DSP
Project #1 , 2

Hagar Osama

4970

Introduction

This program is designed to plot certain signals in both time domain and frequency domain. The objective of this program is to be familiar with frequency domain and fourier transform. The Frequency domain is widely used in signals analysis for two main reasons : first, calculations are often easier. Secondly, some properties of signals are clearer in frequency domain than in time domain.

The project is now spread to include LTI channel and sound processing

Code Description

The code is implemented in MATLAB and has 6 main parts:

1. welcome.m (function)
2. select_signal.m (function)
3. Signals_project_1 (the main file / interface)
4. exp_4.m
5. DSP_project2.m
6. DSP_project2_2.m

Part I

welcome.m is a simple function which welcomes the user and displays the instructions. This function is void.

Part II

- Select_signal.m is a core function that returns Xs and Ys (ts and y(t)s) to each signal to be plotted. This function requires four inputs :

- Input1 : is the choice of user (Impulse - DC Signal - Ramp - Exponential - Sinusoidal wave) represented by numbers from 1 : 5 , any numbers are neglected as an invalid input.
- start_point: the point a subplot starts with.
- breakpoint: the point a subplot end with.
- Sample_freq: the reciprocal of numbers of samples in 1 second
- The return value is two vectors one represents the t axis which I call here new_x and the other represents the y(t) which I call new_y.

Part III

- Signals_project_1.m is the interface of the program. Here the user is asked to give the sampling frequency , the start point , the end point , the break-points and their positions

    - Two vectors are initialized with empty size y and t to hold the entire value of each plot together using concatenation for both m(t) and h(t).
    - A loop is held to ask for the position of break-points to create a vector bk
    - Another loop is held to execute the function select_signal for each subplot
    - After configuring the t and y vectors, we are ready to plot in both time and frequency domain :
        - The plotting in time domain is straightforward
        - The plotting in frequency domain requires fft function. The plot would start from zero so we have to shift the transform using fftshift first this plot the graph.
- The program has the ability to add noise and asks the user to enter a ratio (if ‘zero’ then no noise)
    - The program convolutes m(t) * h(t) using conv function and plots the convolution

- Then m(t) is obtained again from the convolution using deconv and noise might be noticed there (see samples)
- Frequency response (mag) is plotted for both m(t) and h(t)

- Experiment 3 is held in this file :
    - To get the difference equation, the program asks the user to enter both numerator and denominator of the transfer function (that should be H(z) =

$$\frac{1}{1 + 0.5 z^{-1} + 0.5 z^{-2}}$$

    - The program then uses 'filter' function with given system to filterize the signal
    - To Reverse the output system we filterize it again with the ricoprical of H(z)
    - The time and frequency response of the output system is plotted
    - The frequency response and impulse response are plotted for the original system and the recover system
    - **[Bonus]** To Check stability : 2 methods are used :
        - The first is function 'tf2latc' that returns k which indicates the stability of the system
        - The second is pzmap which plots poles and zeros on unit circle. If all zeros are inside the circle, then the system is stable

Part IV

- A sound is read in MATLAB using function 'audioread'
- The sound signal is plotted in both time and frequency domain
- Then the input is filtered using T.F $H(Z) = 1 + Z^{-1}$ using filter function
- The sound is save using 'audiowrite' and played using 'sound' function

- **[Bonus]** To Filter the input sound at 4 KHz, Butterworth filter is used with order 7 and cutoff frequency 4000 Hz and the output is plotted in both time and frequency domain

Part V

- Dual tone signal : is two composed signal of sines or cosines
    - Two sinusoidal signals are generated with frequencies 3 KHz and 5 KHz and composed together with sampling frequency 16 KHz
    - Signal is plotted in both time and frequency domain
    - A LPF is applied with the help of Butterworth filter order 3 and the output is plotted (also spectgram)
- A voice signal is read using audioread function
    - The signal is filtered as the previous signal and plotted in both time and frequency domain

Part VI

- A triple tone sinusoidal signal is generated the same way as in Part V
    - It must be noticed that the filter given here is FIR filter
    - A LPF is applied using the given coefficients
    - The output of the filter is plotted as well as the spectrum of both input and output

Samples :

