Assignment 6B: The Laplace Transform

Katari Hari Chandan [EE19B032]

May 2021

1 Introduction

In this assignment, we will look at how to analyze "Linear Time-invariant Systems" using the scipy.signal library in Python .We limit our analysis to systems with rational polynomial transfer functions. More specifically we consider 3 systems: A forced oscillatory system, A coupled system of Differential Equations and an RLC low pass filter

1.1 Assignment Questions

1.1.1 Question 1

We first consider the forced oscillatory system(with 0 initial conditions):

$$\ddot{x} + 2.25x = f(t) \tag{1}$$

We solve for $X(s)$ using the following equation, derived from the above equation.

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

We then use the impulse response of $X(s)$ to get its inverse Laplace transform.

```
import scipy.signal as sp
#this function returns X(s) from equation(2)
def transfer_spring(frequency,decay):
    p= np.polymul([1.0,0,2.25],[1,-2*decay,frequency*frequency
                                + decay*decay])
    return sp.lti([1,-1*decay],p)
#Taking impulse response of X(s) and plotting
t,x = sp.impulse(transfer_spring(1.5,-0.5),None,np.linspace(0,50,5001))
plotter(t,x,"Forced Damping Oscillator with decay = 0.5","t","x")
```

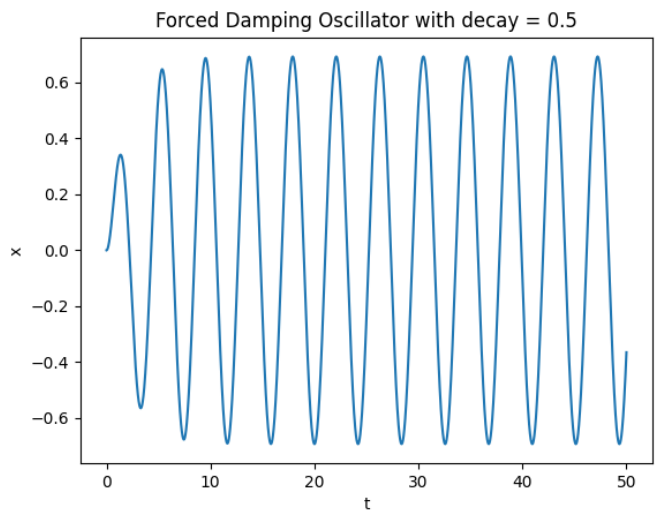

Figure 1: System Response with Decay = 0.5

1.1.2 Question 2

We now see what happens with a smaller Decay Constant.

```
t,x = sp.impulse(transfer_spring(1.5,-0.05),None,np.linspace(0,50,5001))
plotter(t,x,"Forced Damping Oscillator with decay = 0.05","t","x")
```

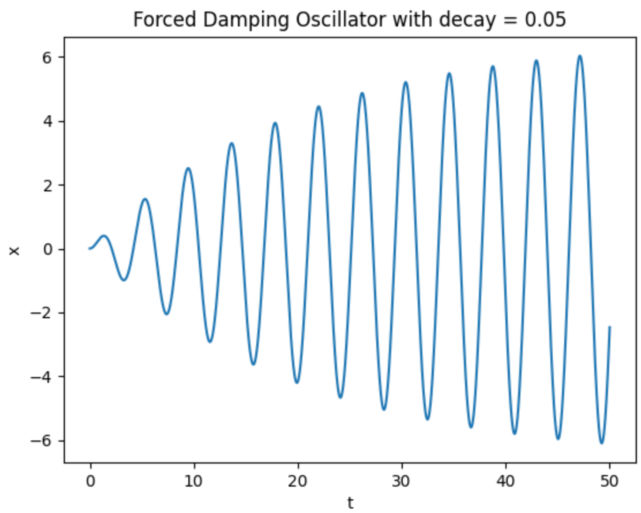

Figure 2: System Response with Decay = 0.05

We notice that the result is very similar to that of question 1, except with a different amplitude. This is because the system takes longer to reach a steady state.

