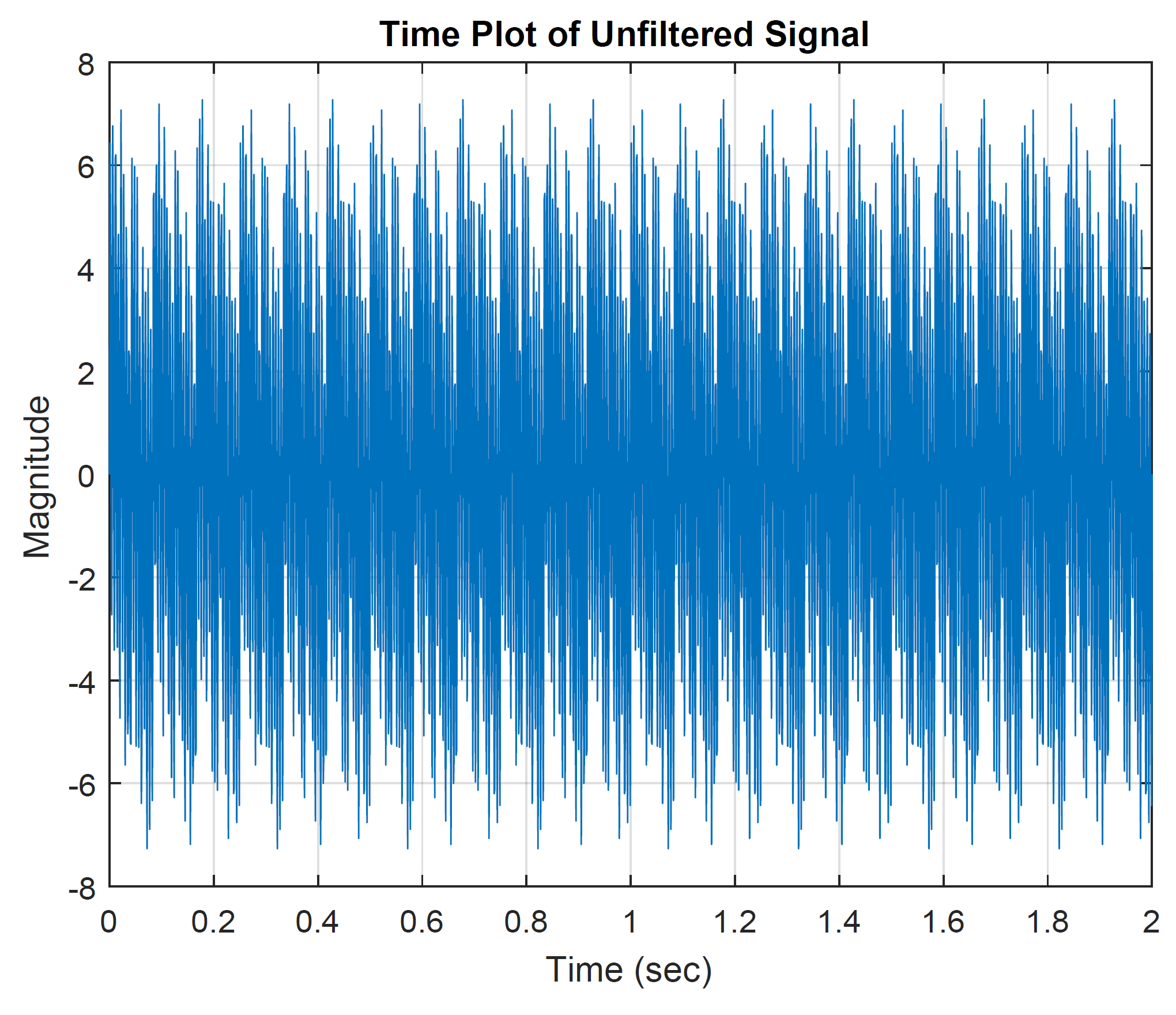
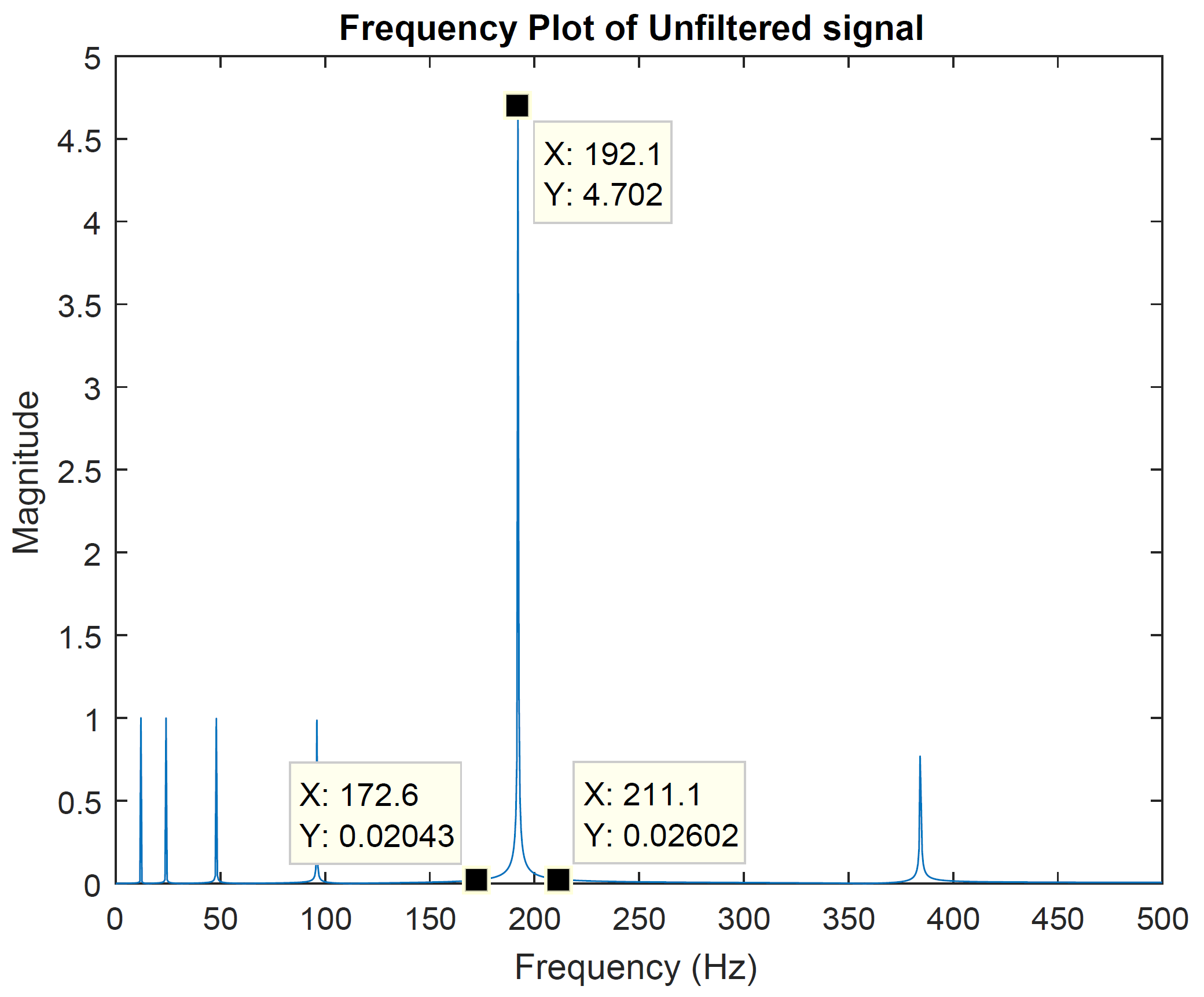
1. Analyze the unfiltered signal in the time domain



