ECE 351-51

April 5, 2022

Lab 10

Submitted By :
Kennedy Beach

Contents

1 Introduction

The purpose of this lab was to use become familiar with frequency response tools and Bode plots using Python.

2 Equations

The transfer function, and the magnitude and phase of the frequency transfer function are shown below.

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

$$H(j\omega) = \frac{\frac{1}{RC}j\omega}{(j\omega)^2 + \frac{1}{RC}j\omega + \frac{1}{LC}}$$

$$|H(j\omega)| = \frac{\frac{\omega}{RC}}{\sqrt{(-\omega^2 + \frac{1}{LC})^2 + (\frac{\omega}{RC})^2}}$$

$$\angle H(j\omega) = \frac{\pi}{2} - \arctan\left(\frac{\frac{\omega}{RC}}{\frac{1}{LC} - \omega^2}\right)$$

The signal to be plotted and passed through the RLC circuit that the transfer function describes is seen below:

$$x(t) = cos(2\pi * 100t) + cos(2\pi * 3024t) + sin(2\pi * 50000t)$$

3 Methodology

The first part involved plotting the magnitude and phase of the transfer function that we found by hand. I did this through a user-defined function.

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sig
import control
import control as con

def H (omega, R, L, C):
    mag = (omega/(R*C)) / np.sqrt(( ((1/(L*C)) - (omega**2) )**2) + (omega/(R*C))**2)
    mag_dB = 20*np.log10(mag)
```

```
    phase_rad = (np.pi/2) - np.arctan(omega/(R*C) / (-omega**2 +
    1/(L*C)))

    for i in range(len(phase_rad)):
        if phase_rad[i] >= np.pi/2:
            phase_rad[i] -= np.pi
        else:
            continue

    phase_deg = phase_rad*180/np.pi

    return mag_dB, phase_deg

R = 1e3
L = 27e-3
C = 100e-9

steps = 10
omega = np.arange(1e3, 1e6, steps)

y1_mag, y1_phase = H(omega, R, L, C)
```

Listing 1: User-defined function for the magnitude and phase

The Bode plot of the transfer function was then plotted with scipy.signal.bode(). The frequency response was also plotted using given code.

```
num = [1/(R*C), 0]
den = [1, 1/(R*C), 1/(L*C)]

y2_freq, y2_mag, y2_phase = sig.bode((num, den), w=omega, n=steps)

sys = con.TransferFunction(num, den)
_ = con.bode(sys, omega, dB = True, Hz = True, deg = True, Plot =
    True)
```

Listing 2: scipy.signal.bode()

The last part involved plotting the given signal, then passing it through the RLC filter using scipy.signal.bilinear() and scipy.signal.1filter().

```
fs = 50000
steps = 1/fs
t = np.arange(0, 0.01+steps, steps)
x = np.cos(2*np.pi*100*t) + np.cos(2*np.pi*3024*t) + np.sin(2*np.pi
    *50000*t)

plt.figure(figsize = (10, 7))
plt.ylabel('x(t)')
plt.xlabel('t')
```

```
plt.plot(t, x)
plt.grid()
plt.suptitle('Plot of x(t) where fs = 50000')

d_num, d_den = sig.bilinear(num, den, fs)
y_out = sig.lfilter(d_num, d_den, x)