1.1.3 Question 3

We now see what happens when we vary the frequency. We note the the amplitude is maximum at frequency = 1.5, which is the natural frequency of the given system

```
freq = np.linspace(1.4,1.6,5)
for f in freq:
    t,x = sp.impulse(transfer_spring(f,-0.05),None,np.linspace(0,150,5001))
    plotter(t,x,"Forced Damping Oscillator with freq = " + str(f),"t","x")
```

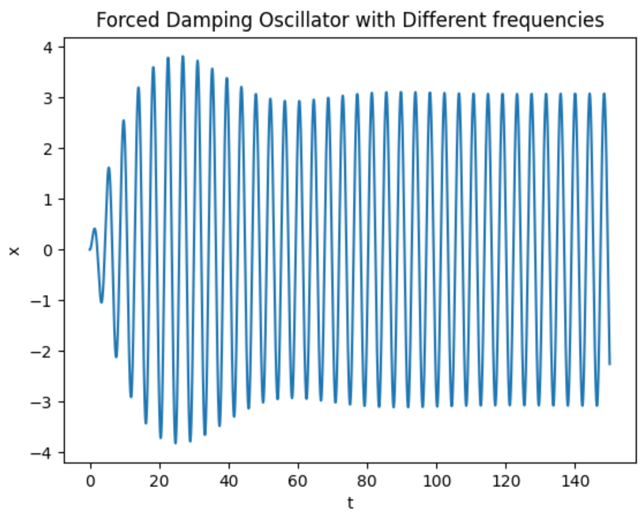

Figure 3: System Response with frequency = 1.4

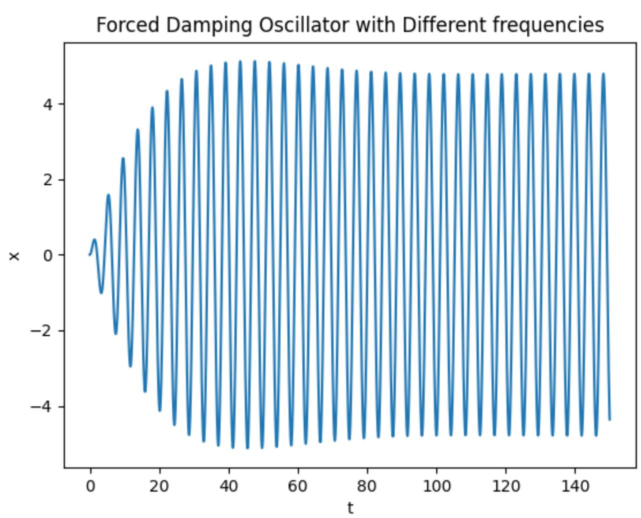

Figure 4: System Response with frequency = 1.45

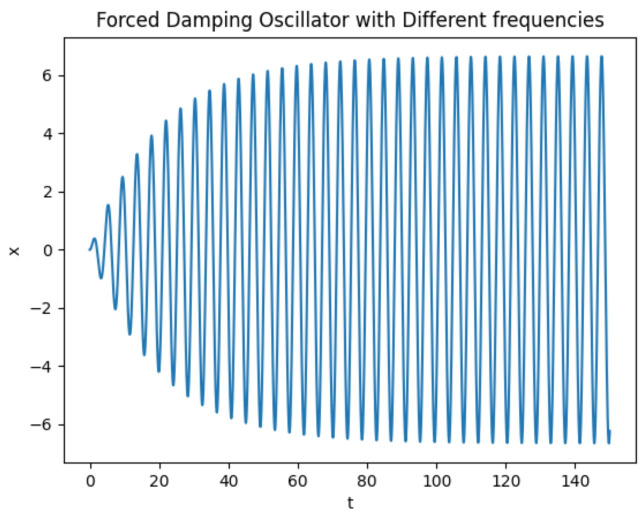

Figure 5: System Response with frequency = 1.5

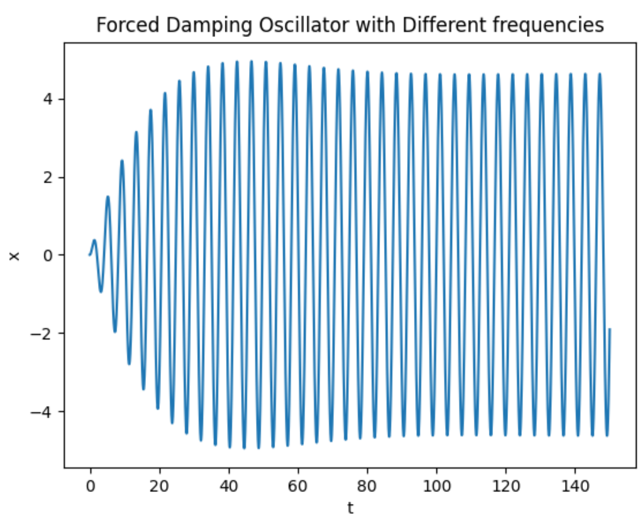

Figure 6: System Response with frequency = 1.55

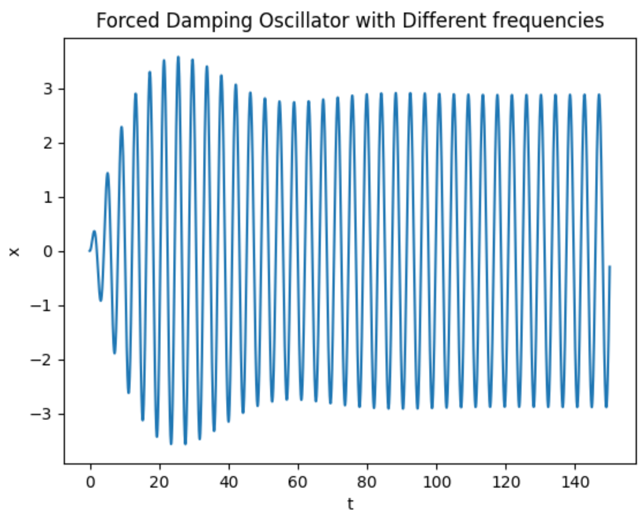

Figure 7: System Response with frequency = 1.6

1.1.4 Question 4

We now consider a coupled Differential system

$$\ddot{x} + (x - y) = 0 \tag{3}$$

and

$$\ddot{y} + 2(y - x) = 0 \tag{4}$$

with the initial conditions: $\dot{x}(0) = 0, \dot{y}(0) = 0, x(0) = 1, y(0) = 0$. Taking Laplace Transform and solving for $X(s)$ and $Y(s)$, We get:

$$X(s) = \frac{s^2 + 2}{s^3 + 3s} \tag{5}$$

$$Y(s) = \frac{2}{s^3 + 3s} \tag{6}$$

```