2. Analyze the unfiltered signal in the frequency domain



3. Design a suitable filter (order, frequencies, etc.) according to the given criteria.

**Given Criteria**

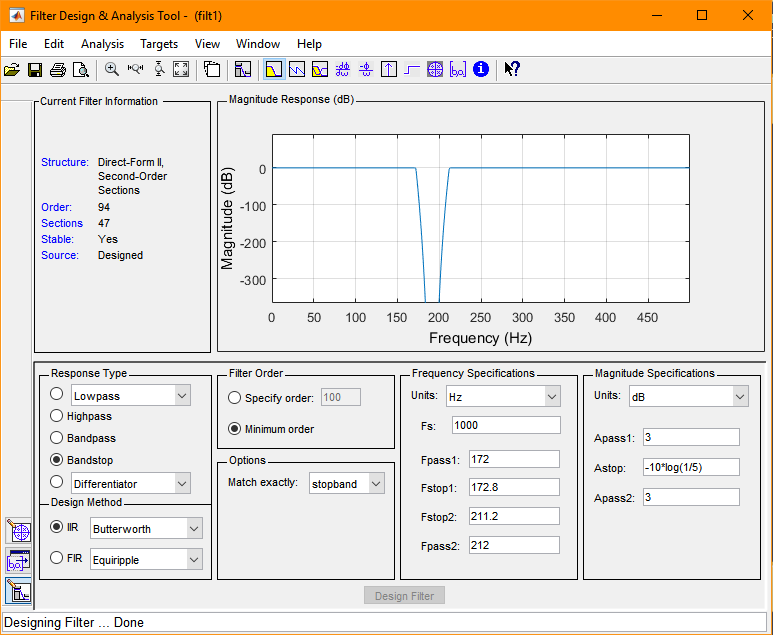
***Undesired Frequency Range:* 172.8-211.2 Hz** for the data file**: *ADC11***

