EE2703 Assignment 6: The Laplace Transform

Anvith Pabba EE19B070

20th April 2021

1 Introduction

In this assignment, we look at how to analyse "Linear Time-invariant Systems" using the scipy.signal library in Python.
We analyse the outputs of 3 systems:

1. A forced spring (oscillatory system)
2. Coupled system of differential equations
3. An RLC low pass filter

2 The Assignment

2.1 Question 1 : Response of a spring system

the time response of the string satisfies

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

where, the function f(t) is given by:

$$f(t) = cos(1.5t)exp^{-0.5t}u(t) \tag{2}$$

The Laplace transform of f(t), i.e F(s) is given by:

$$F(s) = \frac{s+0.5}{(s+0.5)^2+2.25} \tag{3}$$

On solving the equations in Laplace domain, we get:

$$X(s) = \frac{s+0.5}{((s+0.5)^2+2.25)(s^2+2.25)} \tag{4}$$

We then use impulse response of X(s) to get x(t),

2.1.1 Plot of x(t)

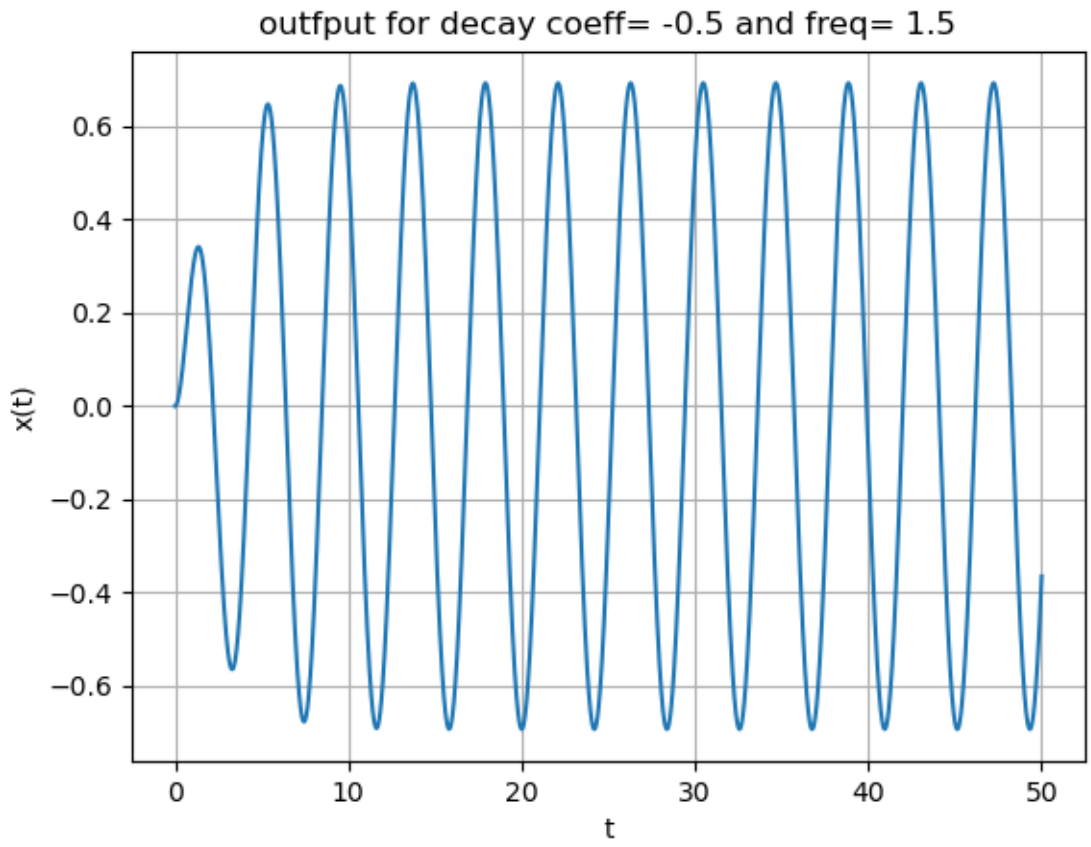


Figure 1: Response of a forced oscillatory system

2.1.2 Code

I defined a function that returns the output based on input decay coefficient and frequency.

```
def H_s(a,b): #defining the Laplace transform, -a is the decay coefficiant, and
                                        b is the frequency
  num = poly1d([1,a])
  den1 = poly1d([1,2*a,a*a + b*b])
  den2 = poly1d([1,0,2.25])
  den = polymul(den1,den2)
  H = sp.lti(num,den)
  return H

H1 = H_s(0.5,1.5)

t,x = sp.impulse(H1,None, linspace(0,50,10000))
plot(t,x)
xlabel('t')
ylabel('x(t)')
title('outfput for decay coeff= -0.5 and freq= 1.5')
grid()
show()
```

2.2 Question 2

now, we find the response for a system given by:

$$f(t) = cos(1.5t)exp^{-0.05t}u(t) \tag{5}$$

In Laplace domain,

$$F(s) = \frac{s + 0.05}{(s + 0.05)^2 + 2.25} \tag{6}$$

Final output in Laplace domain,

$$X(s) = \frac{s + 0.05}{((s + 0.05)^2 + 2.25)(s^2 + 2.25)} \tag{7}$$

2.2.1 Plot

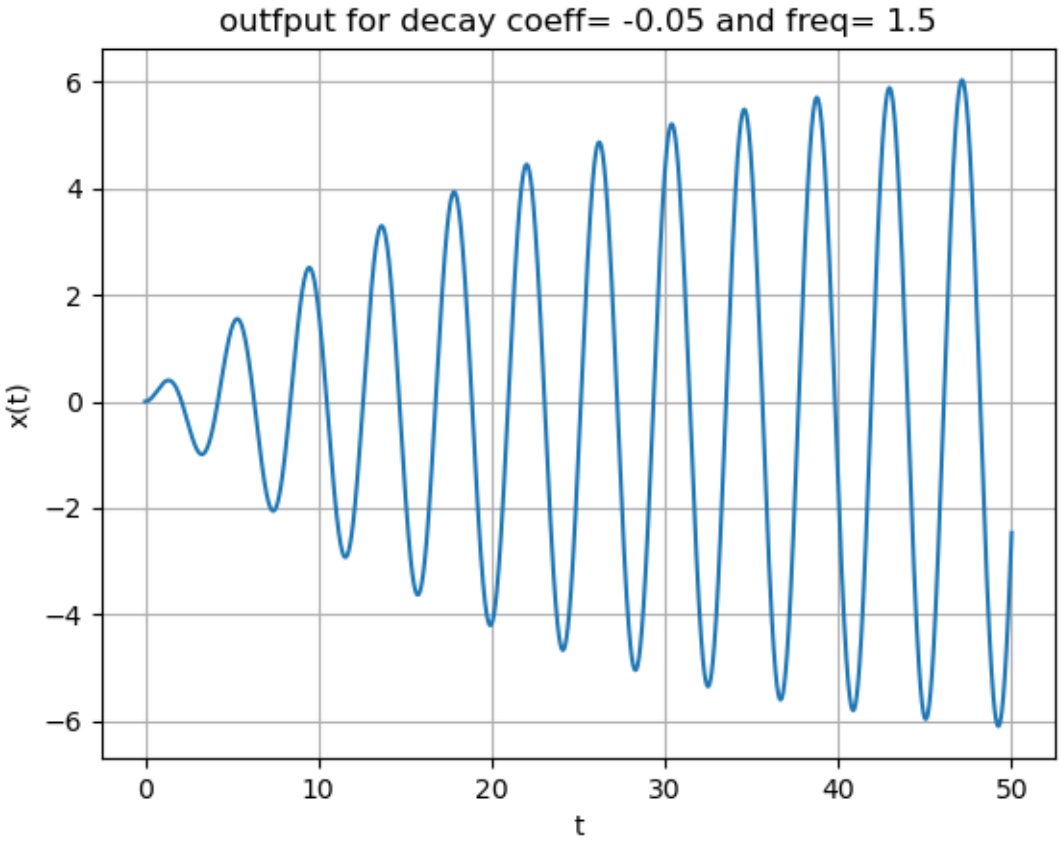


Figure 2: Response of a forced oscillatory system

2.2.2 Code

```
#Question 2:

H2 = H_s(0.05,1.5)

t2,x2 = sp.impulse(H2,None, linspace(0,50,10000))
plot(t2,x2)
xlabel('t')
```

```
ylabel('x(t)')
title('outfput for decay coeff= -0.05 and freq= 1.5')
grid()
show()
```

2.2.3 Observations

We can see that Plot2 and Plot1 are very similar, but plot 2 takes a longer time to reach steady state as its decay coefficient is lower.

2.3 Question 3: over a range of frequencies

We now find the output if the frequency varied from 1.4 to 1.6 in increments of 0.05.