#solve for X in coupling equation
X = sp.lti([1,0,2],[1,0,3,0])
t,x = sp.impulse(X,None,np.linspace(0,50,5001))
plotter(t,x,"Coupled Osccillations: X","t","x",show = False)
#solve for Y in coupling equation
Y = sp.lti([2],[1,0,3,0])
t,y = sp.impulse(Y,None,np.linspace(0,50,5001))
plotter(t,y,"Coupled Oscillations: X and Y","t","y",leg = 1,legend = ['x','y'])
```

We notice that the outputs of this system are 2 sinusoids which are out of phase.This system can be realized by creating an undamped single spring double mass system.

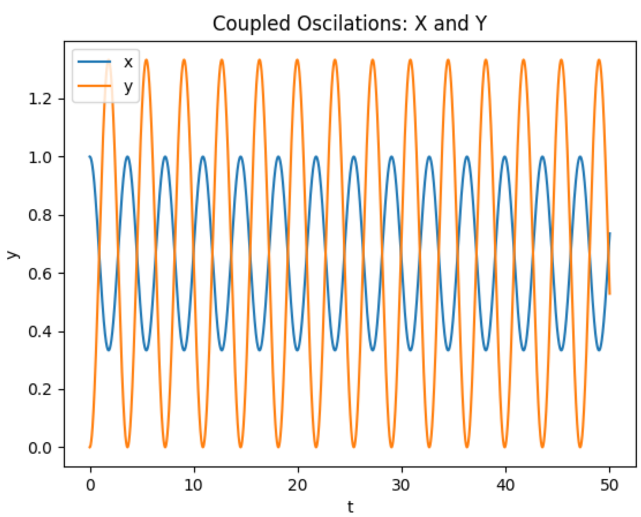

Figure 8: Coupled Oscillations

1.1.5 Question 5

Now we try to create the bode plots for the low pass filter defined in the question

```
#returns Low pass filter response for given input and time period
def RLC(time,inp=func,bode = False,R=100,L=1e-6,C=1e-6):
    H = sp.lti([1],[L*C,R*C,1])
    #plotting bode if required
    if bode:
        w,S,phi = H.bode()
        fig,(ax1,ax2) = plt.subplots(2,1)
        ax1.set_title("Magnitude response")
        ax1.semilogx(w,S)
        ax2.set_title("Phase response")
        ax2.semilogx(w,phi)

        plt.show()

    return sp.lsim(H,inp(time),time)
t=np.linspace(0,30e-6,10000)
t,y,_ = RLC(t,bode = 1)
```

