# ELECENG 3TR4 Lab 1: Fourier Analysis

Aaron Pinto
pintoa9

Raeed Hassan
hassam41

February 5, 2021

# Design Problem

The output of a function generator is a square wave with 50% duty cycle and the period of the wave is 0.1 ms. You may assume that the amplitude of the square wave is 1 Volt. Design a second order Butterworth filter so that any harmonics in the output is at least 23 dB below the fundamental sinusoid at 10 kHz, and the loss of the fundamental sinusoid due to filtering should be less than 2 dB. The purpose of this experiment is to generate a high amplitude sinusoidal signal out of a square wave (to have a high quality sinusoid, the amplitudes of harmonics should be sufficiently low). You should choose the cutoff frequency of the filter (i.e. 3 dB bandwidth of the filter) such that the above design constraints are met.

# Determining the Frequency Response of the Filter

The first step we took was to implement a second order Butterworth filter in MATLAB and examine its frequency response. To simulate the filter, the transfer function of a second order Butterworth filter was used. The transfer function that was used for the second order Butterworth filter was provided in the textbook.

$$H(s) = \frac{1}{s^2 + 1.414s + 1}$$

The MATLAB script used to simulate the filter, frequencyResponse.m, generated the frequency response of the filter with its transfer function using the freqs function. A portion of the MATLAB code used to generate and plot the frequency response of the filter is shown below in Listing 1.

Listing 1: Generating the Frequency Response of Second Order Butterworth Filter

```
23 range = ceil(third_harm_freq/fc) + 1;
24 w = linspace(-range,range,500);
25 h = freqs(b,a,w); % a for poles, b for zeros, w for normalized
      freq range from -5 to 5 rad/s
26 mag = 20*log10(abs(h)); %% convert magnitude to dB
27 %phase = angle(h);
28 %phasedeg = phase*180/pi;
29
30 %% plot frequency response
31 % normalized frequency response
32 figure(1)
33 subplot(2,1,1)
34 plt1 = plot(w,mag);
35 xlim([w(1) w(length(w))]);
36 grid on
37 title('Normalized Frequency Response (Magnitude)');
38 xlabel('Frequency (rad/s)')
39 ylabel('Magnitude (dB)')
```

```
xline(1,'k',{'Cutoff frequency'}); % 3dB frequency (at cutoff
    frequency)

% frequency response
subplot(2,1,2)
plt2 = plot(fc*w,mag);
xlim([fc*w(1) fc*w(length(w))]);
plt2ax = ancestor(plt2(1), 'axes'); plt2ax.XAxis.Exponent = 0;
grid on
title('Frequency Response (Magnitude)');
xlabel('Frequency (Hz)')
ylabel('Magnitude (dB)')
xline(fc,'b',{'Cutoff frequency'}); % 3dB frequency (at cutoff
    frequency)
xline(fund_freq,'r',{'Fundamental frequency'}); % first
    harmonic
xline(third_harm_freq,'m',{'3rd harmonic'}); % third harmonic
```

The MATLAB script was used to visualize the normalized frequency response of the filter, and to determine the attenuation provided by the filter at frequencies of interest (the fundamental frequency, and the frequency of the third harmonic). An example of plots generated by script is shown in Figure 1.