plt.figure(figsize = (10, 7))
plt.ylabel('y(t)')
plt.xlabel('t')
plt.plot(t, y_out)
plt.grid()
plt.suptitle('Plot of y(t)')
```

Listing 3: Filter Output

4 Results

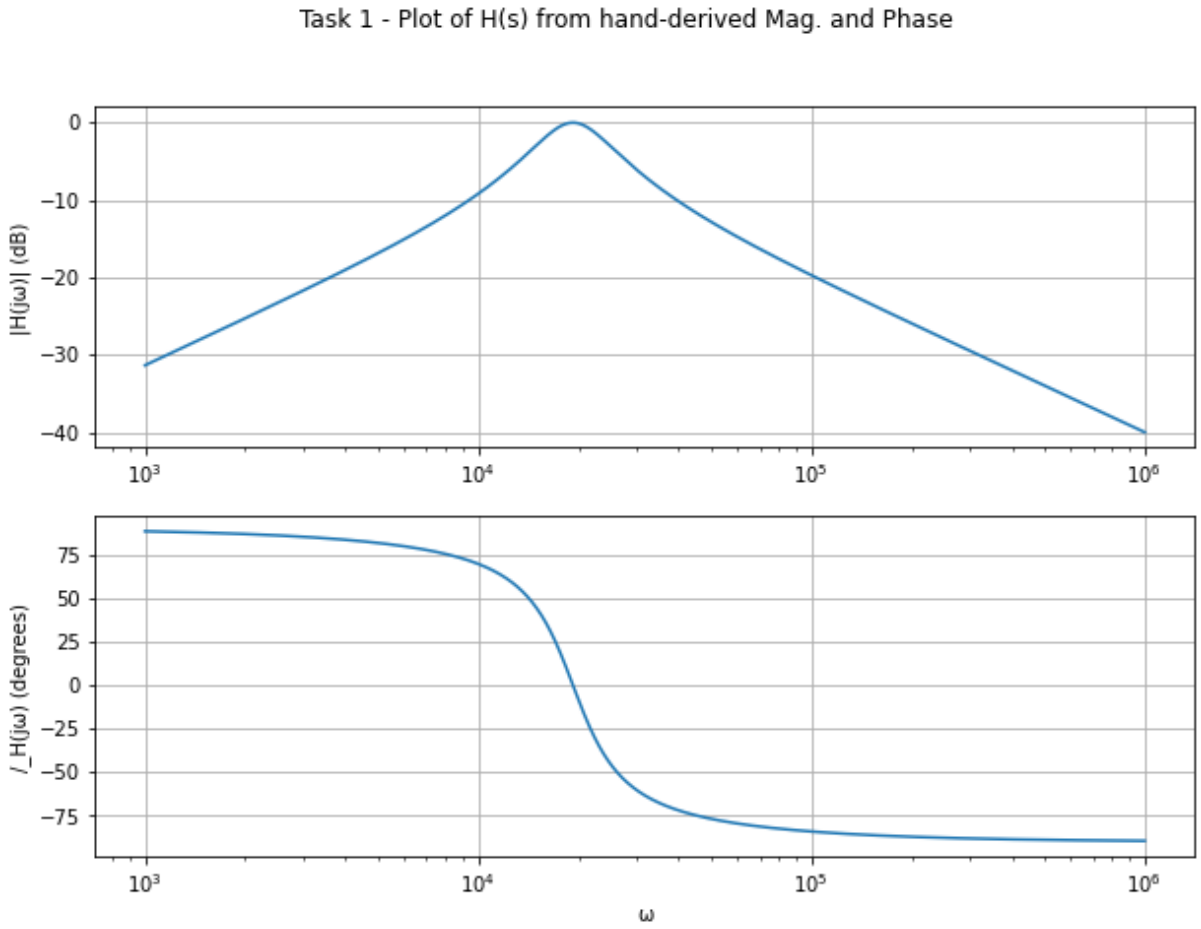

Figure 1: Hand-calculated plot

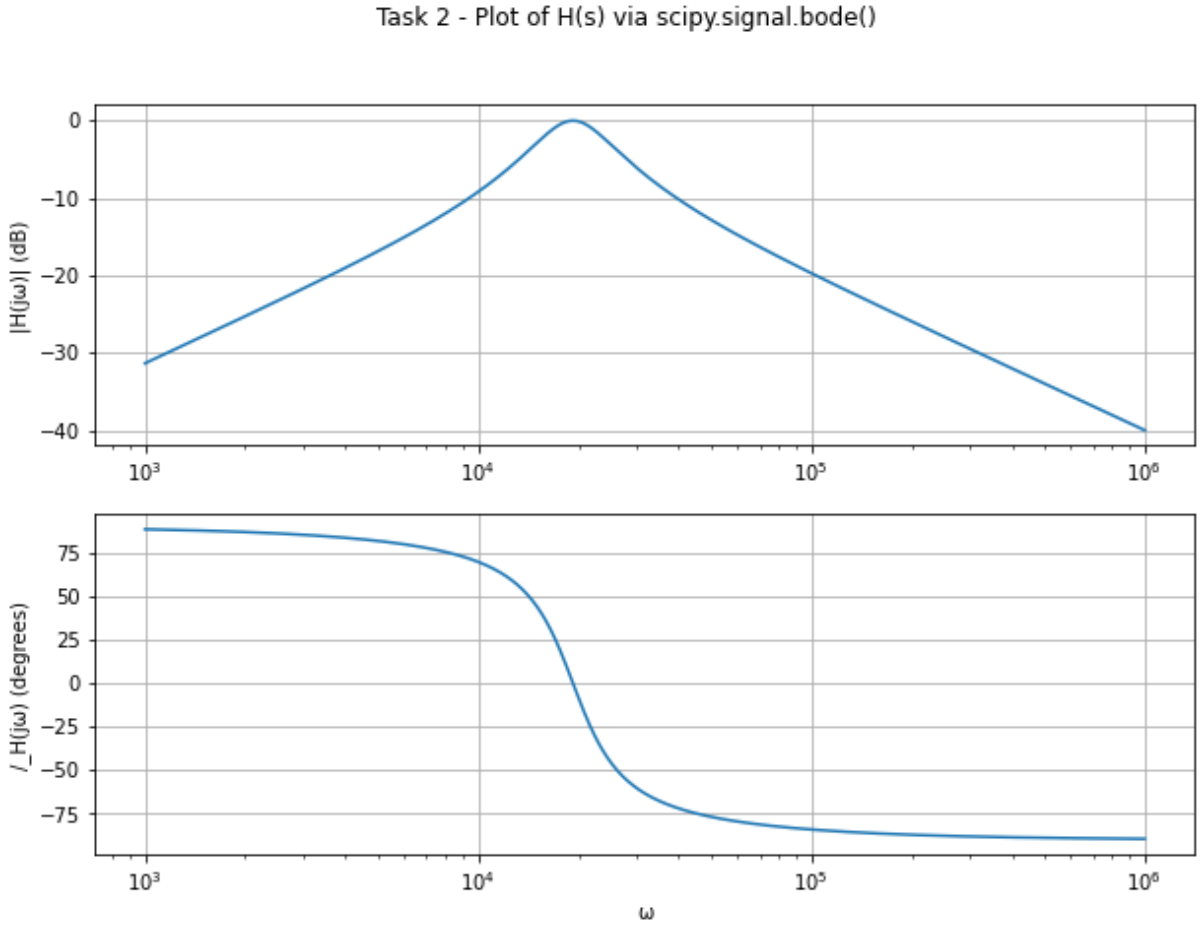

Figure 2: scipy.signal.bode()

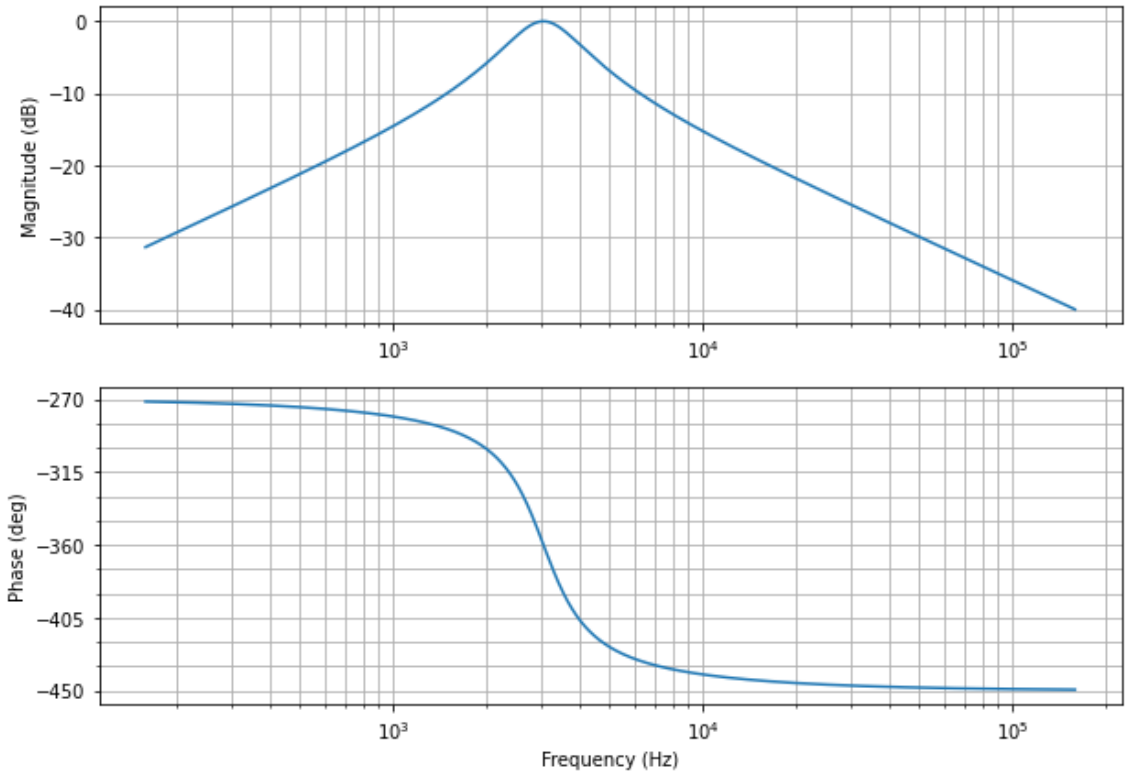

Figure 3: Bode Plot

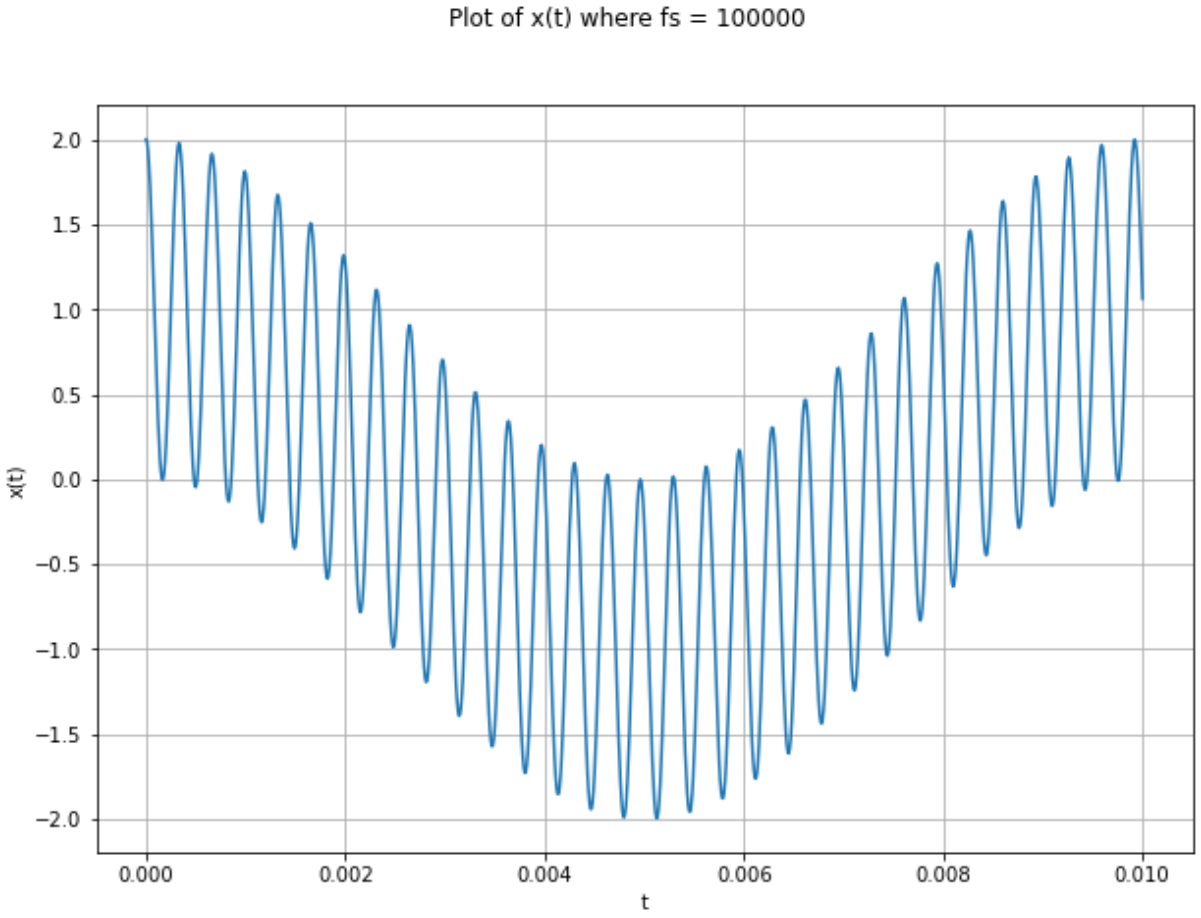
Plot of x(t) where fs = 100000
x(t)
t