*This requires us to design a ‘bandstop/notch’ type filter, for which*

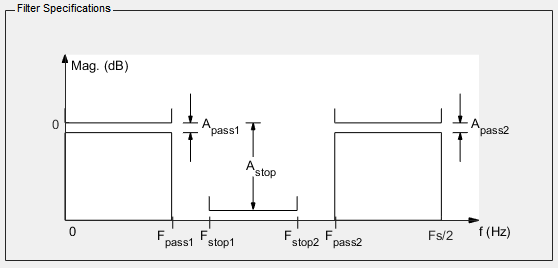
**Stopband Attenuation ()**

**[This corresponds to a maximum allowable gain of: ]**

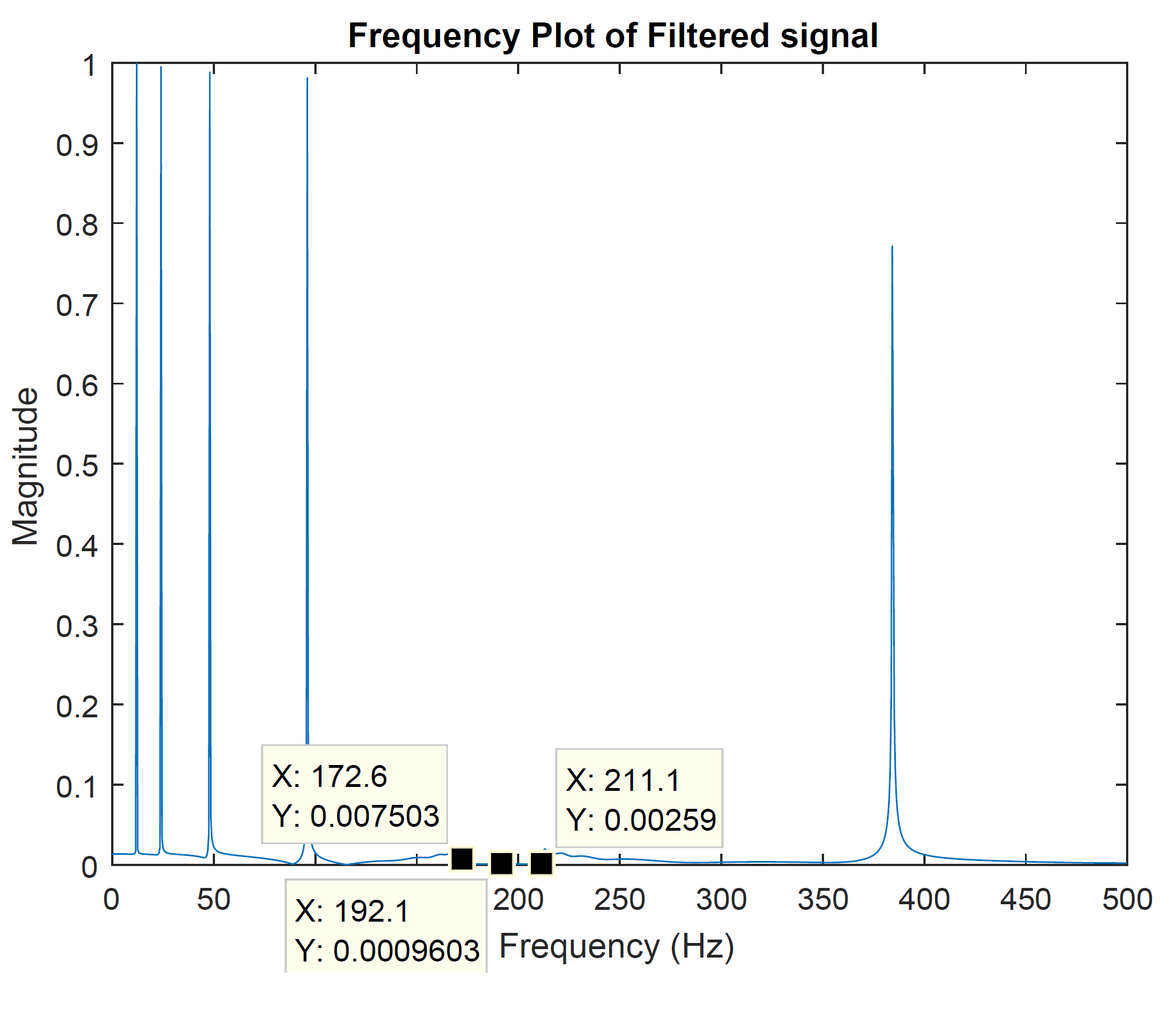
