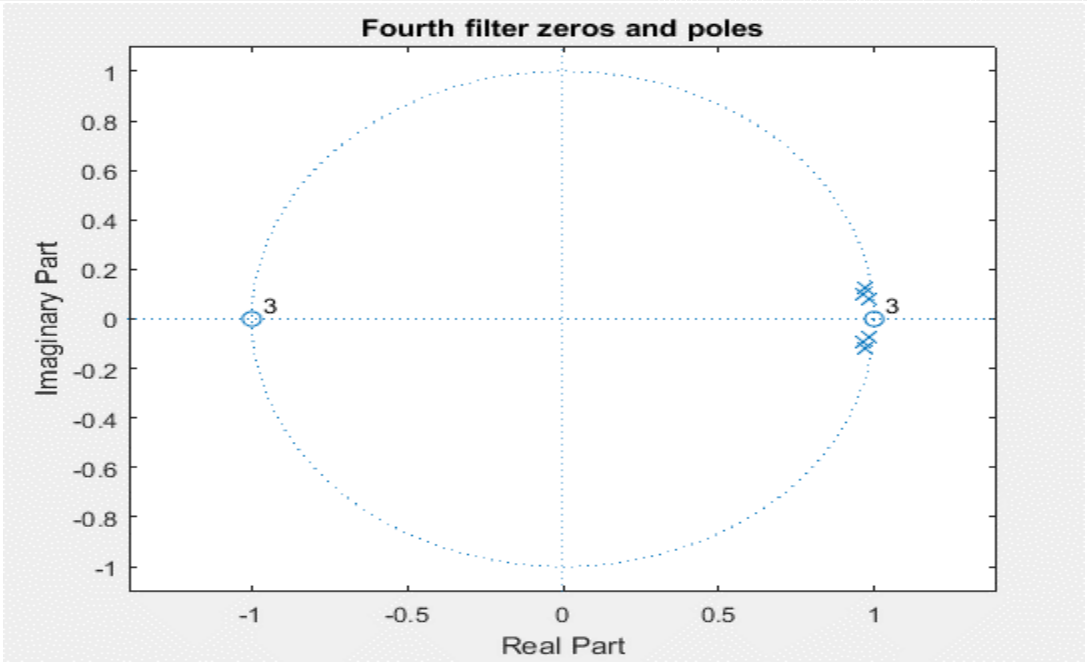

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^23
Amplitude
Time (seconds)
Impulse Response
×10^23
Amplitude
Time (seconds)

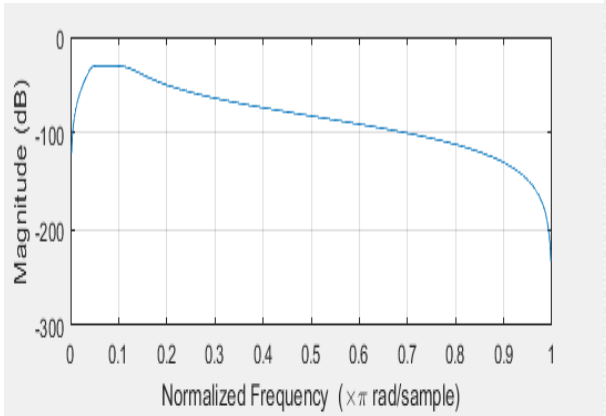

Second filter zeros and poles
Imaginary Part
Real Part
3
3

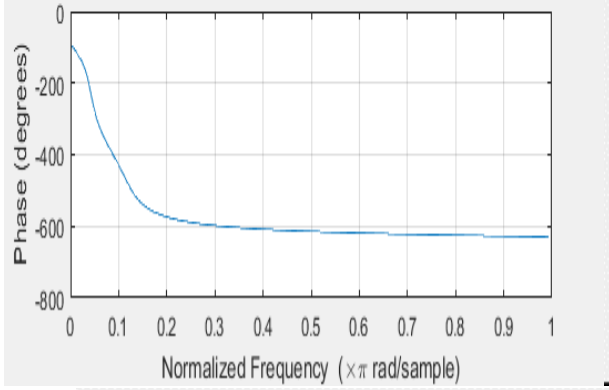

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^23
Amplitude
Time (seconds)
Impulse Response
×10^23
Amplitude
Time (seconds)

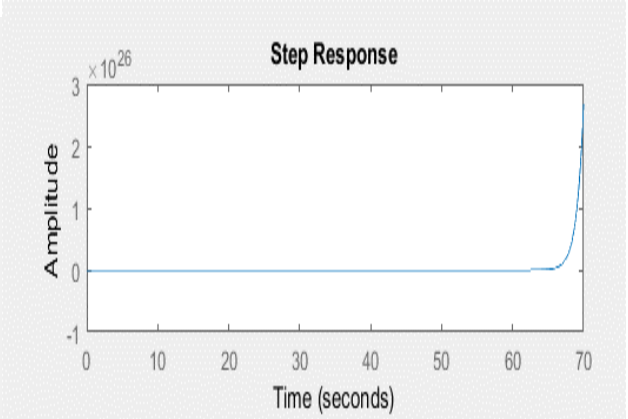

Third filter zeros and poles
Imaginary Part
Real Part
3
3

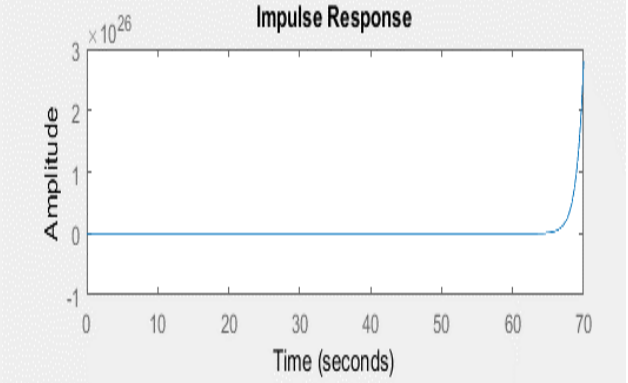

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^10
Amplitude
Time (seconds)
Impulse Response
×10^10
Amplitude
Time (seconds)

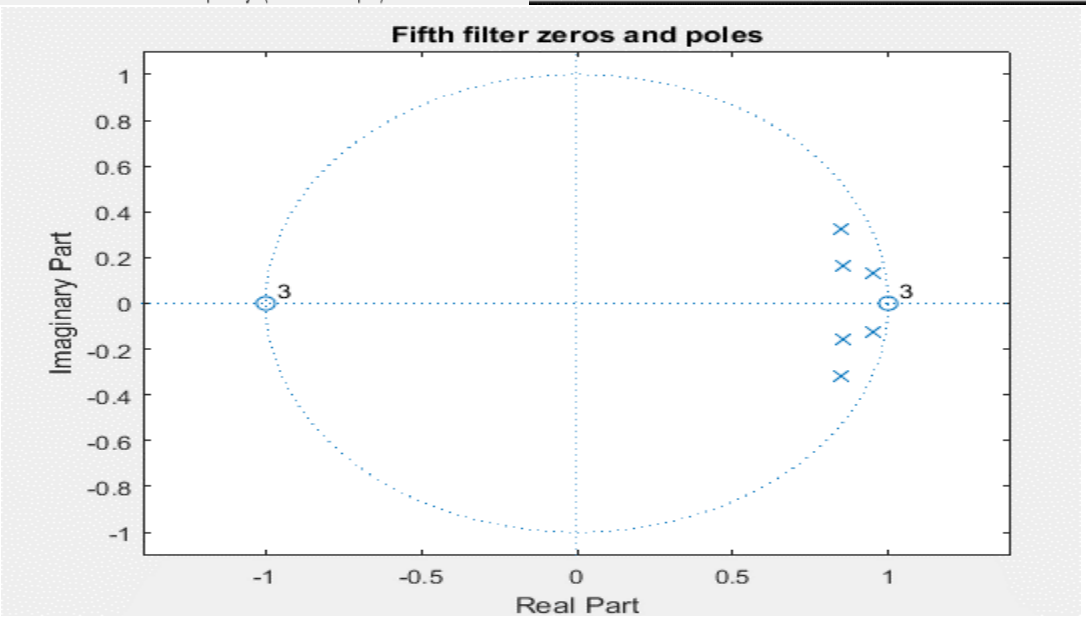

Fourth filter zeros and poles
Imaginary Part
Real Part
3
3

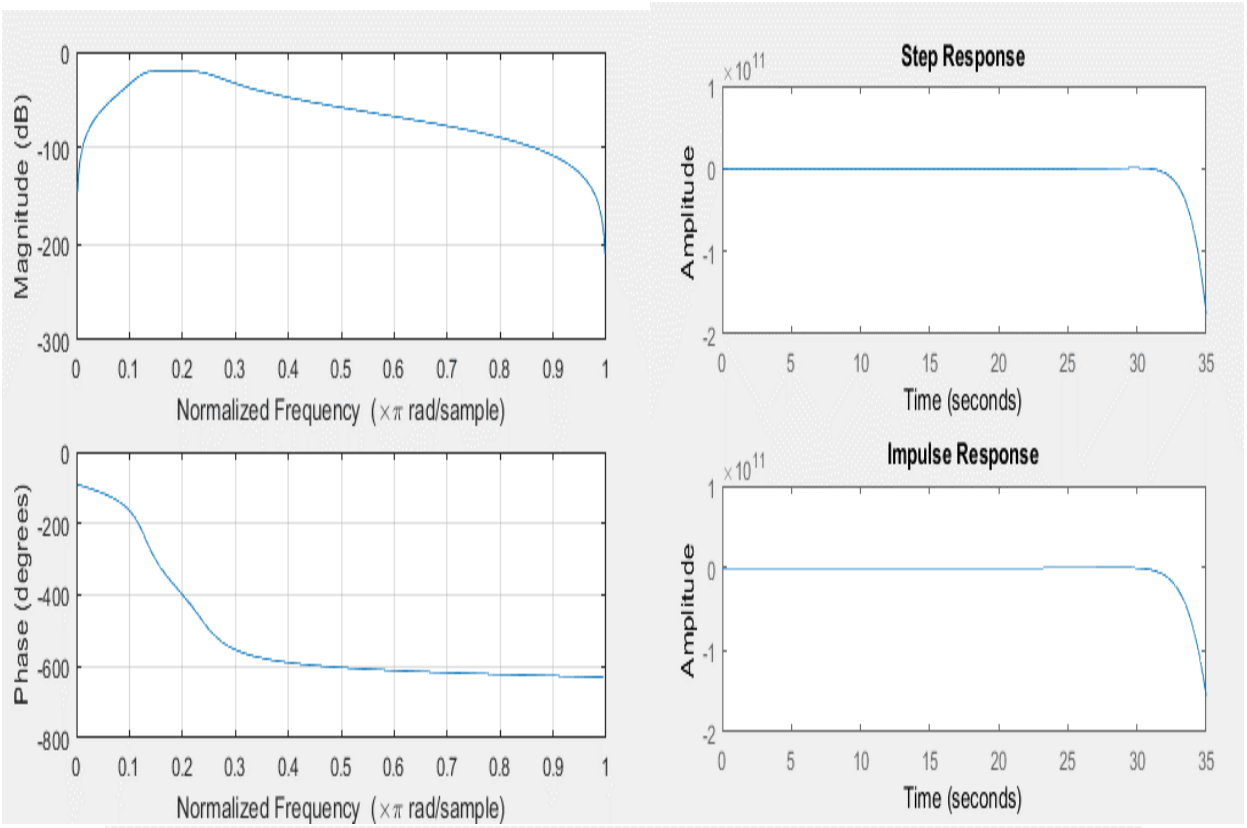

Magnitude (dB)
0
-100
-200
-300
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)

