Assignment No 6

M Srinivas Reddy

March 8, 2020

1 Abstract

Linear Time Invariant Systems are the most analysed systems for Engineers. If an LTI system is given, we most often try to solve for the output given the input or magnitude, phase plots of the transfer function. Using a special module in python called **scipy.signal** we try to solve for the above mentioned quantities.

2 Introduction

In this assignment we are going to use python to solve for continuous time signal given an input and transfer function, and try to find transfer functions of systems and plot the magnitude and phase plots. We also vary the frequency of cos in input to a particular system and try to observe the change in output signals.

3 Useful Commands

The first thing to be done is to import the required modules. In this assignment we are going to use a module called **scipy.signal**.

3.1 Generation and Arithmetic of Polynomials

- To define polynomials we use the command **poly1d([a,b,c]**.
- To add polynomials we use **polyadd([a,b],[c,d,f])**.
- To multiply polynomials we use **polymul([a,b,c],[d,f])**.

Example Code:

```
H=sp.lti([1,0.5],polymul([1,1,2.5],[1,0,1]))
p=polymul([1,0.1,0.025+(1.4+0.05*i)**2],[1,0,2.25]))
```

3.2 Operations on LTI Systems

- To define a transfer function we use **sp.lti(Num,Den)**
- To get the magnitude and phase of transfer function as a function of frequency we use **w,S,phi=H.bode()**, where **H** is a transfer function.
- To find impulse response of a transfer function we use **t,x = sp.impulse(H,None,linspace(0,10,101)**.
- To find output of a system given the transfer function and input, we use **t,y,svec=sp.lsim(H,u,t)**, where **u(t)** is the input signal.

Example Code:

```
H=sp.lti([1,0.5],polymul([1,1,2.5],[1,0,1]))
t,x=sp.impulse(H,None,linspace(0,50,500))

t=linspace(0,10**-2,10**5)
u=cos((10**3)*t)-cos((10**6)*t)
t,y,svec=sp.lsim(H,u,t)
```

4 Problem 1

We have an input $f(t) = cos(\omega t)e^{\alpha}u_0(t)$.
The laplace transform is given by

$$F(s) = \frac{s-\alpha}{(s-\alpha)^2+\omega^2}$$

The relation between input and output is given by

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

Therefore the frequency response of **x(t)** given the initial conditions, x(0)=0, $\dot{x}(0)$=0 is

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

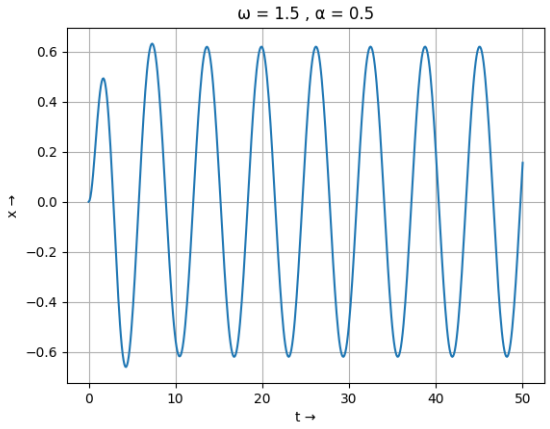
Using the command **sp.impulse** we can find **x(t)**.
In the first case we have ω = 1.5 and α = -0.5.
In the second case we change α to - 0.05

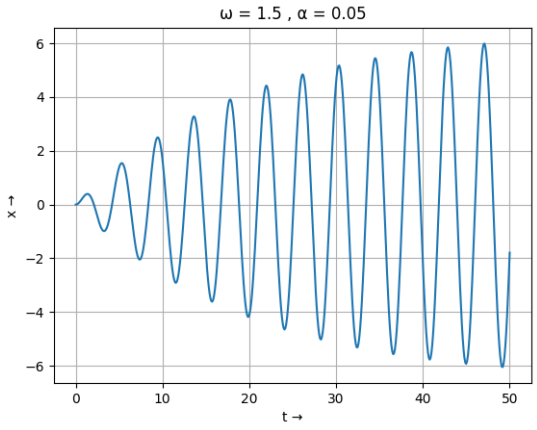
Case 1:

```
H=sp.lti([1,0.5],polymul([1,1,2.5],[1,0,1]))
t,x=sp.impulse(H,None,linspace(0,50,500))
plot(t,x)
```

Case 2:

```
H=sp.lti([1,0.05],polymul([1,0.1,2.275],[1,0,2.25]))
t,x=sp.impulse(H,None,linspace(0,50,500))
plot(t,x)
```

