CIE 442 Project 1 Part 2 Report

Spectrum Analysis and Filter Design Tool

Name: Ahmed Muhammad Tarek

ID: 201500885

E-mail: s-ahmed.tarek@zewailcity.edu.eg

Contents

Introduction

Spectrum analyzers are one of the most important tools in signal analysis and characterization of electronic devices. Their main functionality is to inspect the spectral properties of the input signal which provides the user with important information like the signal bandwidth, dominant frequencies and power of the signal. There are various architectures for spectrum analyzers; the implemented architecture is a Fast-Fourier-Transform based Architecture.

Theory

For discrete aperiodic signals, we know that we can acquire information about its frequency spectrum through its Discrete-Time Fourier Transform (DTFT) $x(\omega)$. However, $x(\omega)$ is continuous which is an inefficient representation to manipulate on computers which are limited by hardware constraints like memory size and the number of floating point operations they are capable of doing in a second.

Discrete Fourier Transform (DFT)

Discrete Fourier Transform (DFT) can be viewed as sampling of the Discrete-Time Fourier Transform.

Consider a discrete aperiodic signal $x(n)$, its Fourier Transform is given by the following DTFT equation:

$$x(\omega) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

$\because$ $x(\omega)$ is periodic with period 2π

$\therefore$ Only N equidistant samples will be considered in each period

This can be represented as follows:

$$x\left(\frac{2\pi k}{N}\right) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\frac{2\pi k}{N}n}$$

$$= \cdots + \sum_{n=-N}^{-1} x(n)e^{-j\frac{2\pi k}{N}n} + \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi k}{N}n} + \sum_{n=N}^{2N-1} x(n)e^{-j\frac{2\pi k}{N}n} + \cdots$$

This can be rewritten as:

$$x\left(\frac{2\pi k}{N}\right) = \sum_{l=-\infty}^{\infty} \sum_{n=lN}^{lN+N-1} x(n)e^{-j\frac{2\pi k}{N}n}$$

$$= \sum_{l=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n - lN) e^{-j\frac{2\pi k}{N}n} e^{j2\pi l}$$

$$= \sum_{l=-\infty}^{\infty} \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi k}{N}n}$$

By interchanging the order of summation, we get:

$$\sum_{n=0}^{N-1} \sum_{l=-\infty}^{\infty} x(n - lN)\, e^{-j\frac{2\pi k}{N}n}$$

$$Let\ x_p(n) = \sum_{l=-\infty}^{\infty} x(n - lN)$$

$x_p(n)$ is a periodic repetition of $x(n)$ with a period N

Consequently, its Discrete-Time Fourier Series can be found as follows

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n)\, e^{-jk\omega_o n}$$

$$= \frac{1}{N} x\left(\frac{2\pi k}{N}\right)$$

Using the inverse Discrete-Time Fourier Series equation:

$$x_p(n) = \sum_{n=0}^{N-1} \frac{1}{N} x\left(\frac{2\pi k}{N}\right) e^{jk\omega_o n}$$

Consequently, we deduce that $x(n)$ can be recovered from $x_p(n)$, if and only if $x(n)$ is time limited to an interval less than or equal to the period N of $x_p(n)$.

Assume a finite duration sequence $x(n)$ which is defined in the interval $0 \leq n \leq N - 1$ and zero elsewhere

If $N < L$, time domain aliasing occurs and $x(n)$ cannot be recovered from $x_p(n)$.

If $N \geq L$,

$$x(n) = \begin{Bmatrix} x_p(n), & 0 \leq n \leq N - 1 \\ 0, & elsewhere \end{Bmatrix}$$

DFT

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi k}{N}n}$$

IDFT

$$x(n) = \frac{1}{N}\sum_{n=0}^{N-1} x(k) e^{j\frac{2\pi k}{N}n}$$

Notice the resemblance between the DFT-IDFT pair and the Discrete-time Fourier Series - Inverse Discrete-time Fourier Series pair. The DFT can be thought of as the Discrete-time Fourier series scaled by a factor N for the periodic extension of the signal which preserves the spectrum at the sampling points in case assuming time domain aliasing does not occur.

Fast Fourier Transform (FFT)

It is an attempt to compute the DFT more efficiently using a divide and conquer approach reducing an N-point DFT where N is a composite number into smaller FFTs from which we can compute the original N-point DFT. One of its examples is the Radix-2 FFT Algorithm will be described in the following section.

Radix-2 FFT Algorithm

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi k}{N}n}$$

By splitting $X(k)$ into even and odd sequences

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(2n) e^{-j\frac{2\pi k}{\frac{N}{2}}n} + e^{-\frac{j2\pi k}{N}} \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) e^{-j\frac{2\pi k}{\frac{N}{2}}n}$$

$$X(k) = DFT\{x(2n)\}_{N/2} + e^{-\frac{j2\pi k}{N}} DFT\{x(2n+1)\}_{N/2}$$

If we continue splitting the sequences, we will end up with 2-point DFTs which can be combined to compute $X(k)$ which reduces the complexity of DFT to $O(N \log N)$

Windowing

The spectrum Analyzer captures the input signal by multiplying it by a window in the time domain which alters the characteristics of the signal in the frequency domain as multiplication in the time domain is equivalent to convolution in the frequency domain which can lead to leakage.

Types of Windows

Rectangular Window

Rectangular Windows can be represented mathematically as follows:

$$w(t) = rect(\frac{t - t_o}{\tau})$$
$$|W(f)| = \tau sinc(\tau f)$$

where t_o is the center of the rectangle and τ is the length of the rectangle

A rectangular window cannot be used in capturing an input signal that is composed of two harmonics that are very close to each other, it might be hard to distinguish those harmonics due to the interference from the side lobes.

Triangular Window

Triangular Windows can be represented mathematically as follows:

$$w(t) = \begin{Bmatrix} 1 - |t|, |t| \leq 1 \\ 0, elsewhere \end{Bmatrix}$$

$$W(f) = sinc^2(f)$$

The side lobes of the Triangular window have lower magnitudes than the Rectangular window which improves their ability to distinguish close spectral components. However, the main lobe of the triangular window is wider which means less selectivity.

Hanning Window

Hanning Windows can be represented mathematically as follows:

$$w(t) = \frac{1}{2}(1 - \cos\frac{2\pi n}{N - 1})$$

where N is the length of the window

The magnitudes of the side lobes are close to that of the triangular window. However, it has a narrower main lobe which increases its selectivity and accuracy in estimating the spectrum.

Hamming Window

$$w(t) = 0.54 - 0.46\ \cos\frac{2\pi n}{N - 1}$$

The main lobe is slightly wider than the main lobes of the triangular window and the Hanning Window. However, the side lobes are considerably lower in magnitude which limits the spectral leakage.

Note: It is recommended to choose an fft size more than or equal to the window size, as choosing a lower one is equivalent to applying a rectangular window the signal in the time domain which can lead to undesirable effects.

Discrete Convolution

The response of a Discrete LTI system whose impulse response is $h(n)$ to an input $x(n)$ is equivalent to the convolution process which is represented mathematically as follows:

$$y(n) = h(n) * x(n)$$

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

An illustration of this process is present in the Discrete Convolution Mode of the implemented Application.

Sampling Frequency

Nyquist Shannon Sampling Theorem states that a signal that is band-limited to $B\ hz$ is completely determined by a numeric sequence of its samples if the sampling rate $f_s \geq 2B$. So, on choosing the sampling frequency, we have to adhere to the $2B\ hz$ rate. However, the effect of windowing time-limits the signal which changes its frequency domain characteristics making it band-unlimited. So, it is advisable to choose a sampling frequency higher than the Nyquist rate $(f_s \geq 4B)$ in order to get reliable results.

Least Squares Filter Design Method

If we assume an even symmetric FIR filter of length $L + 1$ samples, it is frequency response can be written as:

$$H(\omega_k) = h_o + 2\sum_{i=1}^{L/2} h_n \cos(\omega_k n)$$

which can be written in matrix form as:

$$\begin{bmatrix} H(\omega_o) \\ H(\omega_1) \\ \vdots \\ H(\omega_{N-1}) \end{bmatrix} = \begin{bmatrix} 1 & 2\cos(\omega_o) & \dots & 2\cos(\omega_o \frac{L}{2}) \\ 1 & 2\cos(\omega_1) & \dots & 2\cos(\omega_1 \frac{L}{2}) \\ \vdots & \vdots & \dots & \vdots \\ 1 & 2\cos(\omega_{N-1}) & \dots & 2\cos(\omega_{N-1} \frac{L}{2}) \end{bmatrix} \begin{bmatrix} h_o \\ h_1 \\ \vdots \\ h_{N-1} \end{bmatrix}$$

which can be solved by least squares method for $h(n)$.

GUI Documentation

Once you run the main application, the following interface appears

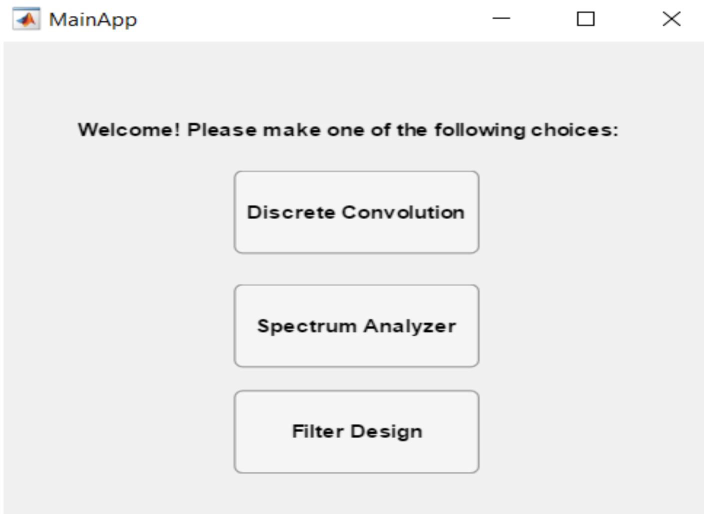

Fig (1) GUI of the main application

Discrete Convolution

If you choose Discrete Convolution, the following interface appears

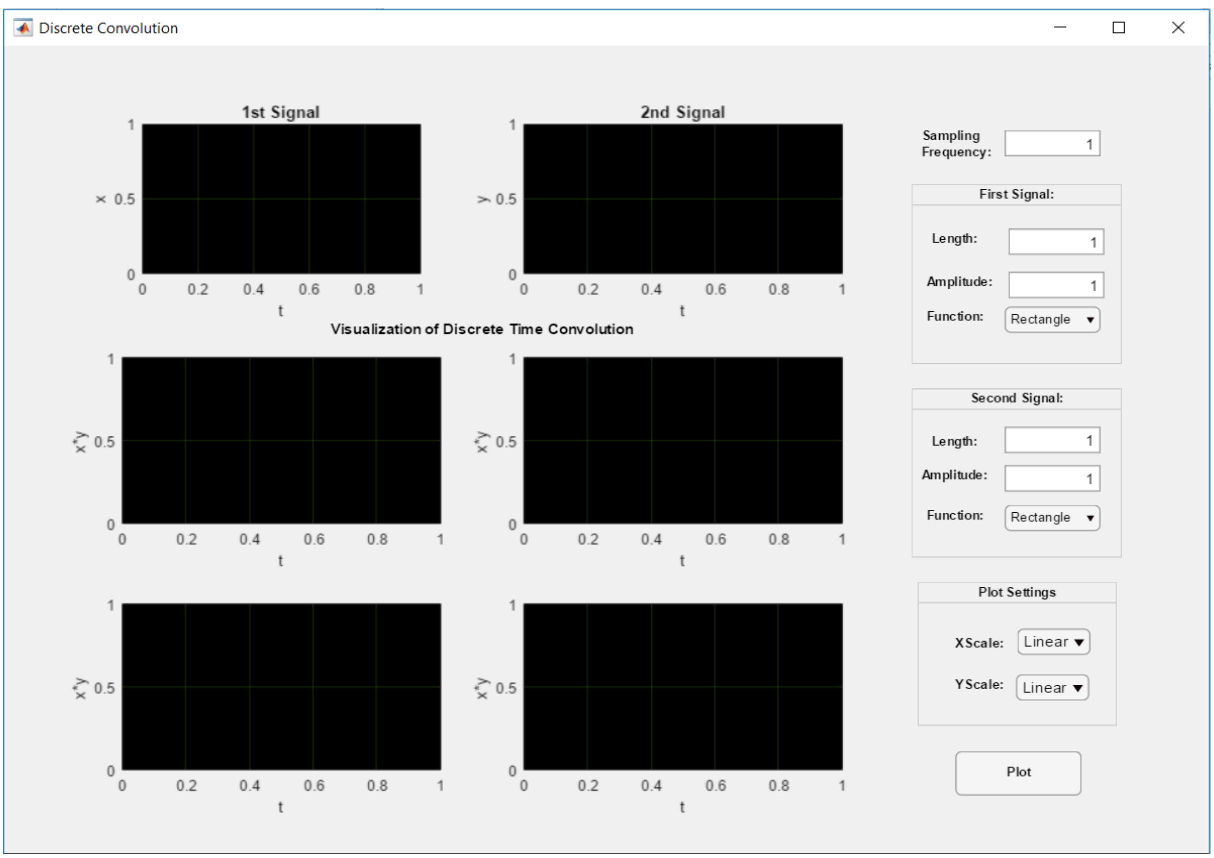

Fig (2) GUI of the Discrete Convolution mode

Capabilities:

This mode computes the discrete convolution of two signals sampled with the same sampling frequency. It plots the two signals in the time domain and four stages of the convolution process in linear or logarithmic scale.