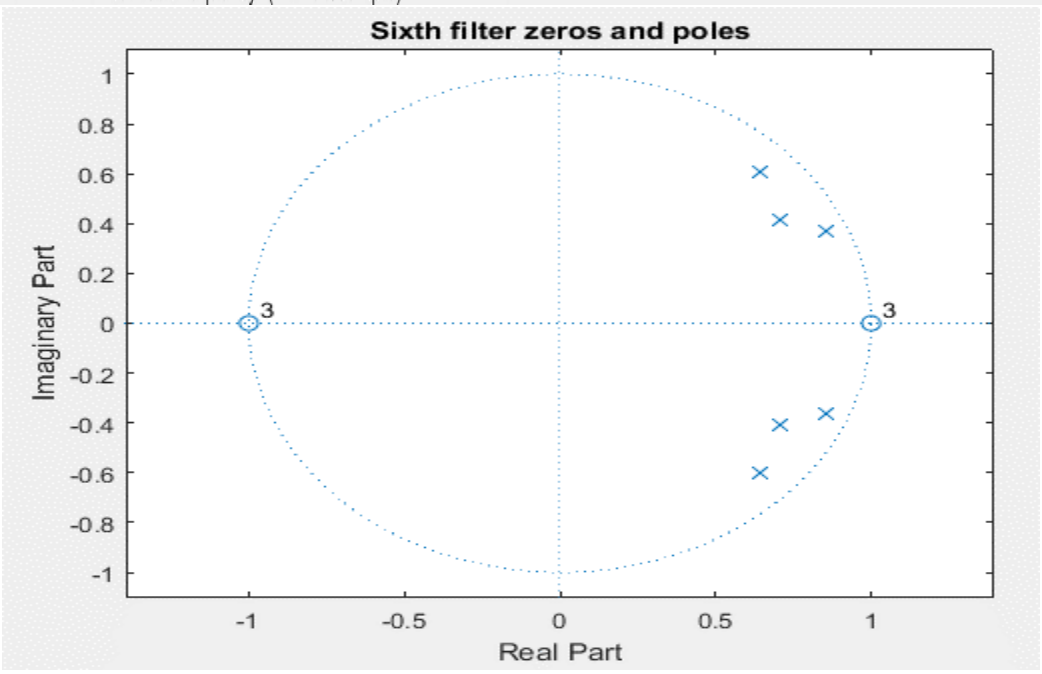

Phase (degrees)
0
-200
-400
-600
-800
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)

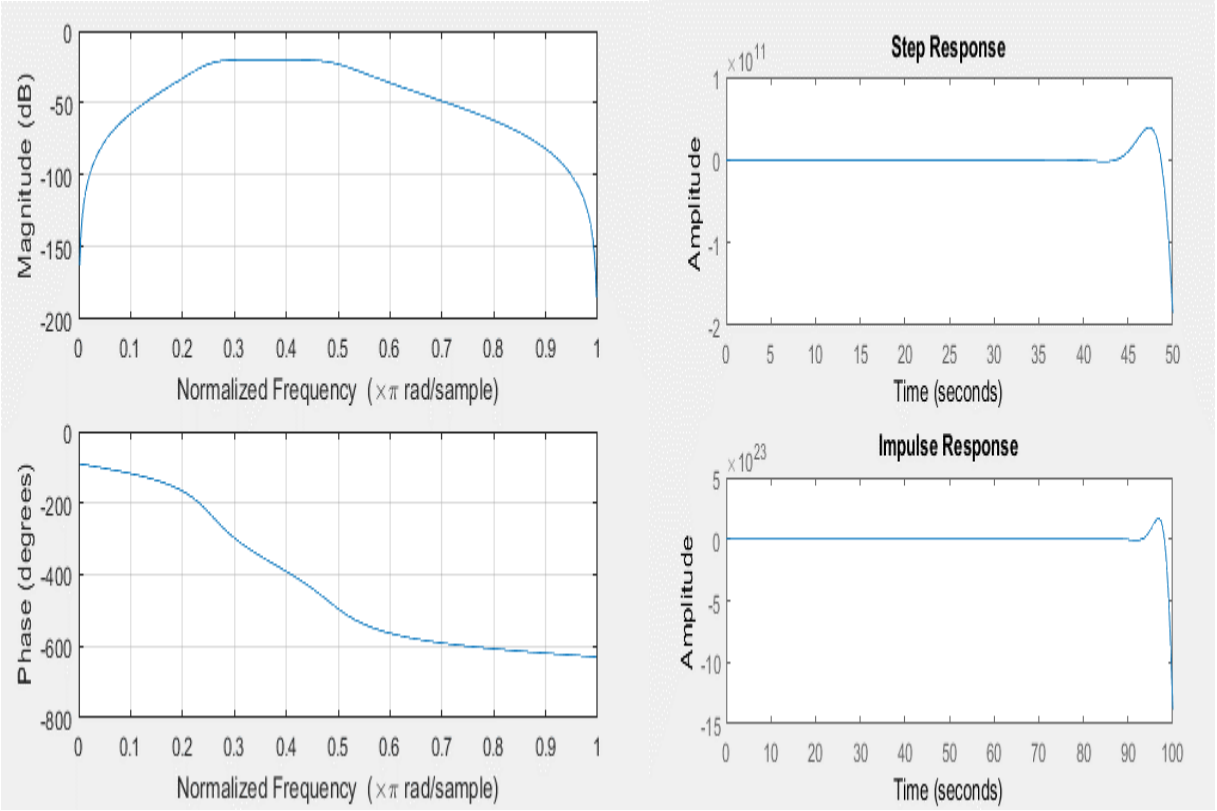

Step Response
×10^26
Amplitude
3
2
1
0
-1
0 10 20 30 40 50 60 70
Time (seconds)

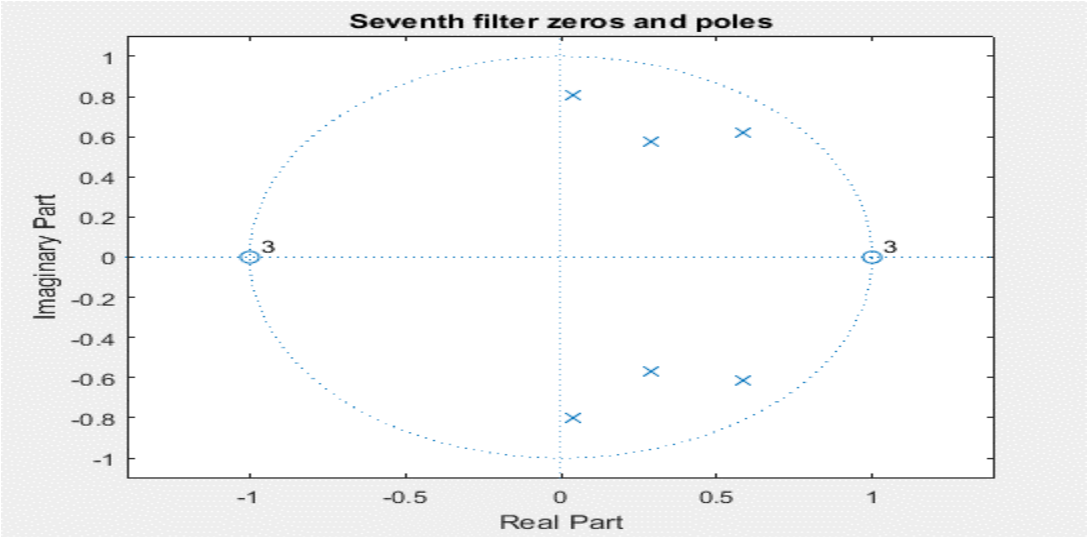

Impulse Response
×10^26
Amplitude
3
2
1
0
-1
0 10 20 30 40 50 60 70
Time (seconds)

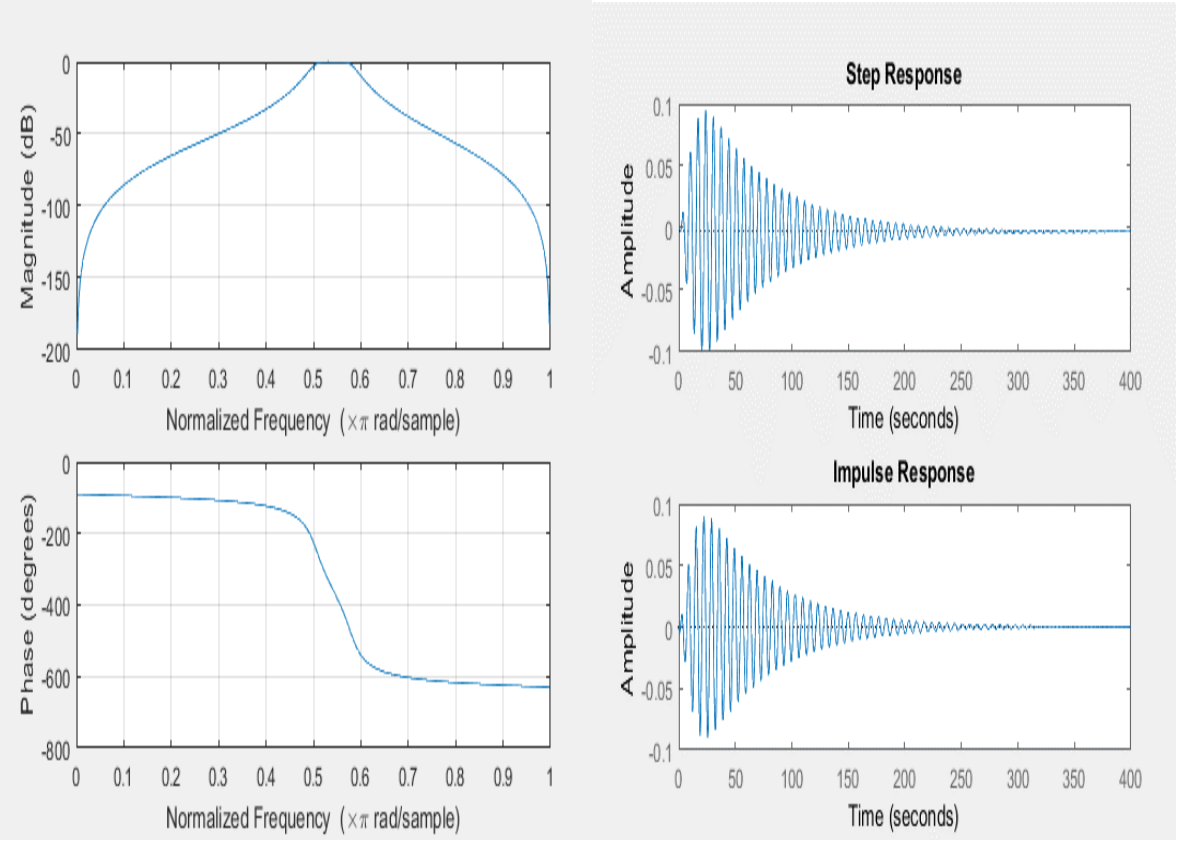

Fifth filter zeros and poles
Imaginary Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
3
3
-1 -0.5 0 0.5 1
Real Part