The MATLAB app, *Signal Anlaysis*, is used to implement the filter designed through the app named *Filter Design & Analysis*.



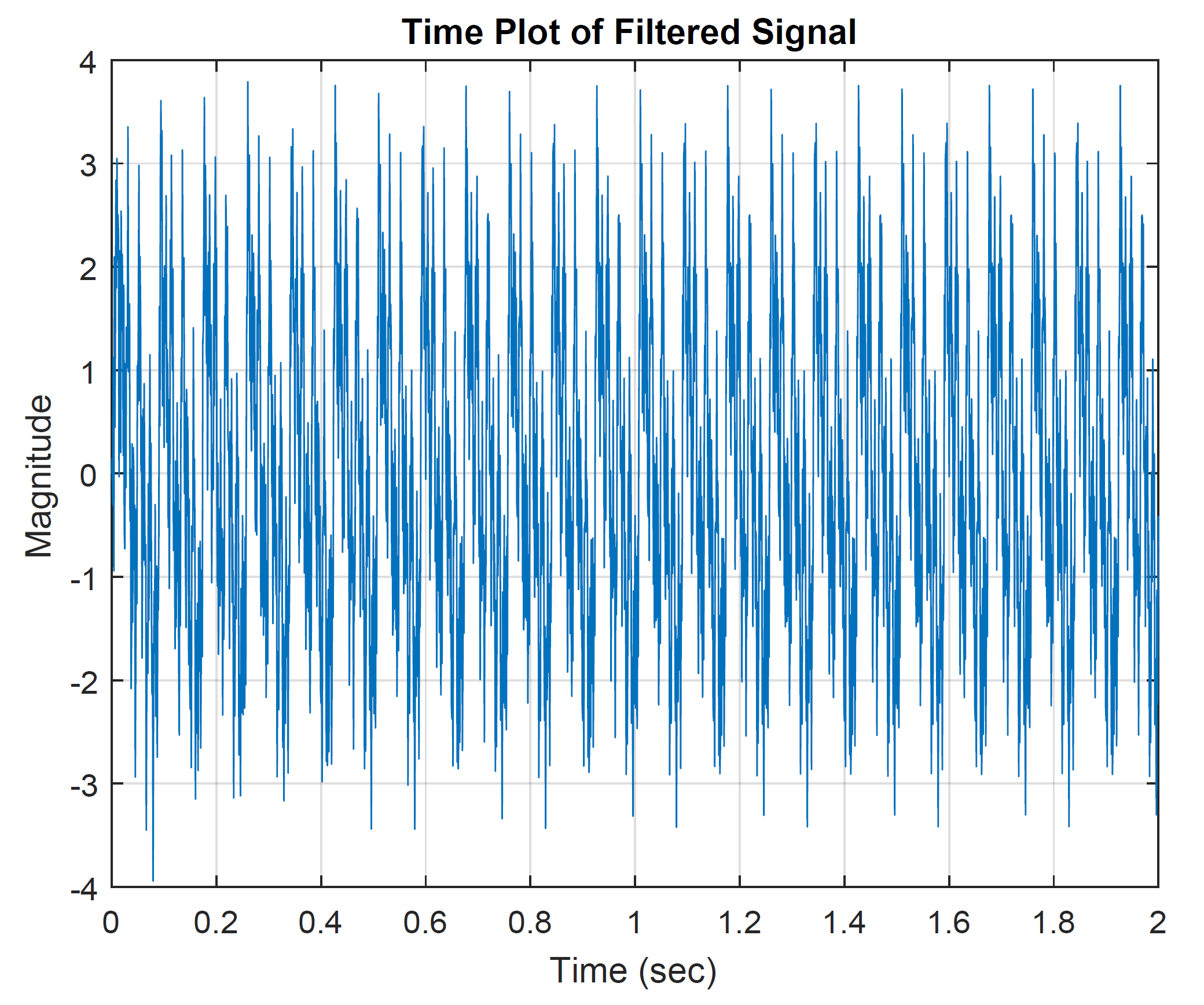
Where the filter specifications above are illustrated as;