The signals that are going to be convolved, the sampling frequency and the plotting scales are chosen by the user.

Provided Signals:

1. Rectangle
2. Triangle
3. Sine

Signal Parameters:

1. Amplitude
2. Length in case of Rectangle and Triangle
3. Period in case of Sine

Step-by-Step Guide:

1. Input the sampling frequency in the appropriate field above the First Signal panel
2. Choose one of the provided signals (Rectangle, Triangle and Sine) in the First Signal panel
3. Input the amplitude and the length of the first signal in case of Rectangle and Triangle or the Amplitude and the Period in case of sine in the appropriate fields in the First Signal panel
4. Do $(2), (3)$ for the second signal
5. Click on the plot button

Example:

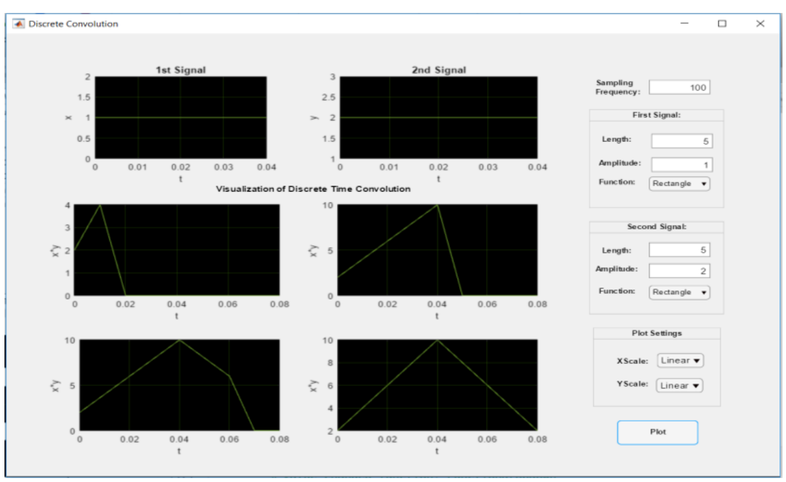

Fig (3) Convolution of two rectangular signals whose lengths are 5 samples and amplitude of the first is 1 and amplitude of the second is 2, both of them are sampled at a rate of $100\ hz$

If you choose Spectrum Analyzer, the following interface appears

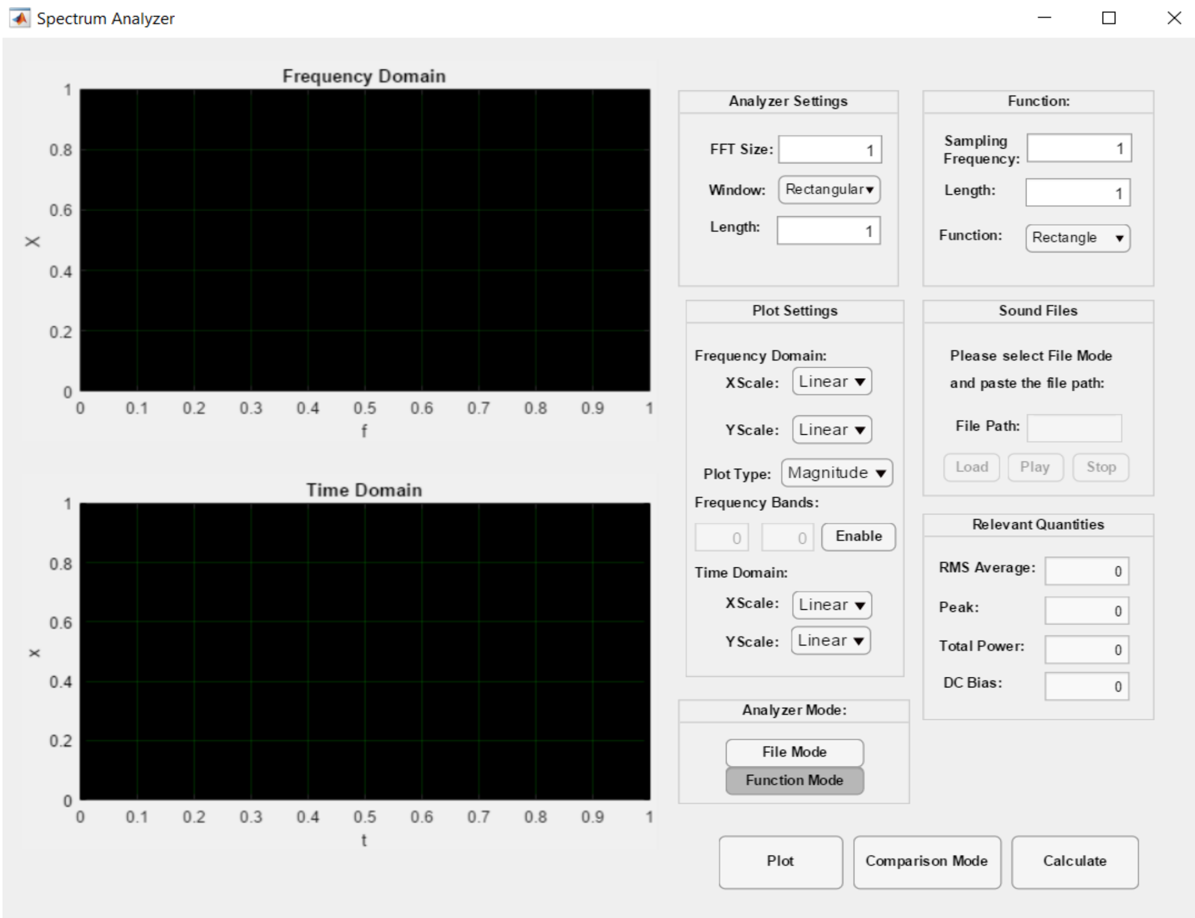

Fig (4) GUI of the Spectrum Analyzer in Function Mode

Main Capabilities:

The mode takes a signal as an input, applies a window to it, then plots the signal in the time domain and the frequency domain on linear or logarithmic scales. It also calculates important quantities like *RMS* Average, Peak of the Signal, Power of the Signal and the Absolute value of the DC bias.

Note: The frequency bands functionality is disabled, it will be added in the next phase

Function Mode

Step-by-Step Guide:

1. Set the Spectrum Analyzer to function mode by toggling the function mode button in the Analyzer Mode panel
2. Choose one of the provided functions (Rectangle, Triangle and Sine) in the function panel
3. Input the sampling frequency in the appropriate field in the function panel
4. Input the length of the function in case of Rectangle and Triangle or the Period

in case of sine in the appropriate field in the function panel
Note: In case of sine, it is preferable to choose a sampling frequency higher than the Nyquist Frequency

5. Choose one of the window types (Rectangular, Triangular, Hamming and Hanning), and type in the window length and the FFT size in the appropriate fields in the Analyzer settings panel.
6. Set the plot settings as desired. You can set the scales to be linear or logarithmic for time domain and frequency domain plots. Also, you can choose to plot the magnitude, phase, real part or imaginary part of the Fourier Transform of the signal.
7. Click on Plot button to display the signal in time domain and frequency domain after windowing
8. After plotting, you have the option to choose frequency bands by pressing the enable button then typing the desired frequency span.
9. Click on Calculate button to display the RMS Average, Peak of the Signal, Power of the Signal and the Absolute value of the DC bias in the Relevant Quantities panel

Examples:

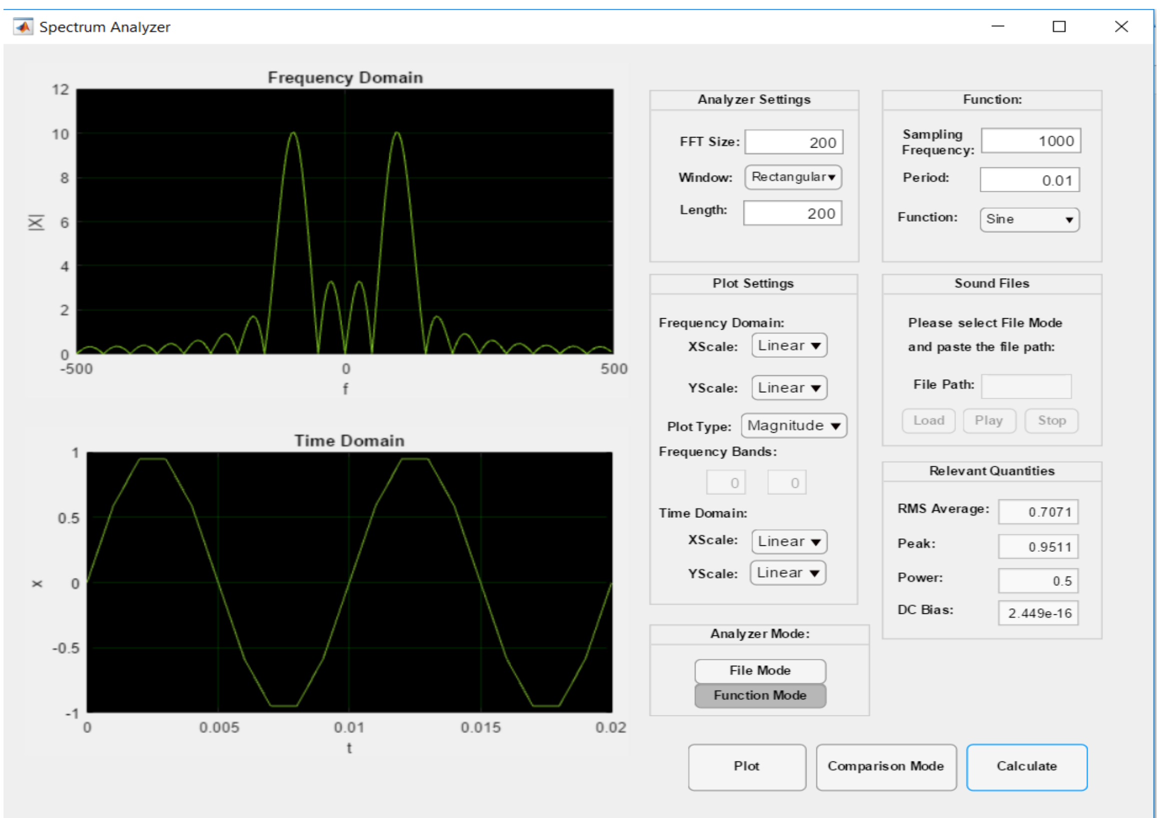

Fig (5) Time domain and Frequency domain plots on linear scales of a sinusoidal function with period 0.01 s sampled at a rate of 1000 hz using a rectangular window whose length is 200

