1 Introduction

The goal of this lab is become familiar with frequency response tools and Bode plots using Python.

2 Equations

For this lab, we use two mean function that we had to solve for the per-lab.

$$|H(jw)| = \frac{wL}{\sqrt{(R - RLCw^2)^2 + (wL)^2}} \tag{1}$$

$$\angle H(Wj) = 90 - tan^{-}1(\frac{wL}{R - RLCw^2}) \tag{2}$$

3 Methodology and Results

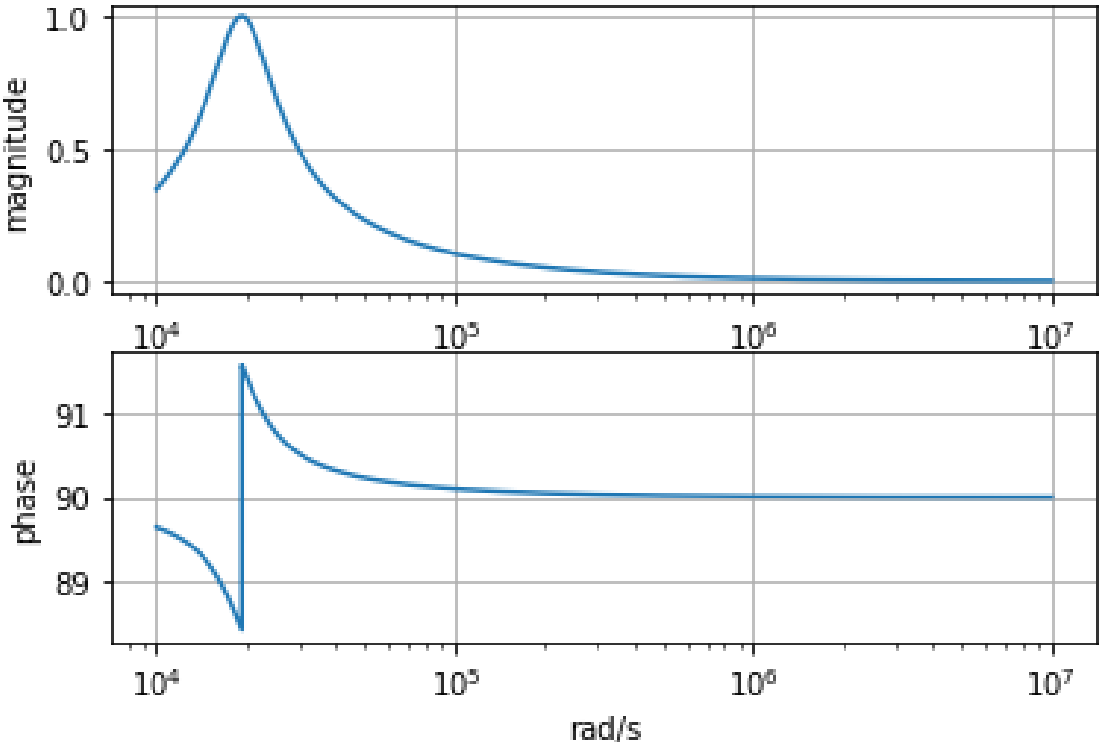
This lab consists of two parts:

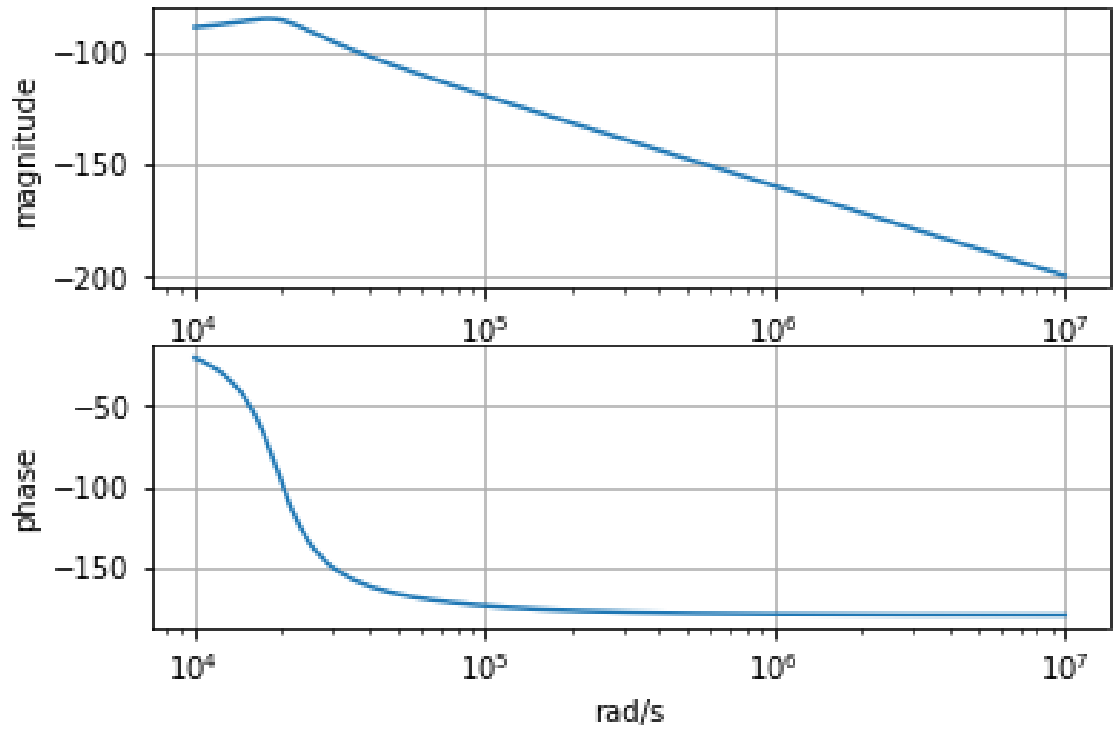
- Part 1: develop a frequency response from the RLC circuit and present the model using Bode plots:

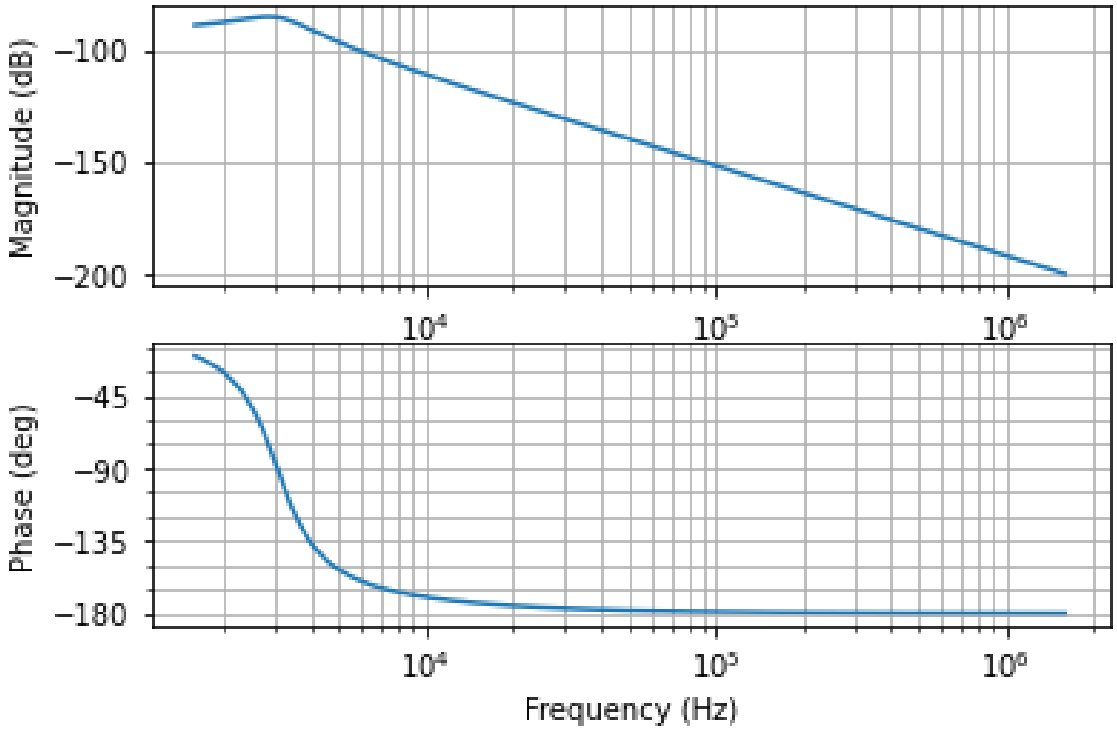
```
#%% part 1 task 1

R = 1000
L = 27e-3
C = 100e-9

w = np.arange(10e3, 10e6, 5)

mag = (w*L)/(np.sqrt(((R-R*L*C*w**2)**2)+(L*w)**2))
phase = 90-np.arctan((w*L)/(R-R*L*C*w**2))

plt.figure()
plt.subplot(2, 1, 1)
plt.semilogx(w, mag)
plt.xlabel('rad/s')
plt.ylabel('magnitude')
plt.grid()
plt.subplot(2, 1, 2)
plt.semilogx(w, phase)
plt.xlabel('rad/s')
plt.ylabel('phase')
plt.grid()

#%% part 1 task 2

num = [1/(R*C)]
den = [1, 1/(R*C), 1/(L*C)]