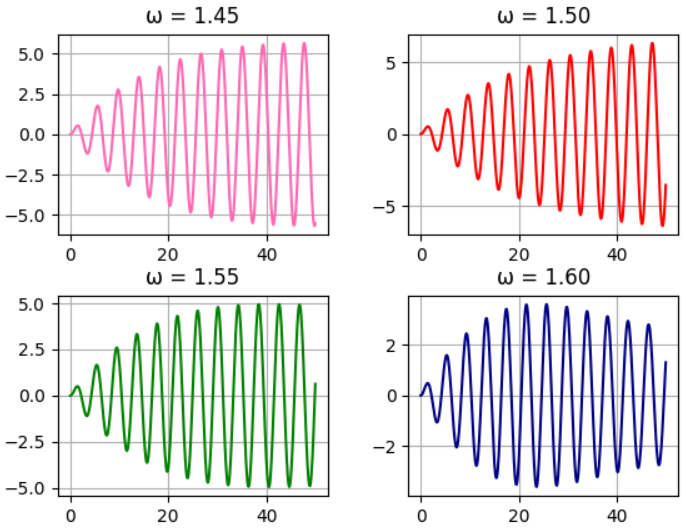




We now vary ω from 1.4 to 1.6 in steps and 0.05 using a for loop and plot output signal in each case.

```
for i in range(1,5):
p=polymul([1,0.1,0.025+(1.4+0.05*i)**2],[1,0,2.25])
H=sp.lti([1,0.5],p)
t,x=sp.impulse(H,None,linspace(0,50,500))
w=1.4+0.05*i
subplots_adjust(wspace=0.3,hspace=0.3)
if i<=2:
```

```
        axes[0][i-1].set_title('\u03C9 = %1.2f'%w)
        axes[0][i-1].plot(t,x,color=color[i-1])
    else:
        axes[1][i-3].set_title('\u03C9 = %1.2f'%w)
        axes[1][i-3].plot(t,x,color=color[i-1])
```



5 Problem 2

In this problem we have a spring coupled in x and y directions. The equations are

$$\ddot{x} + (x - y) = 0 \tag{1}$$

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

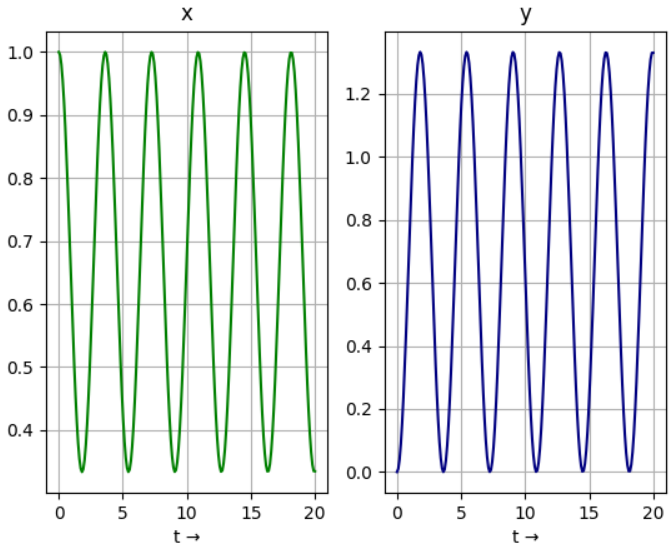
The initial conditions are x(0) = 1 and the rest are zeros.
On solving for x and y we get

$$X(s) = \frac{s^2 + 2}{s^4 + 3s^2} \tag{3}$$

$$Y(s) = \frac{2}{s^4 + 3s^2} \tag{4}$$

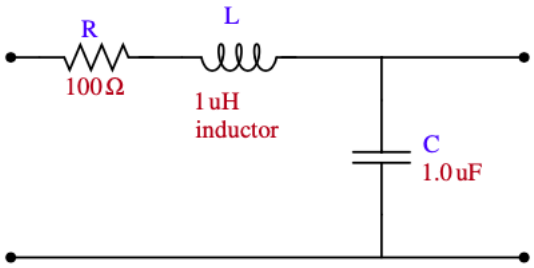
We can now find x(t) and y(t).

```
H1=sp.lti([1,0,2],[1,0,3,0])
t,x=sp.impulse(H1,None,linspace(0,20,200))
H2=sp.lti(2,[1,0,3,0])
t,y=sp.impulse(H2,None,linspace(0,20,200))
```



6 Problem 3

We now have to solve for transfer function of the below RLC circuit.