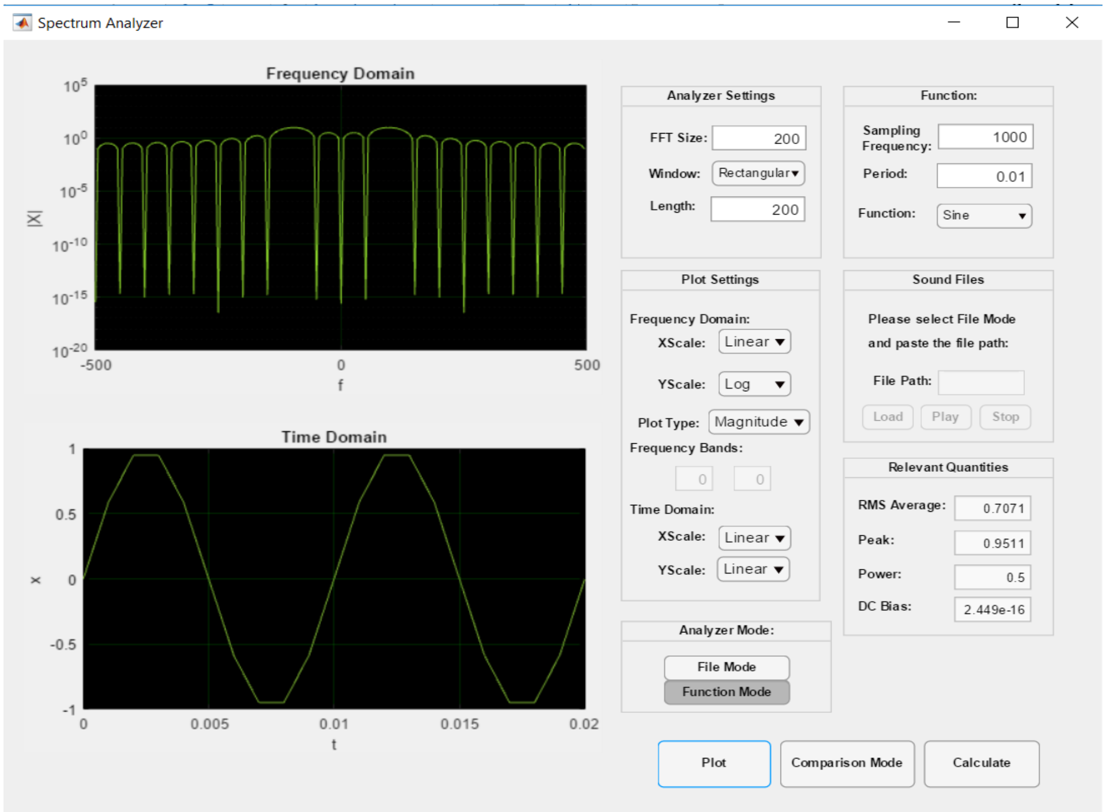

Fig (6) Same as Fig (5) with a logarithmic y-scale for the frequency domain plot

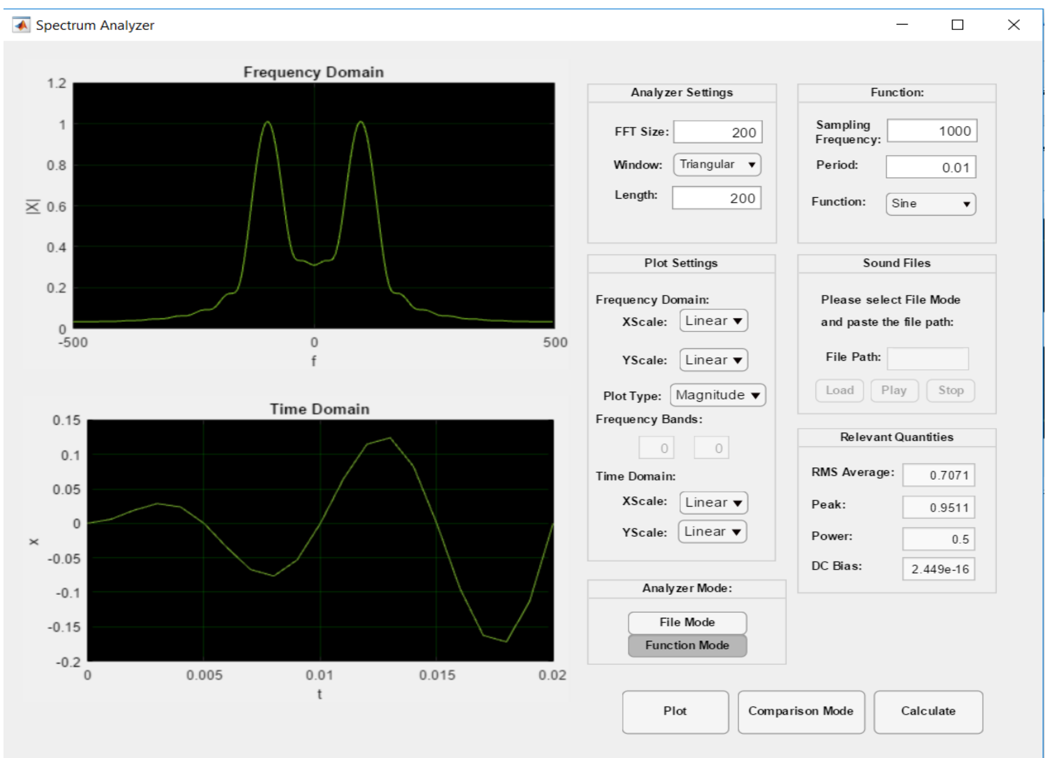

Fig (7) Same as Fig (5) with a triangular window of length 200

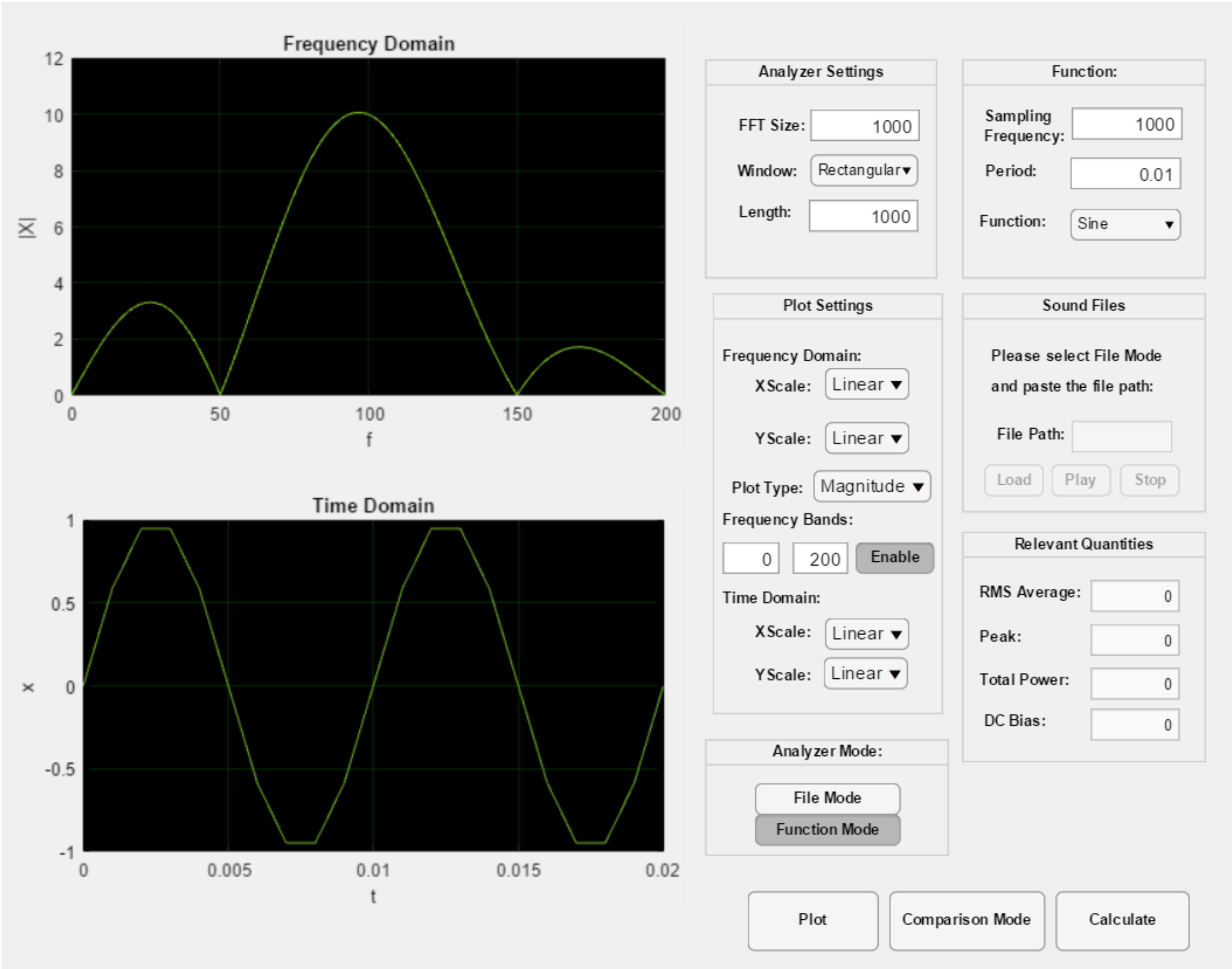

Fig (8) Time domain and Frequency domain plots on linear scales of a sinusoidal function with period $0.01\ s$ sampled at a rate of $1000\ hz$ using a rectangular window whose length is 1000, the frequency band is $[0hz, 200hz]$

File Mode
Step-by-Step Guide:

1. Set the Spectrum Analyzer to file mode by toggling the function mode button in the Analyzer Mode panel, the interface will look as follows:

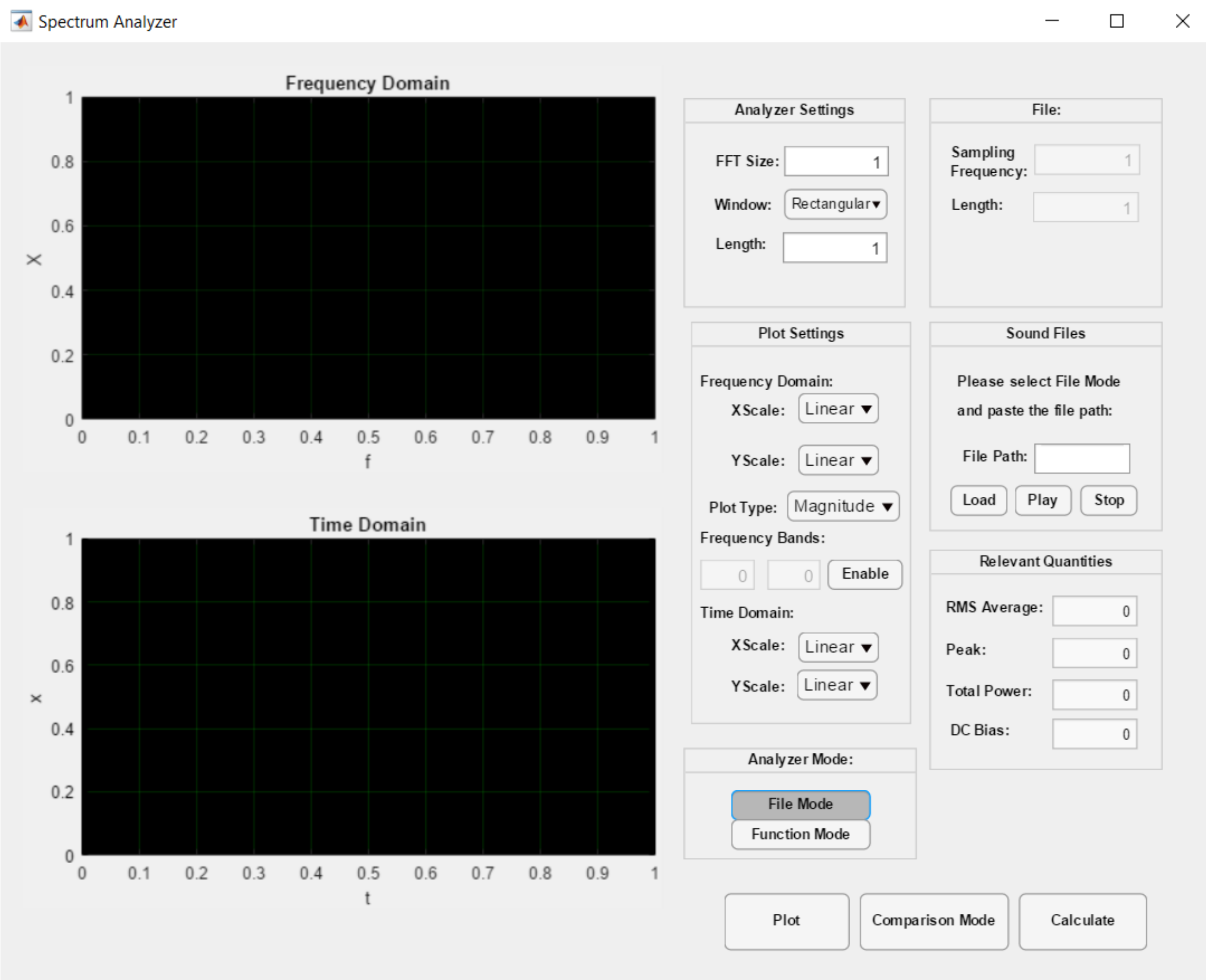

Fig (9) GUI of the Spectrum Analyzer in File Mode

2. Input the file path in the appropriate field in the Sound Files panel and click the load button (You can play and stop the sound if the file is loaded successfully by clicking on play and load buttons)
3. After loading the sound file, notice that the sampling frequency and the length of the file are displayed in the file panel.
4. Choose the window type and the window length and set the FFT size to an appropriate value.
5. Click on Plot button to display the sound signal in time domain and frequency domain after windowing

