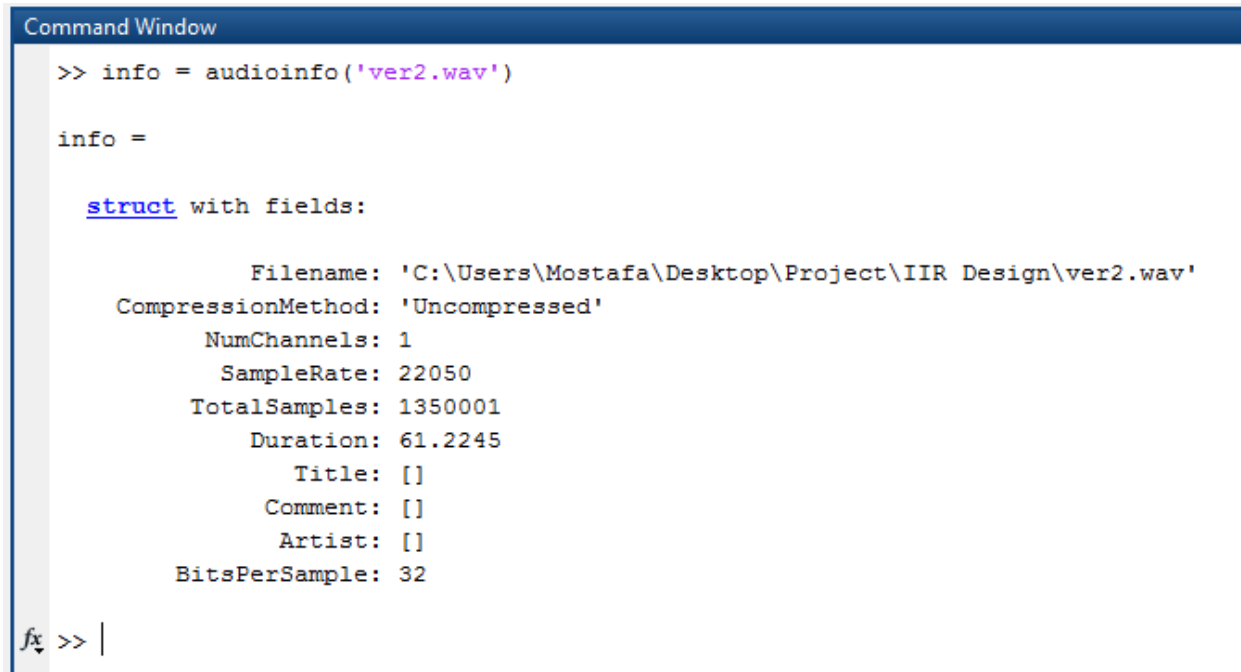


IIR Filter Design.

Intelligence obtained a speech communication recording of a terrorist. However, the signal is corrupted by sinusoidal noises from a siren.

In this part, we have to analyze the recording and remove the corruption as much as possible by designing and implementing Matlab programs to process the signal.

First, we used the method *audioinfo* to get all the information needed about the corrupted wav file as shown in figure below.



```
Command Window
>> info = audioinfo('ver2.wav')

info =

    struct with fields:

        Filename: 'C:\Users\Mostafa\Desktop\Project\IIR Design\ver2.wav'
    CompressionMethod: 'Uncompressed'
        NumChannels: 1
        SampleRate: 22050
    TotalSamples: 1350001
        Duration: 61.2245
        Title: []
        Comment: []
        Artist: []
    BitsPerSample: 32

fx >> |
```

Fig.2.1: Shows the information of the corrupted file.

We used the method *audioread* that takes the audio file as input to get the samples vector and the sampling rate F_s , it is defined by:

$$[x, F_s] = \text{audioread}('ver2.wav')$$

So x now is the sampling vector and F_s is now the sampling rate.

We used the method *sound*($x(1:5 * F_s), F_s$) to listen for 5 seconds of the corrupted audio file.

Now we have to compute the magnitude spectrum of the signal in dB using DFT.

L now is the length of vector x using $L = \text{length}(x)$ and $NFFT = 2\text{nextpow2}(L)$ to get the suitable size for the DFT of x .

