EE2703: Assignment 6

Sagar (ee20b115)

March 12, 2022

1 Aim

In this assignment, we will look at how to analyse *Linear Time-invariant Systems (**LTI**)* with numerical tools in Python.

In this assignment we will use mostly Mechanical examples, We will analyse **LTI** systems in continuous time using *Laplace Transforms* to find the output of the system to a given input with the help of a Python library, namely *scipy.signal* toolbox.

2 Procedure and Observations

2.1 Time Response of a Spring for 2 different Decay using *signal.impulse*

First we'll model it as simply a differential equation and solve it using *scipy.signal.impulse.*
The Laplace transform of $f(t) = e^{-at}cos(\omega t)u(t)$ is given as:

$$\mathcal{L}\{f(t)\} = \frac{s+a}{(s+a)^2+\omega^2}$$

From the property of Laplace transforms, we know:

$$x(t) \to \mathcal{X}(s)$$
$$\dot{x}(t) \to s\mathcal{X}(s) - x(0^-)$$
$$\ddot{x}(t) \to s^2\mathcal{X}(s) - sx(0^-) - \dot{x}(0^-)$$

From the above equations, we get, for $a = 0.5$ and $\omega = 1.5$:

$$\mathcal{F}(s) = \mathcal{L}\{f(t)\} = \frac{s+0.5}{(s+0.5)^2+2.25}$$

So, the equation of the spring oscillator can be written as:

$$s^2\mathcal{X}(s) - sx(0^-) - \dot{x}(0^-) + 2.25\mathcal{X}(s) = \frac{s+0.5}{(s+0.5)^2+2.25}$$

Given that the **IC**s $x(0)$ and $\dot{x}(0)$ are 0, we get:

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

or,

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

Now this can be solved and plotted using *scipy.signal.impulse.*

2.1.1 CODE:

```
plt.figure(0)
plt.title("Plot of $\ddot{x} + 2.25x = e^{-0.5t}cos(1.5t)u(t)$")

t = np.linspace(0,50,500)
H1 = sp.lti([1],[1,0,2.25])
H2 = sp.lti([1,0.5],np.polymul([1,0,2.25],[1,1,2.5]))
t = np.linspace(0,50,500)
t,x1 = sp.impulse(H2,None,t)
```

```
plt.plot(t,x1)
plt.xlabel("t")
plt.ylabel("x(t)")
plt.xlim(0,50)
plt.grid(True)
plt.savefig("figure_0")
plt.show()

plt.figure(1)
plt.title("Plot of $\ddot{x} + 2.25x = e^{-0.05t}cos(1.5t)u(t)$")

H3 = sp.lti([1,0.05],np.polymul([1,0,2.25],[1,0.1,2.2525]))
t,x2 = sp.impulse(H3,None,t)

plt.plot(t,x2)
plt.xlabel("t")
plt.ylabel("x(t)")
plt.xlim(0,50)
plt.grid(True)
plt.savefig("figure_1")
plt.show()
```

2.1.2 PLOTS:

First we Plot the graph for decay = 0.5:

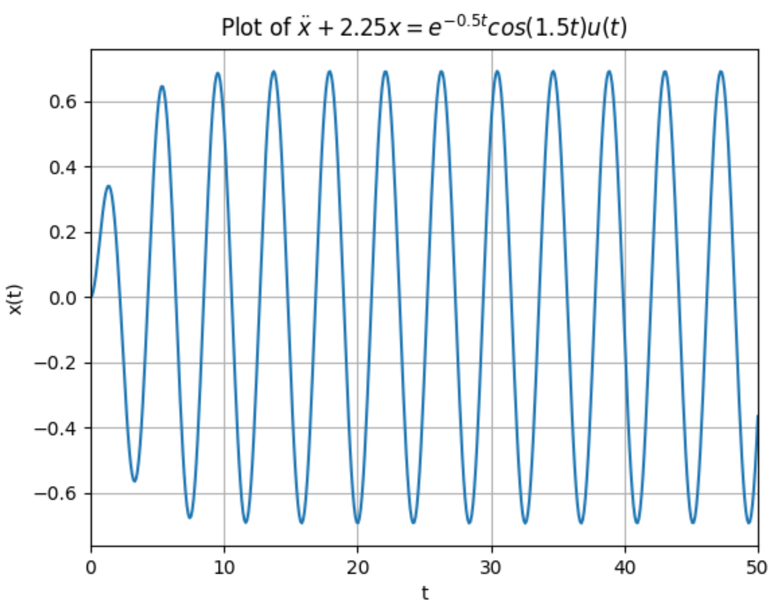

Figure 1: Plot for decay = 0.5

Now we Plot the graph for decay = 0.05:

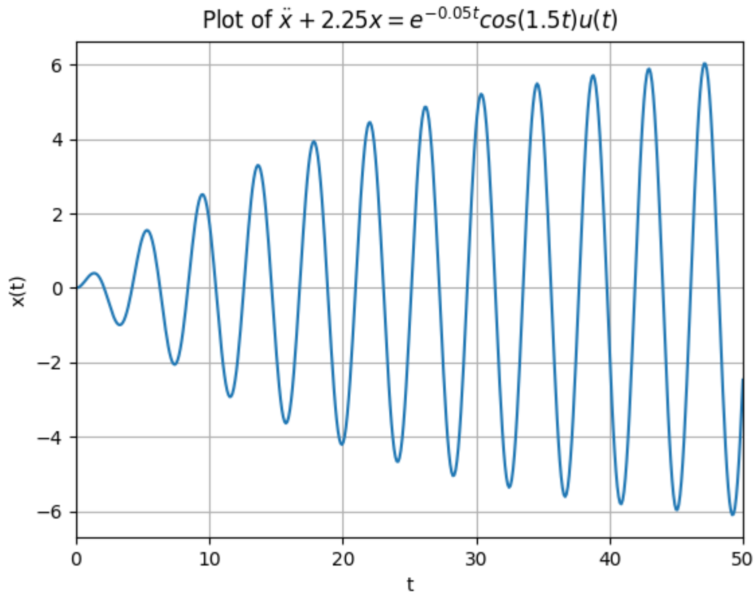

Figure 2: Plot for decay = 0.05

2.1.3 OBSERVATIONS:

1. We can see from the Plots that both of them have the same frequency $\frac{1.5}{2\pi}$Hz.
2. First Plot has Amplitude around 0.7 whereas 2nd Plot has around 6. So we can say that it around 10 times of 1st Plot.

2.2 Time Response of a Spring for 5 different Frequencies using *signal.lsim*

Now we model it as a **LTI** system and Plot the graph for 5 different frequencies using *scipy.signal.lsim*.

2.2.1 CODE:

```
plt.figure(2)
plt.title("Plot of $\ddot{x} + 2.25x = e^{-0.05t}cos(wt)u(t)$")

t = np.linspace(0,50,500)
k = np.linspace(1.4,1.6,5)

for i in k:
    u = np.cos(i*t)*np.exp(-0.05*t)*np.heaviside(t,1)
    t,y,svec=sp.lsim(H1,u,t)
    plt.plot(t,y)

plt.grid(True)
plt.xlabel("t")
plt.ylabel("x(t)")
plt.xlim(0,50)
plt.legend(["w=1.40","w=1.45","w=1.50","w=1.55","w=1.60"])
plt.savefig("figure_2")
plt.show()
```

2.2.2 PLOT:

The Plot of decay = 0.05 and 5 frequencies:

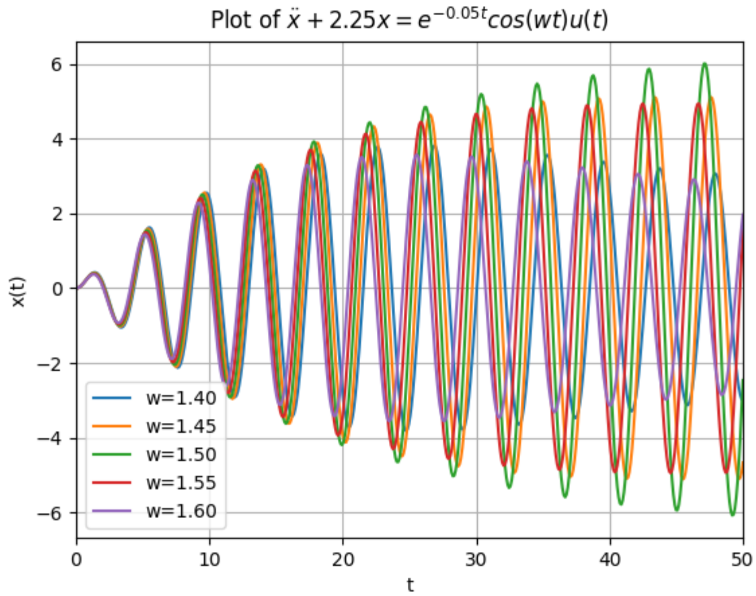

Figure 3: Plot for decay = 0.05 and several frequencies

2.2.3 OBSERVATIONS:

1. The Green Plot ($\omega = 1.5$) has maximum Amplitude out of all the Frequencies.
2. We can conclude that $\frac{1.5}{2\pi}$Hz is the resonant Frequency as it has maximum Amplitude/