```

```
    w, mag, phase = sig.bode((num, den), w = w)

    plt.figure()
    plt.subplot(2, 1, 1)
    plt.semilogx(w, mag)
    plt.xlabel('rad/s')
    plt.ylabel('magnitude')
    plt.grid()
    plt.subplot(2, 1, 2)
    plt.semilogx(w, phase)
    plt.xlabel('rad/s')
    plt.ylabel('phase')
    plt.grid()

    #%% part 1 task 3

    plt.figure()
    sys = con.TransferFunction ( num , den )
    mag, phase, omega = con.bode(sys , w , dB = True , Hz = True , deg =
True , Plot = True )


```

- The output of this code:





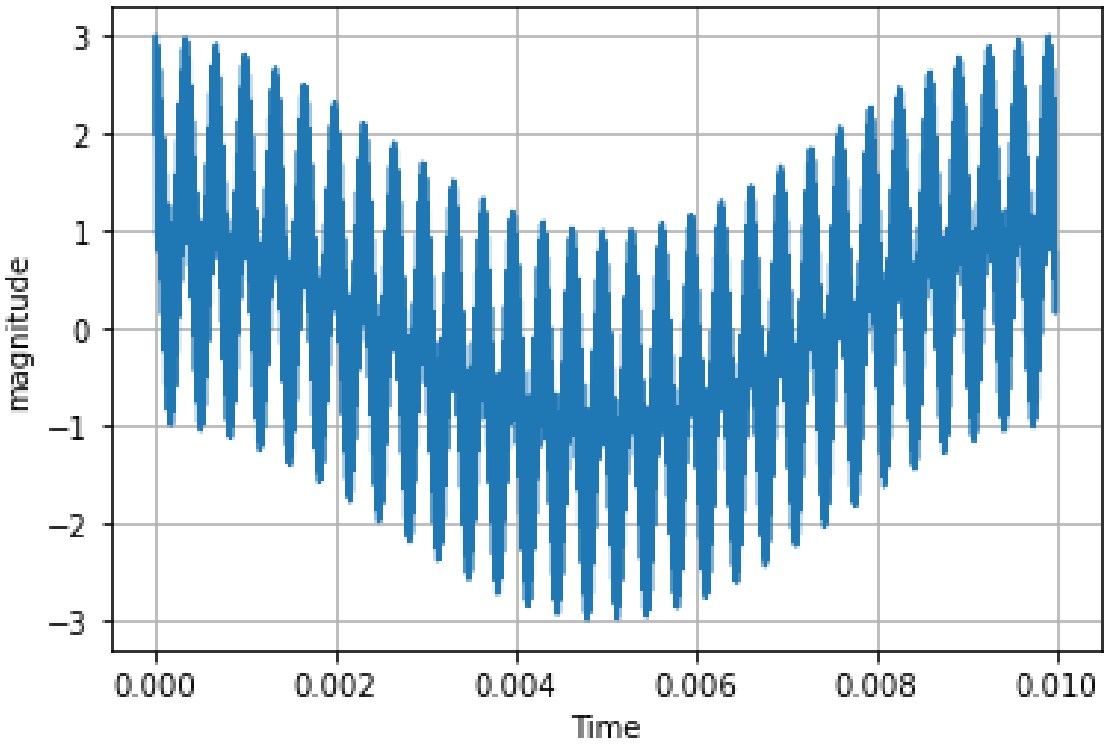


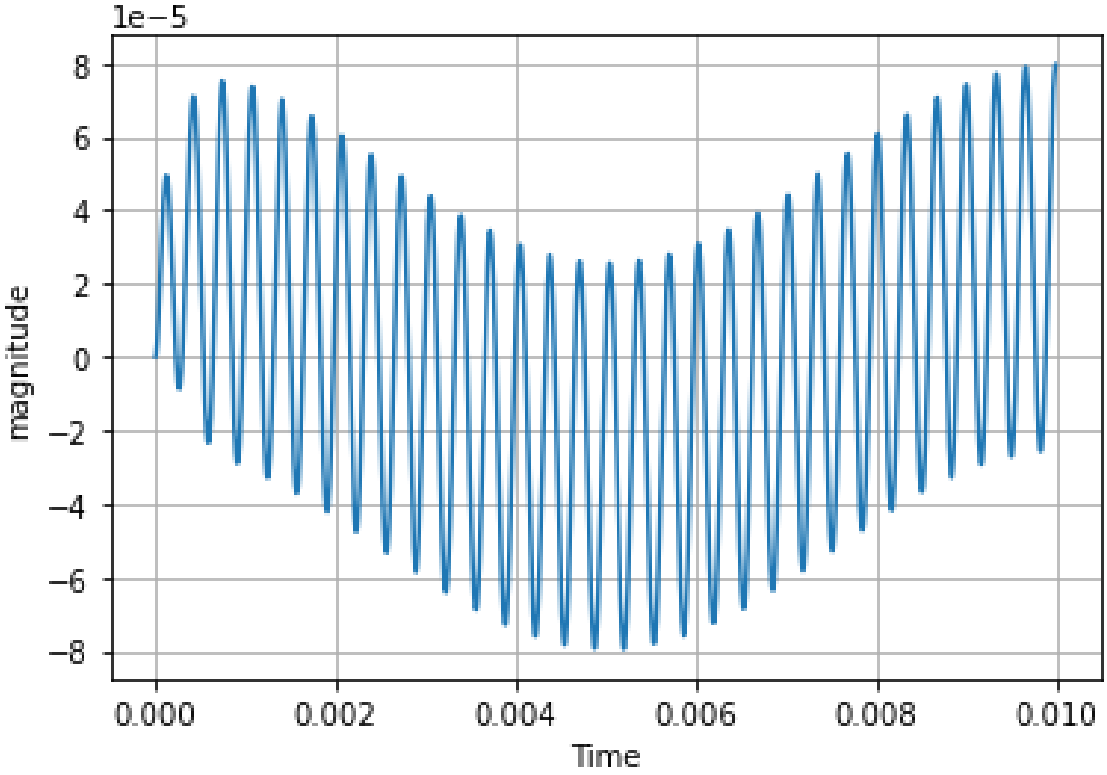
- Part 2: use the frequency response model developed in Part 1 as a filter for a multi-band

input signal:

```
1
2       #%% part 2 task 1
3
4       fs = 1000000
5       T = 1/fs
6       t = np.arange(0, 0.01, T)
7       x = np.cos(2*np.pi*100*t) + np.cos(2*np.pi*3024*t) + np.sin(2*np.pi
    *50000*t)
8
9       plt.figure()
10      plt.plot(t, x)
11      plt.xlabel('Time')
12      plt.ylabel('magnitude')
13      plt.grid()
14
15      #%% part 2 task 2,3 and 4
16
17      z, p = sig.bilinear(num, den, fs = fs)
18      y = sig.lfilter(z, p, x)
19
20      plt.figure()
21      plt.plot(t, y)
22      plt.xlabel('Time')
23      plt.ylabel('magnitude')
24      plt.grid()
25      plt.show()
26
27
```

- The output of this code:


3
2
1
0
−1
−2
−3
magnitude
0.000
0.002
0.004
0.006
0.008
0.010
Time


1e−5
8
6
4
2
0
−2
−4
−6
−8
magnitude
0.000
0.002
0.004
0.006
0.008
0.010
Time

4 Questions