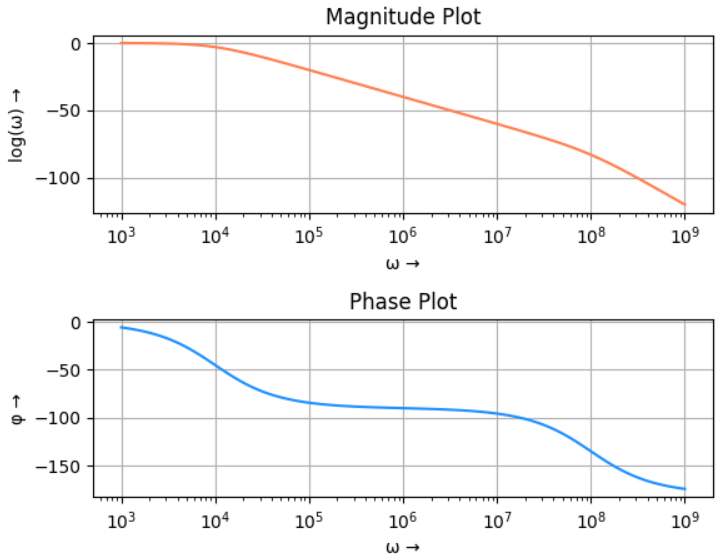


The transfer function of the circuit is

$$H(s) = \frac{1}{10^{-12}s^2 + 10^{-}4s + 1}$$

We now can find the magnitude and phase of H(s) as a function of frequency using **H.bode()** command.

```
H=sp.lti(1,[10**-12,10**-4,1])
w,S,phi=H.bode()
subplot(2,1,1)
semilogx(w,S,color='coral')
subplot(2,1,2)
semilogx(w,phi,color='dodgerblue')
```

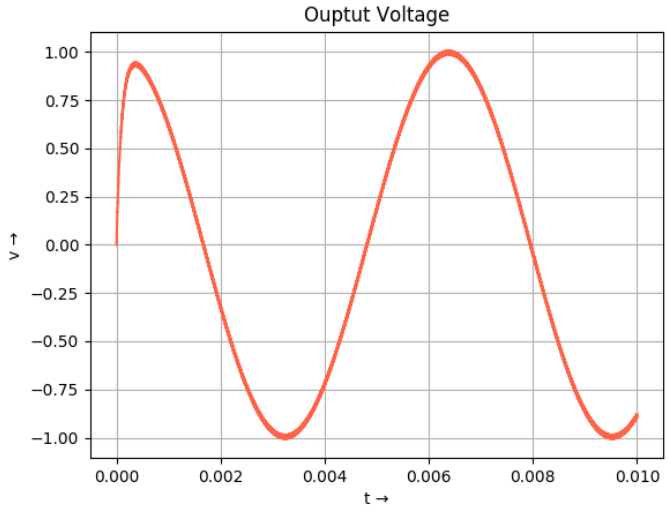


Now input to the system is

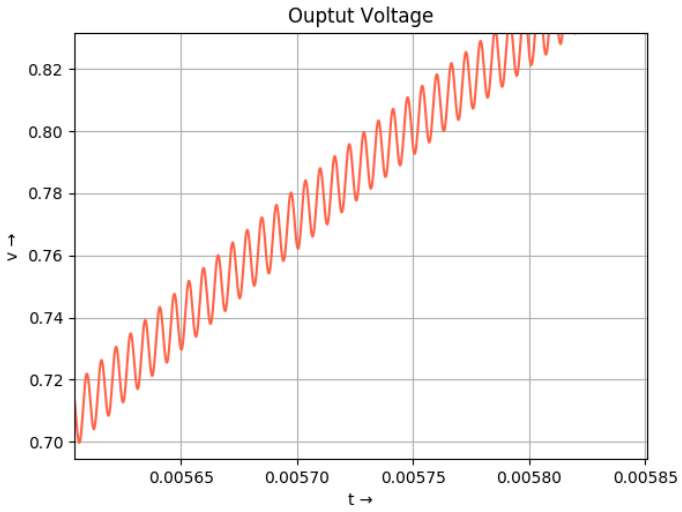
$$v_i(t) = cos(10^3 t)u(t) - cos(10^6 t)u(t)$$

The output can be found using **sp.lsim** command. The range of t should be high enough to observe the ripples in resultant sine wave. Due to the presence of a resistor in the circuit, the initial conditions are zero.

```
t=linspace(0,10**-2,10**5)
u=cos((10**3)*t)-cos((10**6)*t)
t,y,svec=sp.lsim(H,u,t)
```



On zooming in we find the presence of small sine shaped ripples within the larger sine wave. These ripples are due to $\omega = 10^6$ frequency. The larger sine wave is due to the other frequency.



7 Observation

In this assignment we can take away the fact that working on LTI system is very easy in python due to the signal toolbox module. Complex circuits can be modelled and outputs can be found out using python. Python is very useful to engineers.