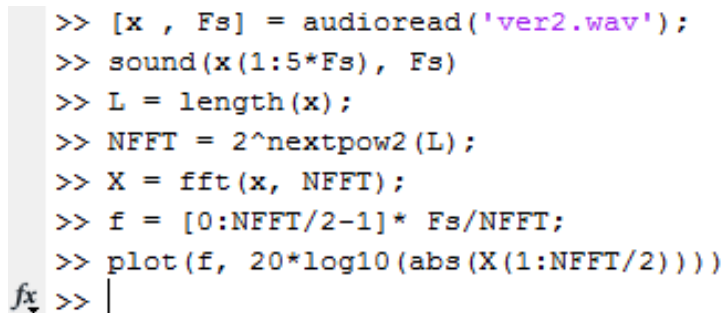
Also the method *fft* is used to compute the discrete Fourier transform (DFT) of *x* using a fast Fourier transform (FFT) and its defined as follows:

$$X = \text{fft}(x, NFFT);$$

We define the frequency domain *f* and plot the single-sided amplitude spectrum in decibel using the MatLab function *plot(vector1, vector2)* as follows:

$$f = [0:NFFT/2-1] * Fs/NFFT;$$
$$\text{plot}(f, 20 * \log_{10}(\text{abs}(X(1:NFFT/2))))$$

Here is the screenshots of the code of these steps, then a screenshot of the function the magnitude spectrum of the corrupted signal.

A screenshot of a MATLAB command window showing the code for computing the magnitude spectrum. The code is as follows:

```
>> [x , Fs] = audioread('ver2.wav');
>> sound(x(1:5*Fs), Fs)
>> L = length(x);
>> NFFT = 2^nextpow2(L);
>> X = fft(x, NFFT);
>> f = [0:NFFT/2-1]* Fs/NFFT;
>> plot(f, 20*log10(abs(X(1:NFFT/2))))
fx >> |
```

A horizontal blue line is drawn below the code.

Fig.2.2.1: Shows the code of computing the magnitude spectrum of the signal.

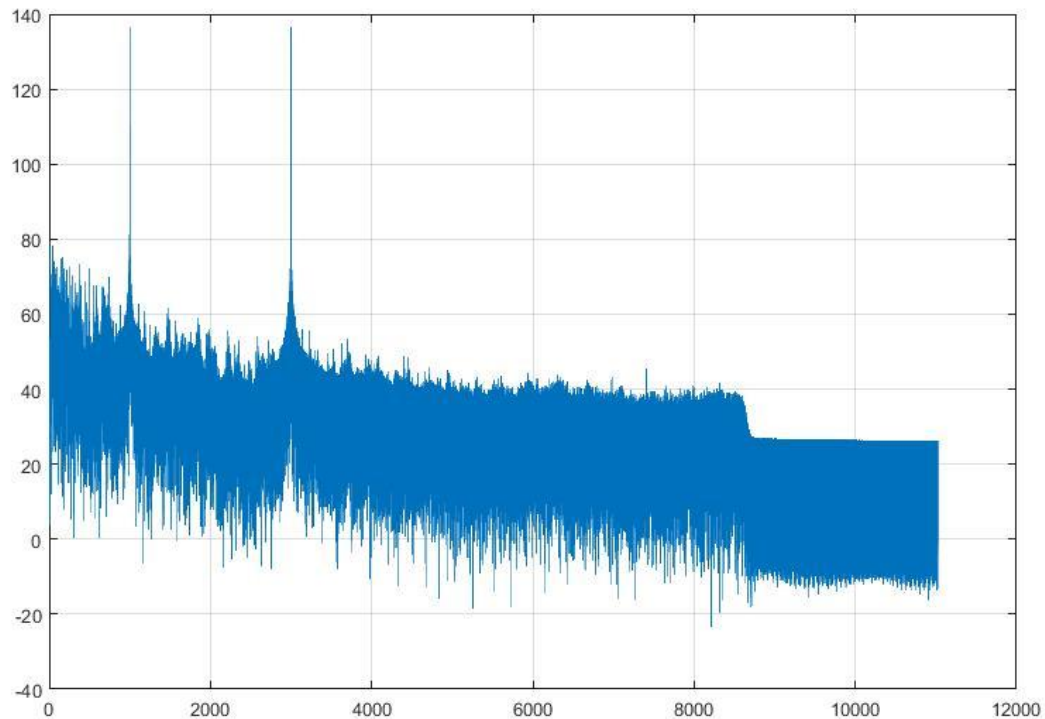


Fig.2.2.2: Shows the magnitude spectrum of the corrupted signal.

