EE2703 : Applied Programming Lab
Assignment: Laplace Transform

P N Neelesh Sumedh
ee19b047

April 18, 2021

1 Introduction

In this assignment, we will look at how to analyse "Linear Time-invariant Systems" with numerical tools in Python. We will use signal toolbox from scipy. We will be using sp.lti, sp.impulse, sp.lsim.

2 Assignment

2.1 Quesion 1

The Laplace Transform of $f(t) = \cos(1.5t)e^{-0.5t}u(t)$ is given as

$$F(s) = \frac{s+0.5}{(s+0.5)^2+2.25}$$

We should solve for the time response of

$$\ddot{x} + 2.25x = f(t)$$

with $x(0) = \dot{x}(0) = 0$.
Taking the laplace transform of the above equation we get

$$s^2X(s) + 2.25X(s) = F(s)$$
$$X(s) = \frac{F(s)}{s^2+2.25}$$

To find x(t) we can find the impulse response of X(s). This impulse response is x(t). The code for this part is

```
def springtransfun(freq,damping):
    return sp.lti([1,damping],np.polymul([1.0,0,freq**2],[1,2*damping,freq
**2+damping**2]))
X = springtransfun(1.5,0.5)
t,x = sp.impulse(X,None,np.linspace(0,50,5001)) # Computes the Impulse
Response
plt.figure(0)
plt.title('Forced Damping oscillator with decay = 0.5')
plt.xlabel('t $\longrightarrow$')
plt.ylabel('x $\longrightarrow$')
plt.plot(t,x)
plt.show()

```

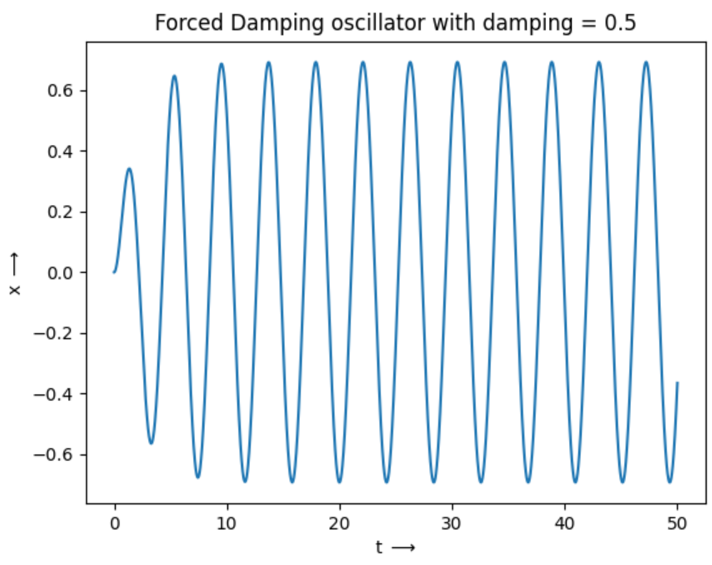

Figure 1: Forced Damping Oscillator, damping = 0.5

2.2 Question 2

Solving the previous with damping = 0.05. When damping = 0.05, the Laplace Transform of f(t) changes to

$$F(s) = \frac{s + 0.05}{(s + 0.05)^2 + 2.25}$$

The code for this part:

```
X = springtransfun(1.5,0.05)
t,x = sp.impulse(X,None,np.linspace(0,50,5001))
plt.figure(1)
plt.title('Forced Damping Oscillator with damping = 0.05')
plt.xlabel('t $\longrightarrow$')
plt.ylabel('x $\longrightarrow$')
```

```
    plt.plot(t,x)
    plt.show()
```