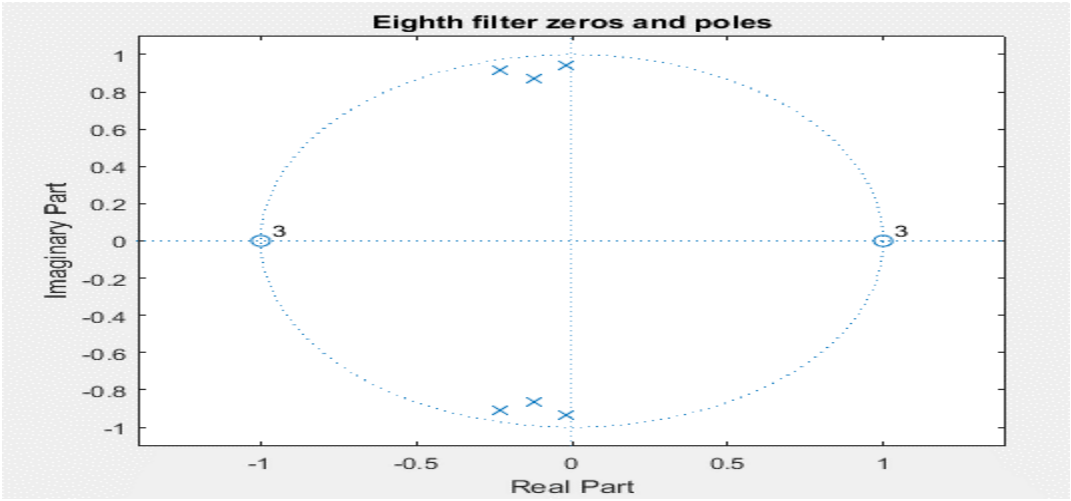

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
Amplitude
Time (seconds)
Impulse Response
Amplitude
Time (seconds)

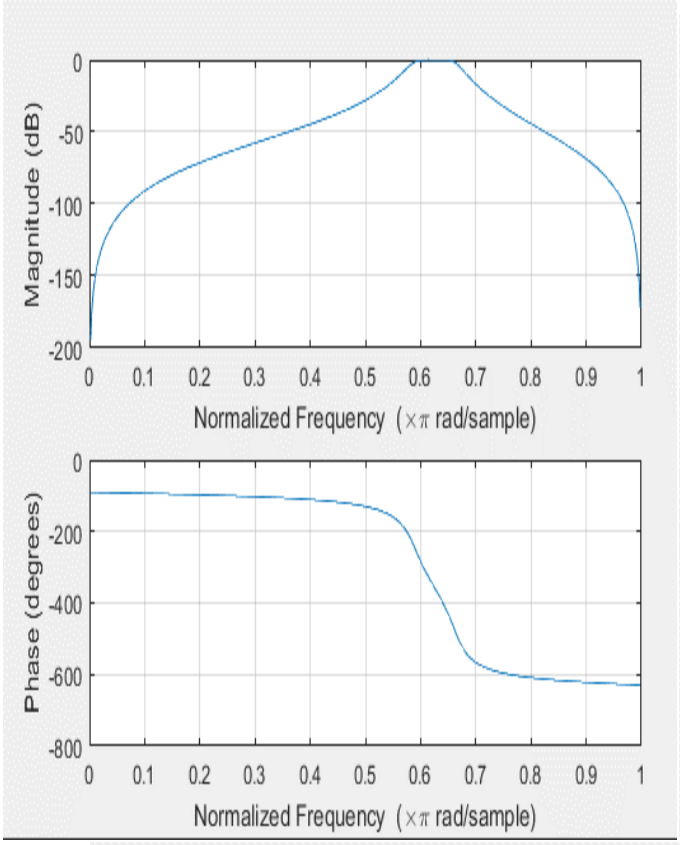

Sixth filter zeros and poles
Imaginary Part
Real Part
3
3

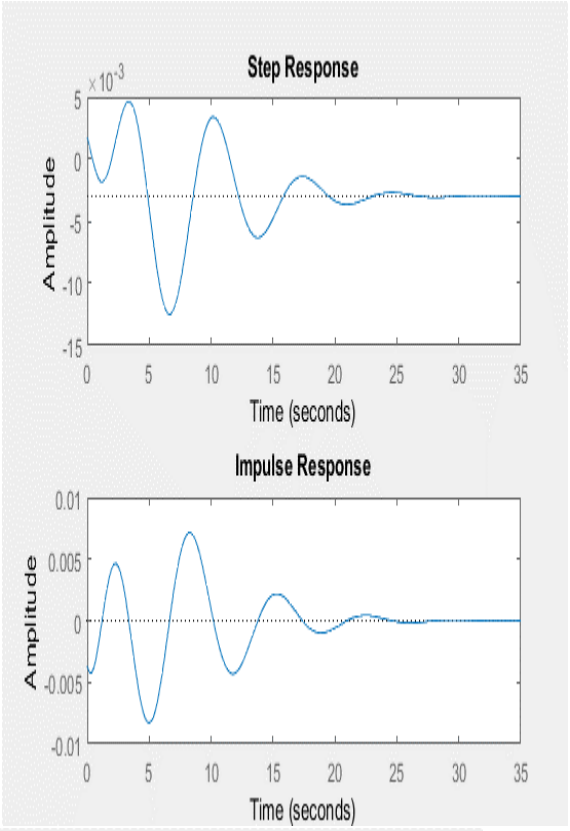

Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Step Response
×10^11
Amplitude
Time (seconds)
Impulse Response
×10^23
Amplitude
Time (seconds)

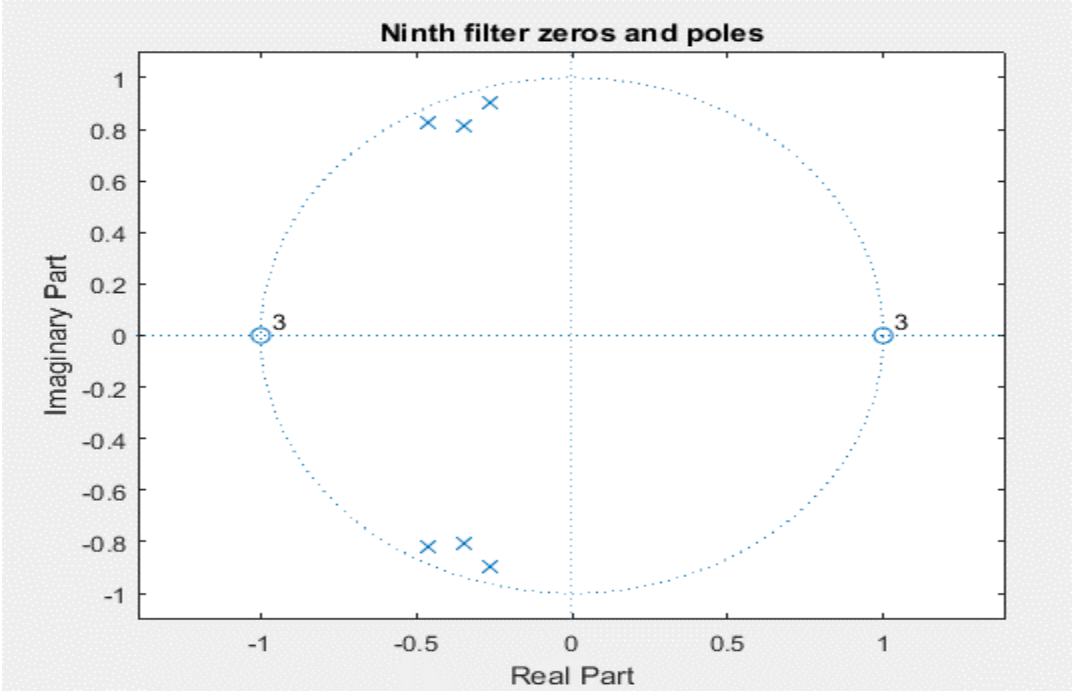

Seventh filter zeros and poles
Imaginary Part
Real Part
3
3


Magnitude (dB)
Normalized Frequency (×π rad/sample)
Step Response
Amplitude
Time (seconds)
Phase (degrees)
Normalized Frequency (×π rad/sample)
Impulse Response
Amplitude
Time (seconds)