After making the grid on, we notice that on the frequency 1000 Hz and 3000 Hz we have a siren noise of 140 dB.

We will design an IIR filter using Butterworth design methods to reject the interference signal and obtain the SOS coefficients of the filter using *fdatool*.

It's a bandstop filter on a sampling rate $F_s=22050$ which is the same sampling rate of the corrupted signal, and centered on the first siren noise spectrum of 1000Hz, and an amplitude stop of 60 dB because the noise on 140 dB and the original signal was on 80 dB. These specs are shown in figure below. Also the filter is attached as .fda file.

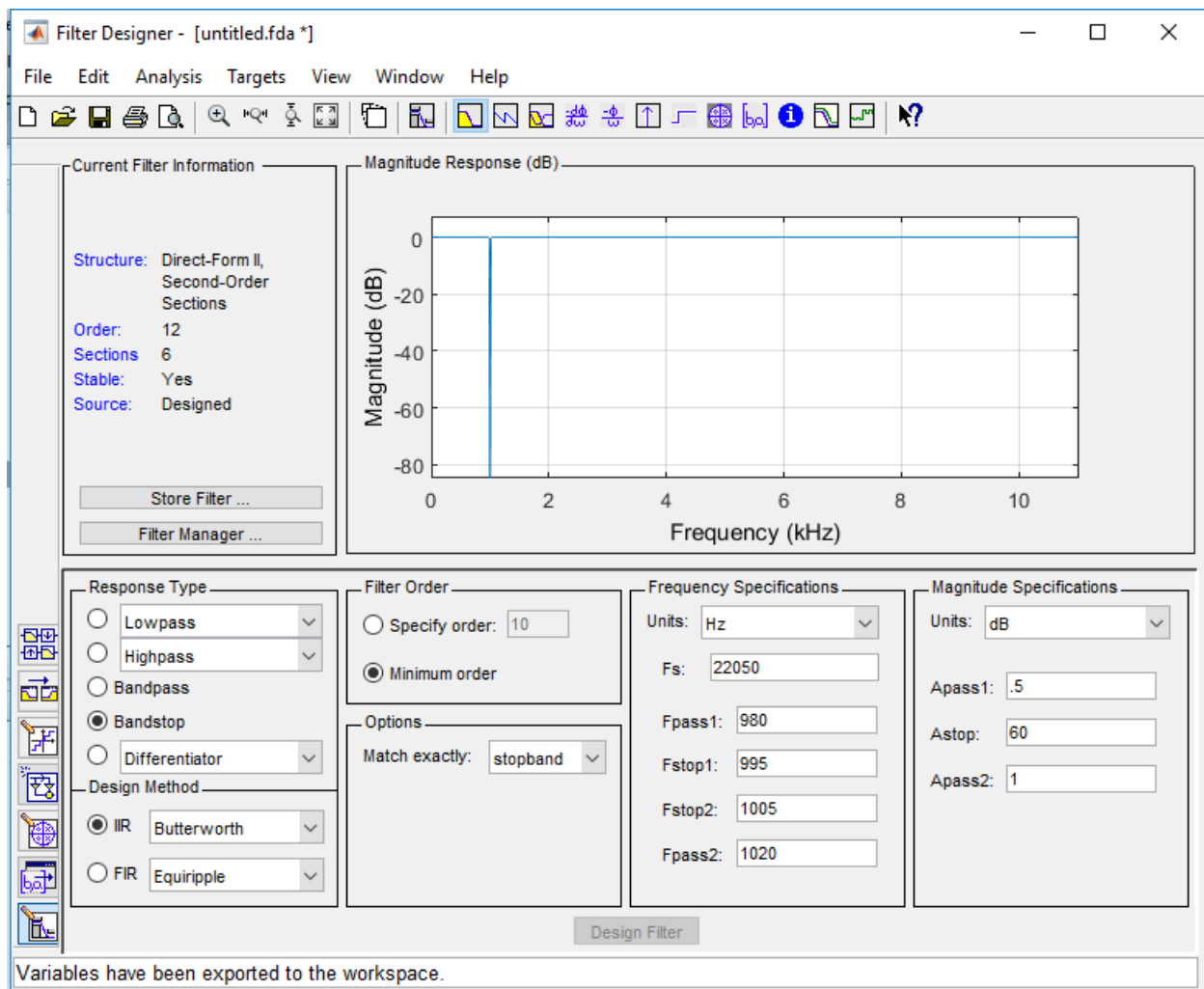


Fig.2.3.1: Shows the First filter specs on fdatool.