2.3.1 Plot

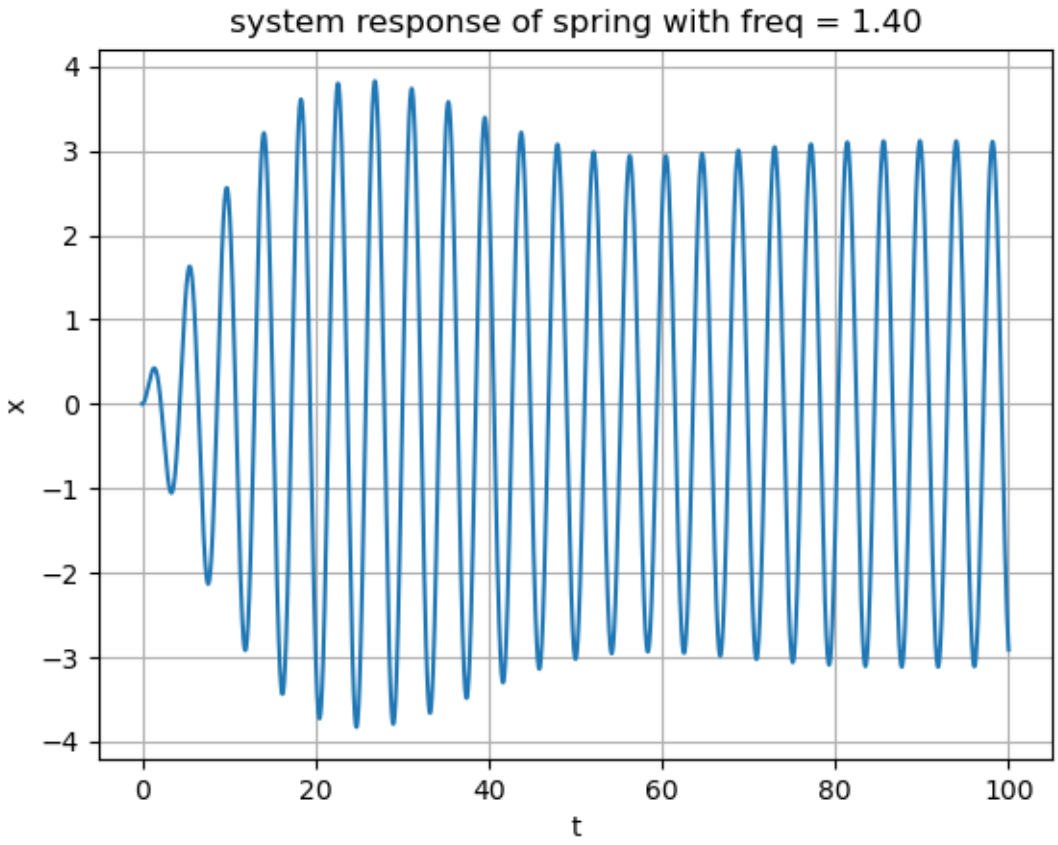


Figure 3: Response of a forced oscillatory system

2.3.2 Plot

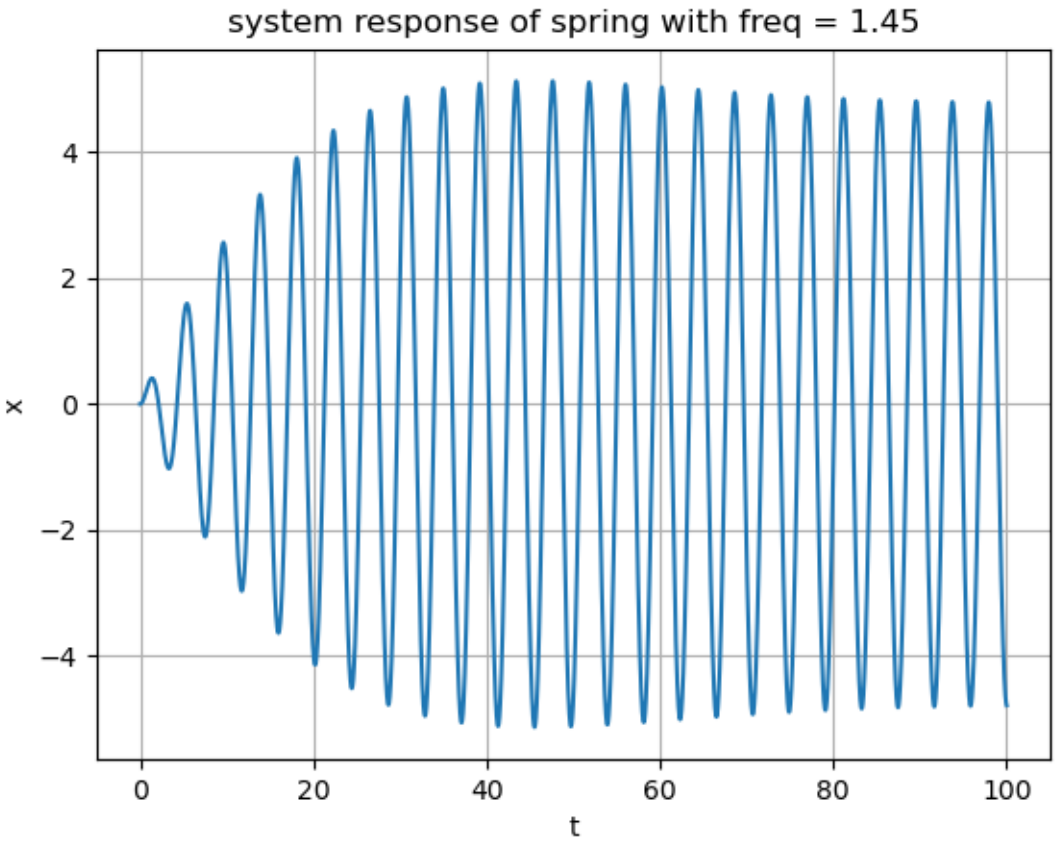


Figure 4: Response of a forced oscillatory system

2.3.3 Plot

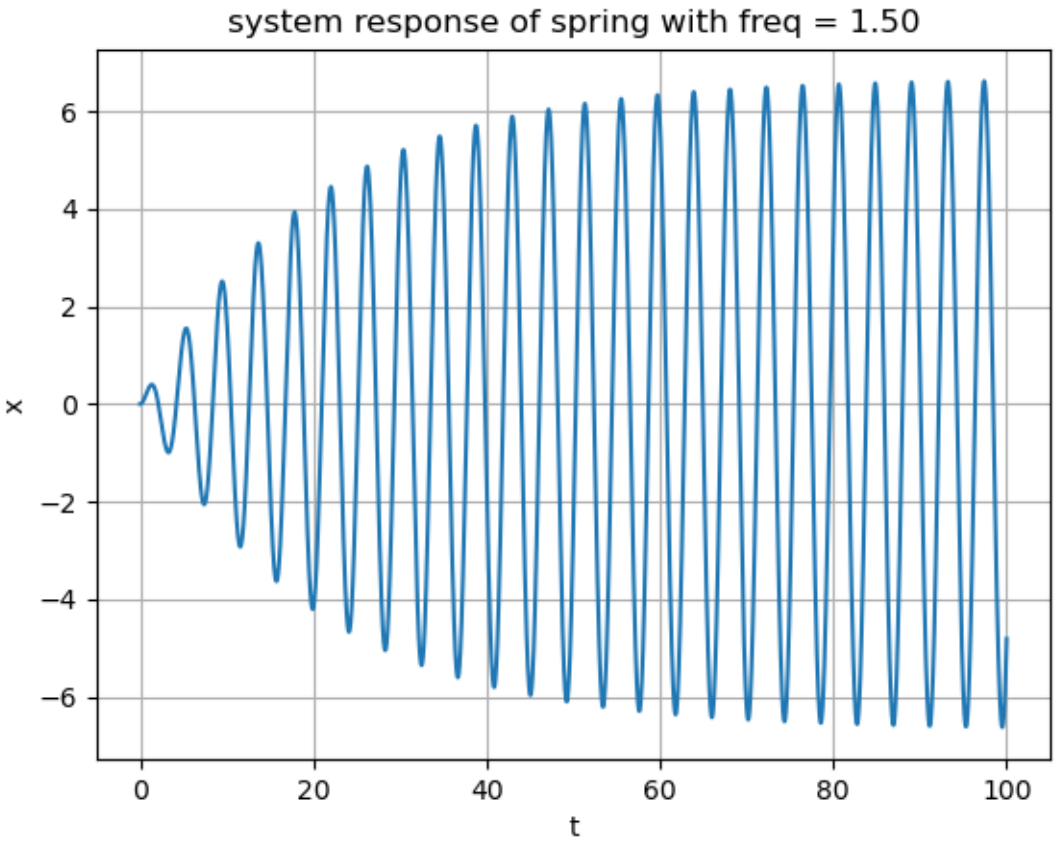


Figure 5: Response of a forced oscillatory system

2.3.4 Plot

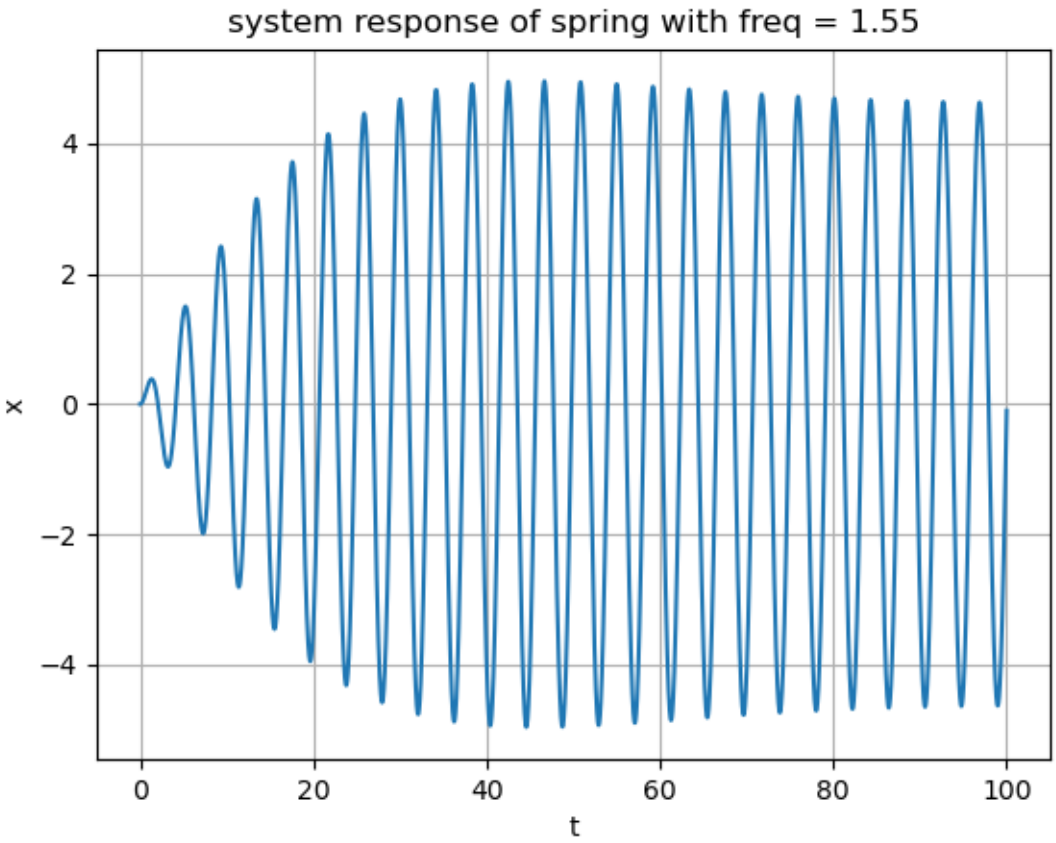


Figure 6: Response of a forced oscillatory system

2.3.5 Plot