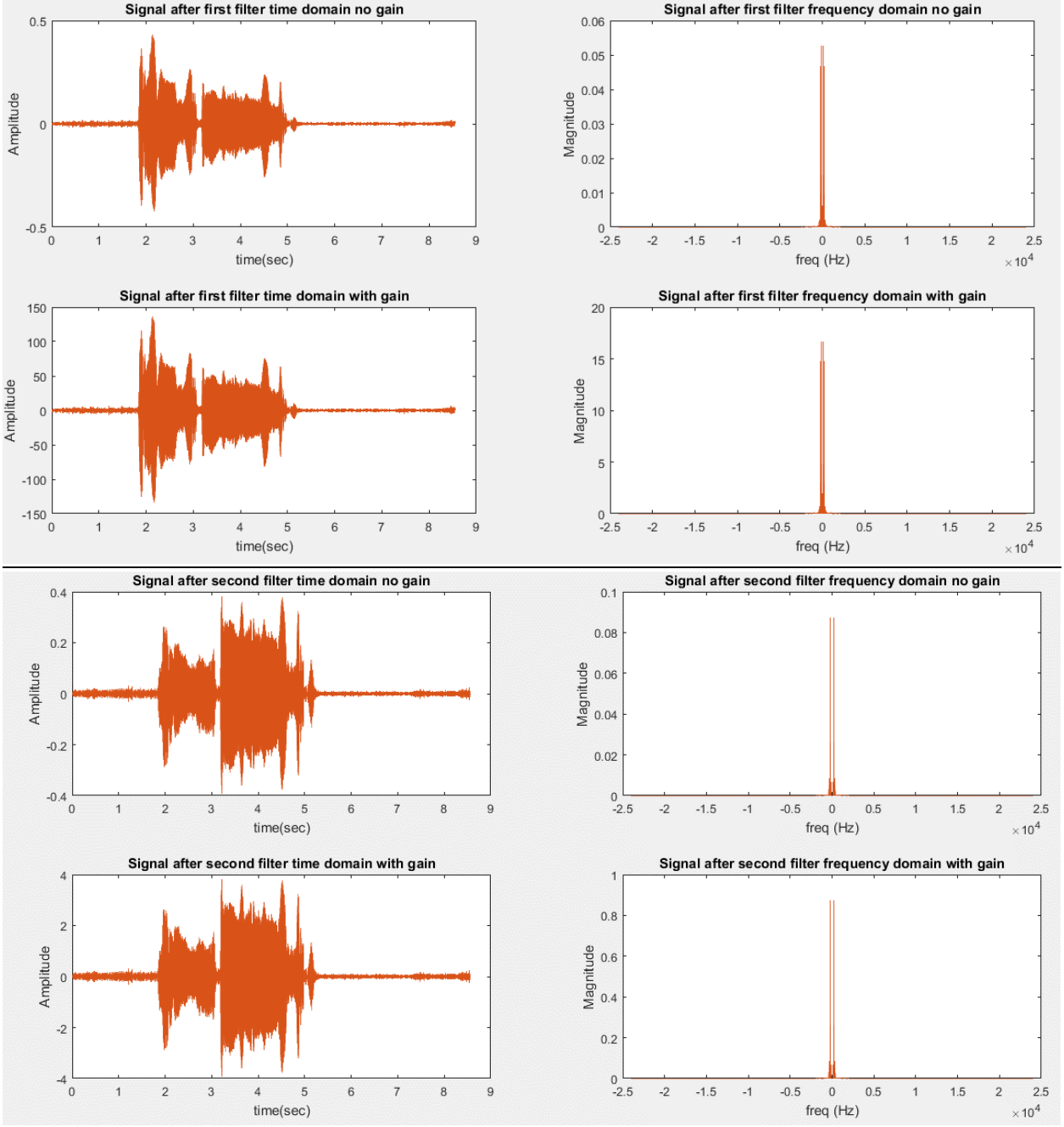

Eighth filter zeros and poles
Imaginary Part
Real Part
3
3

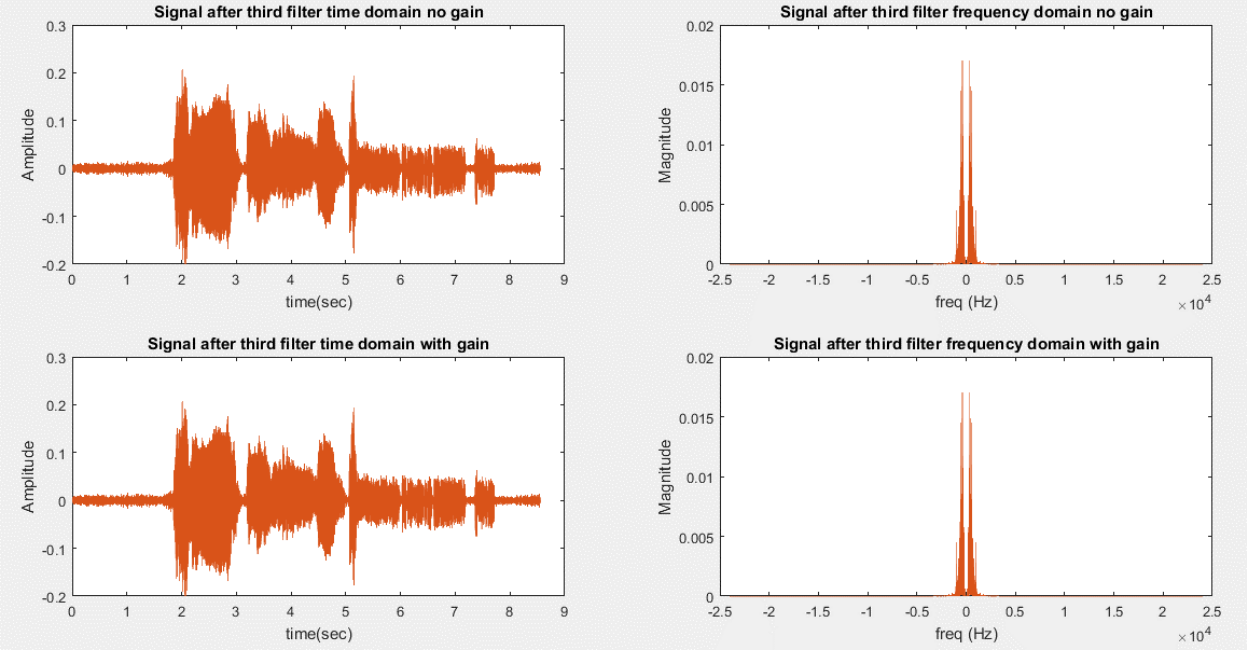

Magnitude (dB)
0
-50
-100
-150
-200
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)
Phase (degrees)
0
-200
-400
-600
-800
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Normalized Frequency (×π rad/sample)

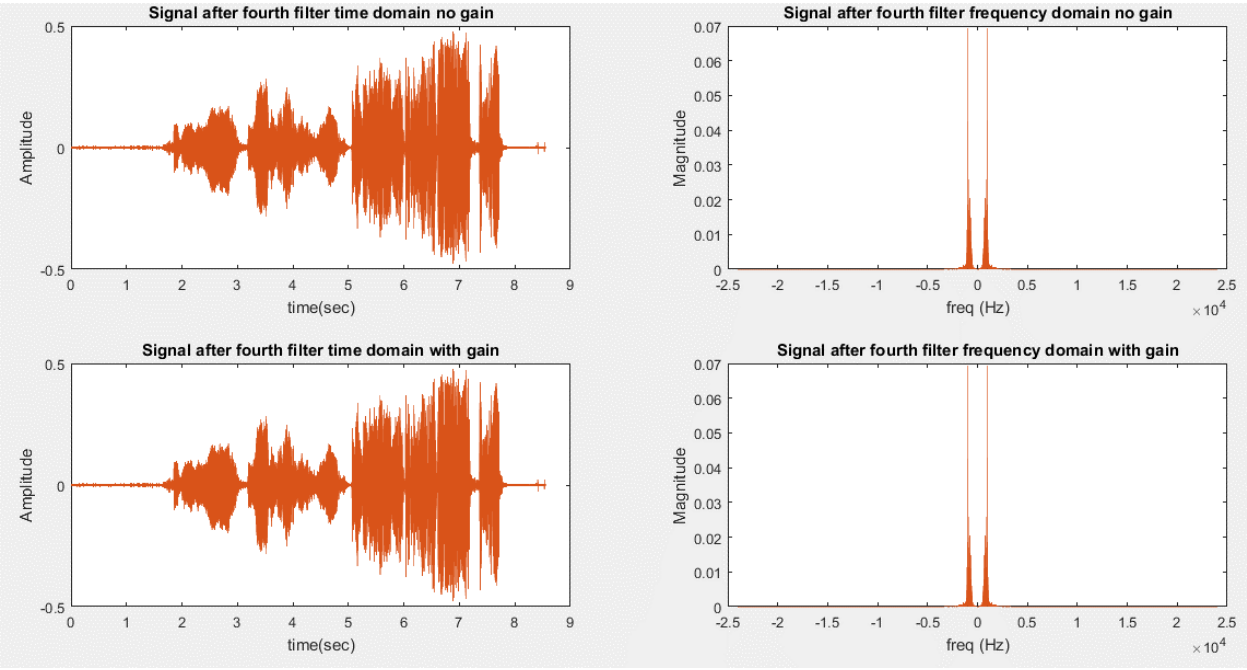

Step Response
×10^-3
5
0
-5
-10
-15
Amplitude
0 5 10 15 20 25 30 35
Time (seconds)
Impulse Response
0.01
0.005
0
-0.005
-0.01
Amplitude
0 5 10 15 20 25 30 35
Time (seconds)

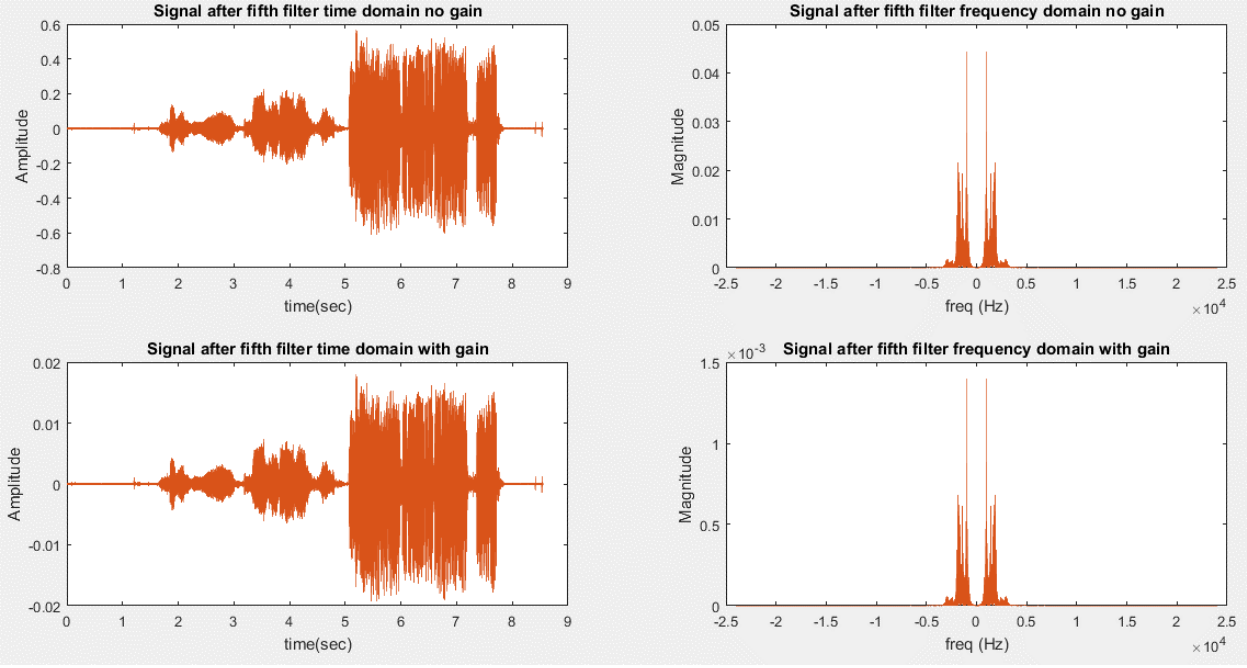

Ninth filter zeros and poles
Imaginary Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
3
3
-1 -0.5 0 0.5 1
Real Part