1. 1. Explain how the filter and filtered output in Part 2 makes sense given the Bode plots from Part 1. Discuss how the filter modifies specific frequency bands, in Hz.

   The bode plot starts filtering frequencies after 10000 Hz on wards and since the time domain signal is a combination of frequencies 100, 3024 and 50000 Hz. The signals after 10000 Hz are attenuated completely but the signals with frequency less than 10000 Hz are attenuated to about -100 dB. Therefore, the signal having frequency of 50000 Hz is completely attenuated but the signals with frequencies 100 and 3024 Hz are attenuated less compared to the high frequency signal and the signal can be seen to have a very small magnitude.

2. 2. Discuss the purpose and workings of scipy.signal.bilinear() and scipy.signal.lfilter().
   The biliniear function converts the numerator and the denominator of the filter from s domain into z domain and it creates an IIR filter. The lfilter takes the z domain filter and applies the signal to it and therefore completing the filtering process.

3. 3. What happens if you use a different sampling frequency in scipy.signal.bilinear() than you used for the time-domain signal?

   If a different sampling frequency is used the filtered signal will not be accurate. It may contain signals that might be filtered originally.

4. 4. Leave any feedback on the clarity of lab tasks, expectations, and deliverables.

5 Conclusion

In this lab, became familiar with frequency response tools and Bode plots using Python.