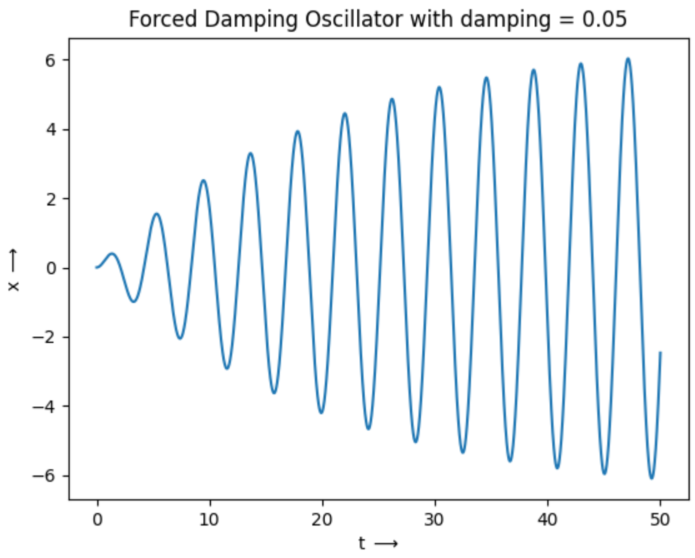

Figure 2: Forced Damping Oscillator, damping = 0.05

The Damping here is less than that in Question 1. Therefore, the time taken to reach steady state is longer.

2.3 Question 3

Now lets consider f(t) as the input of the system and x(t) as the output of the system. To find the output for the given input, we have to convolve the impulse response of the system with the input of the system. To find the convolution, we use the command *scipy.lsim*. Then, we change the frequency of the input from 1.4 to 1.6 with a step of 0.05 to observe the output.

```
    X_F = sp.lti([1],[1,0,2.25])
    i = 6
    for f in np.arange(1.6,1.35,-0.05):
        t = np.linspace(0,150,5001)
        inp = np.cos(f*t)*np.exp(-0.05*t)
        t,y,svec = sp.lsim(X_F,inp,t)
        plt.figure(i)
        plt.plot(t,y)
        plt.title('Forced Damping Oscillator with frequency = '+str(f))
        plt.xlabel('t $\longrightarrow$')
        plt.ylabel('x $\longrightarrow$')
        i -= 1
```