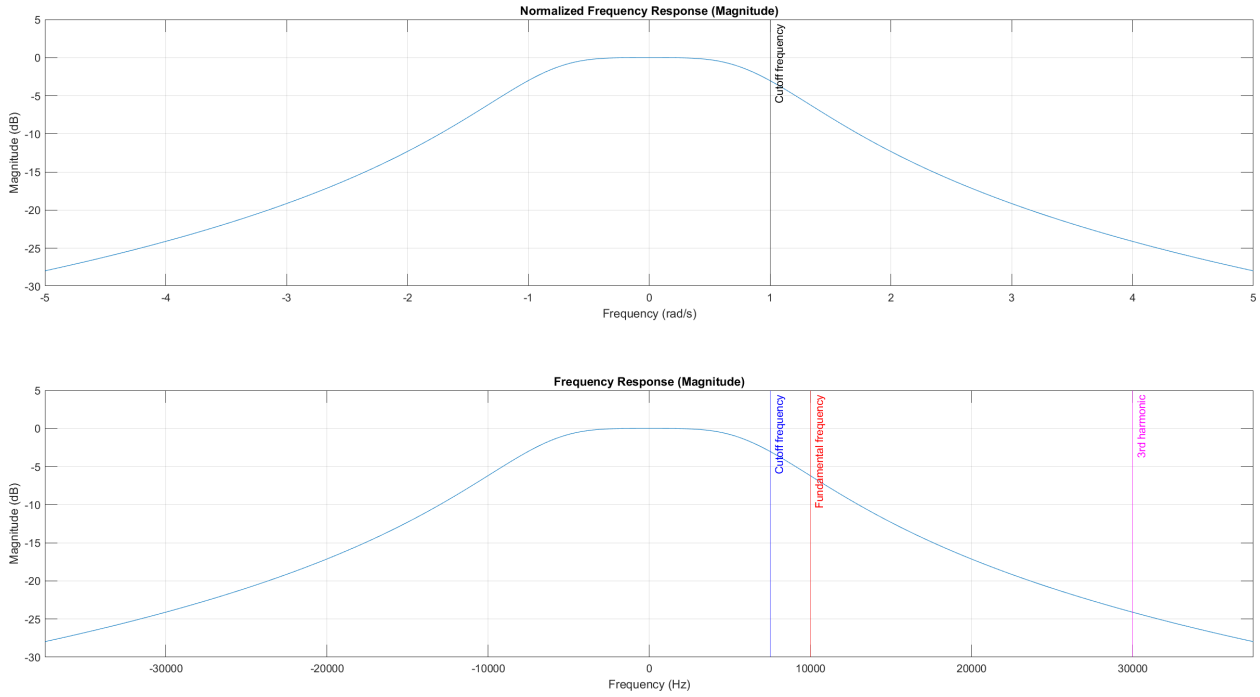


Figure 1: The frequency response plots with the cutoff frequency at 7.5 kHz

# Determining the Cutoff Frequency

The MATLAB script was used to determine the appropriate cutoff frequency to meet the design constraints. The first design constraint that we investigated required the loss of the fundamental sinusoid due to filtering to be less than 2 dB. We determined that this occurred when the cutoff frequency was greater than 11430 Hz. We selected a cutoff frequency of 11500 Hz, where the attenuation at the fundamental frequency was approximately 1.96 dB, as our starting point to continue and explore the other design constraints.

The other design constraint that had to be verified was the attenuation of harmonics other than at the fundamental frequency. The design problem requires any harmonics in the output to be at least 23 dB below the fundamental sinusoid. The square wave has natural attenuation that results the third harmonic being 9.54 dB below the first harmonic as calculated: $20\log_{10}\left(\frac{|C_3|}{|C_1|}\right) = 20\log_{10}\left(\frac{0.1061}{0.3183}\right) = -9.54$ dB. Therefore, the filter should provided an additional $23 - 9.54 = 13.46$ dB attenuation. When the cutoff frequency is 11.5 kHz, the attenuation provided by the filter is found to be approximately 14.79 dB, which is greater than the required 13.46 dB. Figure 2 shows the frequency response of the filter at 11.5 kHz. Therefore, we can conclude the minimum value of the cutoff frequency required to meet all design constraints is around 11.5 kHz.