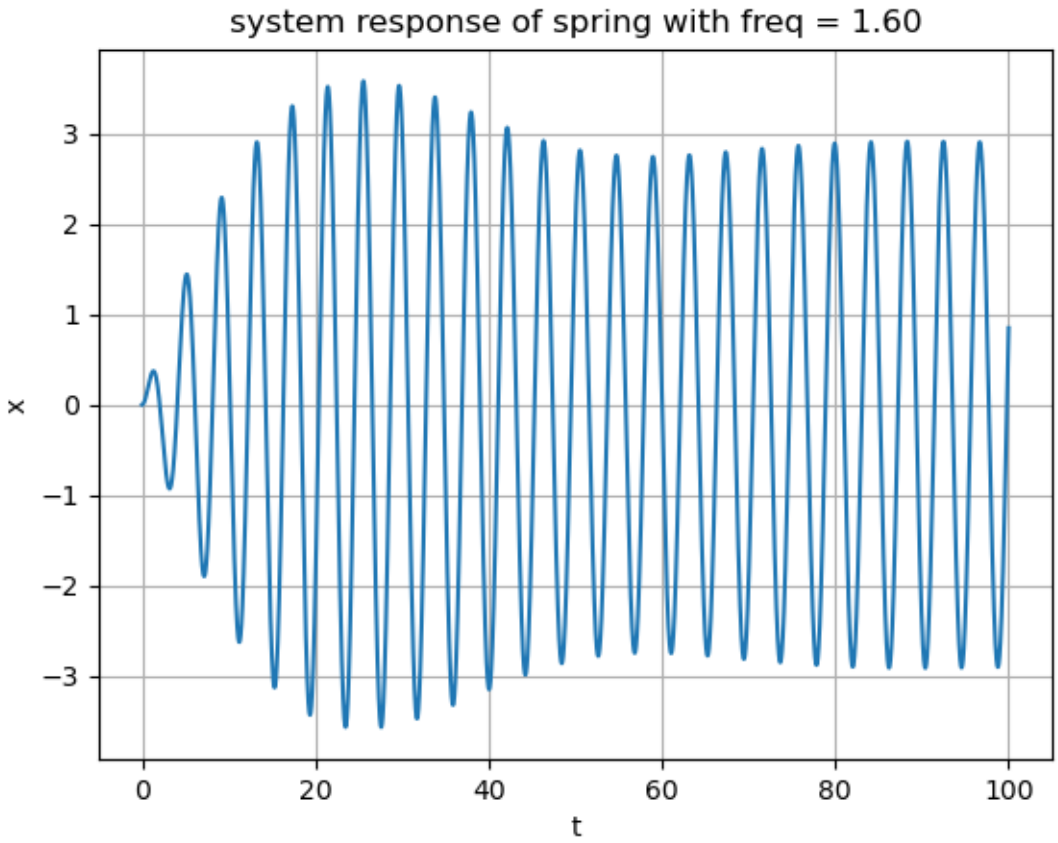


Figure 7: Response of a forced oscillatory system

2.3.6 Observations and Analysis

We can clearly see that the closer the frequency is to 1.5, the higher the amplitude at steady state. The reason for this is because the resonant frequency of the system is 1.5. (As given in the system equation). This is the reason at frequency = 1.5, the graph is extremely smooth and has the highest amplitude at steady state.

