EE2703 : Applied Programming Lab
Assignment 6
The Laplace Transform

Debojyoti Mazumdar
EE20B030

March 27, 2022

0.1 Abstract

Aim of this assignment is to understand how to use scipy.signal library to analyze "Linear Time-invariant systems"

0.2 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 systemswith 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

0.3 Assignment

0.3.1 Question 1

We first import the standard libraries.

```
import scipy.signal as sp
import numpy as np
import matplotlib.pyplot as plt
```

Then for the following differential equation:

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

$$f(t) = \cos(1.5t)\, e^{-0.5t} u_0(t)$$

And the Laplace transform of the given f(t) is:

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

We use the following code to generate x(t) and then we plot the function.

```
# generates transfer function for forced damped spring system
def transfer_func_generator(decay, freq):
    p = np.polymul([1, 0, 2.25], [1, 2*decay, decay*decay+freq*freq])
    return sp.lti([1, decay], p)

# question-1
t = np.linspace(0, 50, 10001)
```

```
x_low_decay = sp.impulse(transfer_func_generator(0.5, 1.5), None, t)[1]
plt.title("plot of spring funtion with decay=0.5/s vs t")
plt.plot(t, x_low_decay, "r")
plt.xlabel("time")
plt.ylabel("x(t)")
plt.show()
```

Then we get the following graph for x(t):

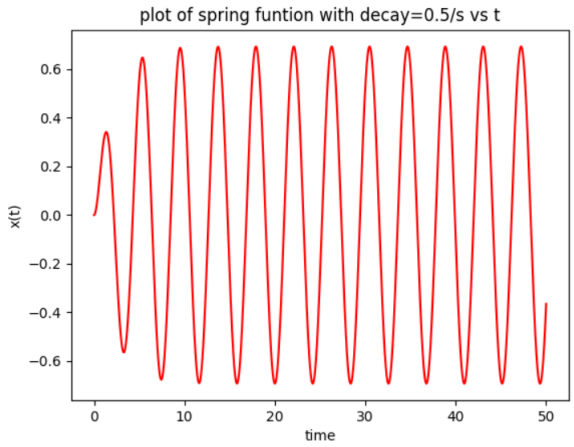


Figure 1: plot of x(t) vs t

0.3.2 Question 2

Here, we do the same as we did in question-1 but with a lower decay(0.05).

```
# question-2
x_high_decay = sp.impulse(transfer_func_generator(0.05, 1.5), None, t)[1]
plt.title("plot of spring funtion with decay=0.05/s vs t")
plt.plot(t, x_high_decay, "r")
plt.xlabel("time")
plt.ylabel("x(t)")
plt.show()
```

Then we get the following graph:

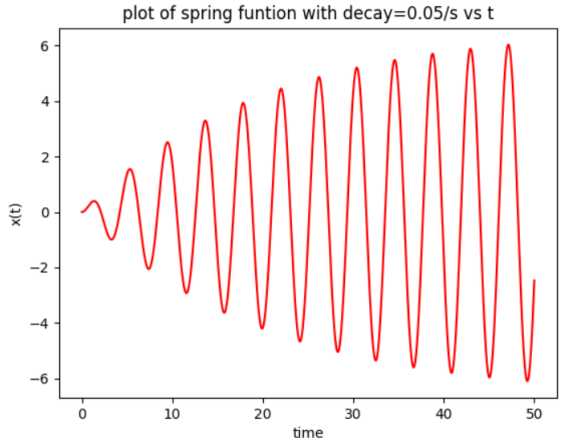


Figure 2: plot of x(t) vs t for decay=0.05

0.3.3 Question 3

Now we try and get the transfer function for the above equation. Then using sp.slim we get the output function with different frequency.We note that the amplitude is maximum at frequency=1.5,which is the natural frequency of the given system and the amplitude decreases symmetrically on either sides of this frequency(1.5). That is why we will get the graphs shown. So we write the following code:

```
# question-3
H = sp.lti(1, [1, 0, 2.25])
freq = np.linspace(1.4, 1.6, 5)
for f in freq:
    f_t = np.cos(f*t)*np.exp(-0.05*t)
    x = sp.lsim(H, f_t, t)[1]
    plt.title("plot of x(t) vs t for freq="+str(f))
    plt.plot(t, x, "r")
    plt.show()
```

Then we get the following graphs:

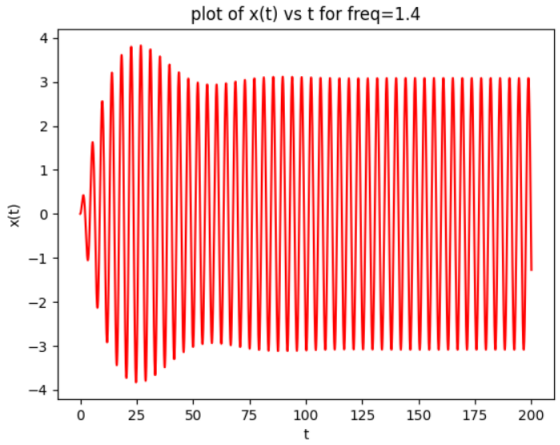


Figure 3: System Response with frequency=1.4

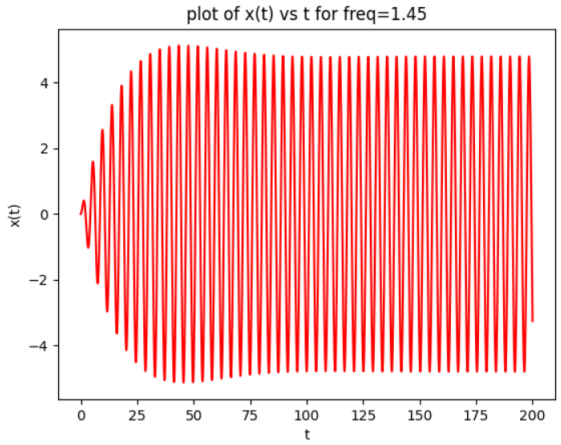


Figure 4: System Response with frequency=1.45

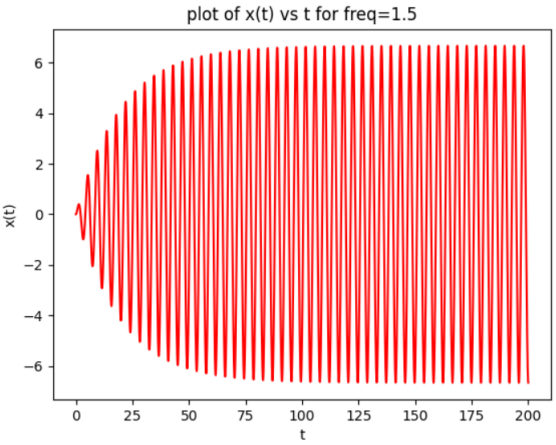


Figure 5: System Response with frequency=1.5

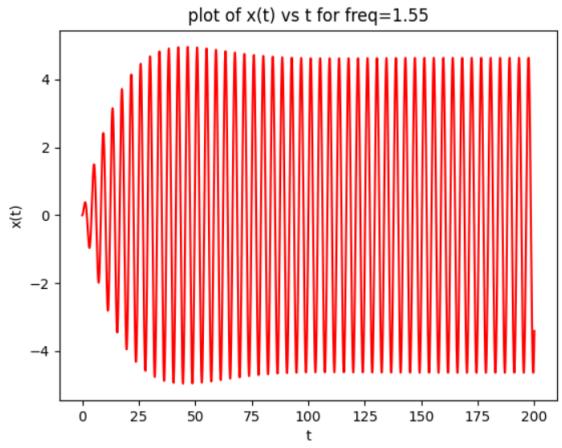


Figure 6: System Response with frequency=1.55

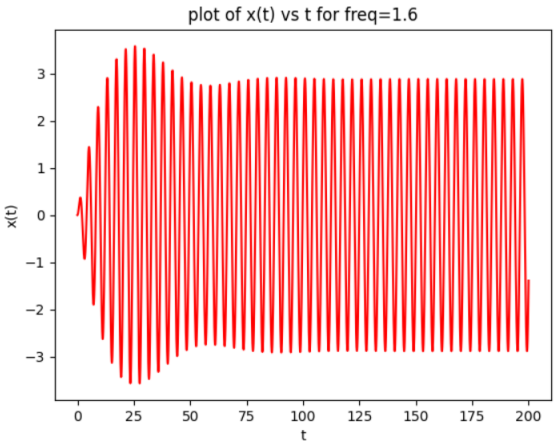


Figure 7: System Response with frequency=1.6

0.3.4 Question 4

Here, we solve the coupled spring problem:

$$\begin{aligned} \ddot{x} + (x - y) &= 0 \\ \ddot{y} + 2(y - x) &= 0 \end{aligned}$$

with the initial conditions:

$$x(0) = 1, \dot{x}(0) = y(0) = \dot{y}(0) = 0$$

Taking the Laplace Transform we get:
So we write the following code to obtain x(t) and y(t) and then plot it.

```
# question-4
X = sp.lti([1, 0, 2], [1, 0, 3, 0])
```

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

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

```
Y = sp.lti([2], [1, 0, 3, 0])

t = np.linspace(0, 20, 5001)
x = sp.impulse(X, None, t)[1]
y = sp.impulse(Y, None, t)[1]

plt.title("x(t) and y(t) vs time")
plt.plot(t, x, "r", label="x(t)")
plt.plot(t, y, "b", label="y(t)")
plt.legend()
plt.show()
```