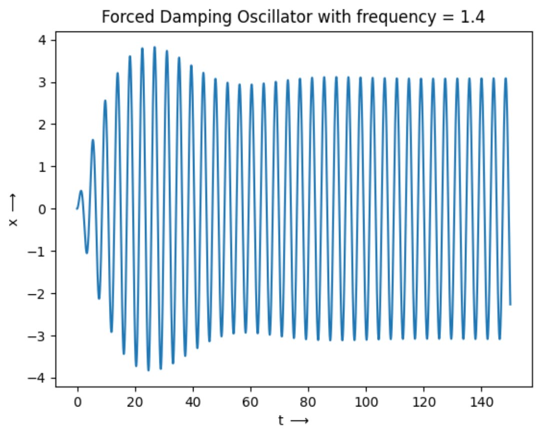

Figure 3: System Response with freq = 1.4

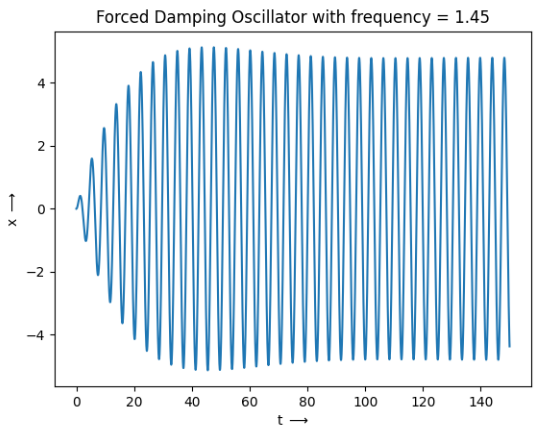

Figure 4: System Response with freq = 1.45

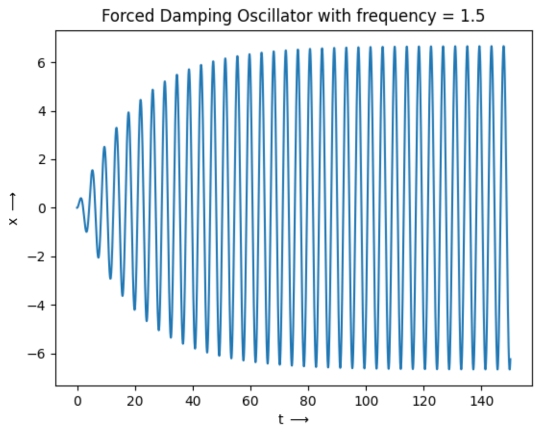

Figure 5: System Response with freq = 1.5

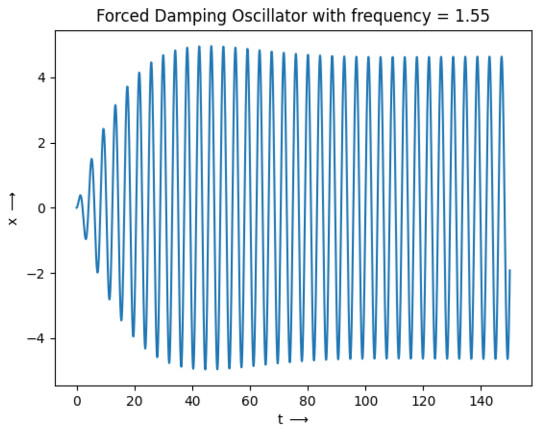

Figure 6: System Response with freq = 1.55

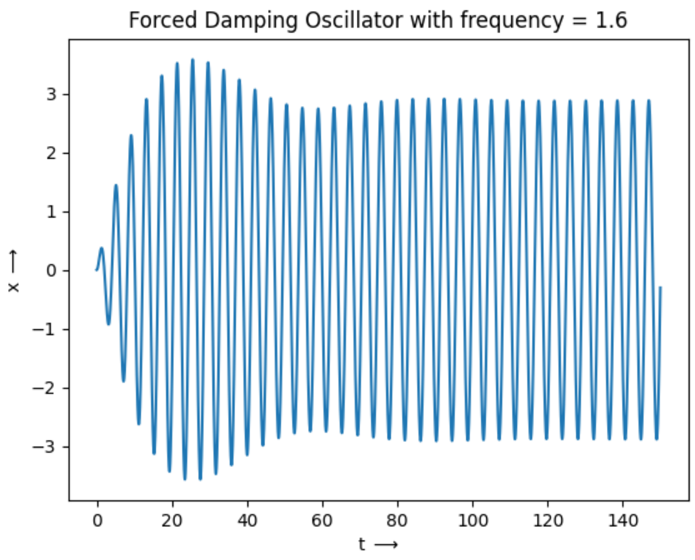

Figure 7: System Response with freq = 1.6

From the system equation, we get to know that the natural frequency of the system is 1.5 . When we give an input with frequency = 1.5, there is resonance and the amplitude is larger than other inputs with different frequencies. AS we can see in the figures, the input with freq = 1.5 has an amplitude greater than 6 while others are around 4.

2.4 Question 4

Given two coupled equations

$$\ddot{x} + (x - y) = 0$$
$$\ddot{y} + 2(y - x) = 0$$

with initial conditions $x(0) = 1, \dot{x}(0) = 0, \dot{y}(0) = y(0) = 0$. Applying Laplace Transform on these equations, we get

$$X(s) = \frac{s^2 + 2}{s^3 + 3s}$$
$$Y(s) = \frac{2}{s^3 + 3s}$$

After finding X(s) and Y(s), we can use *scipy.impulse* to find x(t) and y(t) and plot the graphs of both.

```
    X = sp.lti([1,0,2],[1,0,3,0])
    Y = sp.lti([2],[1,0,3,0])
    t,x = sp.impulse(X,None,np.linspace(0,20,5001))
    t,y = sp.impulse(Y,None,np.linspace(0,20,5001))
    plt.figure(7)
    plt.plot(t,x,label = 'x')
    plt.title('Coupled Oscillations X and Y')
    plt.xlabel('t $\longrightarrow$')
    plt.ylabel('Amplitude $\longrightarrow$')
    plt.plot(t,y, label = 'y')
    plt.legend()
```