5. Analyze the filtered signal in the frequency domain

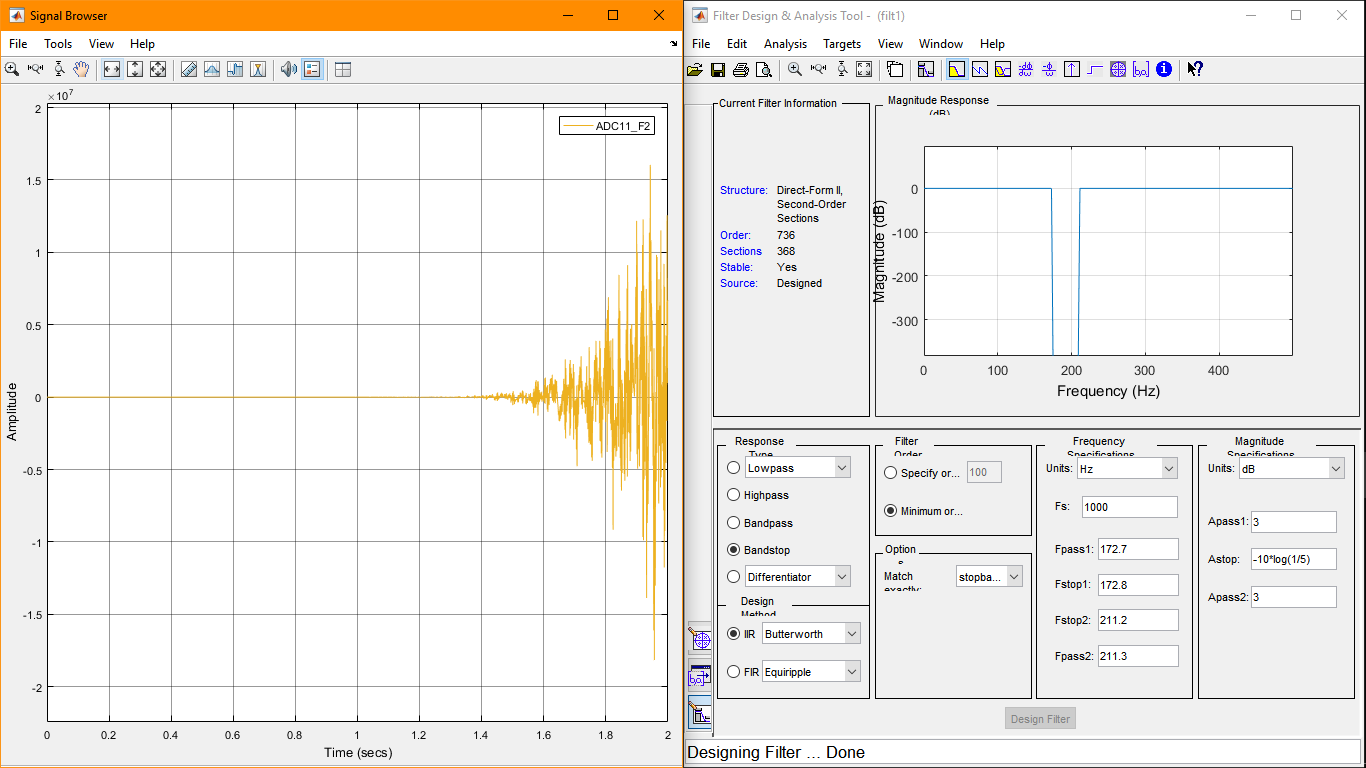


6. Analyze the filtered signal in the time domain

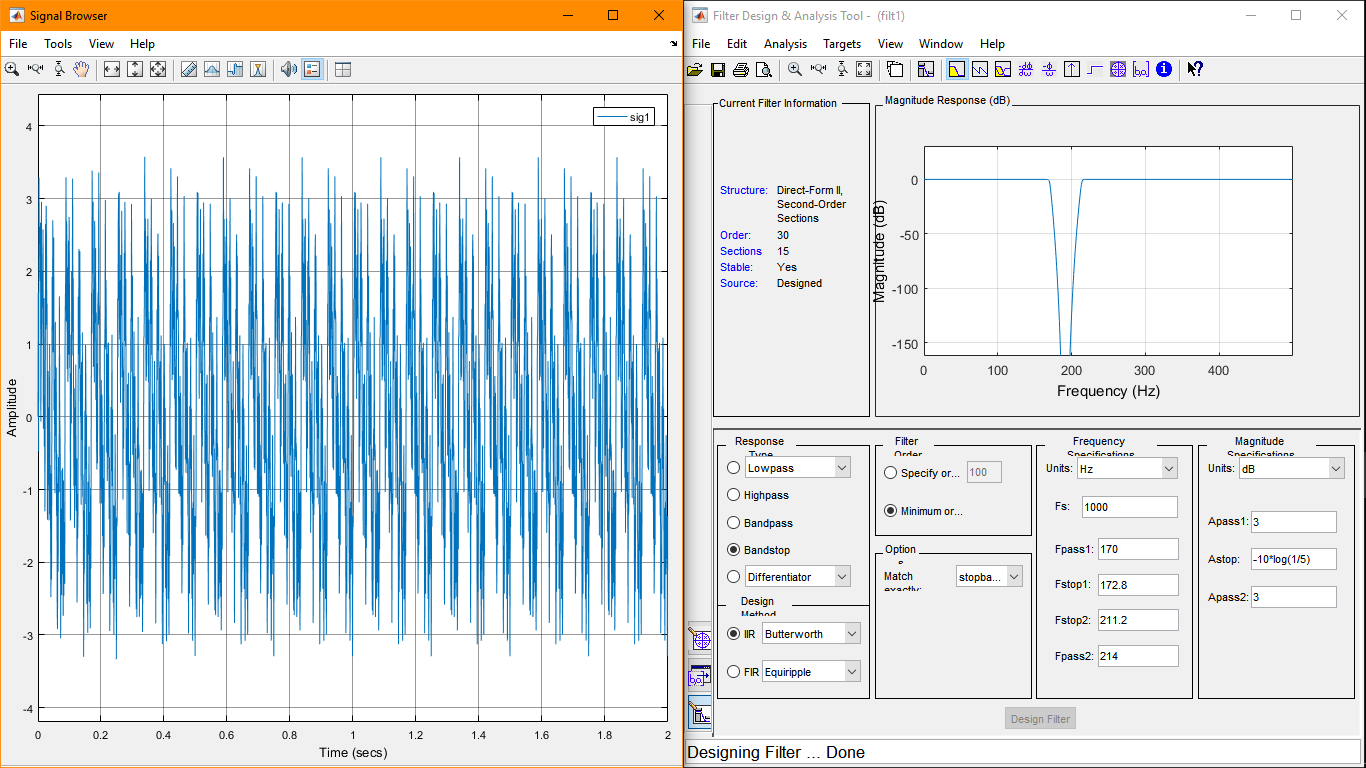


7. Redesign the filter if required.

* By narrowing the filter is noticed to become unstable.