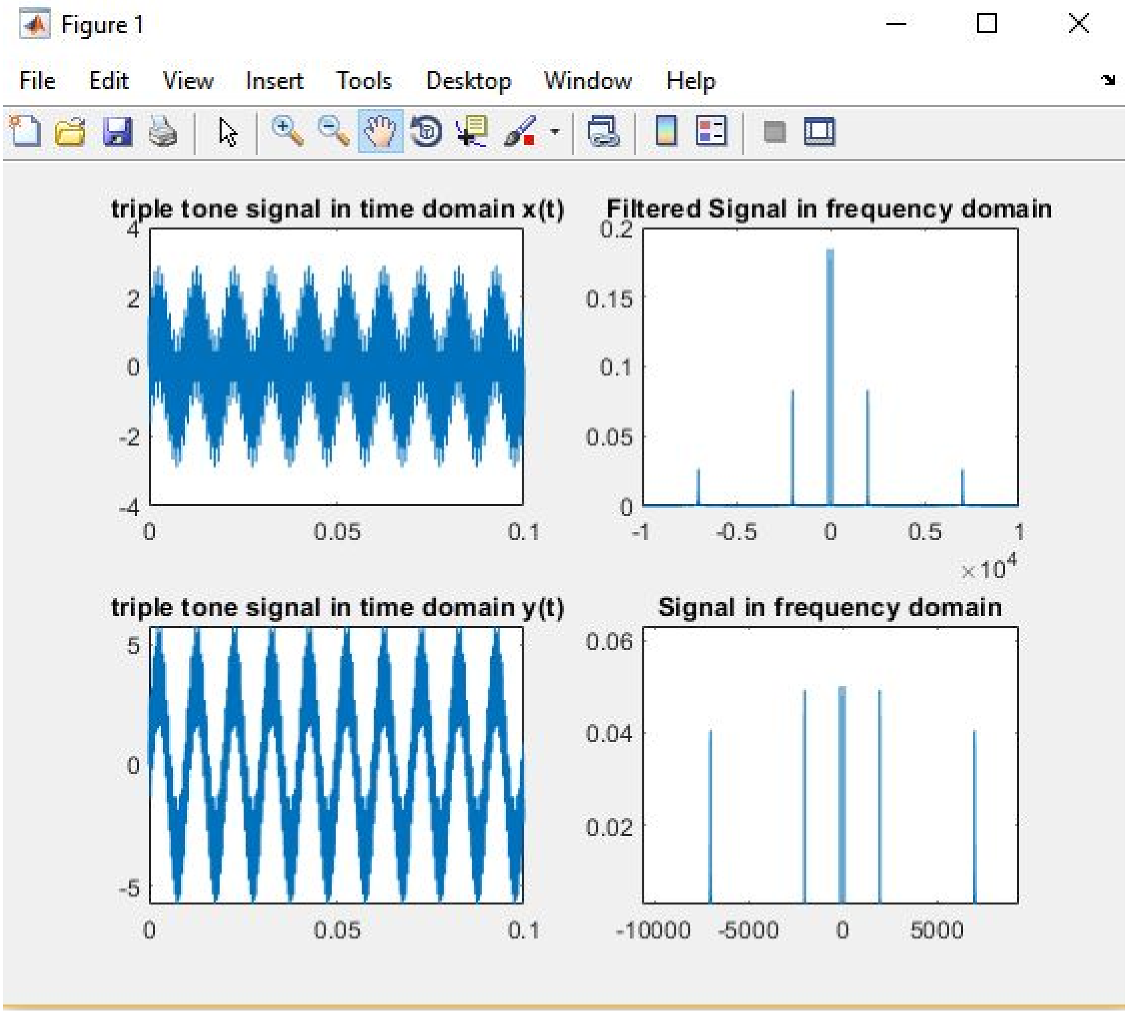


Figure 1
File Edit View Insert Tools Desktop Window Help
triple tone signal in time domain x(t)
Filtered Signal in frequency domain
triple tone signal in time domain y(t)
Signal in frequency domain