After setting the specs of filter we plot the frequency response (magnitude) in decibel of the filter as shown in Fig.2.3.2 using the method $\text{Freqz}(\text{SOS1})$ where SOS1 are the coefficients of the filter.

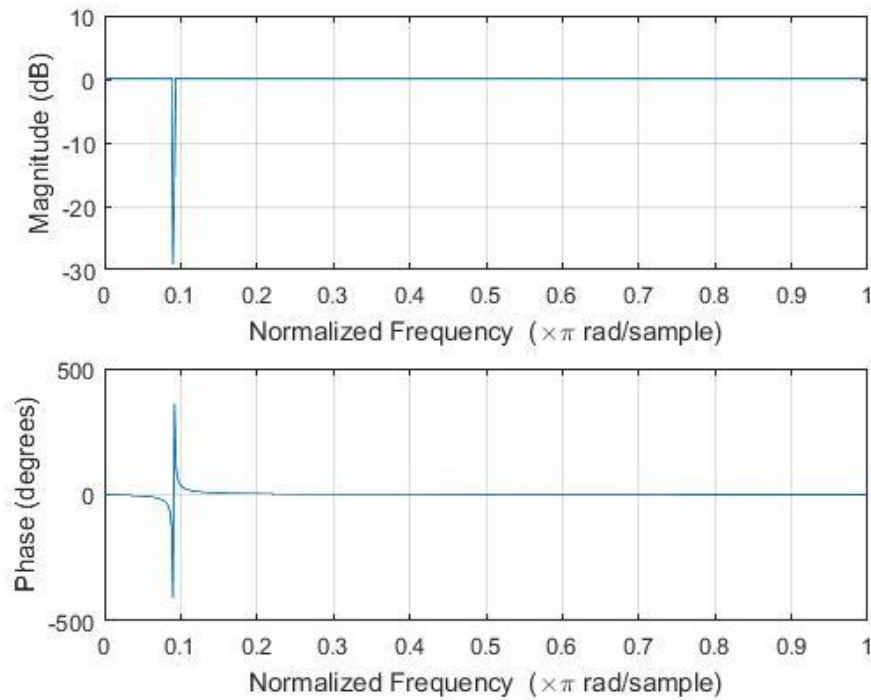


Fig.2.3.2: Shows the the frequency response (magnitude) in decibel of the filter1.

Using *sosfilt(SOS1,x)* method we process x and plot the magnitude spectrum in dB of the filtered signal on the same figure as the corrupted signal using *fft* and *plot* methods as shown below.

```
y1=sosfilt(SOS1,x);
Y1= fft(y1, NFFT);
plot(f, 20*log10(abs(X(1:NFFT/2))) , f , 20*log10(abs(Y1(1:NFFT/2))))
```

Here is the corrupted and the filtered signal after processing the first filter and ignoring the first spectrum of the siren noise.