6. After plotting, you have the option to choose frequency bands by pressing the enable button then typing the desired frequency span.
7. Click on Calculate button to display the RMS Average, Peak of the Signal, Power of the Signal and the Absolute value of the DC bias in the Relevant Quantities panel

Example:

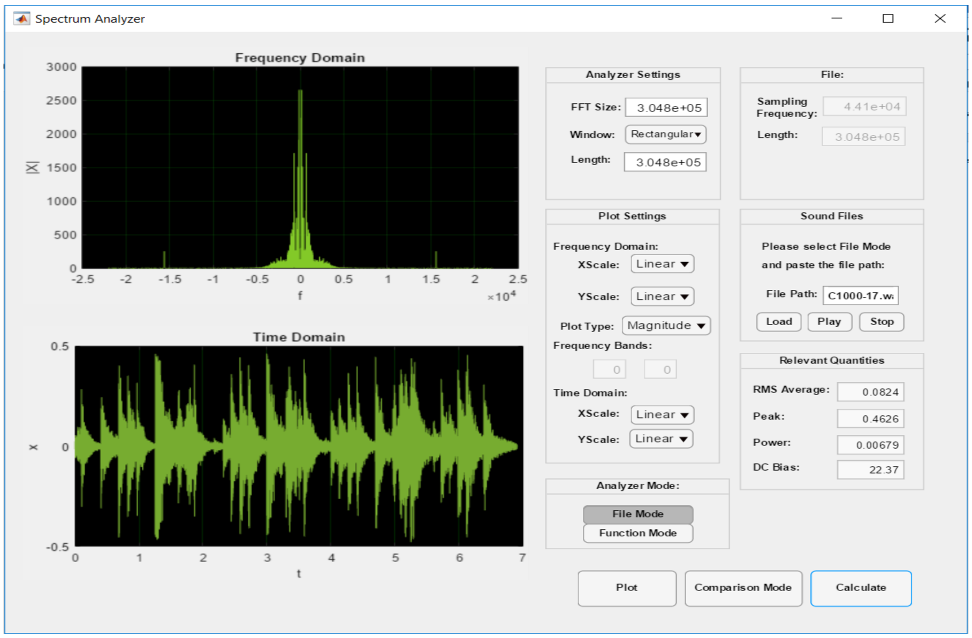

Fig (10) Time domain and Frequency domain plots on linear scales of the given sound file $C1000-17.wav$ using a rectangular window with the same length of the sound file and same for the FFT Size

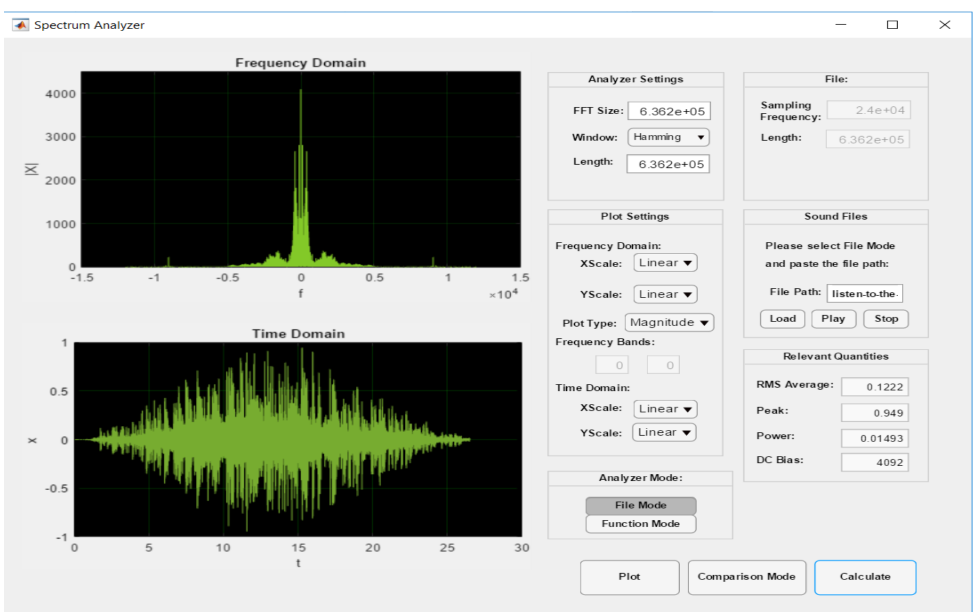

Fig (11) Time domain and Frequency domain plots on linear scales of the given sound file $listen-to-the-ancient-egyptions-tv.wav$ using a Hamming window with the same length of the sound file and same for the FFT Size

Comparison Mode

Main Capabilities:

The Mode takes two signals as an input, applies a certain window to each of them, then plots the two signals in the time domain and the frequency domain on a linear or a logarithmic scale. It also calculates important quantities like RMS Average, Peak of the Signal, Power of the Signal and the Absolute value of the DC bias of each signal

It can be used in the same way the main Spectrum Analyzer is used. However, calculation mode is not available in this mode.

If you click on the Comparison Mode Button, the following interface appears

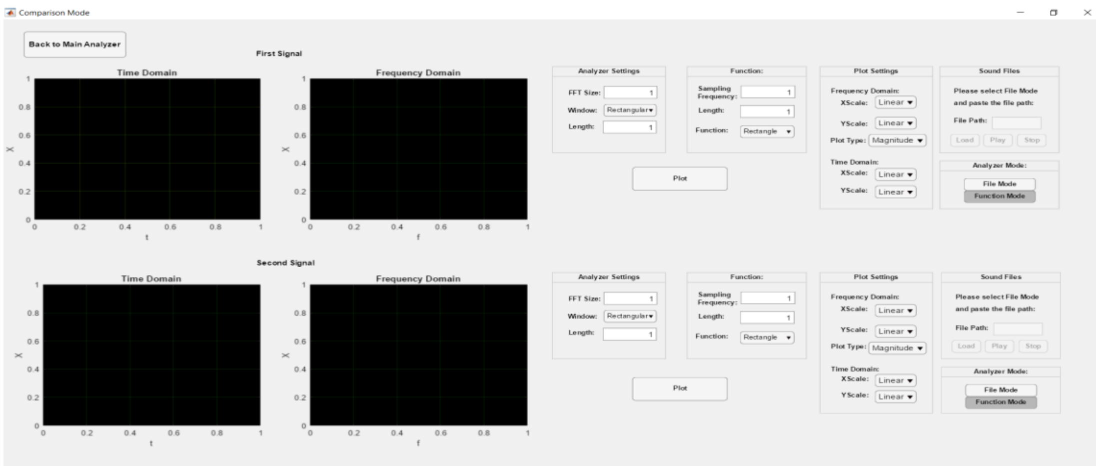

Fig (12) GUI of the Comparison Mode

Example:

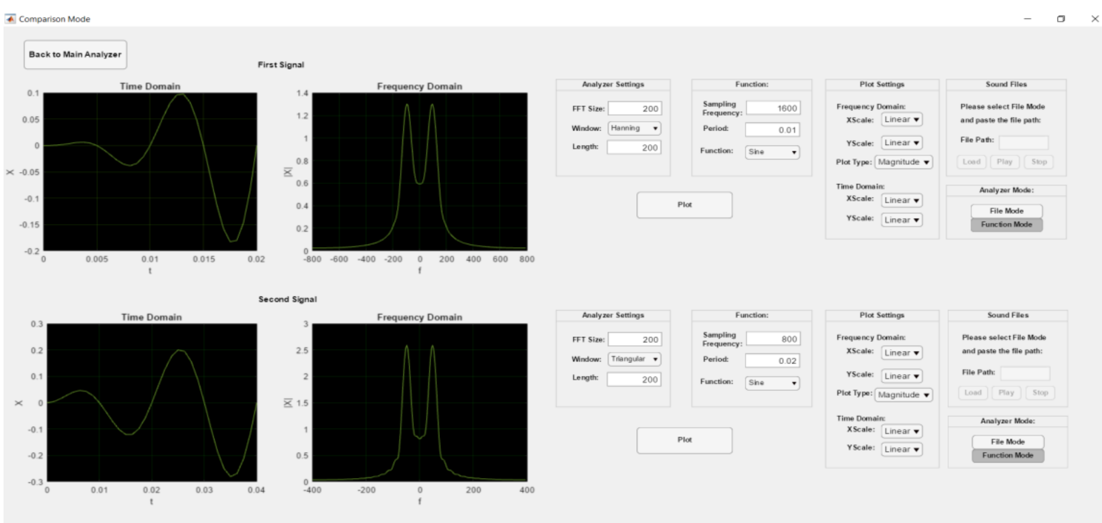

Fig (13) Comparison between two sinusoidal signals with period 0.01 & 0.02 respectively sampled with frequencies $1600hz$ & $800hz$ using a Hanning window of length 200 for the first signal and a triangular window of length 200 for the second signal

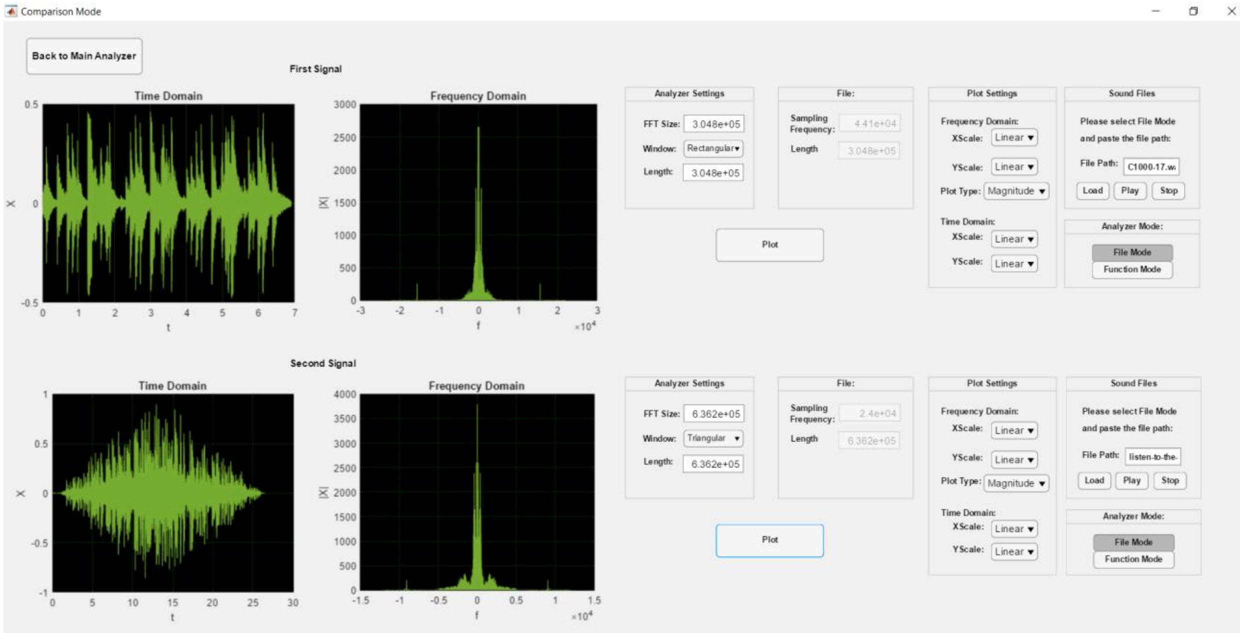

Fig (14) Comparison between the two given sound files app on applying a rectangular window to the first file and a triangular window to the second file, the windows sizes are equal to the length of the file in each case