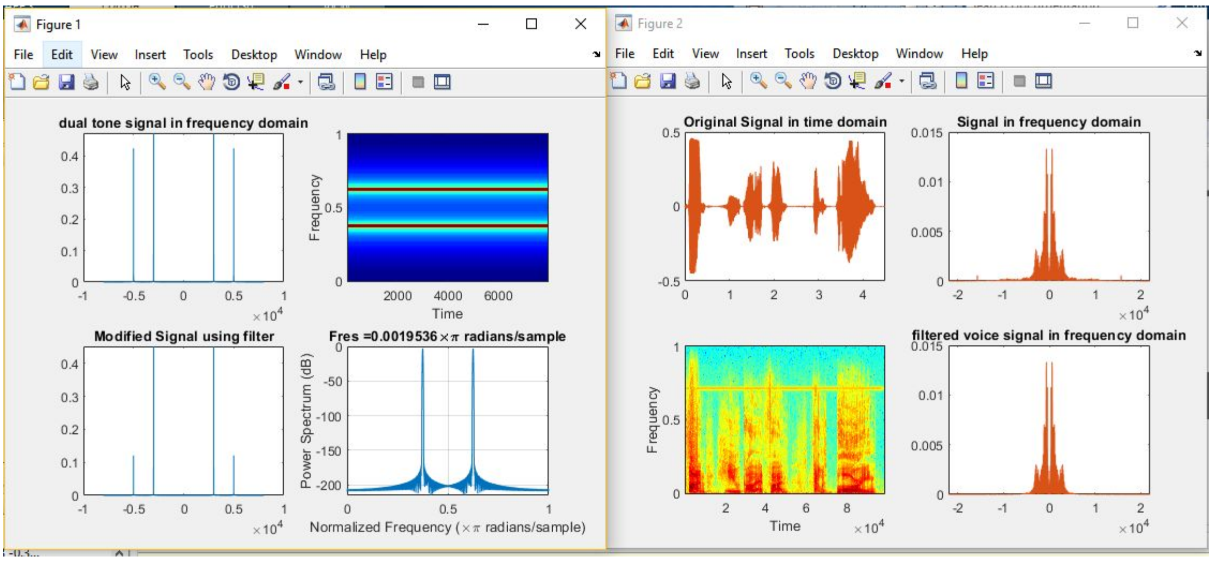
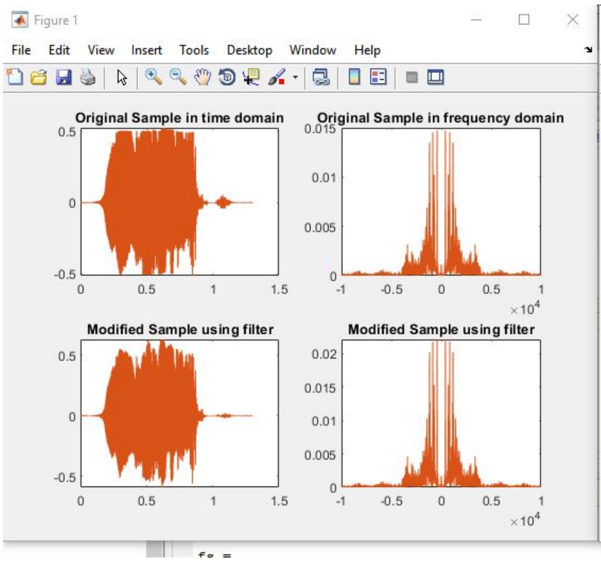
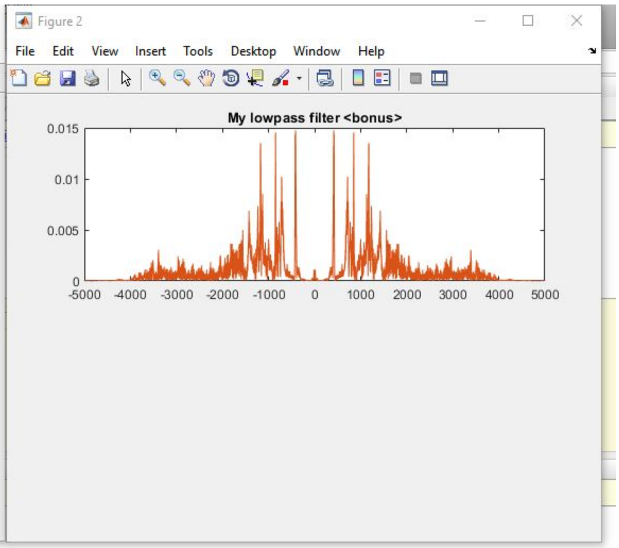
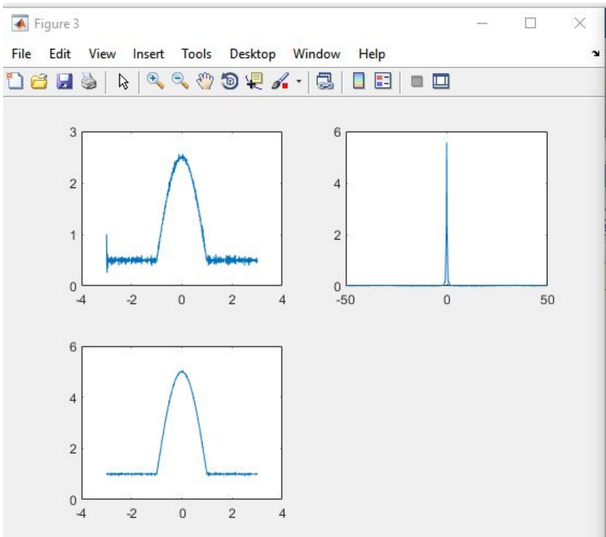