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0.000
0.002
0.004
0.006
0.008
0.010

Figure 4: x(t)

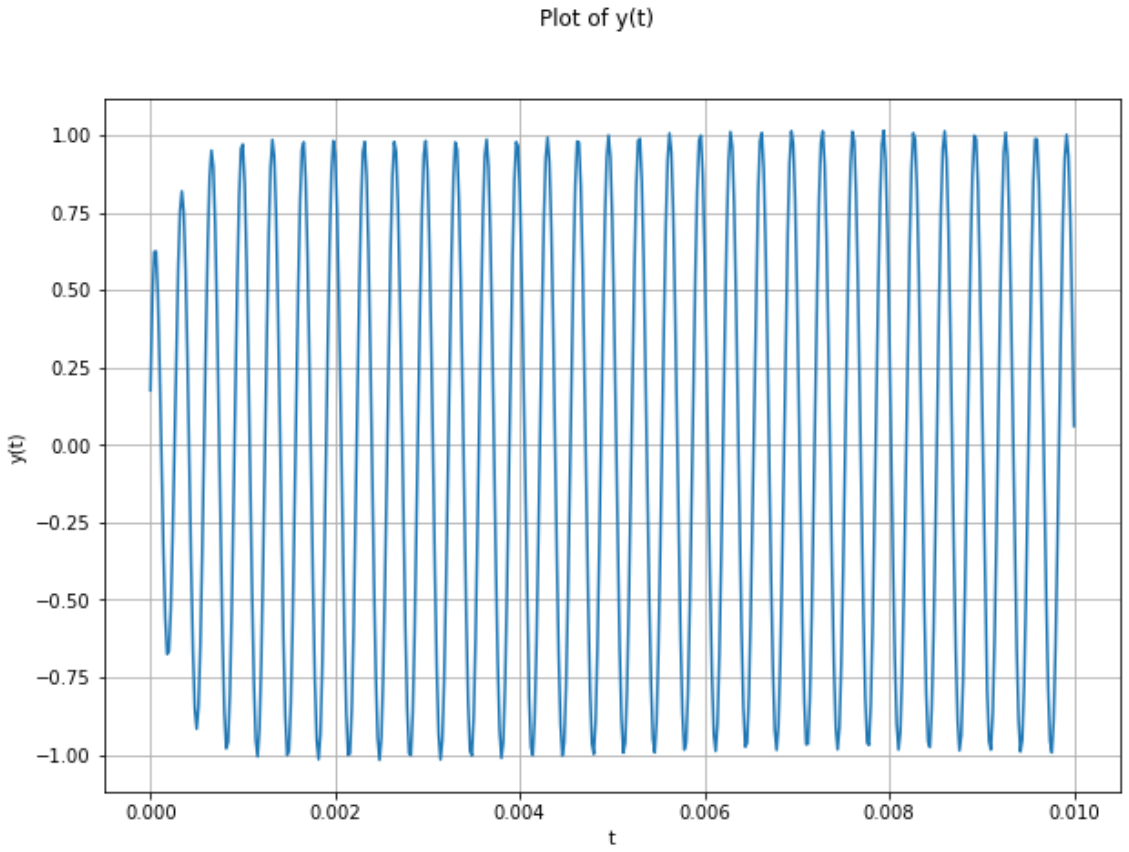

Figure 5: Output of x(t) through filter

5 Error Analysis

There weren't any errors besides some simple mistypes.

6 Questions

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.

Since the Bode plot descibed a bandpass filter, only certain frequencies are accepted while others are filtered out. This is seen in the output where the amplitude is mostly consistent with a uniform frequency.

2. Discuss the purpose and workings of scipy.signal.bilinear() and scipy.signal.lfilter().

The sig.bilinear function takes functions in the s-domain and transforms them into the z-domain. Once in the z-domain, the numerator and denominator were put into the sig.1filter function to run them through an electronic filter.

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

If I used a smaller frequency, the output would become more compact with waves that had smaller amplitudes. When I had a larger frequency, the output started to form a rough sine wave.

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

Everything was clear on what needed to be done and turned in.

7 Conclusion

This lab delved into Bode plots and outputs from a filter. The Python and LaTeX code are seen in `https://github.com/Eniac618/ECE351_Code` and `https://github.com/Eniac618/ECE351_Reports` respectively.