2.3.7 Code

```
#Question 3:

for i in range(5): #finding output for a range of frequencies from 1.4 to 1.6
                                        with 0.05 increments
  f = 1.4 + i*0.05
  H = H_s(0.05,f)
  t,x = sp.impulse(H,None, linspace(0,100,10000))
  xlabel('t')
  ylabel('x')
  title('system response of spring with freq = %0.2f'%f)
  plot(t,x)
  grid()
  show()
```

2.4 Question 4: Coupled spring problem

The equations are:

$$\ddot{x} + (x - y) = 0 \tag{8}$$

$$\ddot{y} + 2(y - x) = 0 \tag{9}$$

after solving, we get the fourth order equation:

$$\dddot{\ddot{x}} + 3\ddot{x} = 0 \tag{10}$$

with initial conditions.

on solving, we get:

$$X(s) = \frac{s^2 + 2}{s^3 + 2s} \tag{11}$$

$$Y(s) = \frac{2}{s^3 + 2s} \tag{12}$$

in time domain, for a time between 0 and 20seconds,

2.4.1 Plot of x(t)

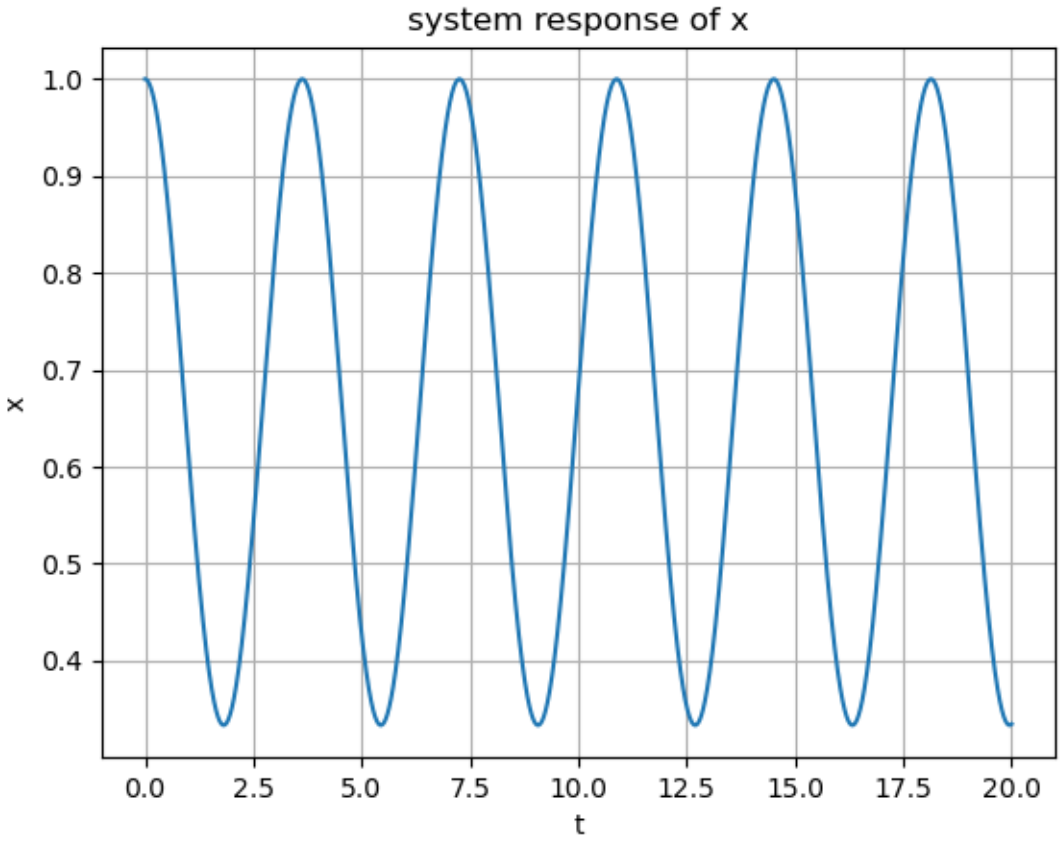


Figure 8: x(t)

2.4.2 Plot of y(t)

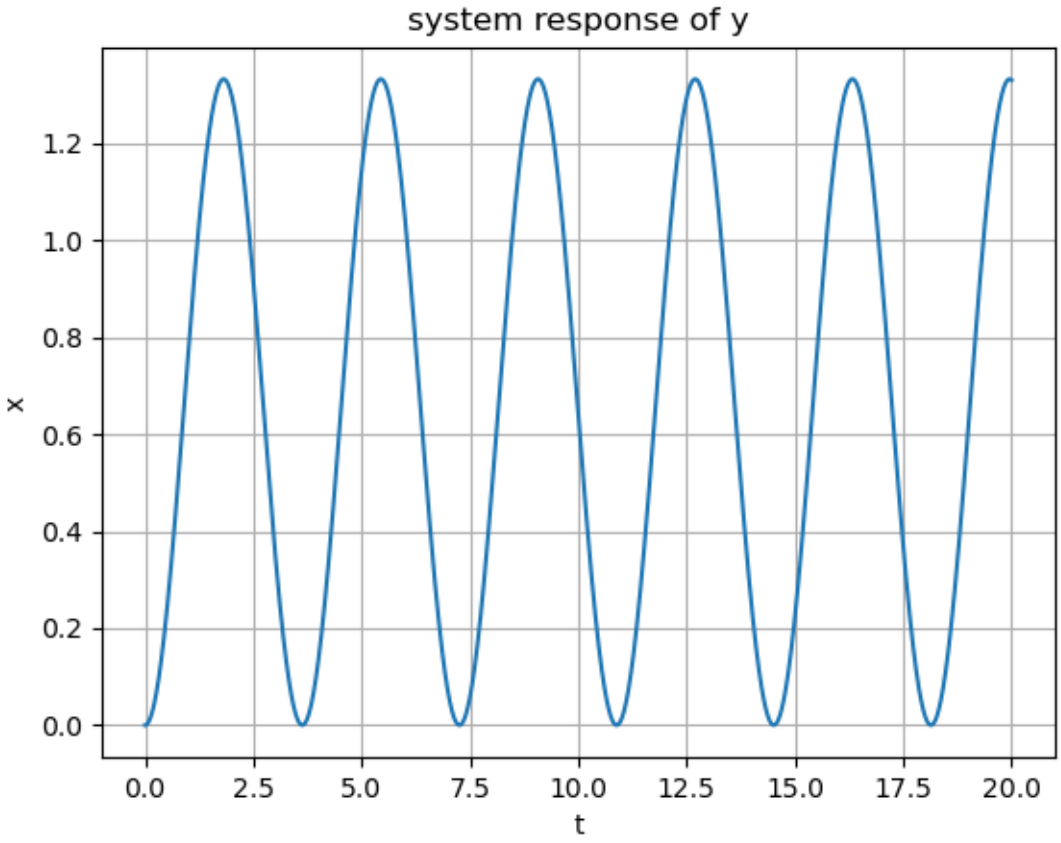


Figure 9: y(t)

2.4.3 Observations

We see that both are sinusoidal, but the amplitudes and phases of both of them are different.