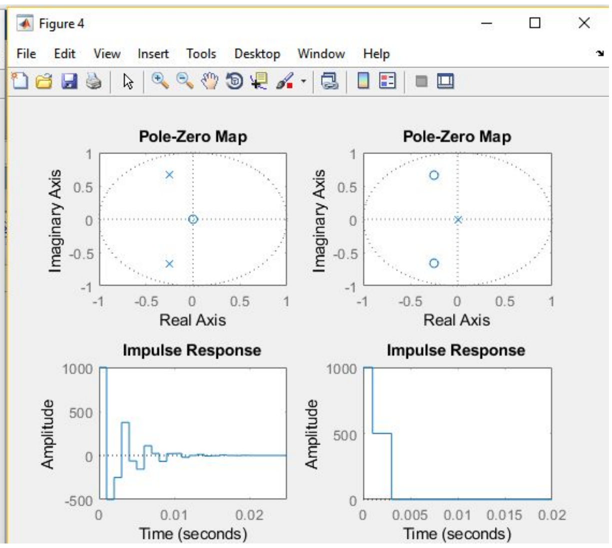
Figure 1
File Edit View Insert Tools Desktop Window Help
dual tone signal in frequency domain
Frequency
Time
Modified Signal using filter
Fres =0.0019536×π radians/sample
Power Spectrum (dB)
Normalized Frequency (×π radians/sample)
Figure 2
File Edit View Insert Tools Desktop Window Help
Original Signal in time domain
Signal in frequency domain
Frequency
Time
filtered voice signal in frequency domain






```
Command Window
New to MATLAB? See resources for Getting Started.
  Select input #3 >> 2
  Enter Amplitude >> 1
  **************h(t)*************
  Enter Start Point >> 0
  Enter End Point >> 5
  Enter # of break-points >> 2
  Break-point #1 position>> 1
  Break-point #2 position>> 3
  Select input #1 >> 3
  Enter Slope >> 1
  Enter Intercept >> 1
  Select input #2 >> 2
  Enter Amplitude >> 2
  Select input #3 >> 4
  Enter Amplitude >> 2
  Enter Exponent >> -1
  Enter Shift >> 3
  Enter Sigma to generate noise (0 if no noise) >> 2
  Enter numerator of Transfer Fucntion>> 1
  Enter denominator of Transfer Fucntion>> [1 0.5 0.5]
  System 1 is stable
  System 2 (recover) is stable
fx >>
```