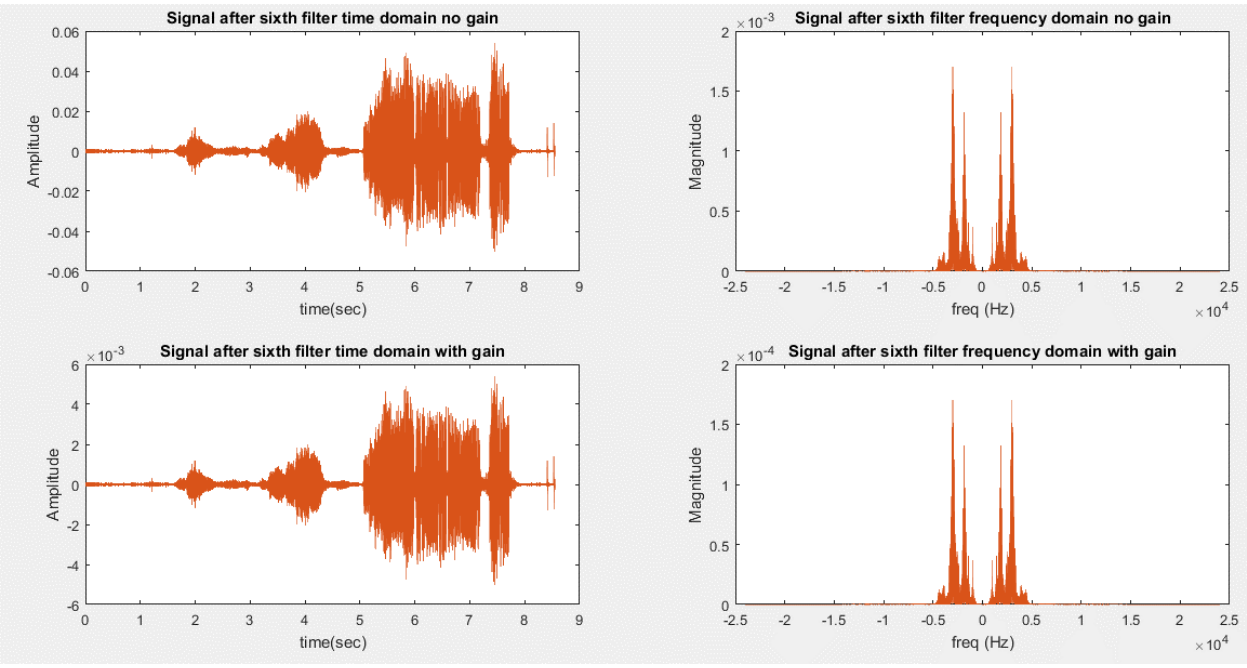
2-b-ii) All figures in time and frequency domain:

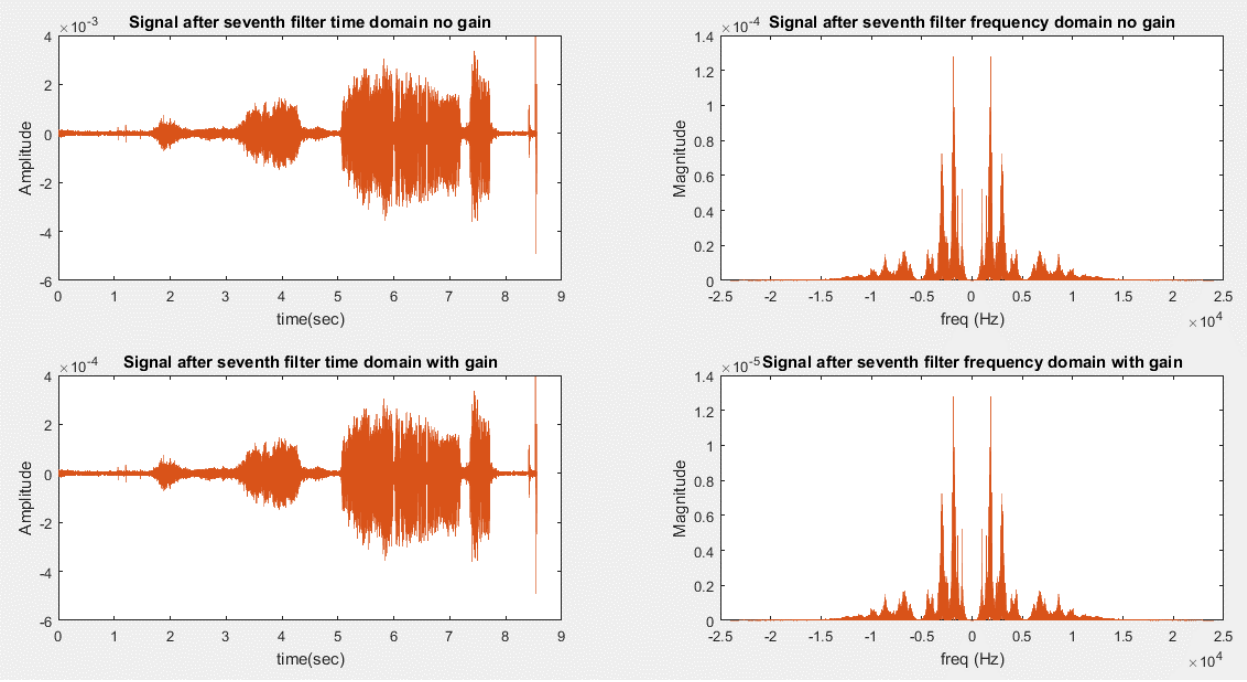


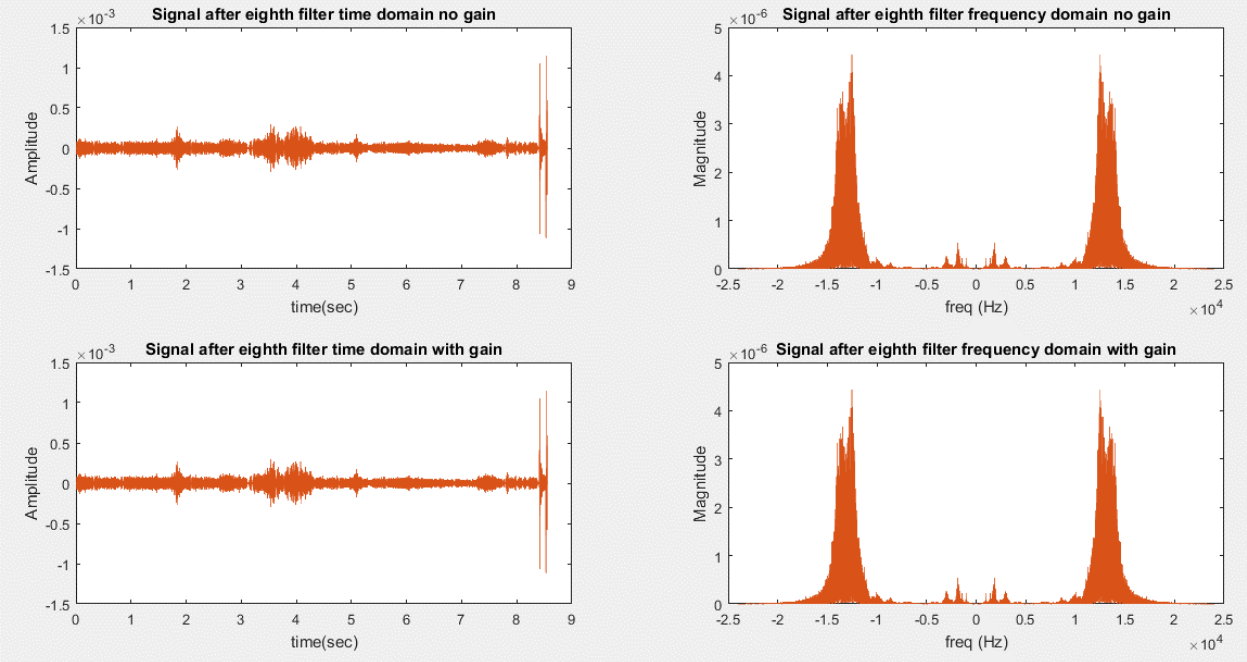

Signal after third filter time domain no gain
Amplitude
time(sec)
Signal after third filter frequency domain no gain
Magnitude
freq (Hz)
Signal after third filter time domain with gain
Amplitude
time(sec)
Signal after third filter frequency domain with gain
Magnitude
freq (Hz)

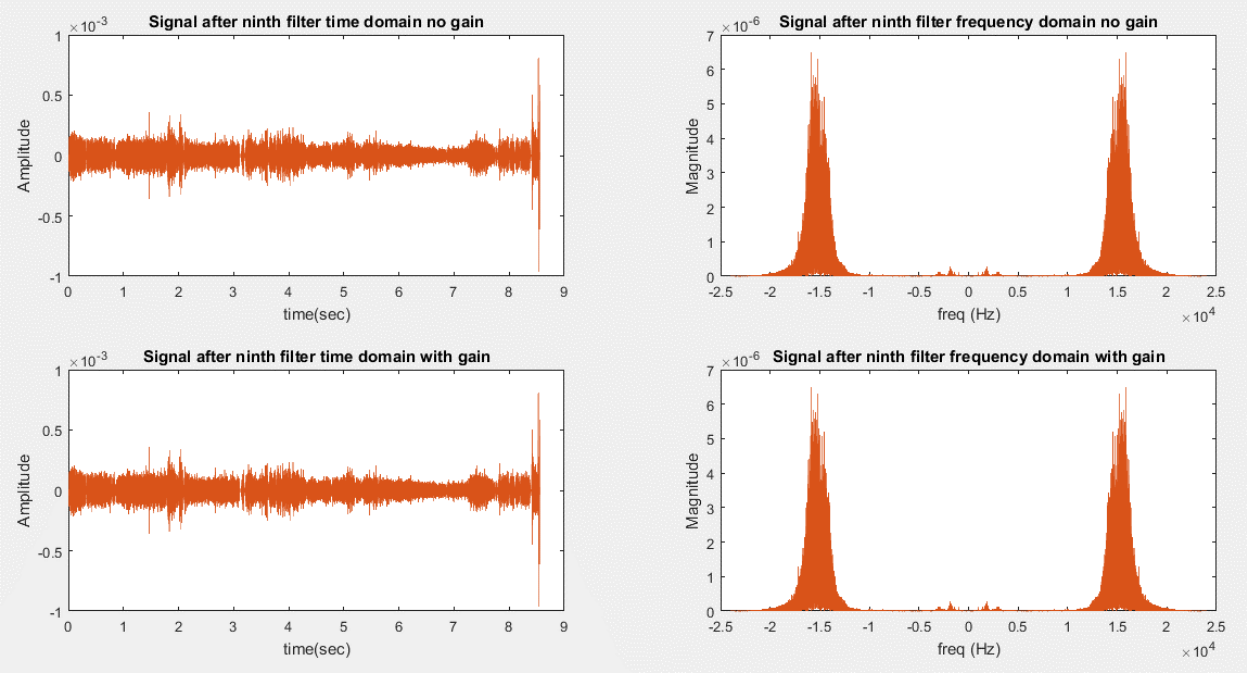

Signal after fourth filter time domain no gain
Amplitude
time(sec)
Signal after fourth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after fourth filter time domain with gain
Amplitude
time(sec)
Signal after fourth filter frequency domain with gain
Magnitude
freq (Hz)