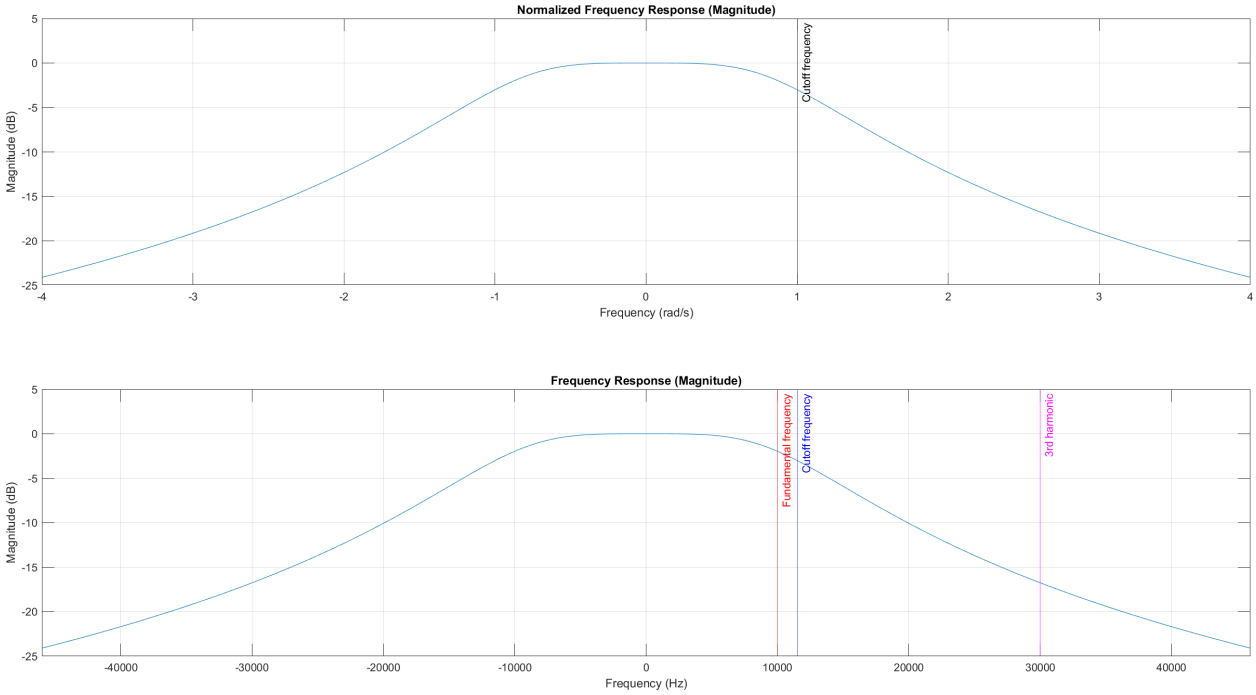


Figure 2: The frequency response plots with the cutoff frequency at 11.5 kHz

# Discussing the Impact of Changing Cutoff Frequency

# Determining the Range of Acceptable Cutoff Frequencies

We have determined through simulation that the minimum cutoff frequency required to satisfy the design constraints is around 11.5 kHz. We utilize the same approach explored earlier to determine what is the maximum cutoff frequency that the filter can use while still satisfying the design constraints, and determined that the maximum cutoff frequency is around 12.9 kHz. When the cutoff frequency is set to be greater than 12.9 kHz, the filter no longer provides the required attenuation between the fundamental sinusoid and the third harmonic. Figure 3 shows the frequency response of the filter at 12.9 kHz.

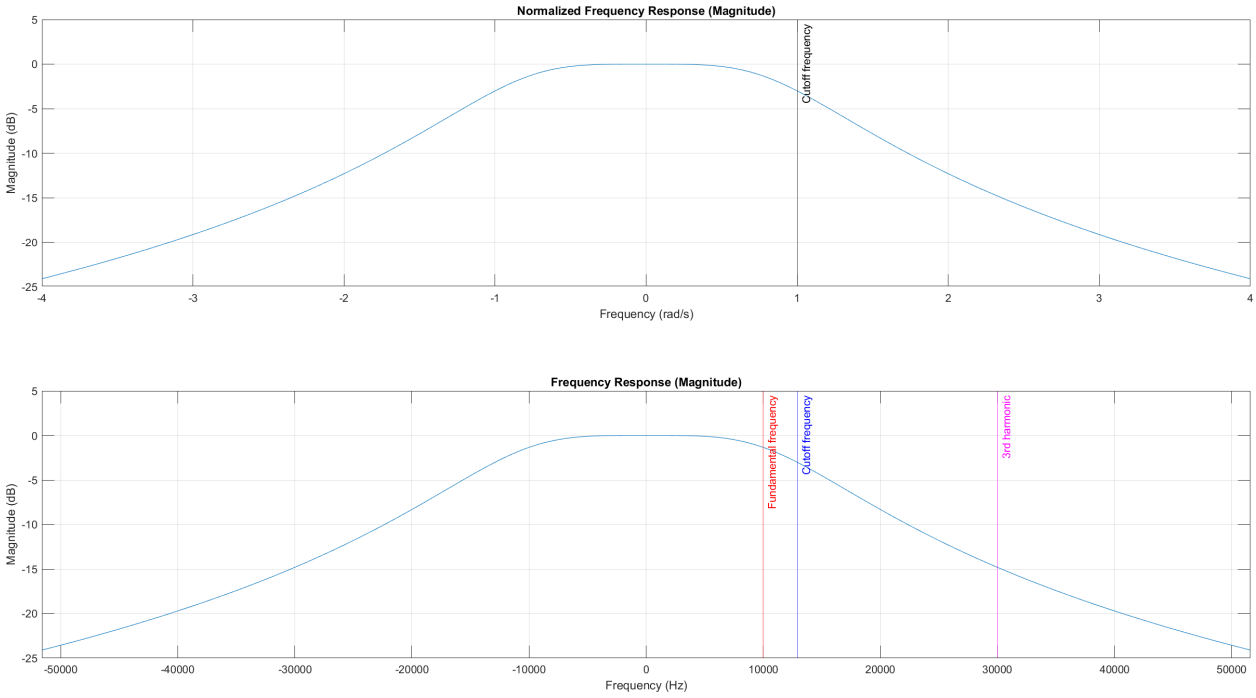


Figure 3: The frequency response plots with the cutoff frequency at 12.9 kHz

We have determined that the acceptable range of cutoff frequencies that satisfy the design constraints of the problem are between 11.5 kHz and 12.9 kHz.

If the cutoff frequency was too large, we would not have been able to satisfy the design constraint without moving to a higher-order Butterworth filter. A larger cutoff frequency would also be detrimental to the efficacy of the filter in removing more frequency components.

We were able to meet the design constraints with a 2nd order filter, thus if we had to build this filter in hardware, it would be cheaper, smaller and use less resources compared a higher order filter.

We used the `square()` function from the Signal Processing toolbox to generate the discrete time square wave function. We generated samples from -3T0 to +3T0 with a step-size of 1e-7, where T0 is 0.1ms.

In order to get the frequency response of the output signal, we took the scaled absolute value DFT of the signal, shifted it to that it was centred on the frequency (x) axis, and then plotted the resultant waveform.