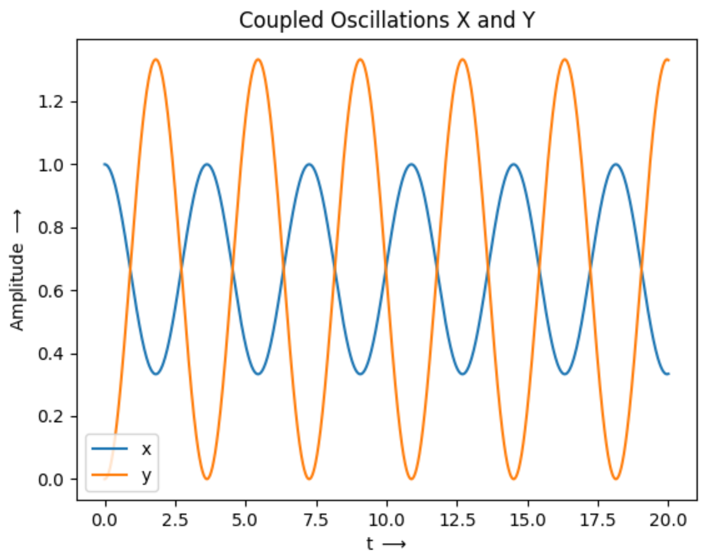

Figure 8: Coupled Oscillations

2.5 Question 5

Given a basic RLC circuit with values $R = 100\Omega, L = 1\mu H, C = 1\mu F$. The transfer function is

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

Now we should plot the Bode plots of the above transfer function. For this we can use *signal.bode* to get the magnitude plots and the phase plots.

```
    w, S, phi = H.bode()
    figure, (ax1,ax2) = plt.subplots(2,1,num=8)
    ax1.semilogx(w,S)
    ax1.set_title('Magnitude')
    ax2.semilogx(w,phi)
    ax2.set_title('Phase')
    figure.tight_layout()
```

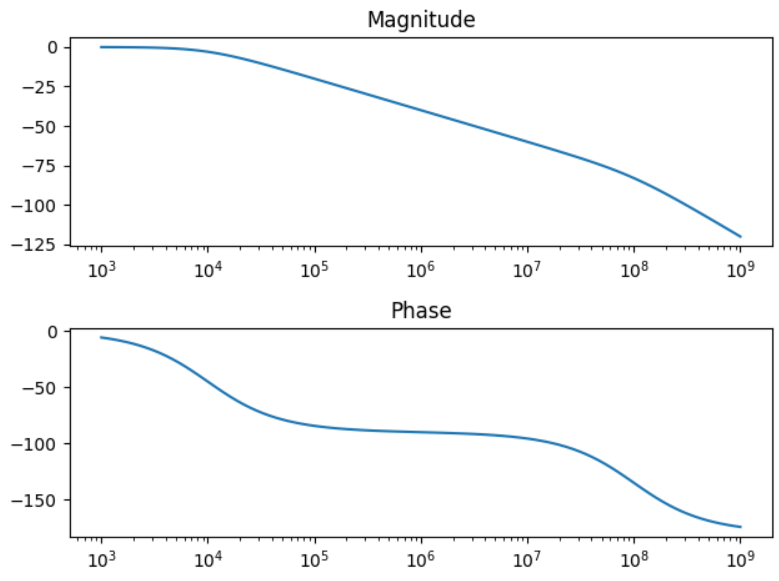

Figure 9: Magnitude and Phase Plots of H(s)

2.6 Question 6

Now the above RLC circuit is fed with the input

$$v_i(t) = \cos(1e3t)u(t) - \cos(1e6t)u(t)$$

To find the output of the system, we should again convolve the input with the impulse response. For this we can use *scipy.lsim*. We solve for two different durations to understand. First we solve for $0 < t < 30\mu s$. Then we solve for $0 < t < 30ms$. Since the lowest time frequency is 10^{-6}, we have to make sure that the time step is less than this.