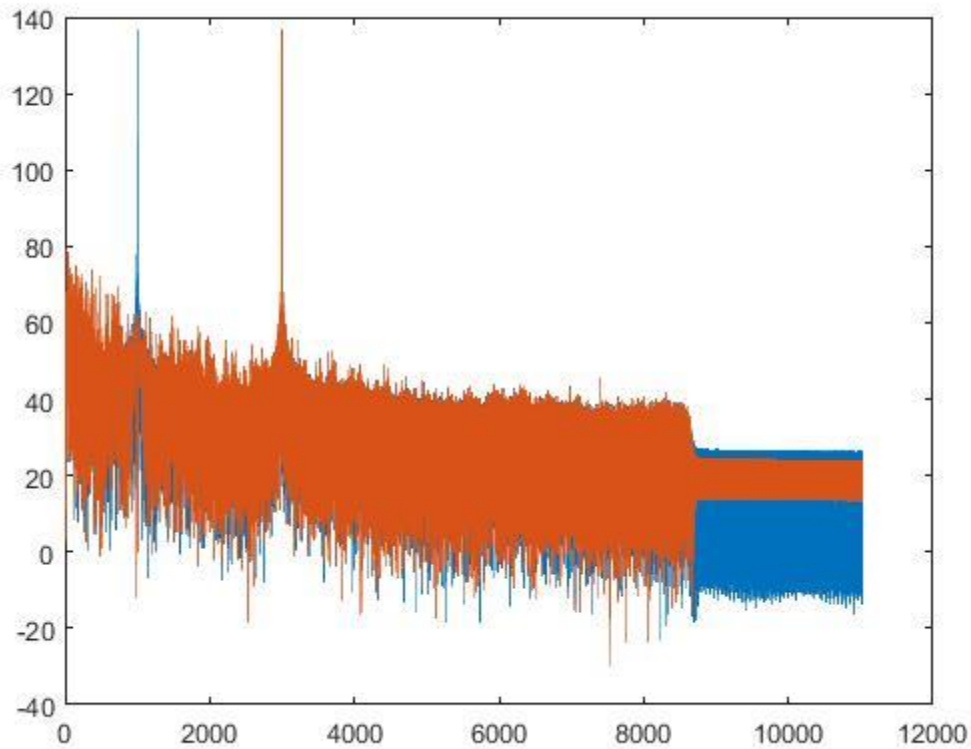


Fig.2.3.3: Shows the corrupted signal and the filtered signal using filter1.

Now we have to ignore the second spectrum of noise, so we repeat the same steps of filter 1 by processing another filter.

It's a bandstop filter on a sampling rate $F_s=22050$ which is the same sampling rate of the corrupted signal, and centered on the first siren noise spectrum of 3000Hz, and an amplitude stop of 60 dB because the noise on 140 dB and the original signal was on 80 dB. These specs are shown in figure below. Also the filter is attached as .fda file.