2.4.4 Code

```
#Question 4:

H_x = sp.lti([1,0,2],[1,0,3,0])
H_y = sp.lti([2],[1,0,3,0])

t,x = sp.impulse(H_x,None, linspace(0,20,10000))
xlabel('t')
ylabel('x')
title('system response of x')
plot(t,x)
grid()
show()

t,x = sp.impulse(H_y,None, linspace(0,20,10000))
xlabel('t')
ylabel('x')
title('system response of y')
plot(t,x)
```

```
grid()
show()
```

2.5 Question 5: Response of the Steady State Transfer function of an RLC network

The steady state transfer function is :

$$H(s) = \frac{10^6}{s^2 + 100s + 10^6} \tag{13}$$

Using H.bode(), we can plot the bode plots.

2.5.1 Magnitude bode plot

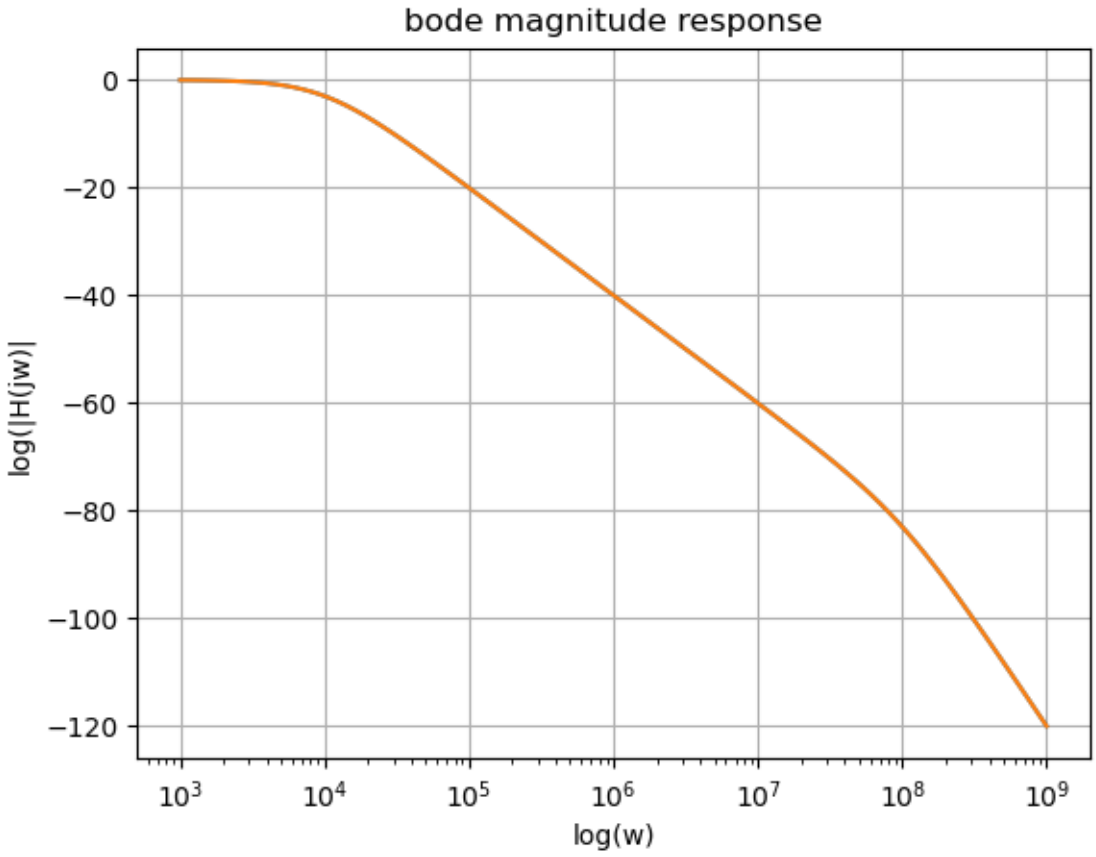


Figure 10: Bode magnitude plot