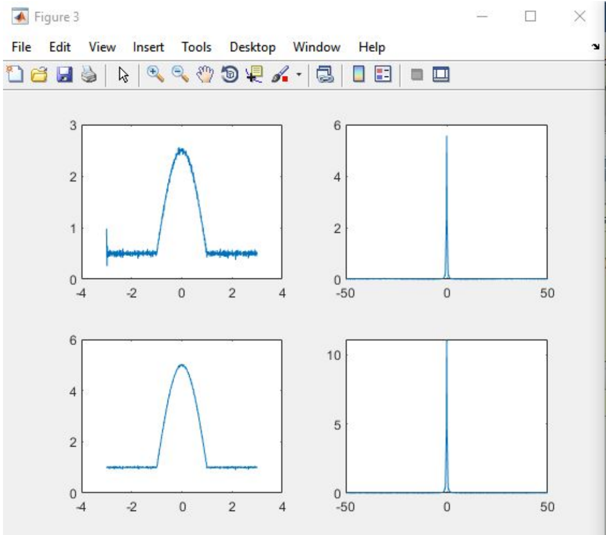


Figure 3
File Edit View Insert Tools Desktop Window Help

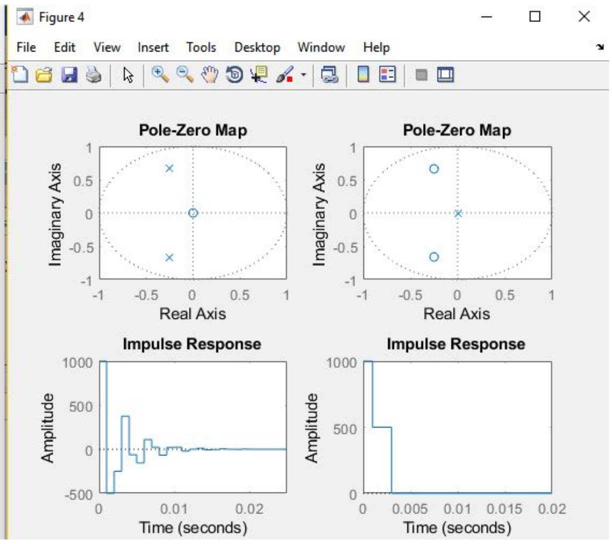


Figure 4
File Edit View Insert Tools Desktop Window Help
Pole-Zero Map
Pole-Zero Map
Imaginary Axis
Imaginary Axis
Real Axis
Real Axis
Impulse Response
Impulse Response
Amplitude
Amplitude
Time (seconds)
Time (seconds)