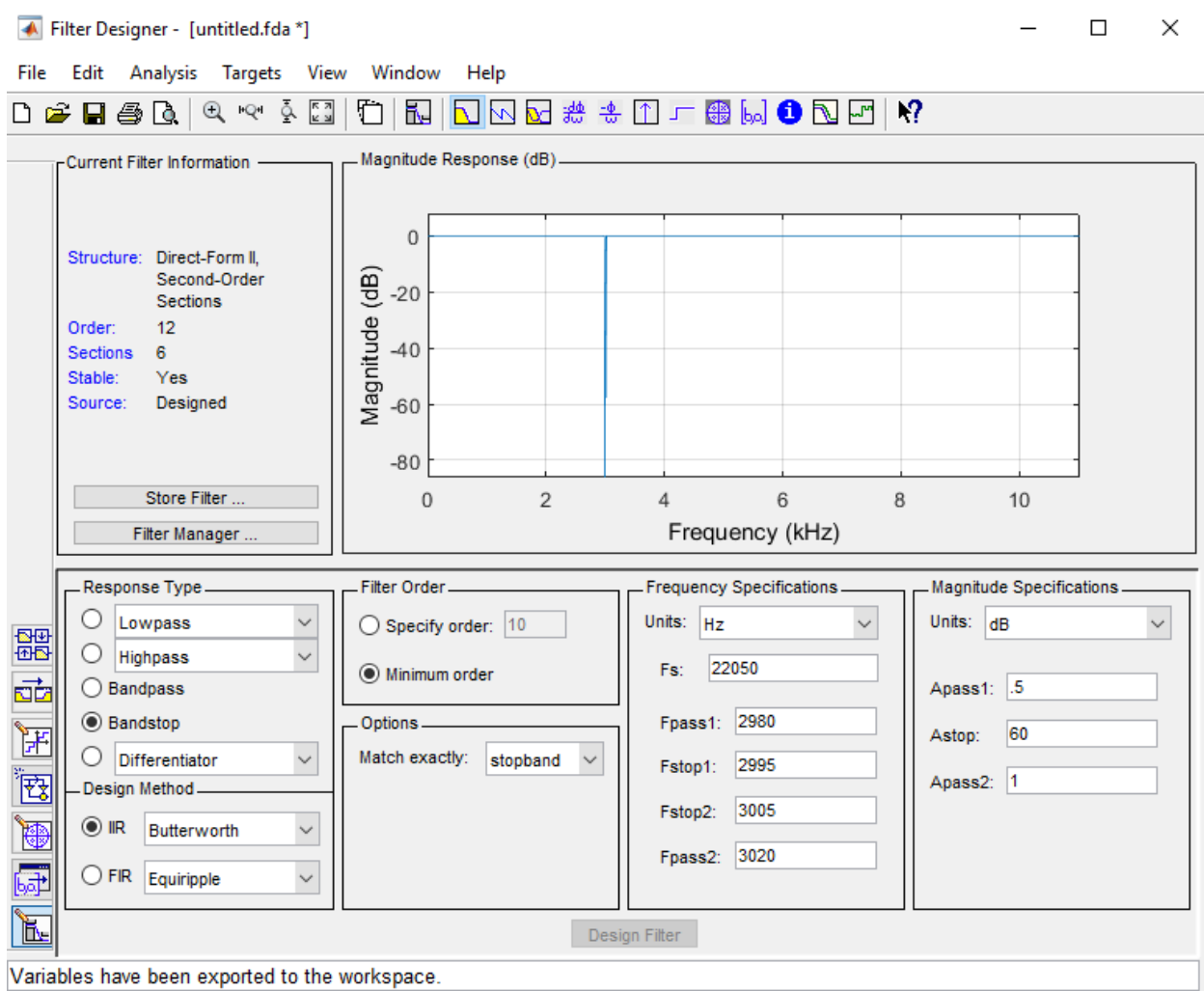


Fig.2.4.1: Shows the First filter specs on fdatool.

After setting the specs of filter we plot the frequency response (magnitude) in decibel of the filter as shown in Fig.2.4.2 using the method $\text{Freqz}(\text{SOS2})$ where SOS2 are the coefficients of the second filter.