2.5.2 Magnitude phase plot

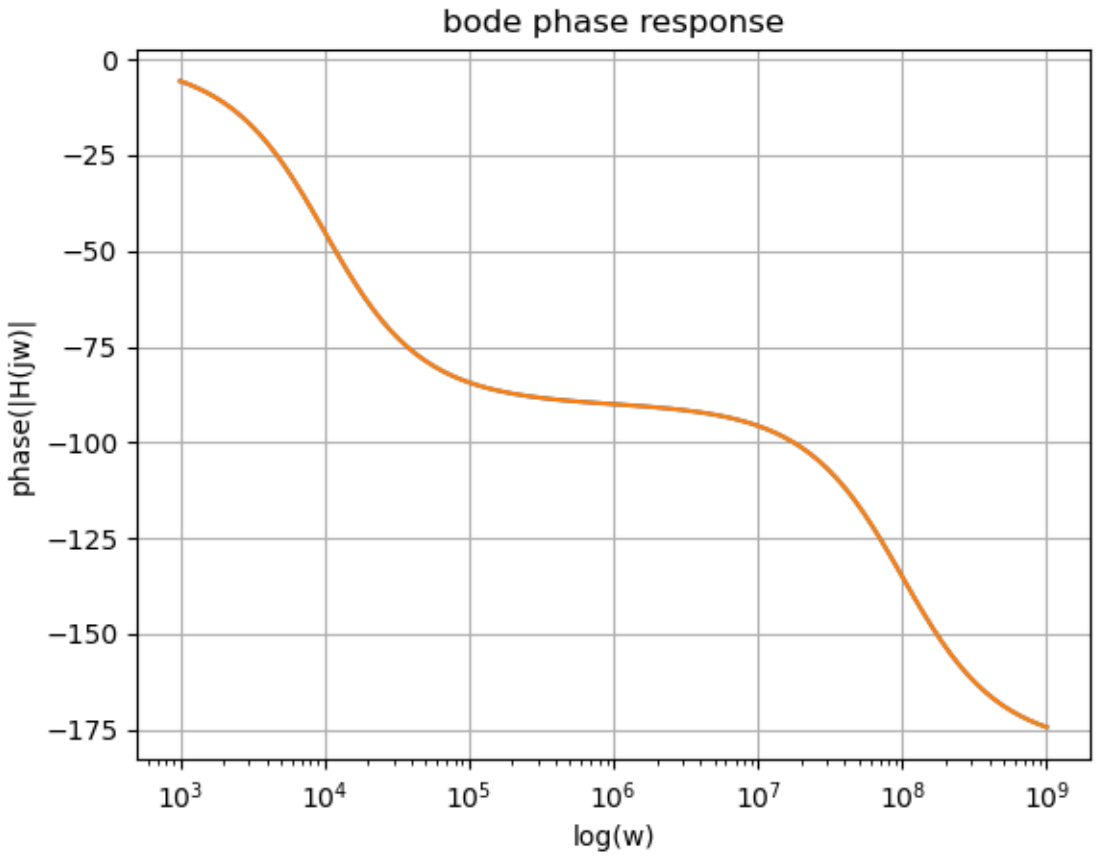


Figure 11: Bode Phase plot

2.5.3 Code

```
#Question 5:

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

H = sp.lti(1,[L*C,R*C,1]) #transfer function
w,S,phi = H.bode()
semilogx(w,S)
xlabel('log(w)')
ylabel('log(|H(jw)|')
title('bode magnitude response')
plot(w,S)
grid()
show()

semilogx(w,phi)
xlabel('log(w)')
ylabel('phase(|H(jw)|')
title('bode phase response')
plot(w,phi)
grid()
show()
```

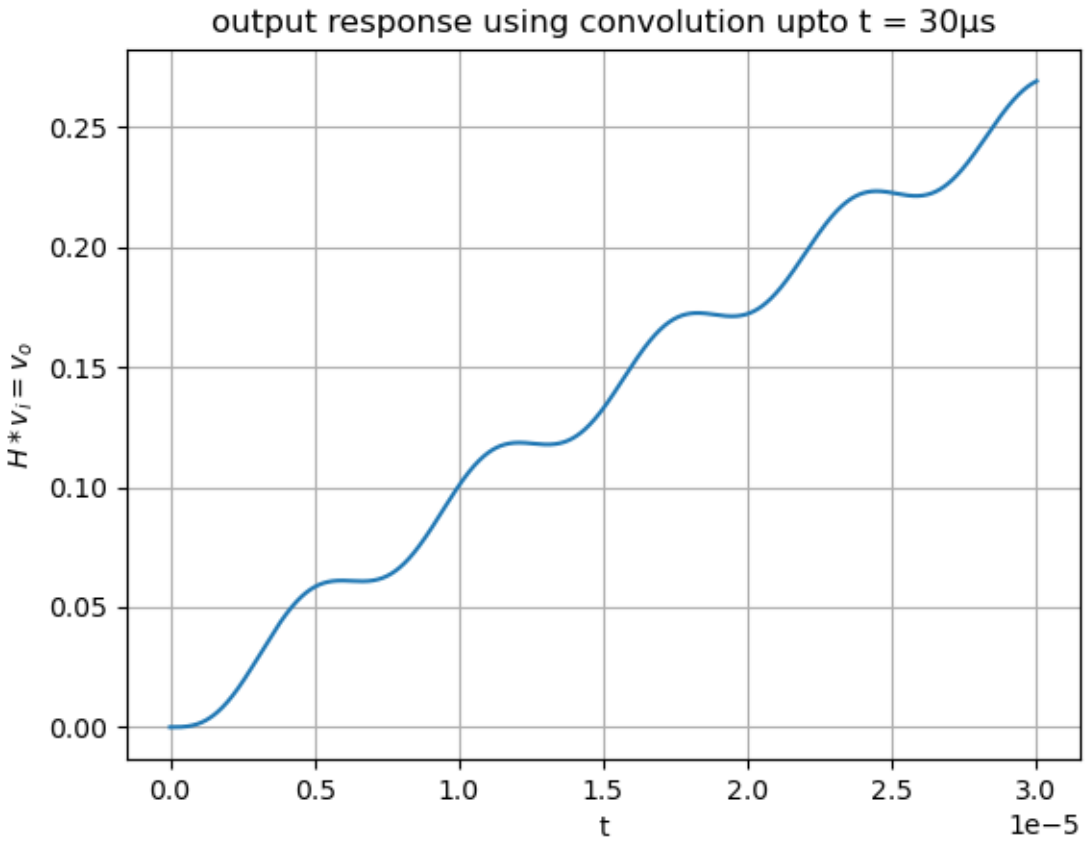
2.6 Question 6:

When input is given by,

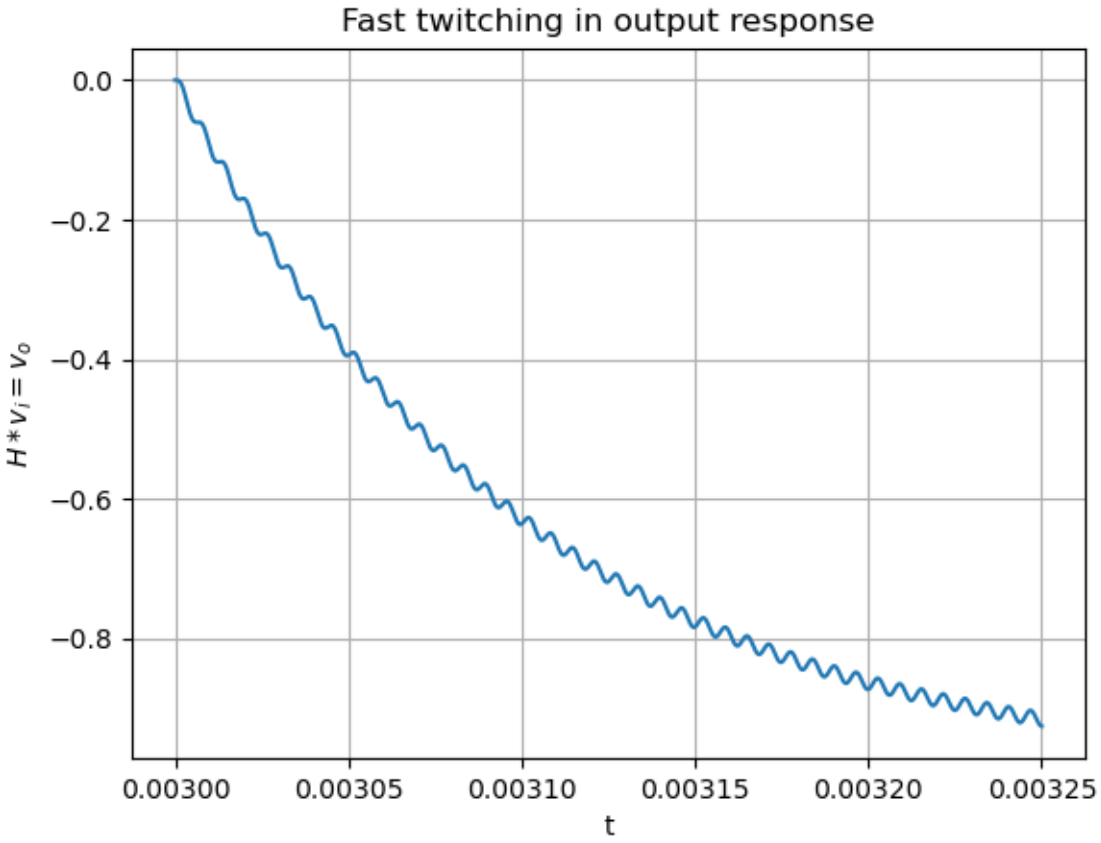
$$v_i(t) = cos(10^3 t)u(t) - cos(10^6 t)u(t) \tag{14}$$