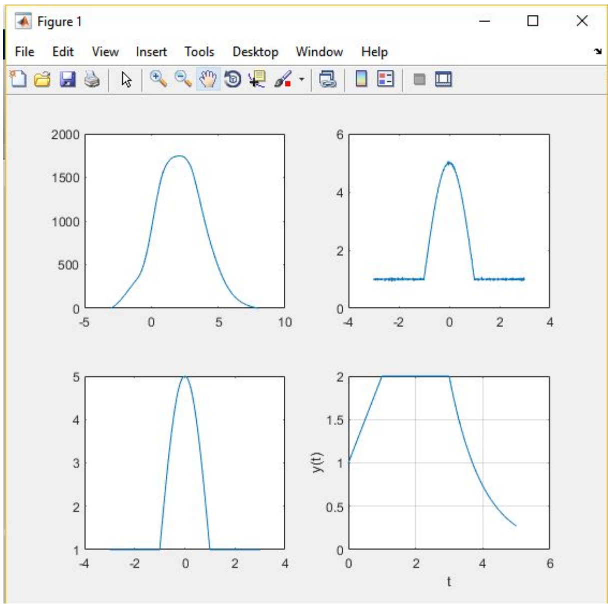


Figure 1
File Edit View Insert Tools Desktop Window Help
y(t)
t

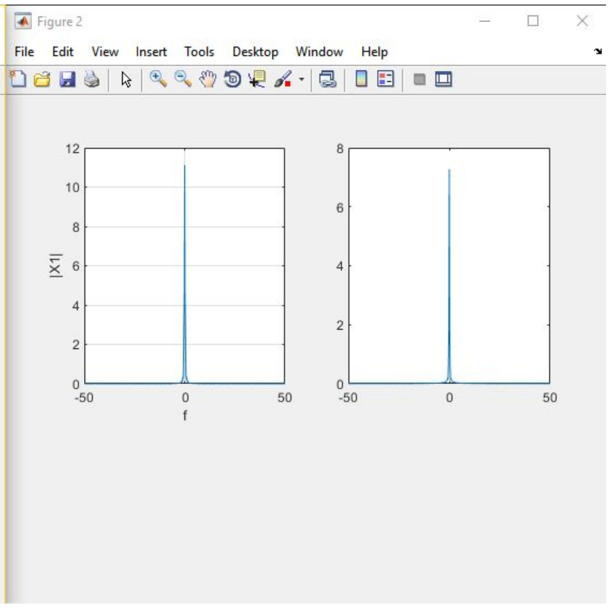


Figure 2
File Edit View Insert Tools Desktop Window Help
|X1|
f

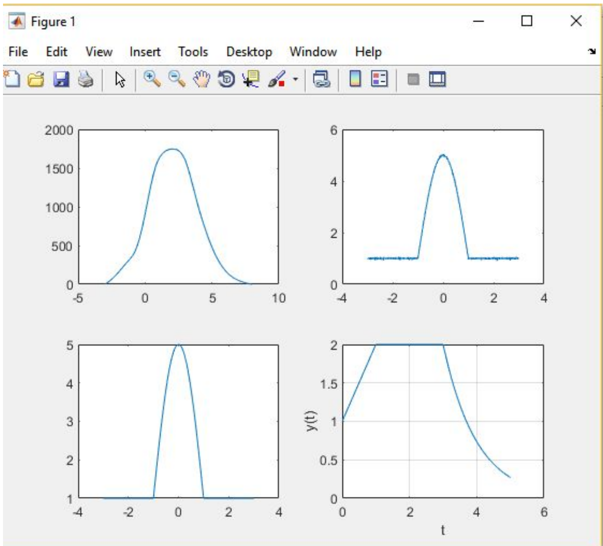


Figure 1
File Edit View Insert Tools Desktop Window Help
y(t)
t

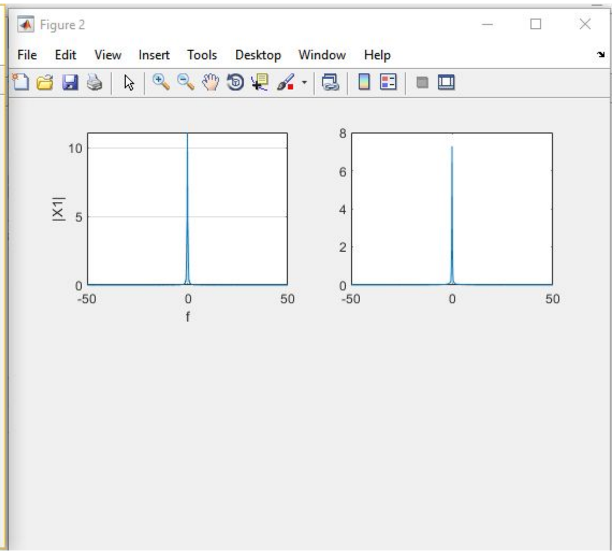
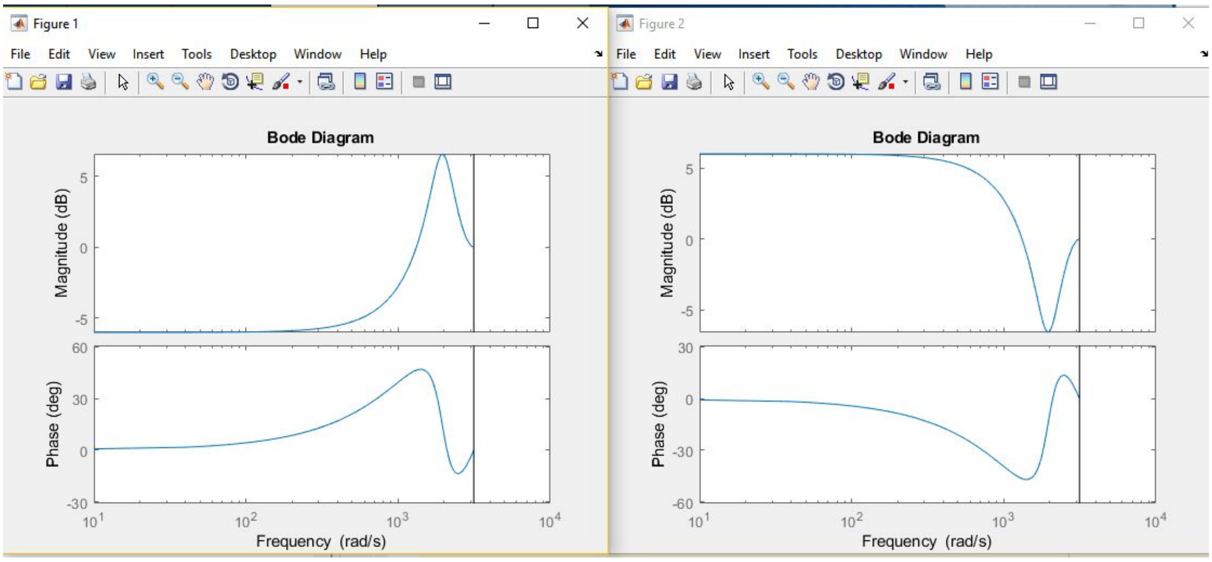


Figure 2
File Edit View Insert Tools Desktop Window Help
|X1|
f



Notes :

- **This program is implemented in octave (an alternative open source to MATLAB) edit(Now implemented in MATLAB)**
- **The fft plot in Figure 2 is the magnitude**
- **For phase angle plot << see main file Signals_project_1.m last comment**