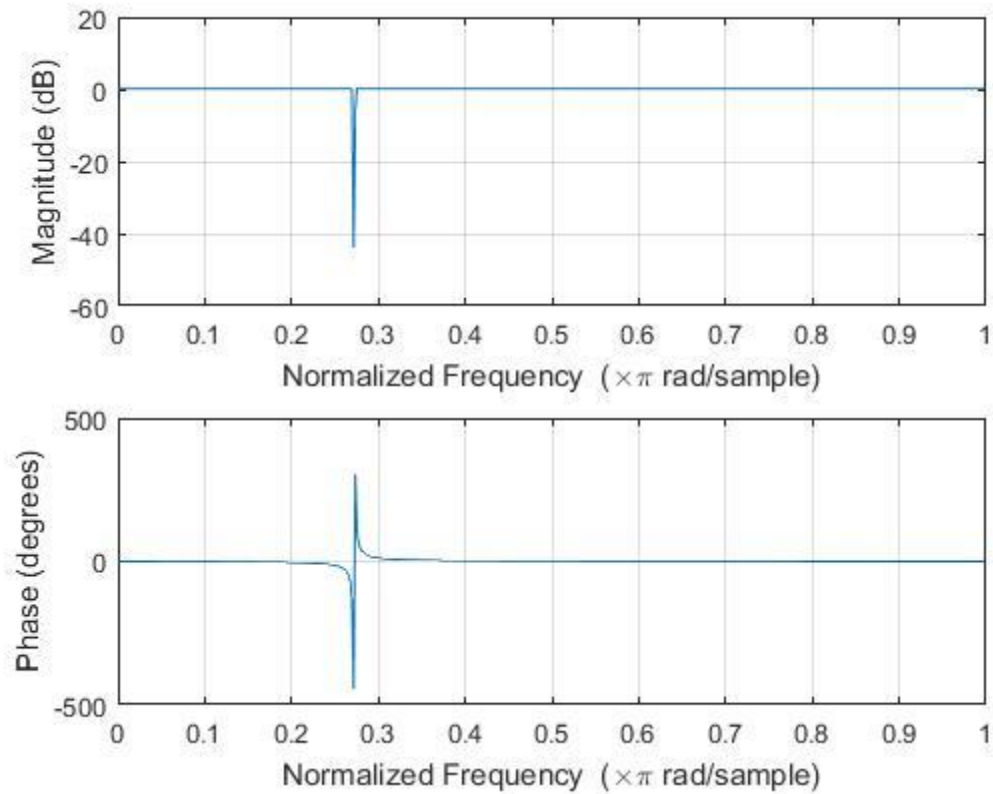


Fig.2.4.2: Shows the the frequency response (magnitude) in decibel of the filter2.

Using `sosfilt(SOS2,y1)` method we process `y1` and plot the magnitude spectrum in dB of the filtered signal on the same figure as first filtered signal using *fft and plot* methods as shown below.

```
y2=sosfilt(SOS2,y1);
Y2= fft(y2, NFFT);
plot(f, 20*log10(abs(Y1(1:NFFT/2))) , f , 20*log10(abs(Y2(1:NFFT/2))))
```

Here is the corrupted and the filtered signal after processing the second filter and ignoring the second spectrum of the siren noise.

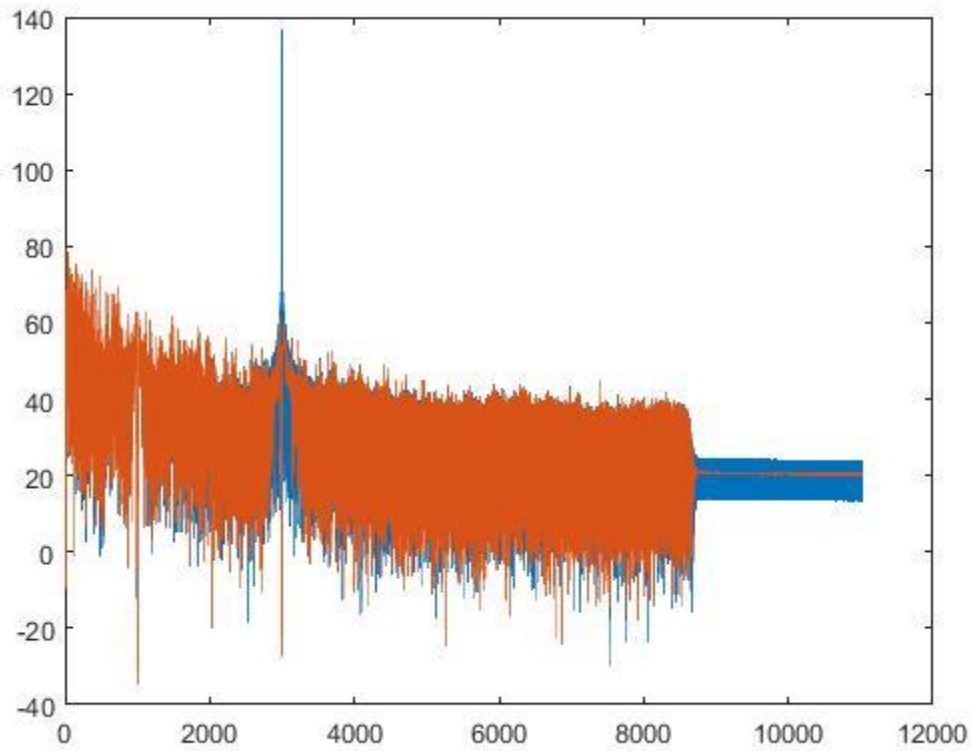


Fig.2.3.3: Shows the corrupted signal and the filtered signal using filter2.

Finally, we convert our result into a wave file named 'Ibrahim-Mostafa-Ghamloush-Moslem.wav' and you can listen to it 😊