For $t \in (0, 30\mu s)$,

```
    L = 1e-6
    C = 1e-6
    R = 100
    H = sp.lti([1],[L*C,R*C,1])
    time = np.linspace(0,30e-6,10000)
    v = np.cos(1e3*time) - np.cos(1e6*time)
    t,y,svec = sp.lsim(H,v,time)
    plt.figure(9)
    plt.title('Output of RLC circuit for t<30 usec')
    plt.xlabel('t $\longrightarrow$')
    plt.ylabel('$V_{out}$ $\longrightarrow$')
    plt.plot(t,y)
```

The output we get is shown below.

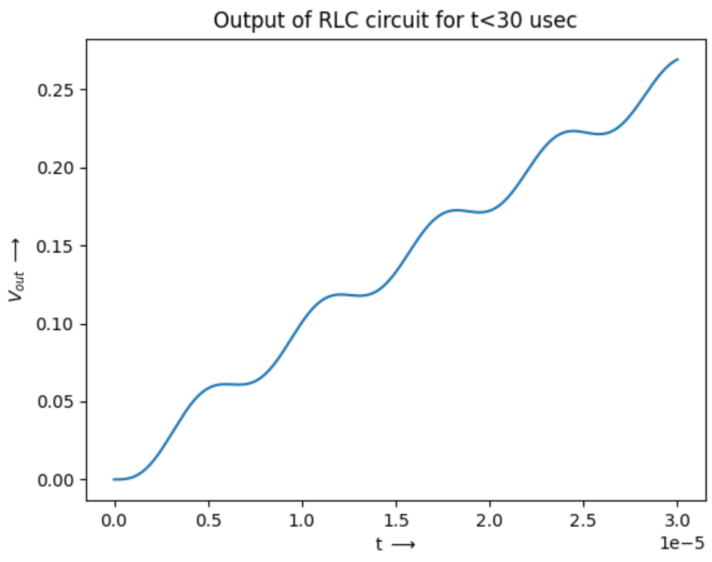

Figure 10: v_o for $t \in (0, 30\mu s)$

For $t \in (0, 30ms)$,

```
    time = np.linspace(0,3e-2,10000)
    v = np.cos(1e3*time) - np.cos(1e6*time)
    t,y,svec = sp.lsim(H,v,time)
    plt.figure(10)
    plt.title('Output of RLC circuit for t<30 msec')
    plt.xlabel('t $\longrightarrow$')
    plt.ylabel('$V_{out}$ $\longrightarrow$')
    plt.plot(t,y)
```

The output for this is shown below.

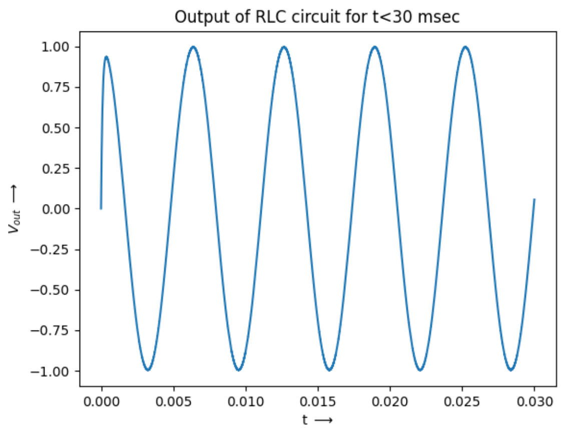

Figure 11: v_o for $t \in (0, 30ms)$

The output as we can see has a frequency of 10^3. Also as we can see from the Magnitude Plot, the lower frequencies have high magnitude, but for the higher frequencies, the amplitude is very small. Therefore the system acts like a LPF. Thus the long time output signal has a frequency of 10^3 and does not contain 10^6 frequency. Now in the small time scale graph, even though the amplitude of 10^6 is very less, it is not zero. This is added to the 10^3 frequency cosine signal, thus there are ripples seen in the short duration. While in the long duration, these ripples are very small and can be neglected.

3 Conclusion

In this assignment, we analysed three LTI systems. We analysed Forced Oscillations, Coupled Oscillations and Low Pass Filters. We used the scipy commands: lti, impulse, lsim. scipy.lti is used to define the transfer function. scipy.impulse is used to find the impulse response of the transfer function and scipy.lsim is used to find the convolution of impulse response and input.