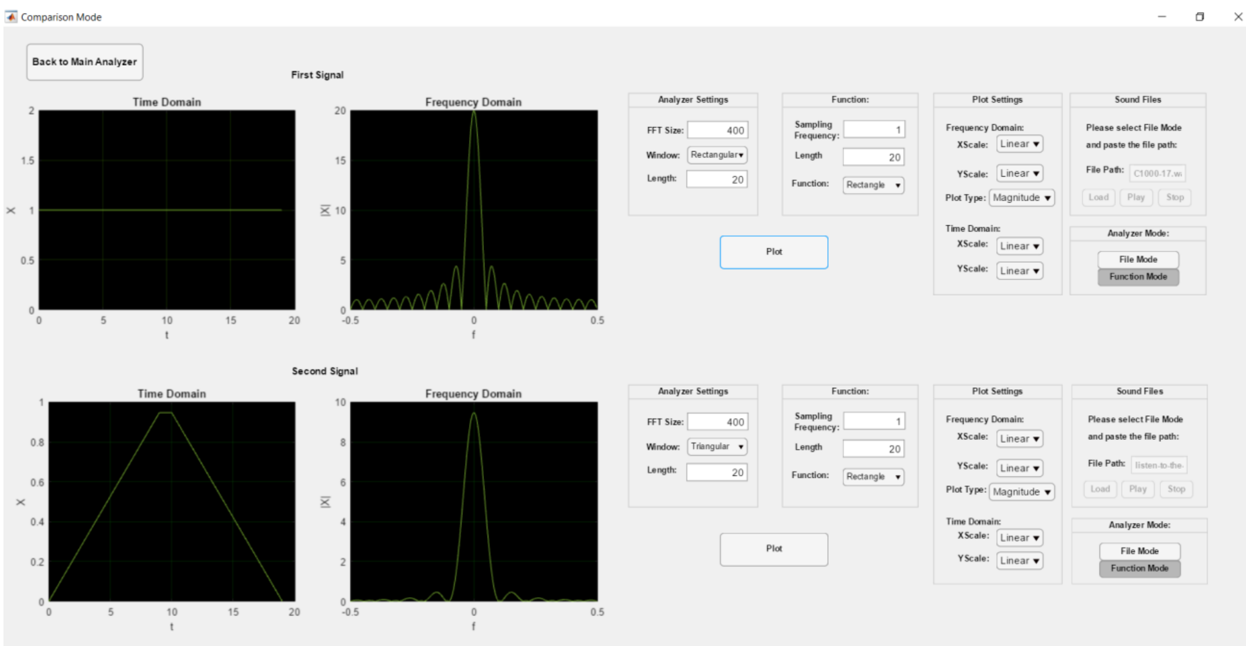

Fig (15) figure shows the effect of different windows on the spectrum a rectangular signal of length 20 sampled at a rate of $1hz$, the windows applied are a rectangular window of length 20 and a triangular window of length 20

Filter Design

If you choose Filter Design, the following interface appears:

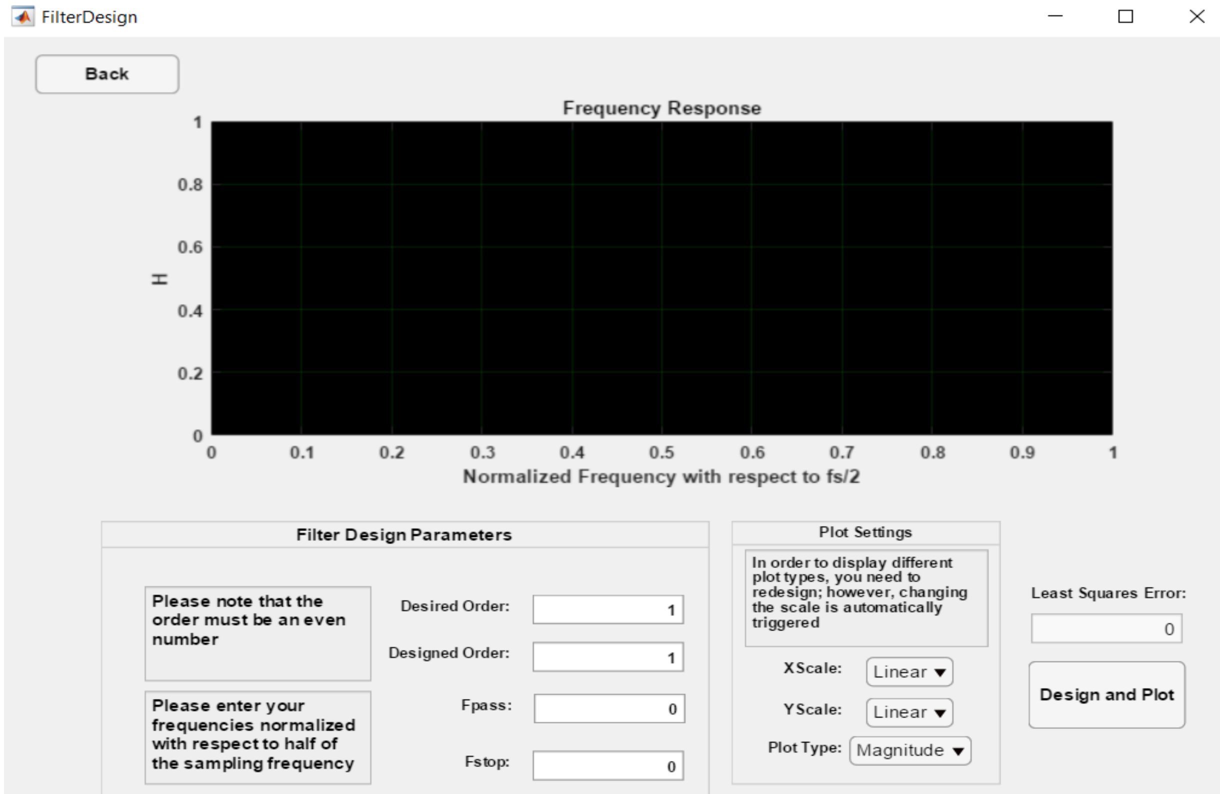

Fig (16) GUI of the Filter Design application

Main Capabilities:

The application takes user specified filter specifications as an input (Filter order, Passband Frequency & Stopband Frequency) and approximates the filter using least squares method and calculates the least squares error. It plots the frequency response of the filter on a linear or a logarithmic scale. It also provides different plotting options (Magnitude, Phase, Real and Imaginary).

Step-by-Step Guide:

1. Input the desired filter order, the designed filter order, the Passband Frequency f_{pass} and the Stopband Frequency f_{stop}.

Notes:

The desired order of the filter and the designed order of the filter must be even.

The filter order has to be even and the frequencies has to be normalized with respect to $\frac{f_s}{2}$ and $f_{pass} < f_{stop}$.

2. Choose the type of the plot from the plot type dropdown menu.

3. Click on Design and Plot Button.
4. You can change the scales after plotting; however, the plot type cannot be changed (a redesign command has to be issued)

Examples:

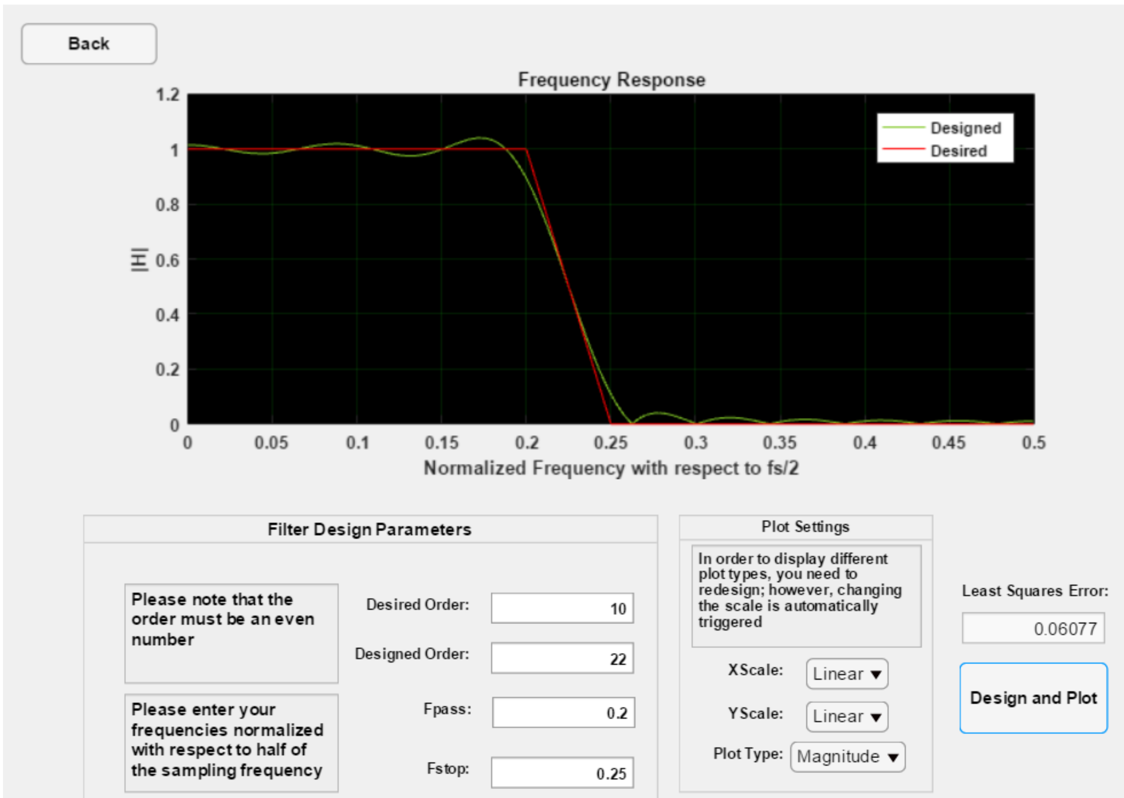

Fig (17) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.2, f_{stop} = 0.25$ & $Order = 22$

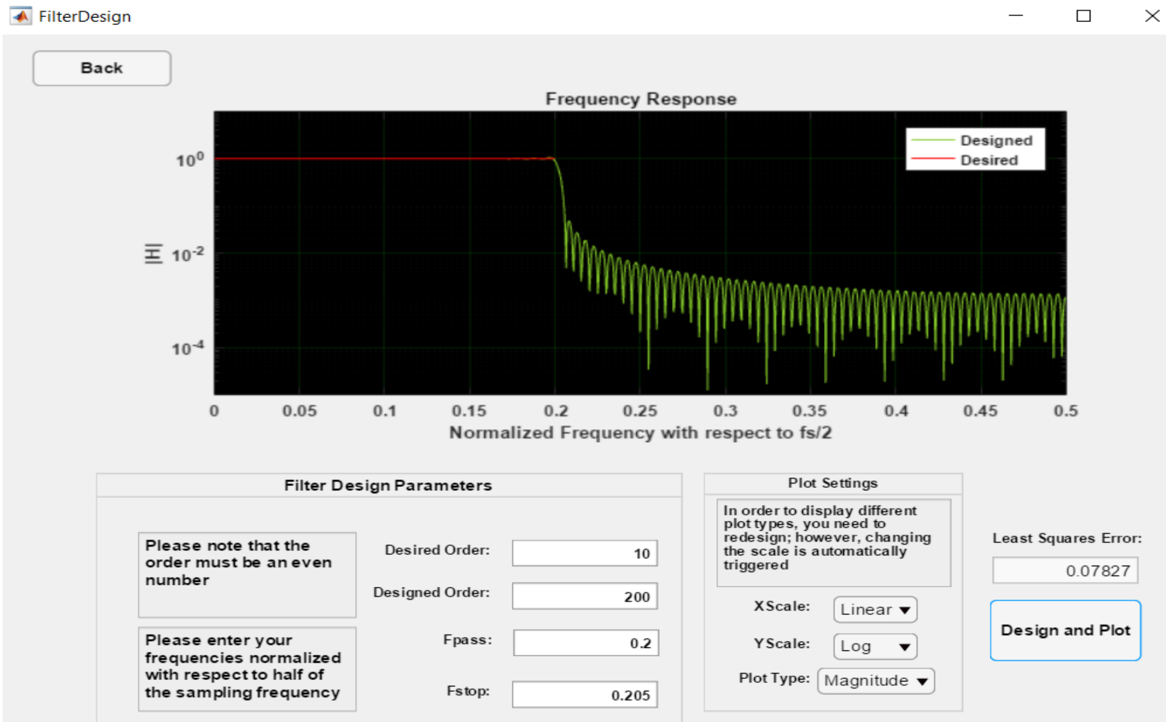

Fig (18) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.2, f_{stop} = 0.205$ & $Order = 200$ plotted on a logarithmic scale