We get the output $v_o(t)$ by finding the convolution of x(t) and $v_i(t)$, we do this using sp.lsim method.

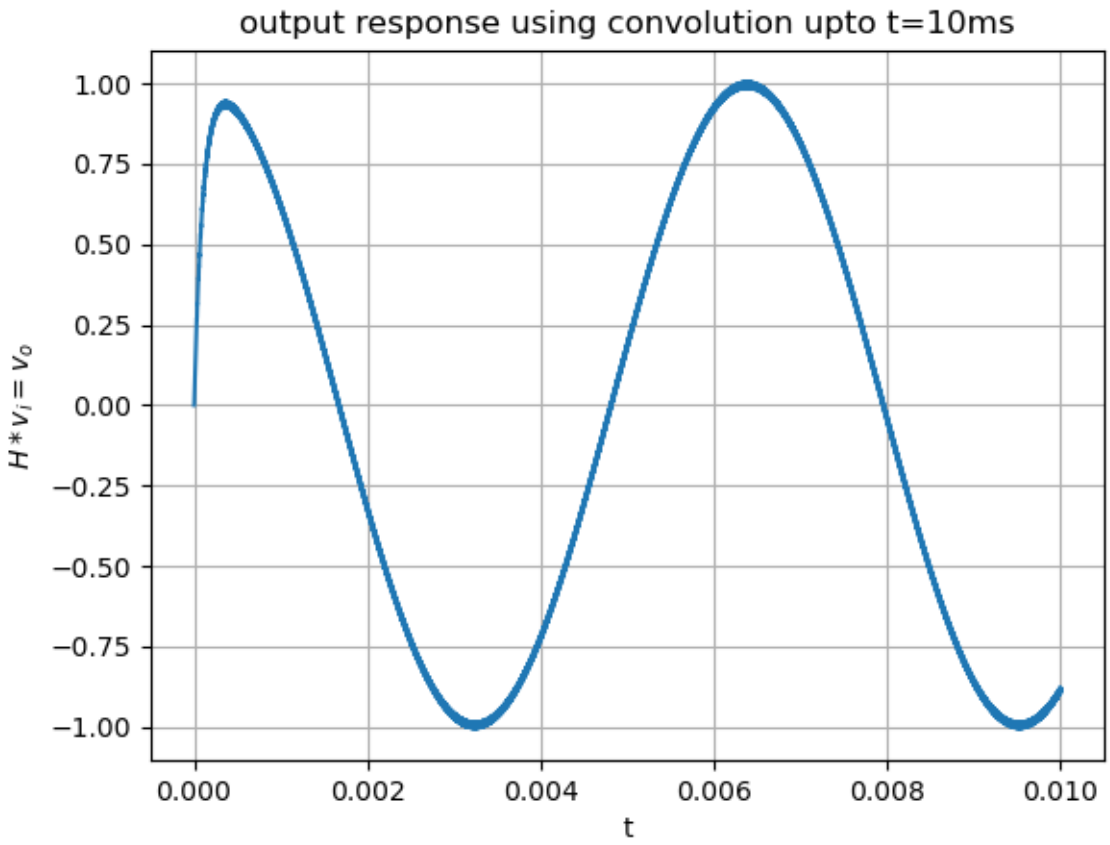
2.6.1 Signal between 0 and 30μs



2.6.2 Signal between 30ms and 32.5ms



2.6.3 Signal between 0 and 10ms



2.6.4 Observations

the reason we see a small periodic change in the output is because the original RLC circuit is a low pass filter, so it filters out the 10^6 component of frequency. and since its not an ideal filter, this just means the magnitude of the low frequency is decreased. Hence the output consists of 10^3 component with a small varying 10^6 component.

2.6.5 Code

```
#Question 6:

t = linspace(0,1e-2,10000)
v_i = cos(1e3*t) - cos(1e6*t)
t,y,svec = sp.lsim(H,v_i,t)
plot(t,y)
xlabel('t')
ylabel('$H * v_i = v_o$')
title('output response using convolution upto t=10ms')
grid()
show()

t = linspace(0,30*1e-6,10000)
v_i = cos(1e3*t) - cos(1e6*t)
t,y,svec = sp.lsim(H,v_i,t)
plot(t,y)
xlabel('t')
```

```
ylabel('$H * v_i = v_o$')
title('output response using convolution upto t = 30\u03BCs')
grid()
show()


t = linspace(3*1e-3,3.25*1e-3,10000)
v_i = cos(1e3*t) - cos(1e6*t)
t,y,svec = sp.lsim(H,v_i,t)
plot(t,y)
xlabel('t')
ylabel('$H * v_i = v_o$')
title('Fast twitching in output response')
grid()
show()
```

3 Conclusions

In this assignment, we have learnt how to use scipy.signal, a Signal analysing tool box in python. We have learnt how to find the response of a rational LTI system along with how to find the convolution of two functions in the time domain.