2.3 Time Response of a Coupled Spring System

The coupled equations are:

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

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

Using the above two equations we get:

$$\ddddot{x} + 3\ddot{x} = 0$$

Using the initial conditions $x(0) = 1$, $\dot{x}(0) = y(0) = \dot{y}(0) = 0$:

$$s^4\mathcal{X}(s) - s^3 + 3(s^2\mathcal{X}(s) - s) = 0$$

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

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

Now thses can be solved and plotted using *scipy.signal.impulse*.

2.3.1 CODE:

```
plt.figure(4)
plt.title("Plot of x(t)/y(t) vs t")

H4 = sp.lti([1, 0, 2], [1, 0, 3, 0])
H5 = sp.lti([2], [1, 0, 3, 0])
t = np.linspace(0, 20, 200)
t, x = sp.impulse(H4,None,t)
t, y = sp.impulse(H5,None,t)

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

```
plt.xlabel("t")
plt.ylabel("x(t)/y(t)")
plt.legend(["$x(t): \ddot{x}+(x-y)=0,\dot{x}(0)=0,{x}(0)=1$",
"$y(t): \ddot{y}+2(y-x)=0,\dot{y}(0)=0,{y}(0)=0$"], loc = "upper right")
plt.grid(True)
plt.xlim(0,20)
plt.savefig("figure_3")
plt.show()
```

2.3.2 PLOT:

Plot of x(t) and y(t) vs t:

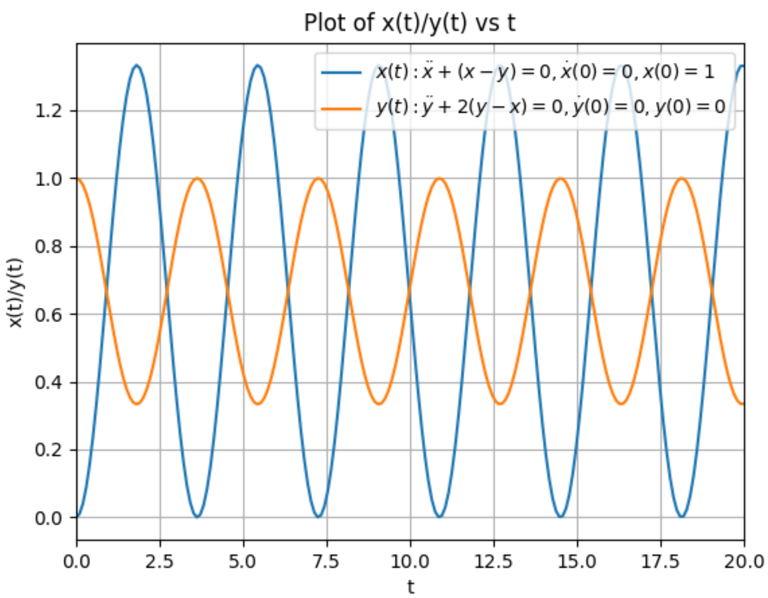

Figure 4: Plot for Coupled Oscillator

2.3.3 OBSERVATIONS:

1. Both the Plots have same Frequency, which means that both springs have same Frequency.
2. But they are out of Phase and have different Magnitudes, so they will be opposite side of equilibrium position and achieve maximum length together.

2.4 Magnitude and Phase Response of Two-Port Network

The transfer function of the given two-port network can be written as:

$$\frac{V_o(s)}{V_i(s)} = \mathcal{H}(s) = \frac{sL}{R + sL + \frac{1}{sC}}$$

For $R = 100, L = 10^{-6}, C = 10^{-6}$ we get:

$$\frac{V_o(s)}{V_i(s)} = \mathcal{H}(s) = \frac{10^6}{s^2 + 100s + 10^6}$$

This can be solved using *scipy.signal.bode*.

2.4.1 CODE:

```
plt.figure(5)
plt.title("Plot of |H(jw)| vs w")
```

```
L = 1e-6
C = 1e-6
R = 100

H6 = sp.lti([1], [L*C, R*C, 1])
w,S,phi = H6.bode()

plt.semilogx(w,S)
plt.xlabel("w")
plt.ylabel("|H(jw)| (in dB)")
plt.xlim(1e3,1e9)
plt.grid(True)
plt.savefig("figure_4")
plt.show()

plt.figure(6)
plt.title("Plot of \N{GREEK CAPITAL LETTER PHI}(H(jw)) vs w")

plt.semilogx(w,phi)
plt.xlabel("w")
plt.ylabel(" \N{GREEK CAPITAL LETTER PHI}(H(jw)) in \N{DEGREE SIGN}")
plt.xlim(1e3,1e9)
plt.grid(True)
plt.savefig("figure_5")
plt.show()
```

2.4.2 PLOTS:

First we Plot the Magnitude Response of the Network:

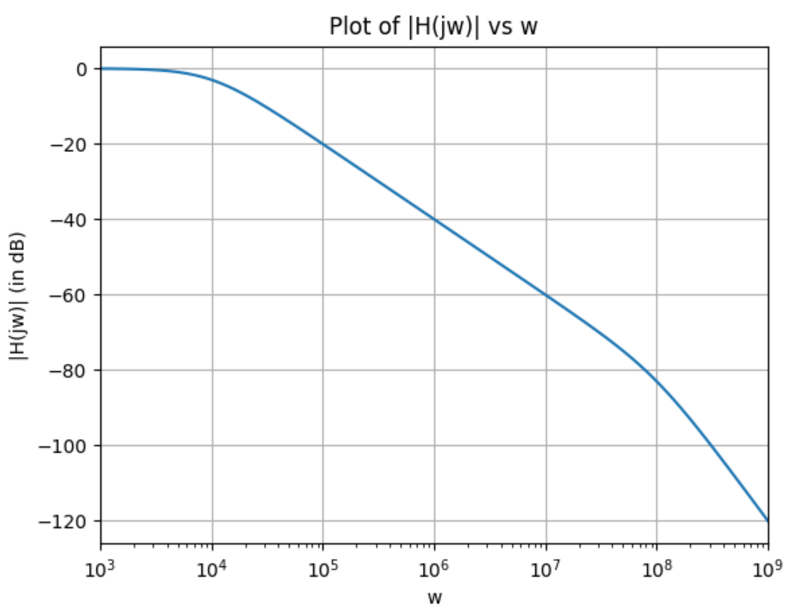

Figure 5: Magnitude Response of the Network

Now we Plot the Phase Response of the Network:

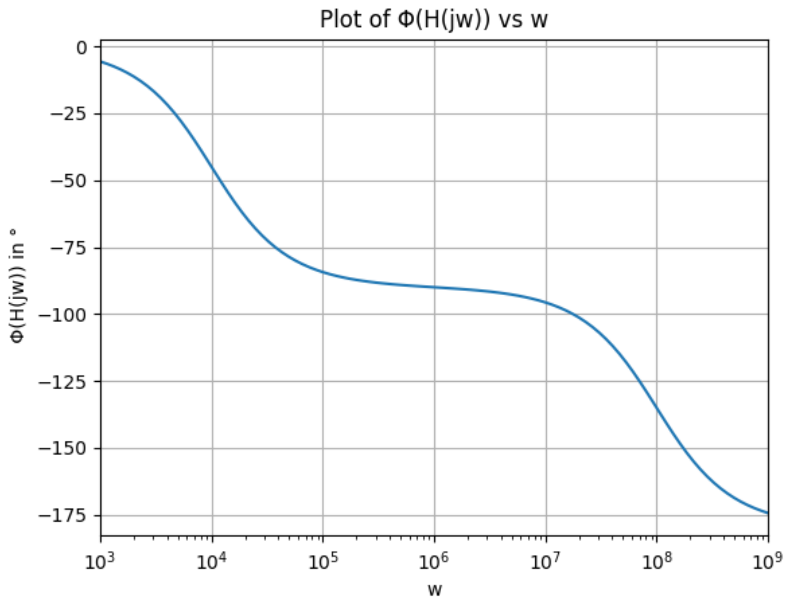

Figure 6: Phase Response of the Network

2.4.3 OBSERVATIONS:

1. We can see from the 1st Plot that the Gain decreases with increase in Frequency.
2. We can see from the 2nd Plot that the Phase difference also decreases with increase in Frequency.

2.5 Output Signal for a given Particular Input using the Two Port Network

Now, when the input to the network, $v_i(t)$ is $cos(10^3t)u(t) - cos(10^6t)u(t)$, the output is given by:

$$V_o(s) = V_i(s)\mathcal{H}(s)$$

where

$$V_i(s) = \mathcal{L}\{v_i(t)\}$$

The Laplace transform of $f(t) = e^{-at}cos(\omega t)u(t)$ is given as:

$$\mathcal{L}\{f(t)\} = \frac{s+a}{(s+a)^2+\omega^2}$$

For a = 0:

$$\mathcal{L}\{f(t)\} = \frac{s}{s^2+\omega^2}$$

Or,

$$V_i(s) = \mathcal{L}\{v_i(t)\} = \frac{s}{s^2+(10^3)^2} - \frac{s}{s^2+(10^6)^2}$$

This can be solved and plotted using *scipy.signal.lsim*.

2.5.1 CODE:

```
plt.figure(7)
plt.title("Plot of $v_o(t)$ vs t")

w1  = int(1e3)
w2 = int(1e6)

t = np.linspace(0, 0.1, w2)
u = np.cos((w1)*t)*np.heaviside(t,1)-
np.cos((w2)*t)*np.heaviside(t,1)
t,y,svec=sp.lsim(H6,u,t)
```

```
plt.plot(t,y)
plt.ylim(-1,1)
plt.xlim(0,0.00003)
plt.xlabel("t")
plt.ylabel("$v_o(t)$")
plt.grid(True)
plt.savefig("figure_6")
plt.show()

plt.figure(8)
plt.title("Plot of $v_o(t)$ vs t")

t2,y2,svec2=sp.lsim(H6,u,t)

plt.plot(t,y)
plt.xlim(0,0.01)
plt.xlabel("t")
plt.ylabel("$v_o(t)$")
plt.grid(True)
plt.savefig("figure_7")
plt.show()
```

2.5.2 PLOTS:

First we'll Plot the Output Signal for $0 < t < 30\mu s$:

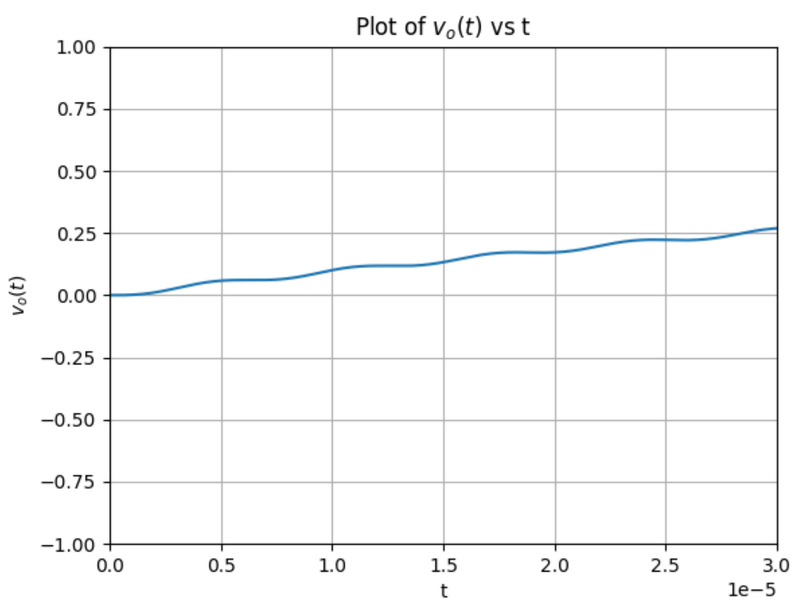

Figure 7: Output Signal for $0 < t < 30\mu s$

Now we'll look at the Long Term Response on the msec scale:

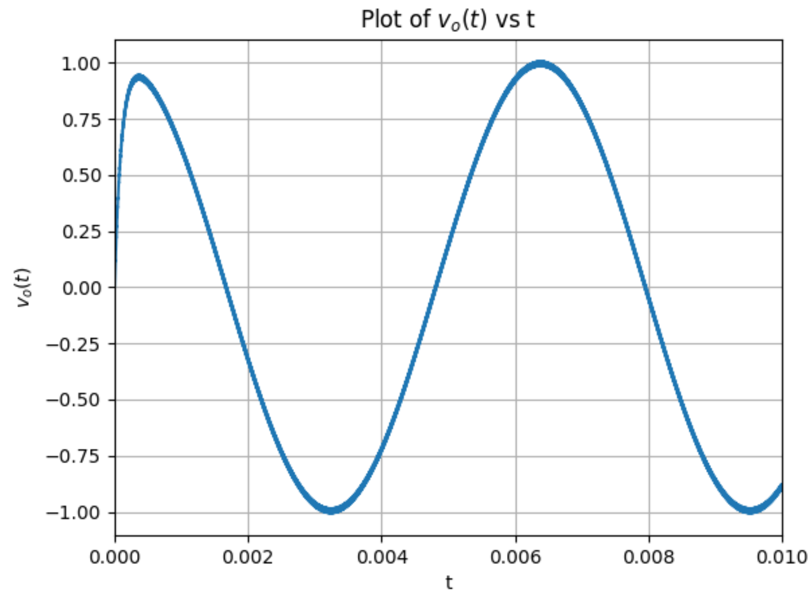

Figure 8: Long Term Response on ms Scale

We zoom in on the Plot to see that this Plot is made up of another High Frequency Signal:

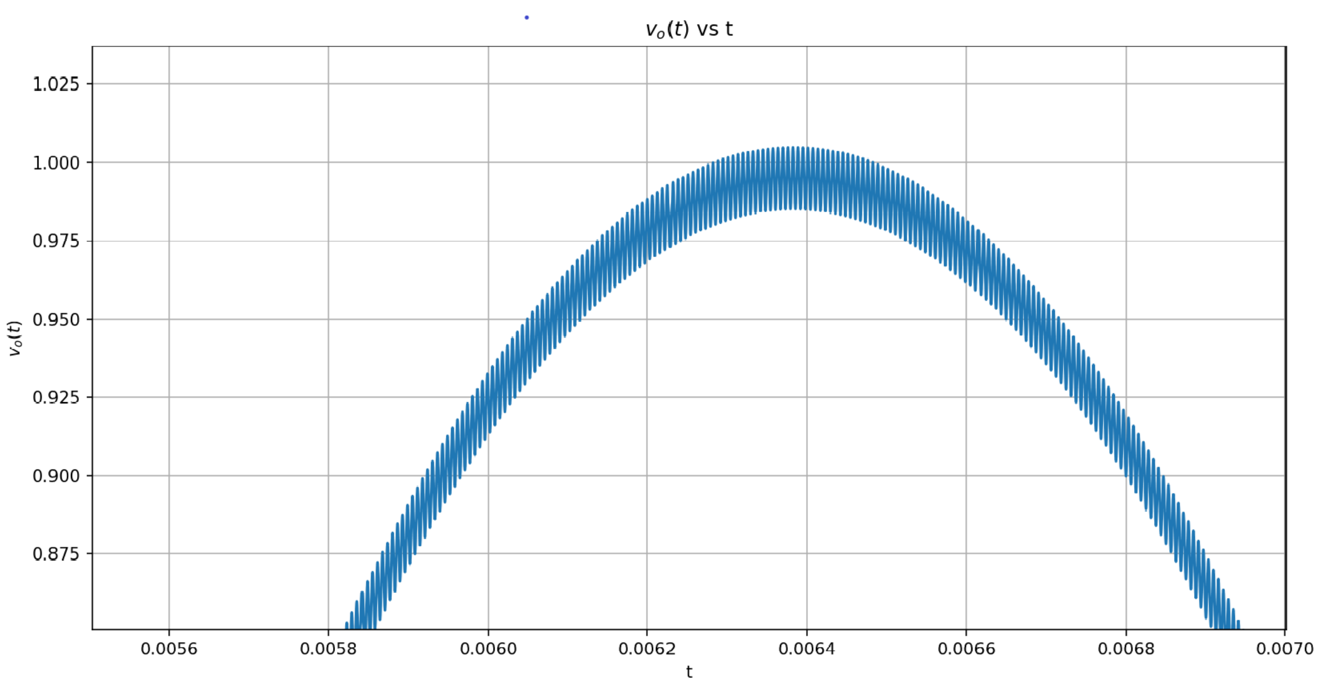

Figure 9: Zoomed in Long Term Response

2.5.3 OBSERVATIONS:

1. In the beginning, the High frequency component has very less effect due to value of time being so less.
2. The Plot has a Frequency of around 160Hz with V_{pp} around 2.
3. When we zoom in, we can see a high frequency signal with around 0.02 V_{pp}.

3 Conclusion

In this assignment, we analysed the solution of various continuous time **LTI** systems using *Laplace transforms* with help of *scipy.signal* toolbox and made observations about he solutions using their Plots.