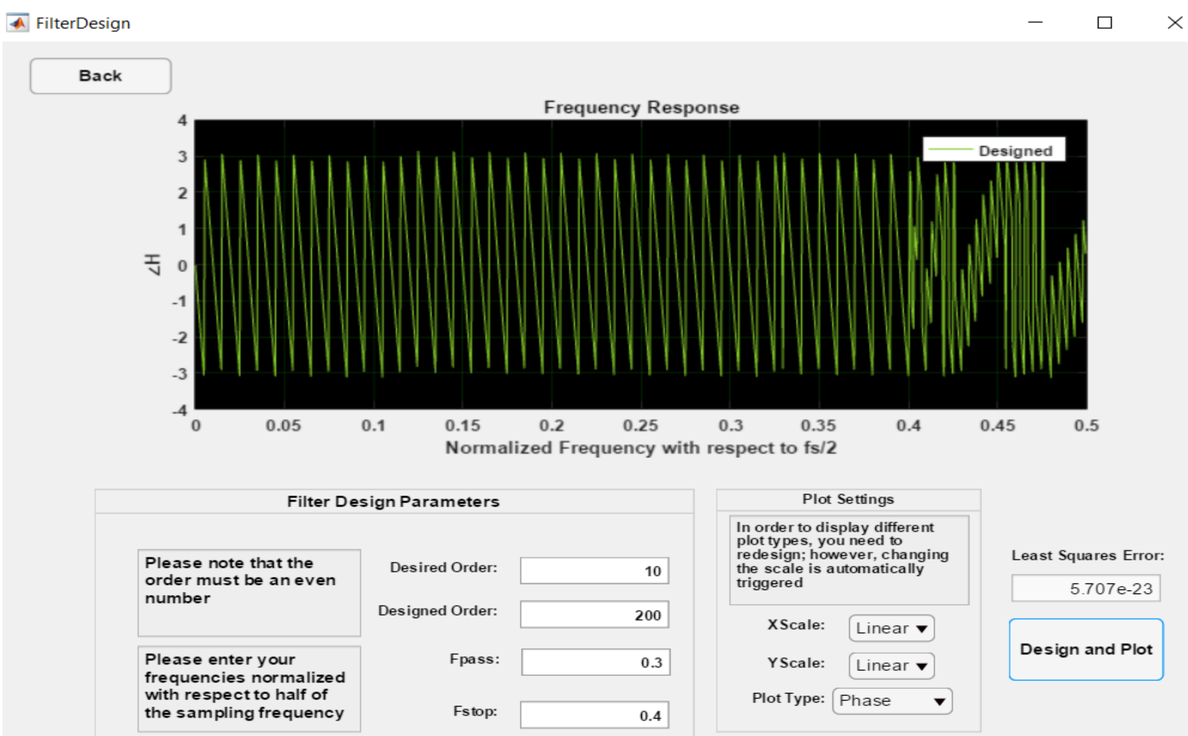

Fig (19) Phase of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.3, f_{stop} = 0.4$ & $Order = 200$

Note: The tool will only design low-pass filters. Other types of filters can be designed using frequency shifting methods.

$$H_{bp}(f) = e^{-j\frac{\pi}{2}} H_{lp}(f)$$

$$H_{hp}(f) = e^{-j\pi} H_{lp}(f)$$

Results and Discussion

Effects of windows on the processed signals

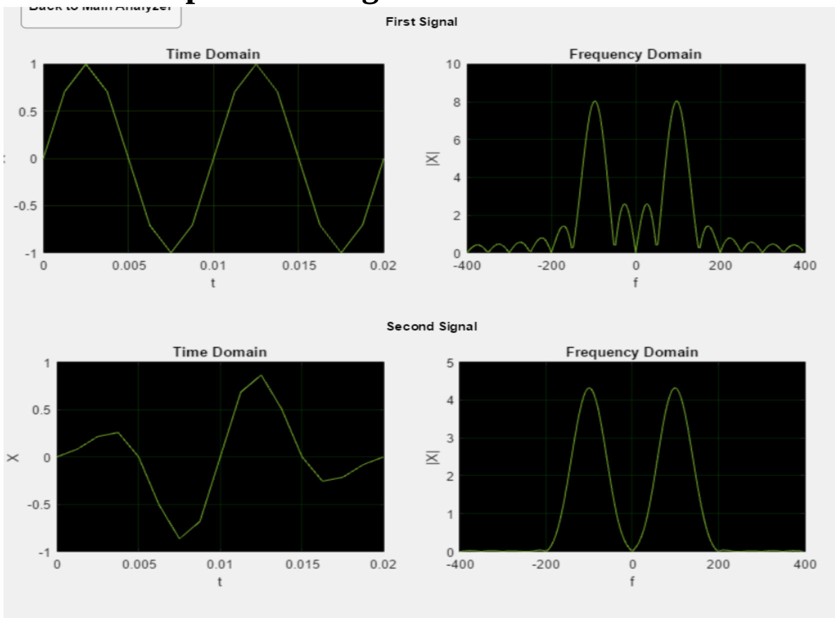

Fig (20) Spectrum of a sinusoid of period 0.01s sampled at a rate $800hz$ using a Rectangular window and a Hamming window of length 17

The spectral leakage in case of the Hamming window is significantly smaller than the case of the rectangular window as the amplitudes of the side lobes of the Hamming window are significantly smaller than those of the rectangular window.

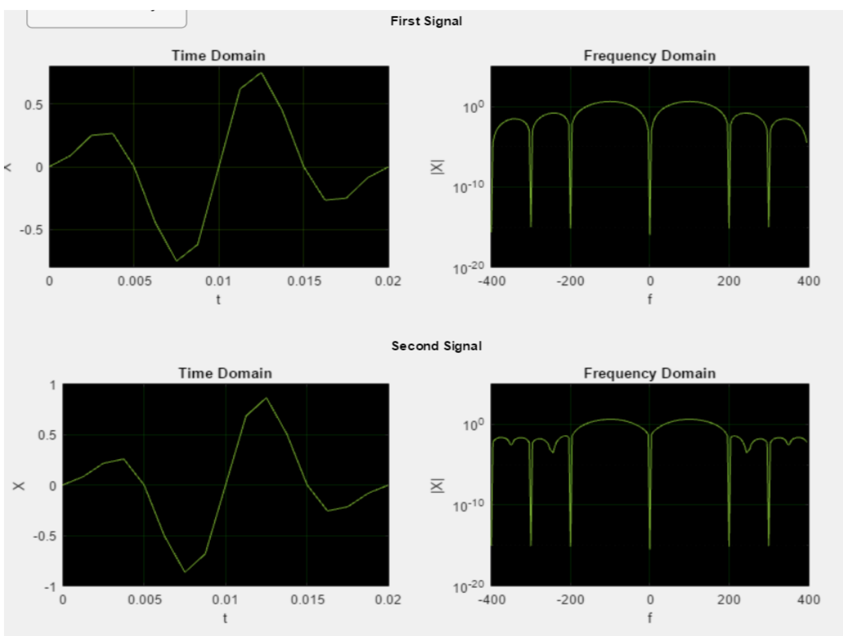

Fig (21) Spectrum of a sinusoid of period 0.01s sampled at a rate $800hz$ using a Triangular window and a Hamming window of length 17 plotted on a logarithmic scale

The spectral leakage in case of the Hamming window is smaller than the case of the Triangular window as the amplitudes of the side lobes of the Hamming window are smaller than those of the Triangular window in exchange for a slightly larger main lobe which makes the hamming window less frequency selective.

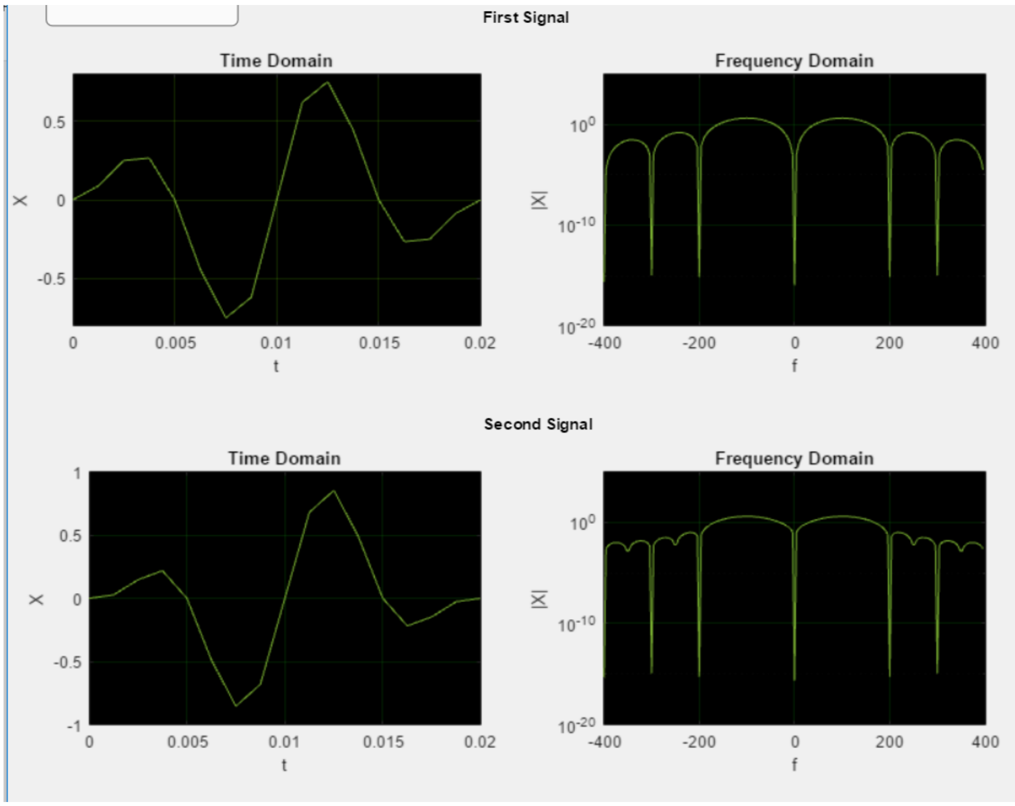

Fig (22) Spectrum of a sinusoid of period 0.01s sampled at a rate $800hz$ using a Triangular window and a Hanning window of length 17 plotted on a logarithmic scale

The spectral leakage in case of the Hanning window is smaller than the case of the Triangular window as the amplitudes of the side lobes of the Hanning window are smaller than those of the Triangular window. Also, their main lobes have the same size so they are equal when it comes to selectivity.

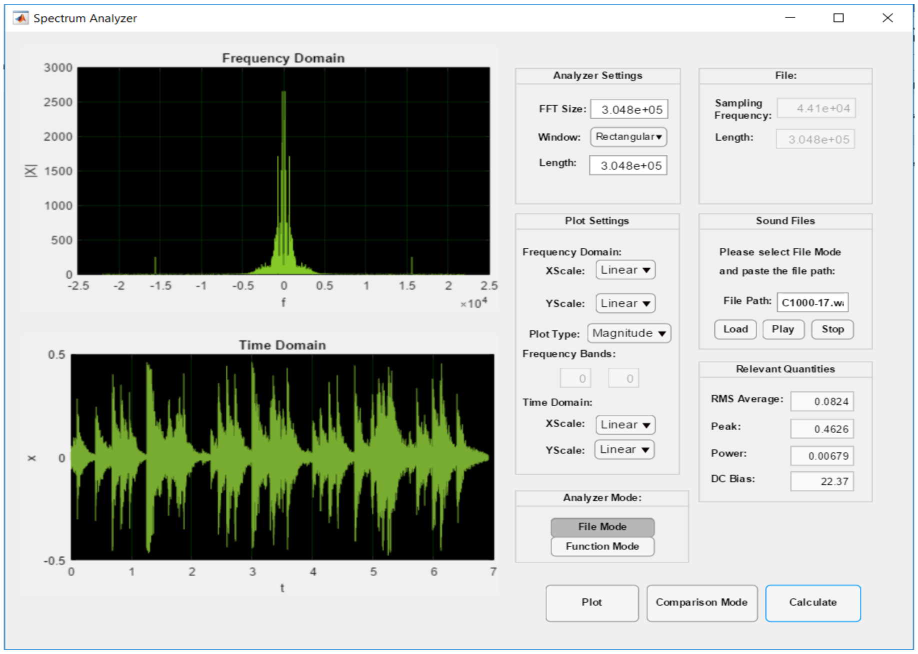

Fig (23) Time domain and Frequency domain plots on linear scales of the given sound file $C1000-17.wav$ using a rectangular window with the same length of the sound file and same for the FFT Size

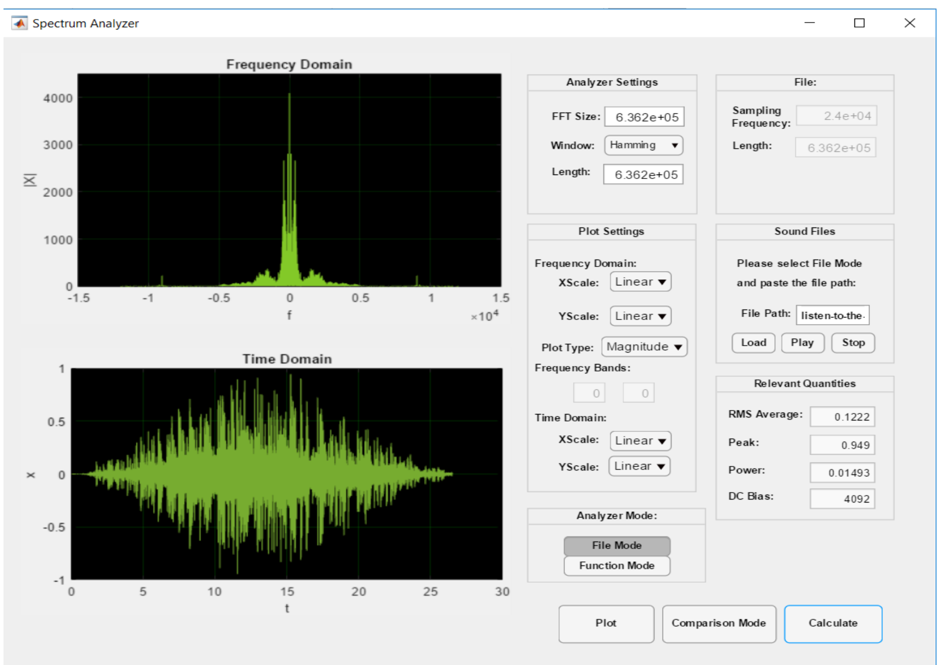

Fig (24) Time domain and Frequency domain plots on linear scales of the given sound file $listen-to-the-ancient-egyptions-tv.wav$ using a Hamming window with the same length of the sound file and same for the FFT Size