* By widening , the filter yields acceptable results and reduced order, hence reduced cost of implementing.



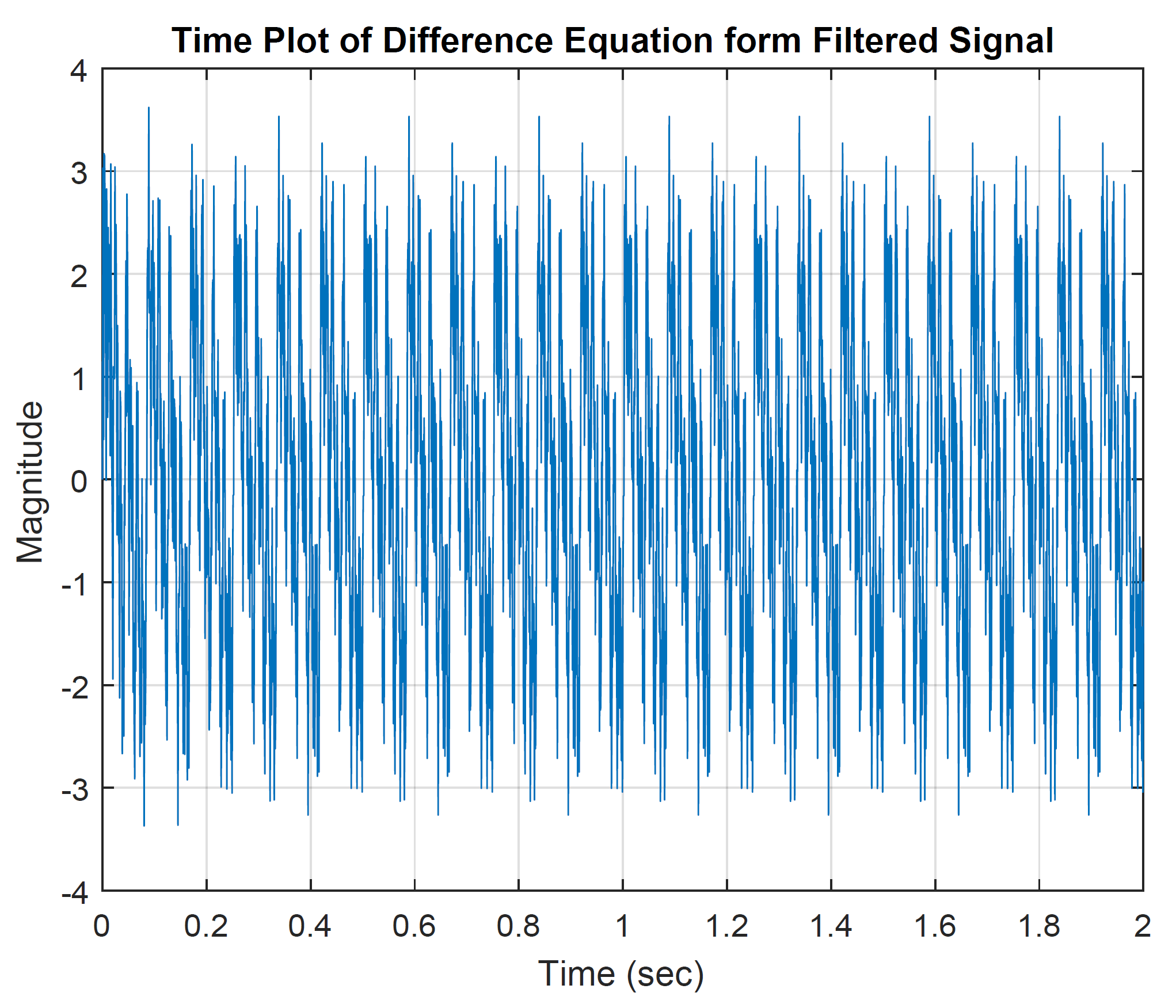
8. Implement the digital filter in the form of a DIFFERENCE EQUATION in Matlab script language and analyze the filtered signal.