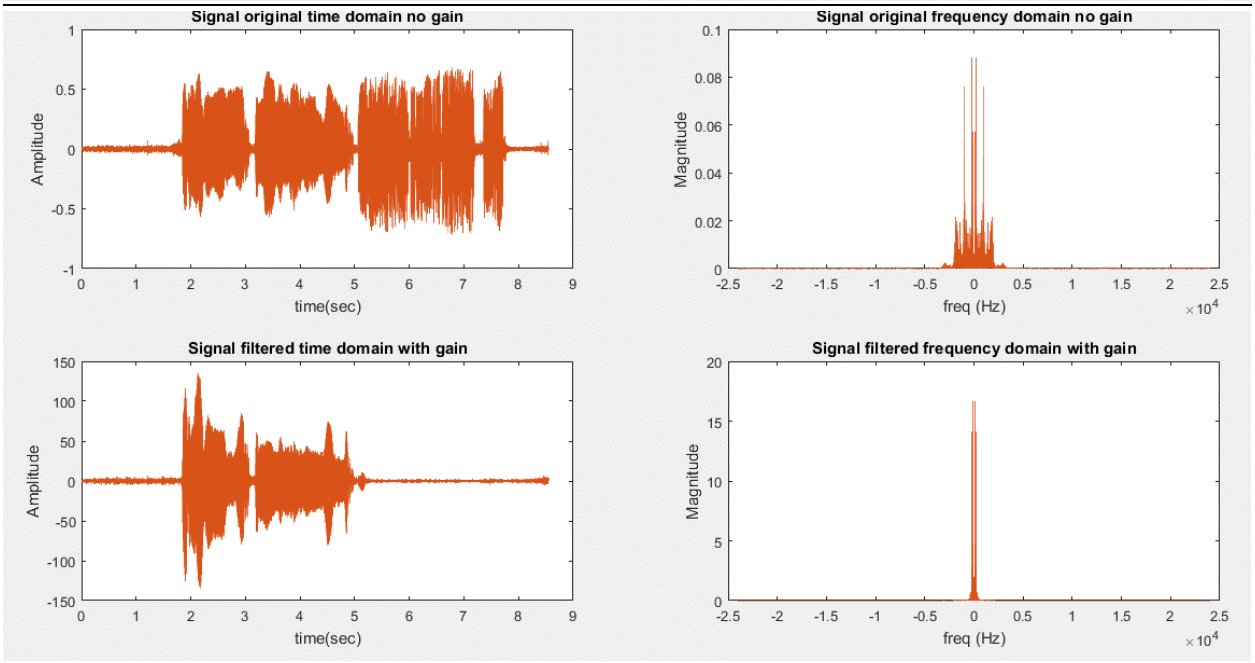

Signal after fifth filter time domain no gain
Amplitude
time(sec)
Signal after fifth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after fifth filter time domain with gain
Amplitude
time(sec)
Signal after fifth filter frequency domain with gain
Magnitude
freq (Hz)


Signal after sixth filter time domain no gain
Amplitude
time(sec)
Signal after sixth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after sixth filter time domain with gain
Amplitude
time(sec)
Signal after sixth filter frequency domain with gain
Magnitude
freq (Hz)

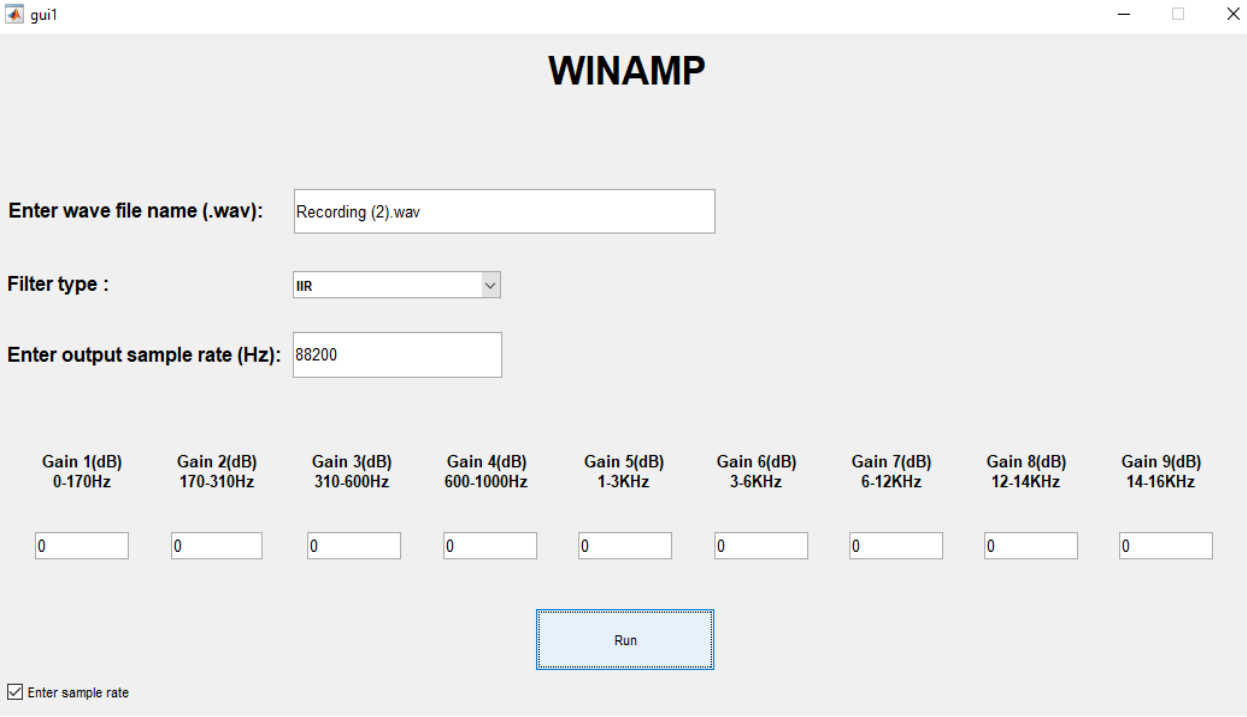

Signal after seventh filter time domain no gain
Amplitude
time(sec)
Signal after seventh filter frequency domain no gain
Magnitude
freq (Hz)
Signal after seventh filter time domain with gain
Amplitude
time(sec)
Signal after seventh filter frequency domain with gain
Magnitude
freq (Hz)


Signal after eighth filter time domain no gain
Amplitude
time(sec)
Signal after eighth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after eighth filter time domain with gain
Amplitude
time(sec)
Signal after eighth filter frequency domain with gain
Magnitude
freq (Hz)

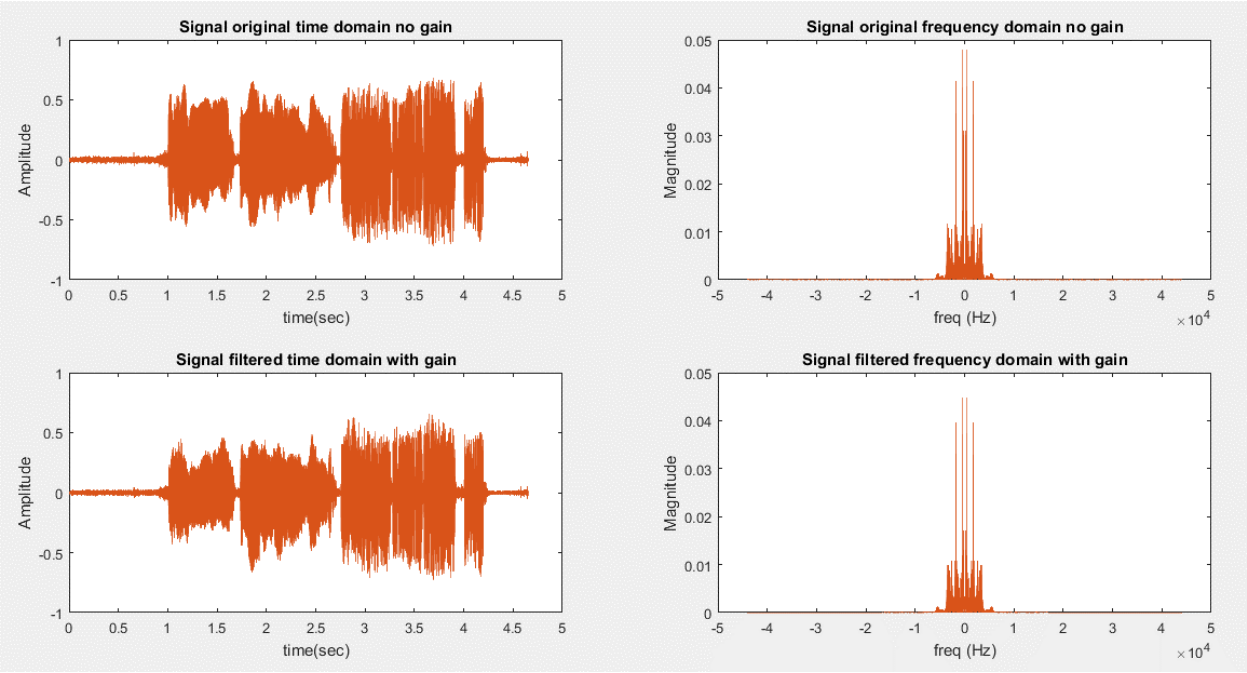

Signal after ninth filter time domain no gain
Amplitude
time(sec)
Signal after ninth filter frequency domain no gain
Magnitude
freq (Hz)
Signal after ninth filter time domain with gain
Amplitude
time(sec)
Signal after ninth filter frequency domain with gain
Magnitude
freq (Hz)


Signal original time domain no gain
Amplitude
time(sec)
Signal original frequency domain no gain
Magnitude
freq (Hz)
Signal filtered time domain with gain
Amplitude
time(sec)
Signal filtered frequency domain with gain
Magnitude
freq (Hz)

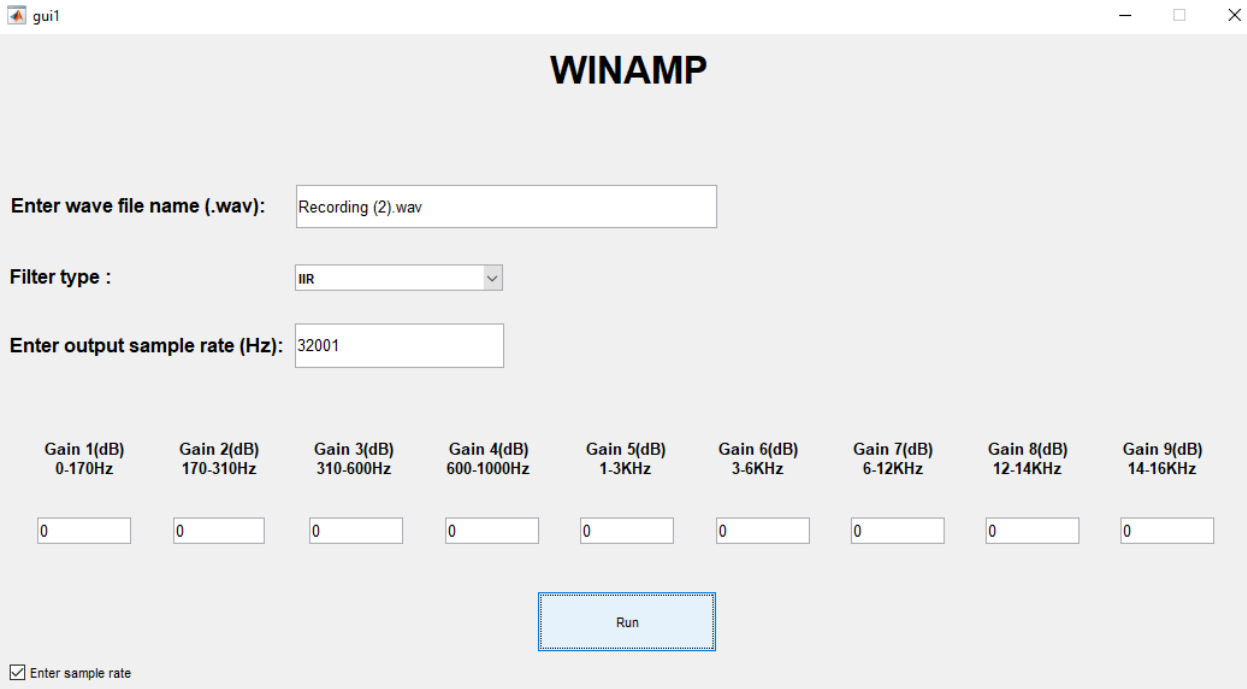
2-c) IIR with file double sample rate:



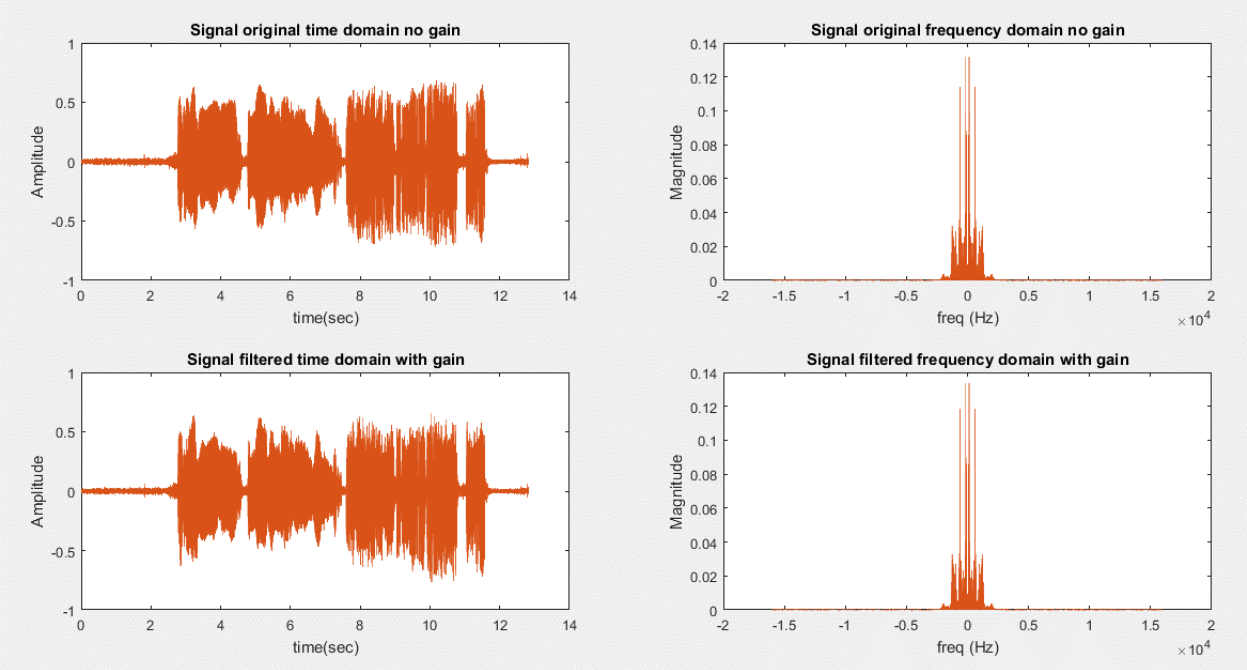
2-c-i) Final figure in time and frequency domain:



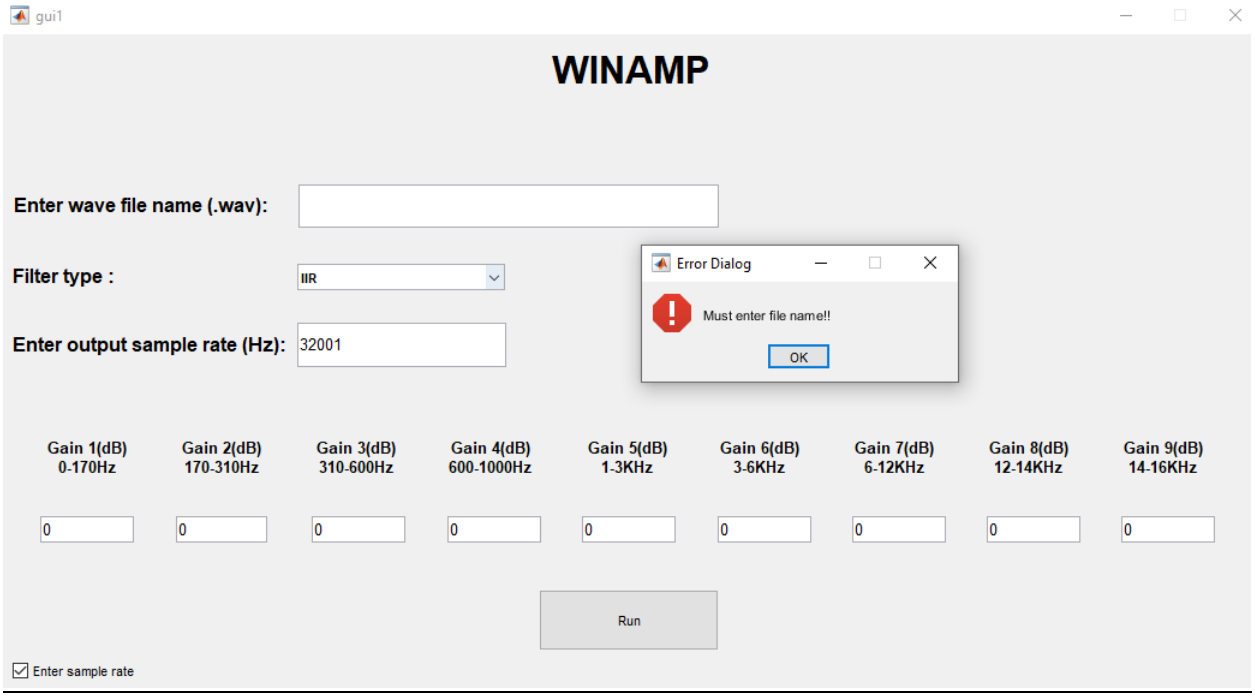
2-d) IIR with file half sample rate:

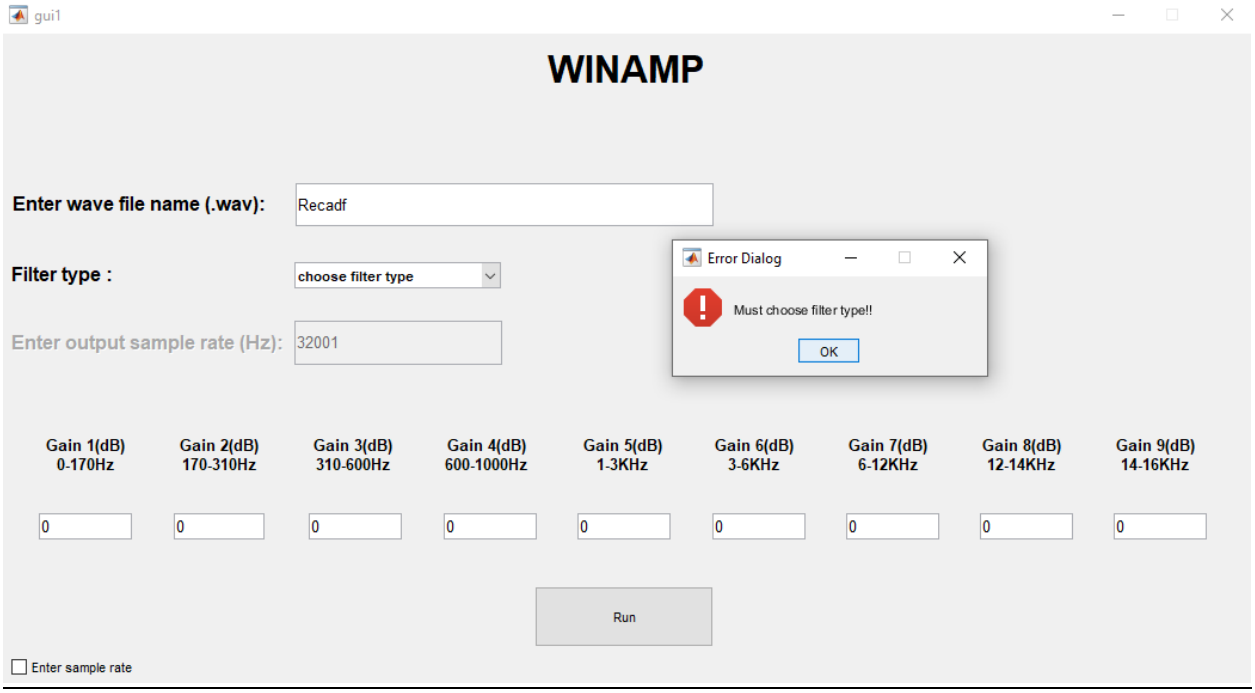


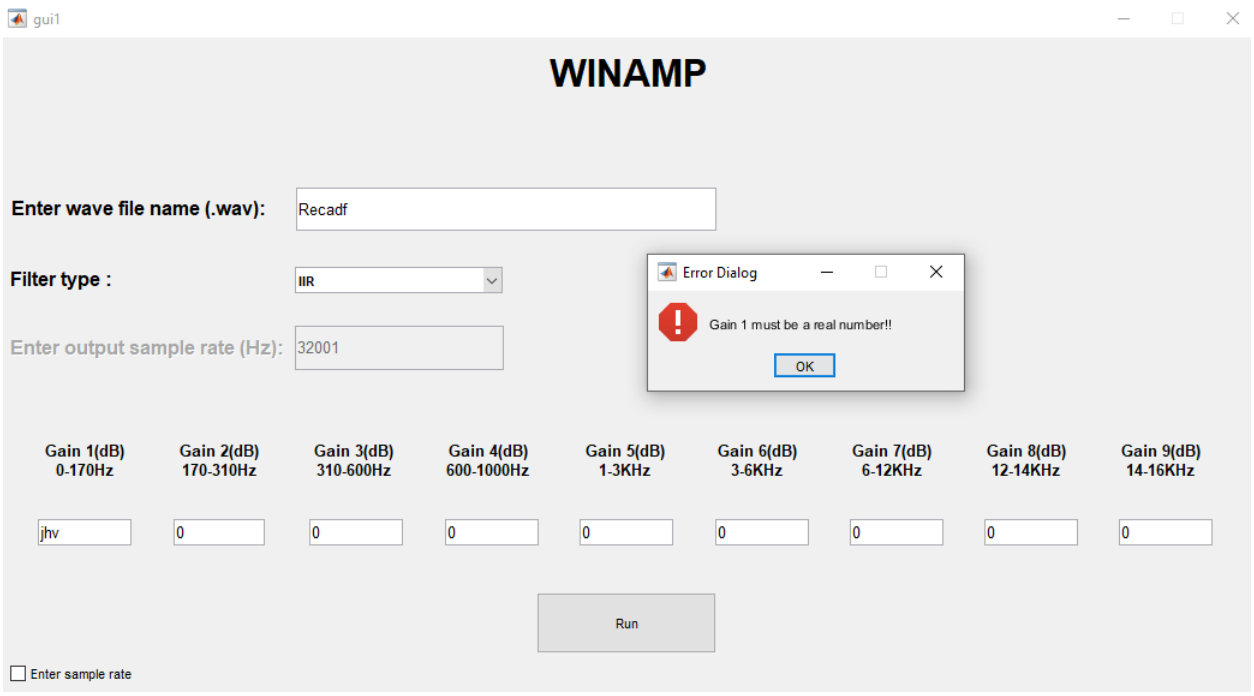
2-d-i) Final figure in time and frequency domain:

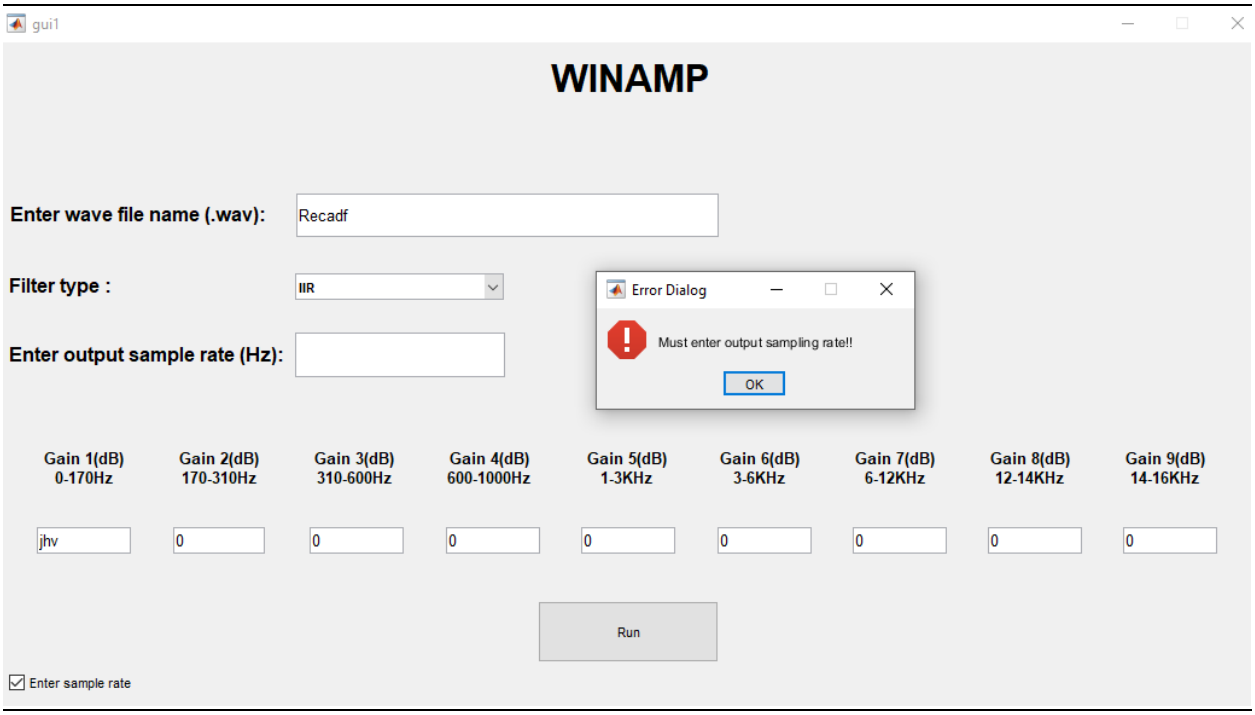


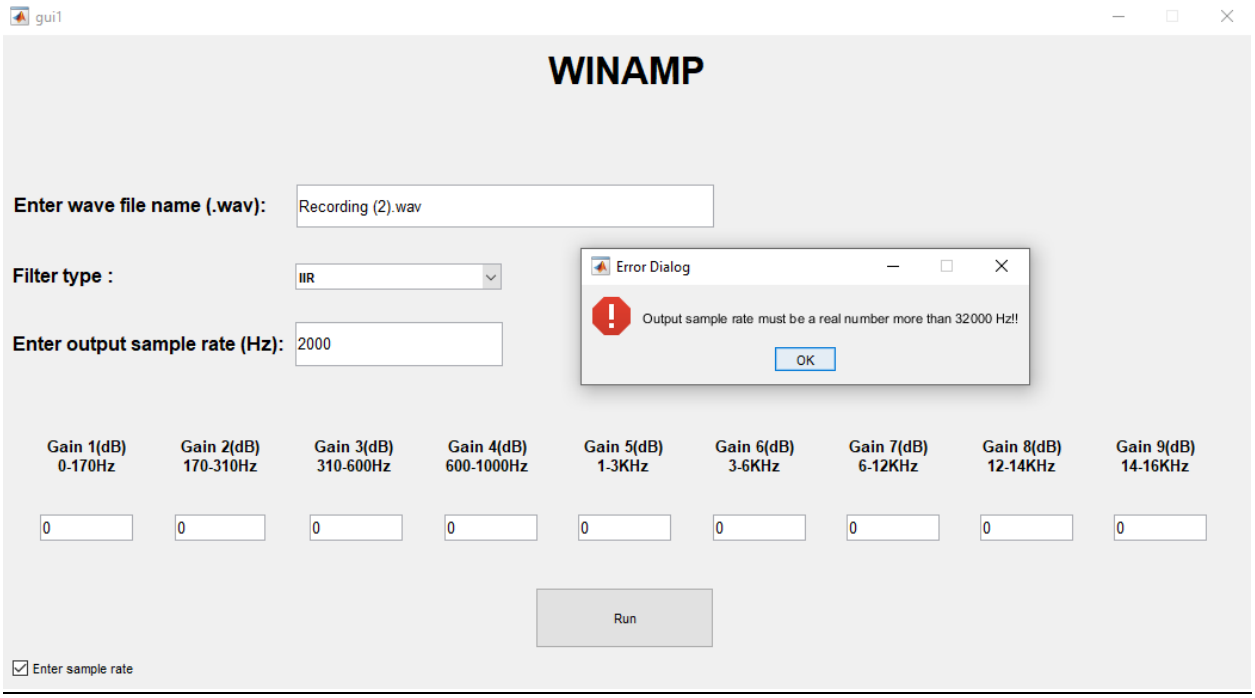
3) Error handling:

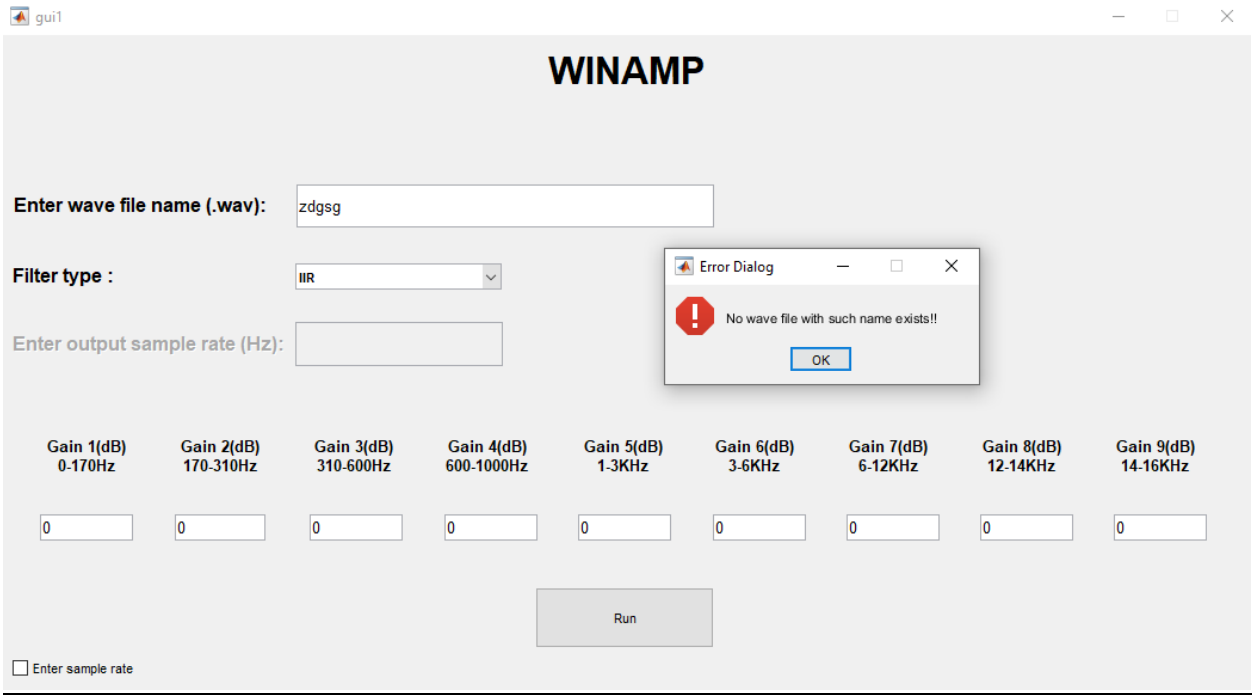





gui1
WINAMP
Enter wave file name (.wav):
Recadf
Filter type :
IIR
Enter output sample rate (Hz):
32001
Error Dialog
Gain 1 must be a real number!!
OK
Gain 1(dB) 0-170Hz
Gain 2(dB) 170-310Hz
Gain 3(dB) 310-600Hz
Gain 4(dB) 600-1000Hz
Gain 5(dB) 1-3KHz
Gain 6(dB) 3-6KHz
Gain 7(dB) 6-12KHz
Gain 8(dB) 12-14KHz
Gain 9(dB) 14-16KHz
jhv
0
Run
Enter sample rate


gui1
WINAMP
Enter wave file name (.wav):
Recadf
Filter type :
IIR
Enter output sample rate (Hz):
Error Dialog
Must enter output sampling rate!!
OK
Gain 1(dB) 0-170Hz
Gain 2(dB) 170-310Hz
Gain 3(dB) 310-600Hz
Gain 4(dB) 600-1000Hz
Gain 5(dB) 1-3KHz
Gain 6(dB) 3-6KHz
Gain 7(dB) 6-12KHz
Gain 8(dB) 12-14KHz
Gain 9(dB) 14-16KHz
jhv
0
Run
Enter sample rate


gui1
WINAMP
Enter wave file name (.wav):
Recording (2).wav
Filter type :
IIR
Enter output sample rate (Hz):
2000
Error Dialog
Output sample rate must be a real number more than 32000 Hz!!
OK
Gain 1(dB) 0-170Hz
Gain 2(dB) 170-310Hz
Gain 3(dB) 310-600Hz
Gain 4(dB) 600-1000Hz
Gain 5(dB) 1-3KHz
Gain 6(dB) 3-6KHz
Gain 7(dB) 6-12KHz
Gain 8(dB) 12-14KHz
Gain 9(dB) 14-16KHz
0
Run
Enter sample rate


gui1
WINAMP
Enter wave file name (.wav):
zdgsg
Filter type :
IIR
Enter output sample rate (Hz):
Error Dialog
No wave file with such name exists!!
OK
Gain 1(dB) 0-170Hz
Gain 2(dB) 170-310Hz
Gain 3(dB) 310-600Hz
Gain 4(dB) 600-1000Hz
Gain 5(dB) 1-3KHz
Gain 6(dB) 3-6KHz
Gain 7(dB) 6-12KHz
Gain 8(dB) 12-14KHz
Gain 9(dB) 14-16KHz
0
Run
Enter sample rate