Notice that the spectrum of the sound files is concentrated in the interval $[0khz, 5khz]$ which is the range of frequencies that carries most of the sound information

Using the Spectrum Analyzer to inspect the frequency spectrum of the window functions

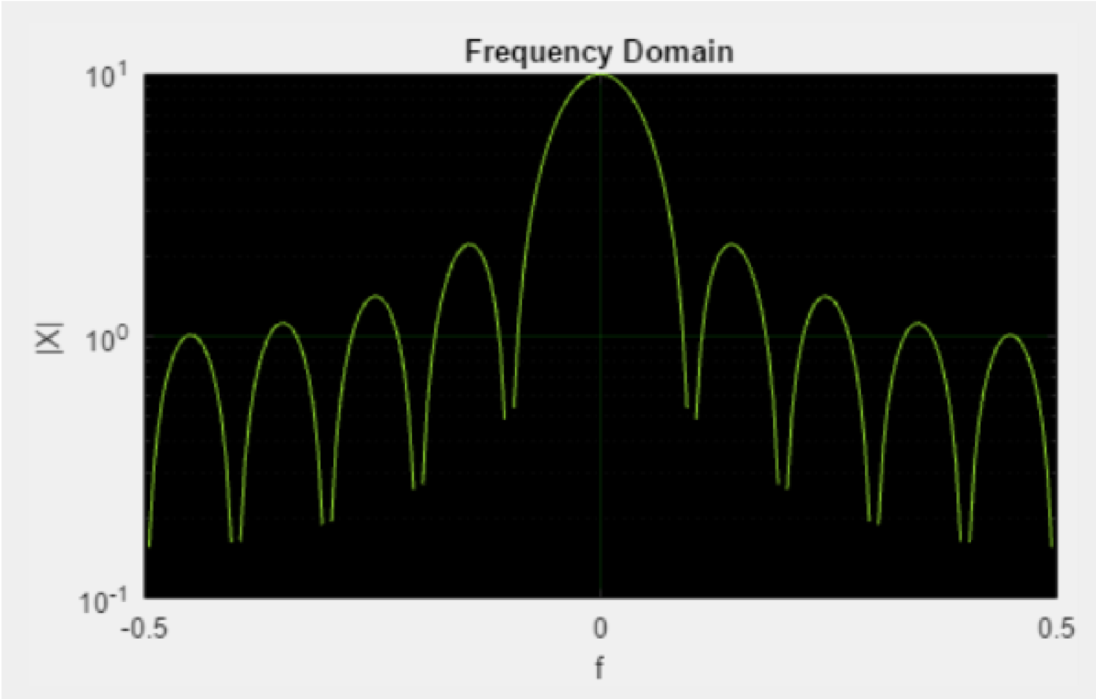

Fig (25) Frequency Spectrum of the rectangular window

Notice, the high amplitudes of the side lobes which results in high spectral leakage.

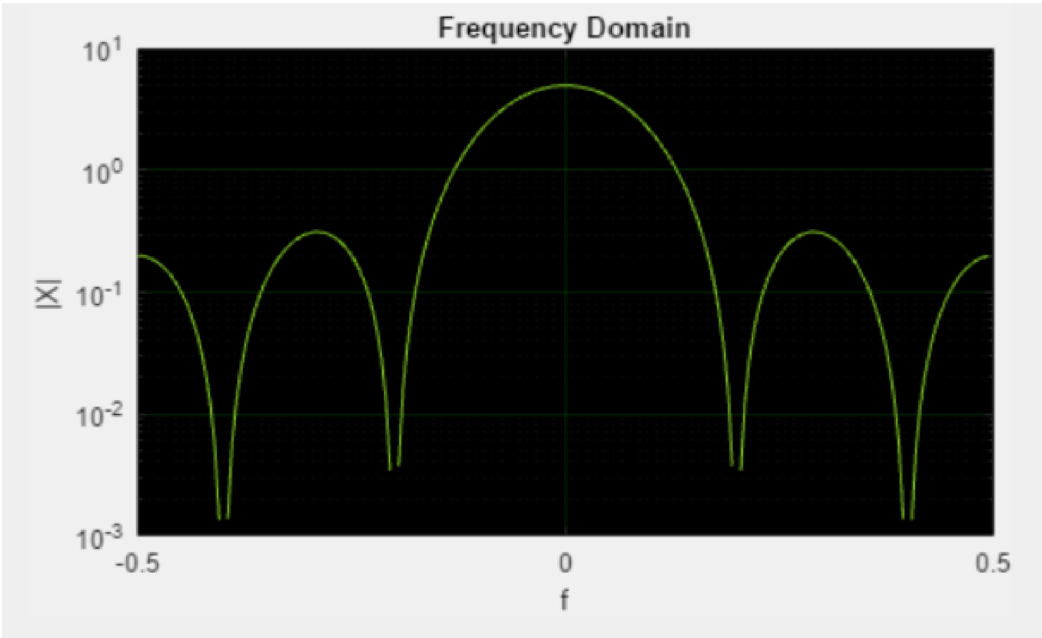

Fig (26) Magnitude of the Frequency Spectrum of the Triangular window on a logarithmic scale

The amplitudes of the side lobes are smaller those of the rectangular window which decreases the spectral leakage in exchange for a wide main lobe which decreases the selectivity of the window.

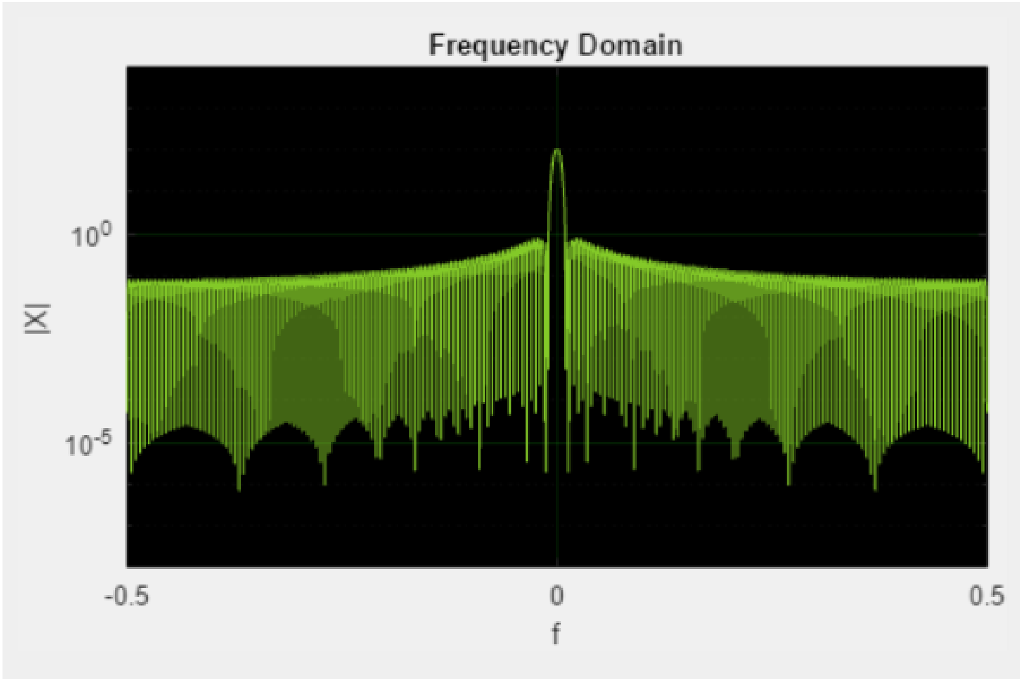

Fig (27) Magnitude of the Frequency Spectrum of the Hamming window on a logarithmic scale

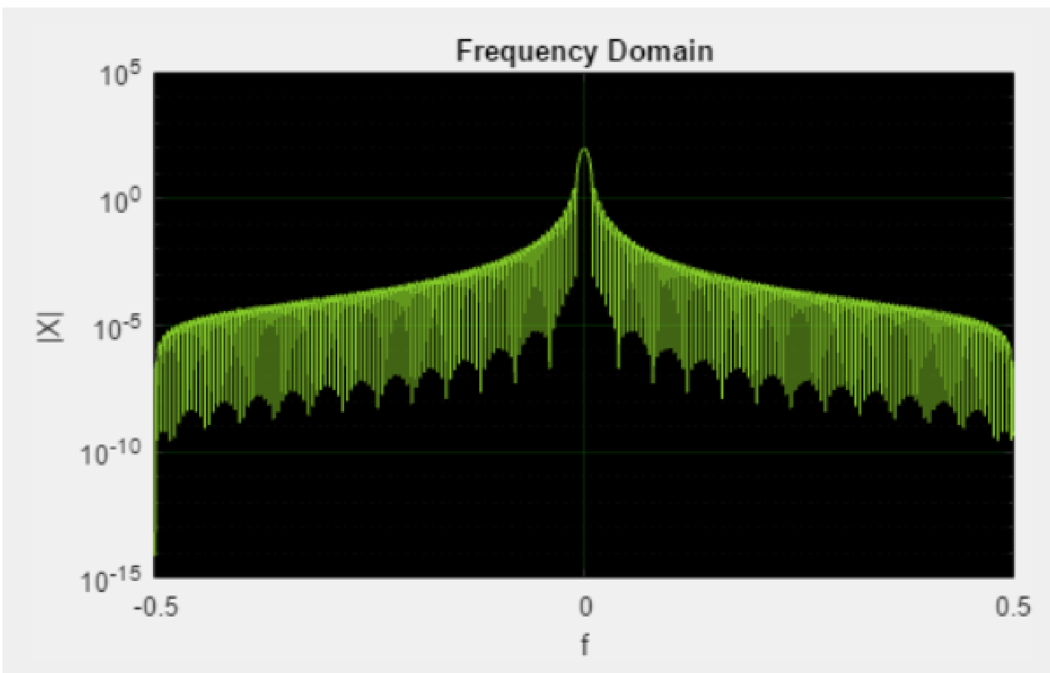

Fig (28) Magnitude of the Frequency Spectrum of the Hanning window on a logarithmic scale

As we are plotting the magnitude of the frequency spectrum, it is not clear how to distinguish between attenuation and gain points in the spectrum in case of Hanning and Hamming windows. However, it is clear that Hamming window can have higher attenuation.