The MATLAB function buttordgives a suitable order as and a normalised *cutoff* frequency range [ for the following parameters:

|  |  |
| --- | --- |
| **Parameter** | **Value(s)** |
| (Normalised) Passband corner frequency () |  |
| (Normalised) Stopband corner frequencies () |  |
| Maximum Passband Ripple () |  |
| Minimum Stopband Attenuation () |  |

These are then used with the function, butter, to generate coefficients for the difference equation.

***[N.B. The same passband frequencies resulted in instability, thus we widened the passband]***



Comparing plots in the result of step 6 and 8. These are close enough.

Analysis code and implemented digital filter code:

% The signal is sampled at 1000 Hz

FS = 1000 ; % Sampling frequency

% According to Nyquist theorm we can only have frequencies upto 500 Hz

% in this signal. The function butter take normalised frequency.

FN = FS/2; % Nyquist frequency

figure(1)

plot (t,x) ; grid

title ('Time Plot of Unfiltered Signal')

xlabel('Time (sec)')

ylabel('Magnitude')

% Plot first half of DFT (normalised frequency)

X\_mags = abs(fft(x))/FS;

num\_bins = length(X\_mags);

figure(2)

plot([0:1/(num\_bins/2 -1):1]\*FN, X\_mags(1:(num\_bins-1)/2))

title ('Frequency Plot of Unfiltered signal')

xlabel('Frequency (Hz)')

ylabel('Magnitude')

format long

x\_dfiltered=(ADC11\_F.data)';

% Plot first half of DFT (normalised frequency)

X\_mags = abs(fft(x\_dfiltered))/FS;

num\_bins = length(X\_mags);

figure(4)

plot([0:1/(num\_bins/2 -1):1]\*FN, X\_mags(1:(num\_bins-1)/2))

title ('Frequency Plot of Filtered signal')

xlabel('Frequency (Hz)')

ylabel('Magnitude')

figure(5)

plot (t,x\_dfiltered) ; grid

title ('Time Plot of Filtered Signal')

xlabel('Time (sec)')

ylabel('Magnitude')

% Design butterworth filter

Wp=[171 213]./FN%Passband (normalised) corner frequencies (rad/sample)

Ws=[172.8 211.2]./FN%Stopband (normalised) corner frequencies (rad/sample)

Rp=3%Maximum Passband Ripple (dB)

Rs=-10\*log10(1/5)%Minimum Attenuation of the Undesired Frequency range (dB)

[n,Wn] = buttord(Wp,Ws,Rp,Rs) %n: Order of the Filter Wn: Normalized cutoff frequencies

[b a] = butter(n, Wn, 'stop');

%plot the frequency response (normalised frequency)

H = freqz(b,a, floor(num\_bins/2));

figure(3);

plot([0:1/(num\_bins/2 -1):1]\*FN, abs(H),'r');grid

title ('Frequency Response of Difference Equation form Filter')

xlabel('Frequency (Hz)')

ylabel('Magnitude')

x\_DEfiltered=filter(b, a, x);

% Plot first half of DFT (normalised frequency)

X\_mags = abs(fft(x\_DEfiltered))/FS;

num\_bins = length(X\_mags);

figure(6)

plot([0:1/(num\_bins/2 -1):1]\*FN, X\_mags(1:(num\_bins-1)/2))

title ('Frequency Plot of Difference Equation form Filtered signal')

xlabel('Frequency (Hz)')

ylabel('Magnitude')

figure(7)

plot (t,x\_DEfiltered) ; grid

title ('Time Plot of Difference Equation form Filtered Signal')

xlabel('Time (sec)')

ylabel('Magnitude')