Then we get the following plot:

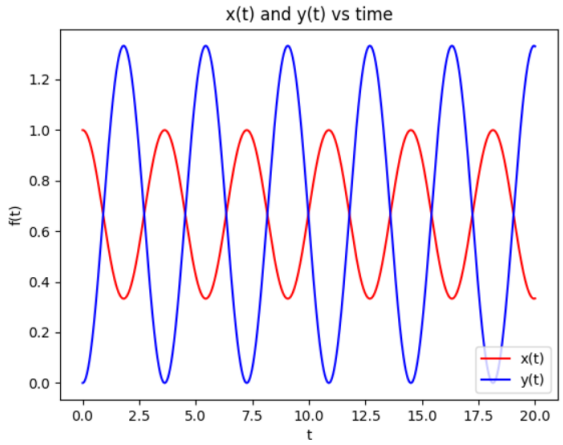


Figure 8: x(t) and y(t) vs t

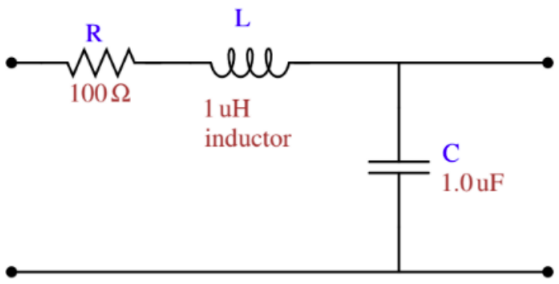
0.3.5 Question 5

Now we plot the transfer function of the following RLC low pass filter circuit.

To do so we write the following code.

```
# question-5
H = sp.lti([10**12], [1, 10**8, 10**12])

# plotting the bode plot
w, mag, phase = sp.bode(H)
```




```
plt.figure()
plt.title("Magnitude response of H(s)")
plt.semilogx(w, mag)    # Bode magnitude plot
plt.figure()
plt.title("Phase response of H(s)")
plt.semilogx(w, phase)  # Bode phase plot
plt.show()
```

Then we get the following bode plots:

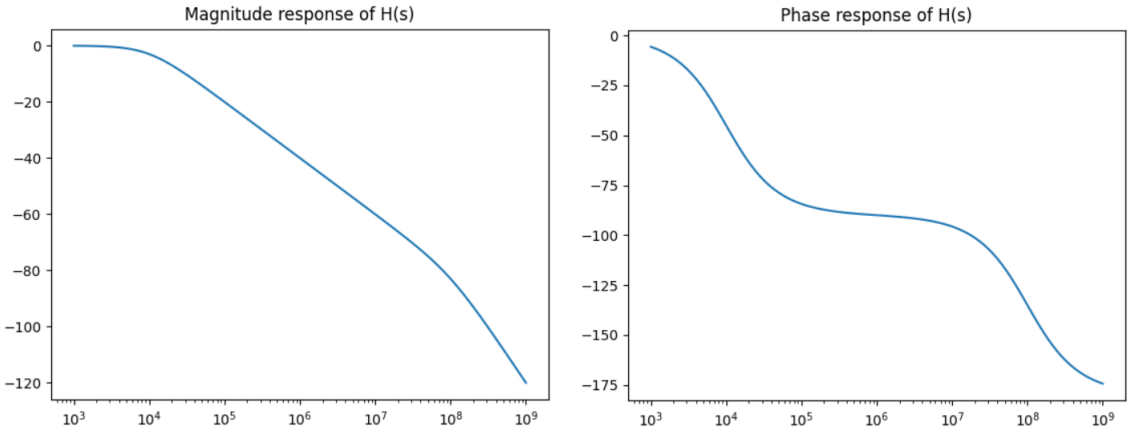


Figure 9: Bode plots for the RLC low pass filter

0.3.6 Question 6

Now we have to plot the output of the low pass filter above with the following input voltage:

$$v_i(t) = \cos\left(10^3 t\right) u(t) - \cos\left(10^6 t\right) u(t)$$

To find the output voltage we use the following code:

```
# question-6
t = np.linspace(0, 30e-6, 10000)
```

```
vi_t = np.cos((10**3)*t)-np.cos((10**6)*t)
vo_t = sp.lsim(H, vi_t, t)[1]

# plotting
plt.title("vo(t) vs time for t<30\u03BCsec")
plt.plot(t, vo_t, "r", label="vo(t)")
plt.show()

t = np.linspace(0, 30e-3, 10000)
vo_t = sp.lsim(H, vi_t, t)[1]

# plotting
plt.title("vo(t) vs time for t<30msec")
plt.plot(t, vo_t, "r", label="vo(t)")
plt.show()
```

Then we get the following graphs for the two time intervals:

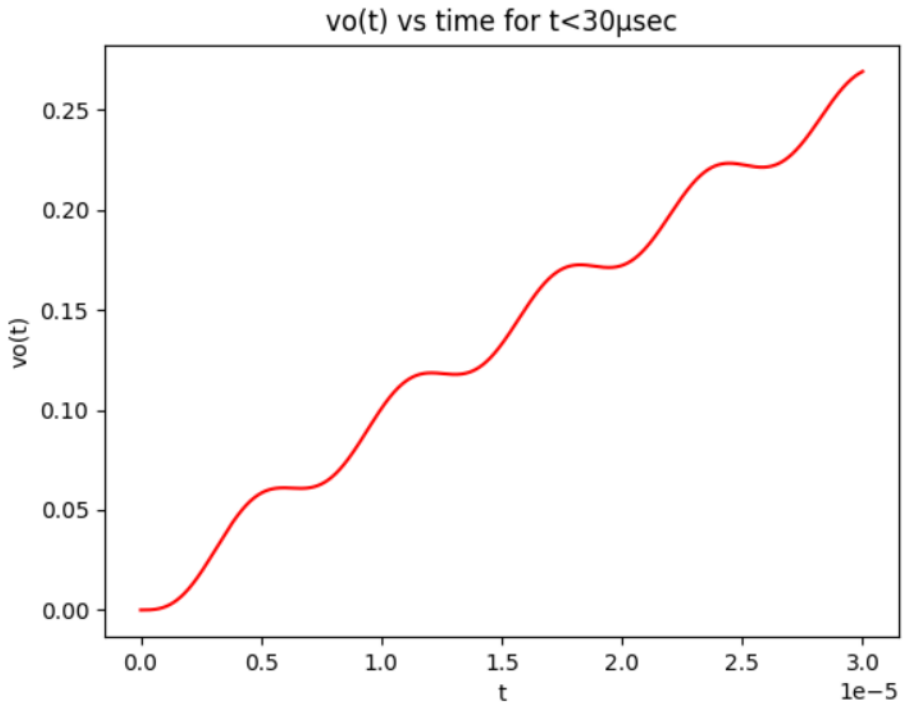


Figure 10: plot of vo(t) for time¡30microsec

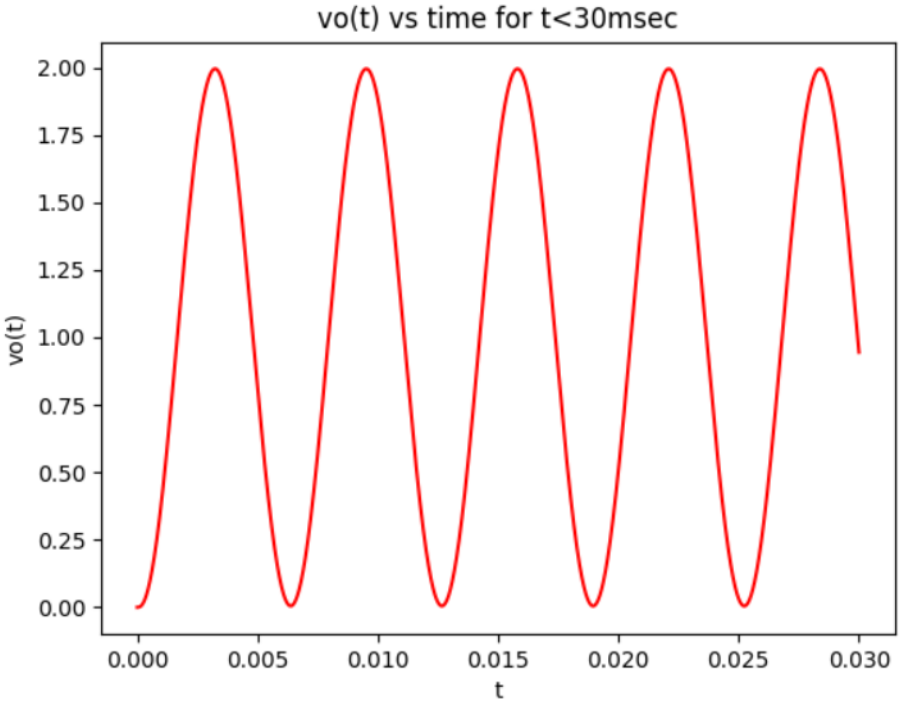


Figure 11: plot of vo(t) for timej30milliosec

The graphs are different for short time interval and long time interval because in short time interval the high frequency component is not totally filtered out so you get the first type of graph. But in the long time interval the 10^6 frequency component is filtered out so you get the second type of graph.

0.4 Conclusions

We understood how to use scipy.signal library to solve LTI system problems. Here, we have solved problems regarding forced damped spring sytem and low pass filter RLC circuit using the scipy.signal library.