Discrete Convolution

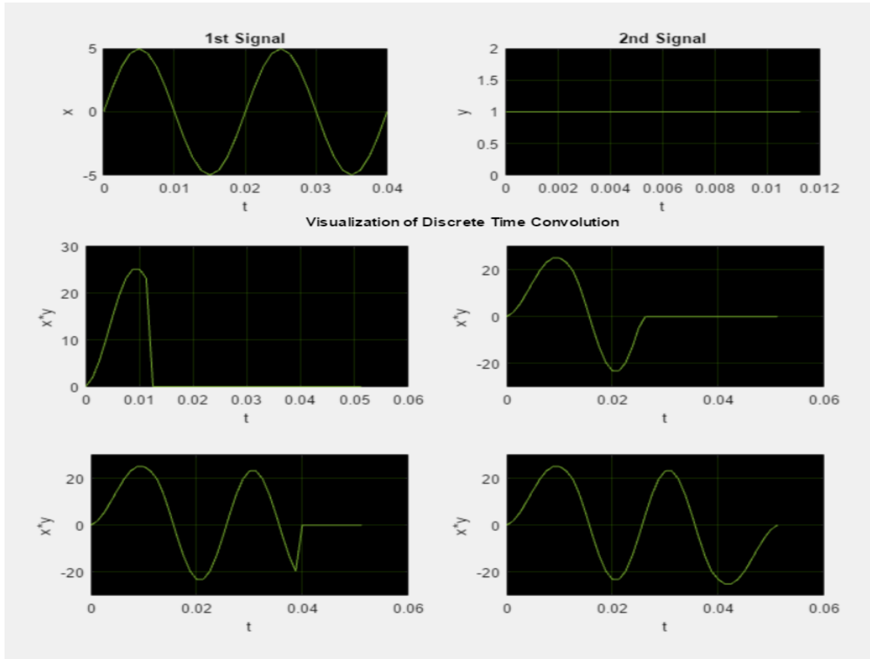

Fig (29) Convolution of a sinusoid with a rectangular function

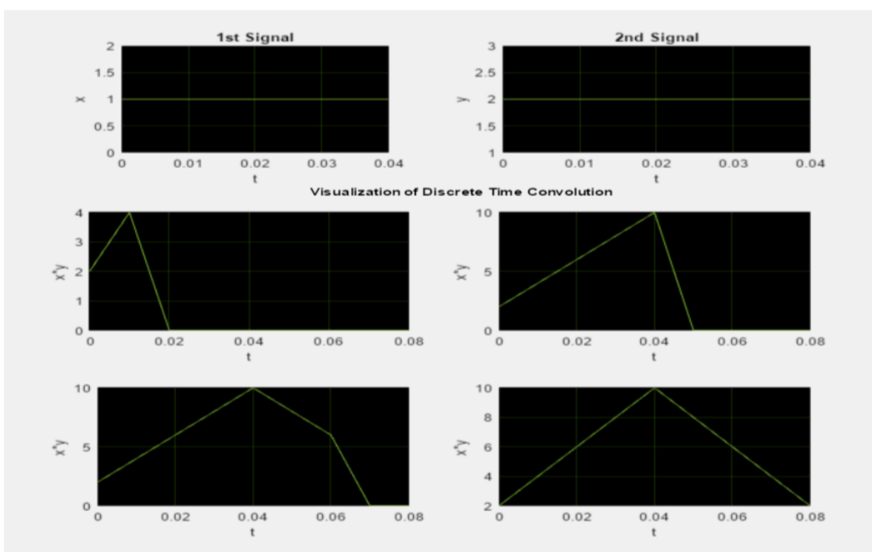

Fig (30) Convolution of a rectangular function with itself

Effect of Filter Order on its Frequency Response

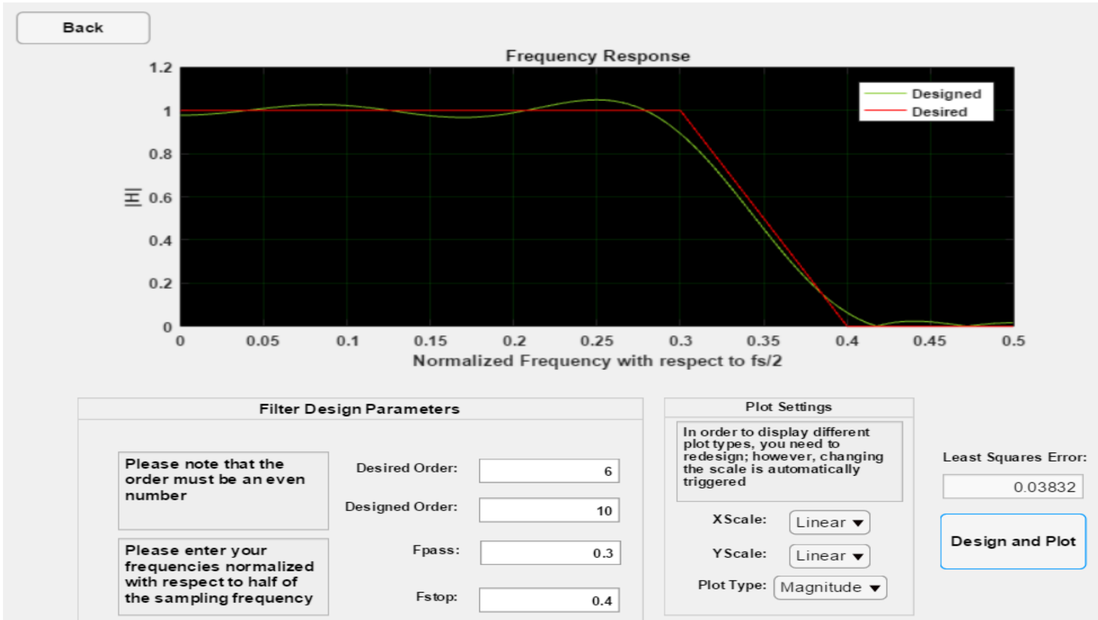

Fig (31) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.3, f_{stop} = 0.4$ & $Order = 10$

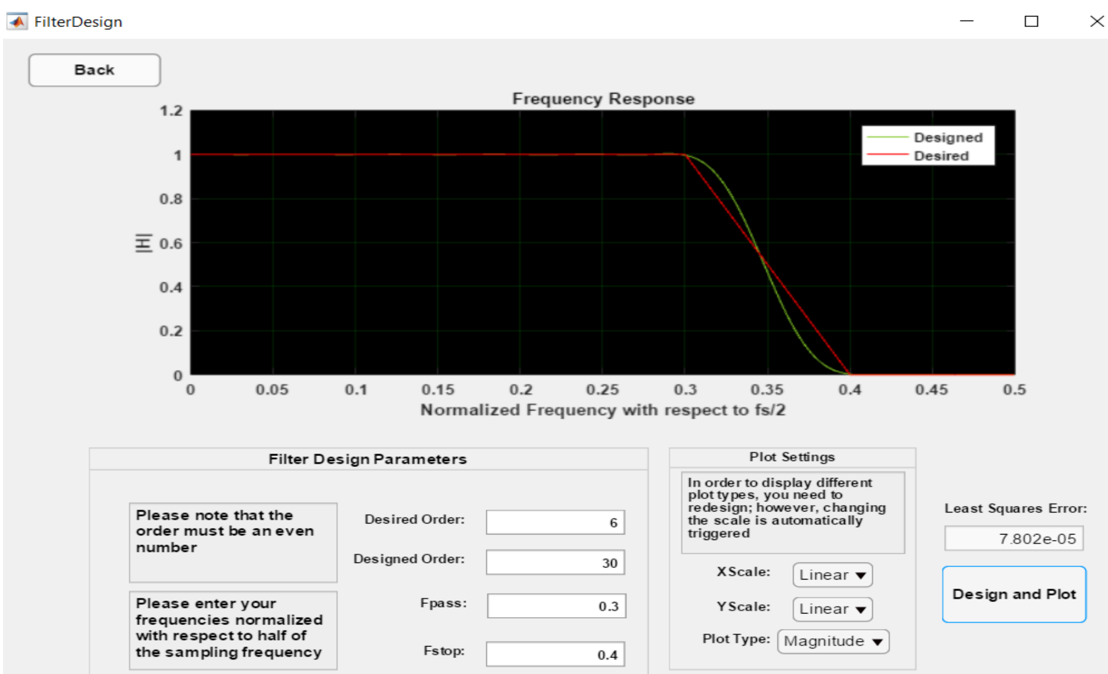

Fig (32) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.3, f_{stop} = 0.4$ & $Order = 30$

Increasing in the filter order leads to better realization of the frequency response which manifests itself in a lower least-squares error. However, the higher the order is, the more complex and expensive is its physical realization.

Effect of the width transition band on the Filter's Frequency Response

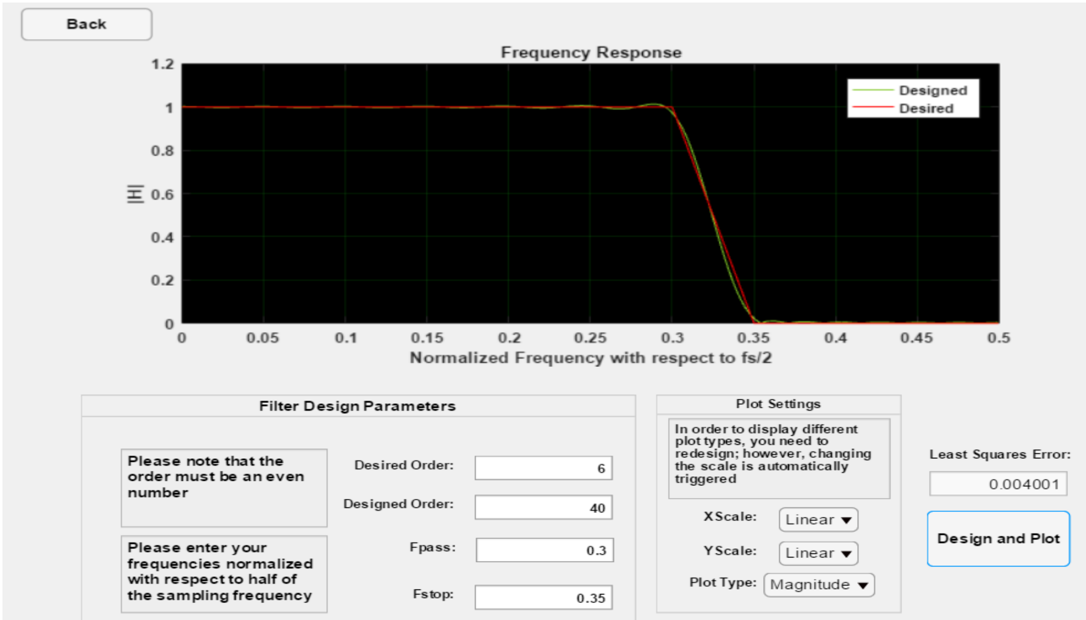

Fig (33) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.3, f_{stop} = 0.35$ & $Order = 40$

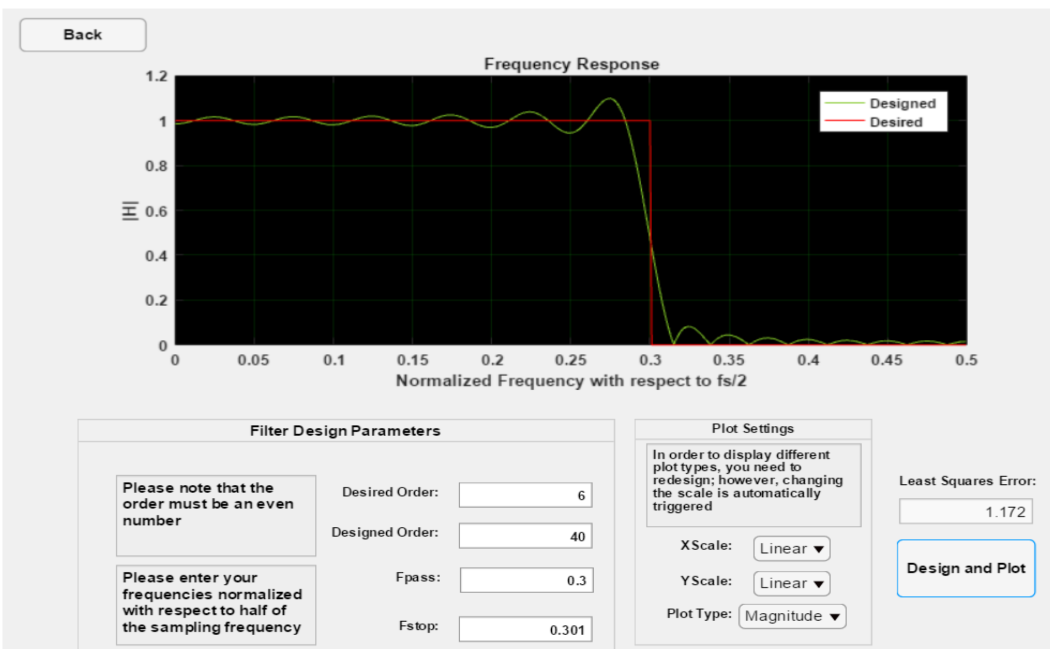

Fig (34) Magnitude of the frequency response of a Low-pass FIR Filter designed using Least-Squares Method $f_{pass} = 0.3, f_{stop} = 0.301$ & $Order = 40$

As the transition band becomes narrower given the same order, the least squares error increases which manifests itself as ripples in the passband and the stopband which compromises the filter performance.

Conclusion

Summarizing our results, we conclude that windowing is a very important tool in spectral analysis, the subtle differences between the windows can lead to disparate inferences about the nature of the spectrum of the signal. So, an appropriate choice of the window is crucial for effective analysis. In Filter Design, choosing the filter order and the width of the transition band are two important parameters that should be carefully fine-tuned in order to meet the performance requirements at a reasonable cost.

References

[1] J. G. Proakis and D. G. Manolakis, *Digital signal processing principles, algorithms and applications*. Upper Saddle River: Pearson Education, 2007.