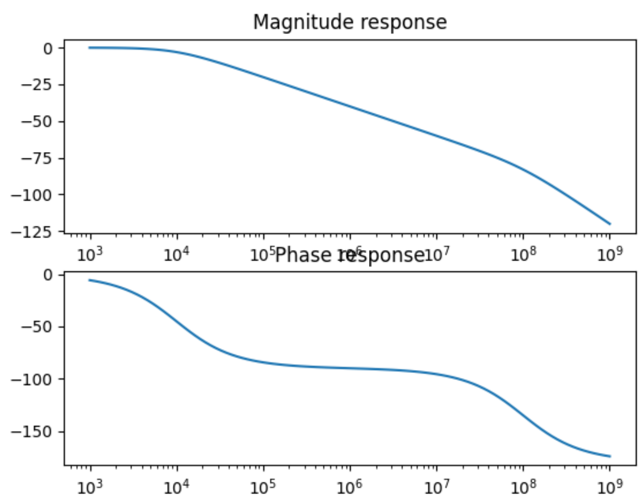

Figure 9: Bode Plots For RLC Low pass filter

1.1.6 Question 6

We know plot the response of the low pass filter to the input:

$$V_i(t) = (cos(10^3t) - cos(10^6t))u(t)$$

for $0 < t < 30\mu s$ and $0 < t < 30ms$

```
#Returns input voltage
def func(t):
    return (np.cos(1000*t) -np.cos(1e6*t))*(t>0)
#for t<30us
t=np.linspace(0,30e-6,10000)
t,y,_ = RLC(t,bode = 1)
plotter(t,y,"Output of RLC for t<30u","t","x")
#for t<30ms
t=np.linspace(0,30e-3,10000)
t,y,_ = RLC(t)
plotter(t,y,"Output of RLC for t<30m","t","x")
```

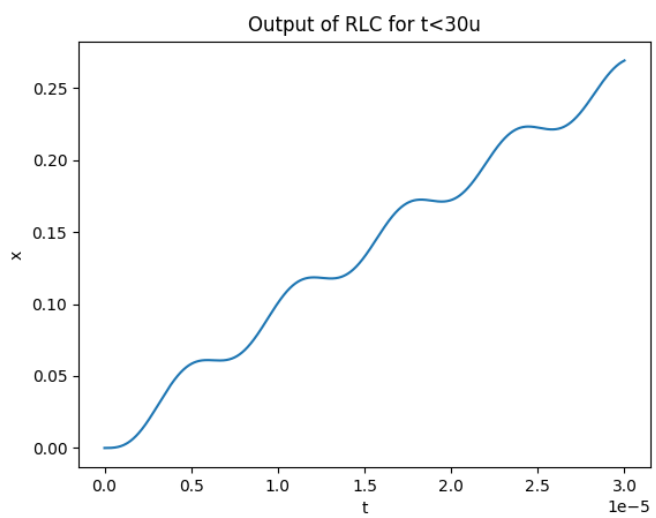

Figure 10: System response for t < 30us

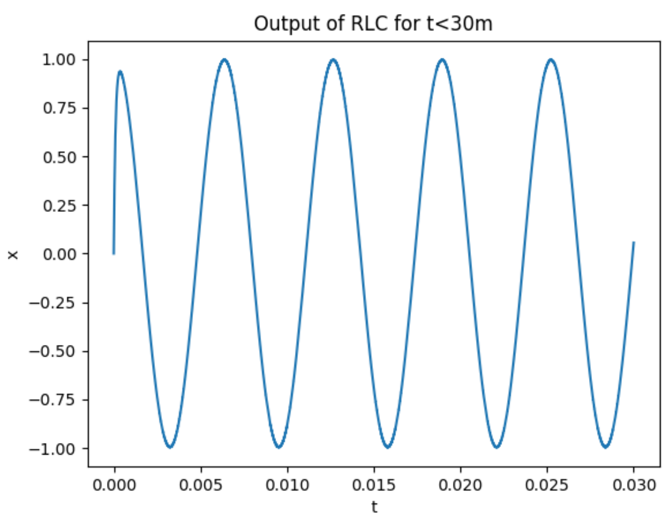

Figure 11: System response for t < 30ms

2 Conclusion

LTI systems are observed in all fields of engineering and are very important. In this assignment, we have used scipy's signal processing library to analyze a wide range of LTI systems. Specifically we analyzed forced oscillatory systems, single spring, double mass systems and RLC filters