Indian Institute of Technology, Madras
Department of Electrical Engineering
Applied Programming Lab

Lab Report
Assignment 6

The Laplace Transform

Nithin Uppalapati
EE18B035

05-03-2020

Contents

1. Abstract

Aim :
The aim of this assignment is to analyse "Linear Time-invariant Systems" with numerical tools in Python.
All the problems will be in "continuous time" and will use Laplace Transforms.
There are 6 sub-parts in this assignment which requires to do various operations with the signals toolbox in python.
There are the sub parts in the assignment:

1. Finding output of a given system in time domain, with two different damping coefficients.
2. Plotting the output for various frequencies of inputs
3. Solving for a coupled spring problem, with given initial conditions
4. Finding the transfer function of given two port network and then finding a output for a given input.

2. Introduction

Assignment 5 is based on:
Analysing Laplace Transforms, by using bode plots, and also finding the system response in time domain.

Implementation :

- The bode plots can be plotted by using the $.bode()$ function in the signals toolbox
- The system response of a given transfer function can be found by the $.impulse()$ function.
- And, any LTI system of the form $\frac{N(s)}{D(s)}$ can be defined by the $.lti()$ function in the signals toolbox.

3. Theory

The Unilateral Laplace Transform :

The unilateral Laplace transform, is of considerable value in analyzing causal systems and, particularly, systems specified by linear constant-coefficient differential equations with nonzero initial conditions (i.e., systems that are not initially at rest).
Some of the Unilateral Laplace Transform pairs are given below:

Property	Signal	Unilateral Laplace Transform
	$x(t)$ $x_1(t)$ $x_2(t)$	$\mathfrak{X}(s)$ $\mathfrak{X}_1(s)$ $\mathfrak{X}_2(s)$
Linearity	$ax_1(t) + bx_2(t)$	$a\mathfrak{X}_1(s) + b\mathfrak{X}_2(s)$
Shifting in the s-domain	$e^{s_0 t}x(t)$	$\mathfrak{X}(s - s_0)$
Time scaling	$x(at), \quad a > 0$	$\frac{1}{a}\mathfrak{X}\left(\frac{s}{a}\right)$
Conjugation	$x * (t)$	$x * (s)$
Convolution (assuming that $x_1(t)$ and $x_2(t)$ are identically zero for $t < 0$)	$x_1(t) * x_2(t)$	$\mathfrak{X}_1(s)\mathfrak{X}_2(s)$
Differentiation in the time domain	$\frac{d}{dt}x(t)$	$s\mathfrak{X}(s) - x(0^-)$
Differentiation in the s-domain	$-tx(t)$	$\frac{d}{ds}\mathfrak{X}(s)$
Integration in the time domain	$\int_{0^-}^{t} x(\tau)\,d\tau$	$\frac{1}{s}\mathfrak{X}(s)$

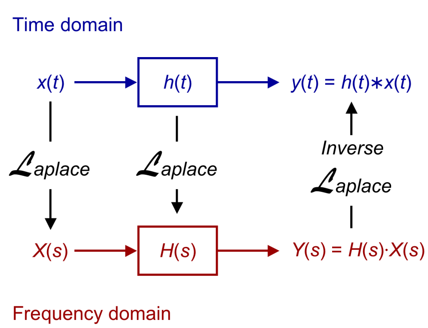
Initial- and Final-Value Theorems

If $x(t)$ contains no impulses or higher-order singularities at $t = 0$, then

$$x(0^+) = \lim_{s\to\infty} s\mathfrak{X}(s)$$

$$\lim_{t\to\infty} x(t) = \lim_{s\to 0} s\mathfrak{X}(s)$$

So, given a system with an input $x(t)$ and output $y(t)$, and if it is known that the system is **Linear and Time-Invariant**, then we can write the relation of $x(t)$ and $y(t)$ in the laplace domain as follows, and let the system response is $h(t)$,



We can also compute the initial state response, final state response and Natural response and Forced response, of we know the Transfer function of a system and the initial conditions of the system.

3.1 Solving the Problem 1:

The Laplace Transform of the function, $f(t) = cos(1.5t)e^{-0.5t}u_0(t)$is to be found and then we have to solve for the system for which $f(t)$ is input and the system is described as,

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

With initial conditions as, $x(0) = 0$ and $\dot{x}(0) = 0$, for t going from zero to 50 seconds.
We can solve this by using system.impulse, as it computes the impulse response of a given transfer function. Thus we input the output's transfer function in-order to get its impulse response, which is the solution required in t-domain.
This can be implemented as follows:

```
################1 ~~~~~~~~~~~~
a=0.5 ## a is the decay
freq=1.5
s=trans(a,freq)
H_1=sp.lti(s[0],s[1])
t_1,x_1=sp.impulse(H_1,None,t)
```

Where $trans()$, is a defined function, by me, as follows:

```
def trans(a,freq):
num=array([1,a])
den=polymul(num,num)
den=polyadd(den,array([freq**2]))

charac=array([1,0,2.25])
den_final=polymul(den,charac)

s=[num,den_final,den,charac]
return s
```

So, in the function $trans()$, I defined the transfer functions, which are required for all the problems.

3.2 Solving the Problem 2:

This problem is nothing but repeating the first, with a smaller decay.
The Laplace Transform of the function, $f(t) = cos(1.5t)e^{-0.05t}u_0(t)$is to be found and then we

have to solve for the system for which $f(t)$ is input and the system is described as,

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

With initial conditions as, $x(0) = 0$ and $\dot{x}(0) = 0$, for t going from zero to 50 seconds.
We can solve this by using system.impulse, as it computes the impulse response of a given transfer function. Thus we input the output's transfer function in-order to get its impulse response, which is the solution required in t-domain.
This can be implemented as follows:

```
################2 ~~~~~~~~~~~~
a=0.05 ## a is the decay
freq=1.5
s=trans(a,freq)
H_2=sp.lti(s[0],s[1])
t_2,x_2=sp.impulse(H_2,None,t)
```

3.3 Solving the Problem 3:

In this problem, we solve the same system as in the problems 1 and 2, but what we do differently than the previous problems is that we vary the frequency of the inputs.
I implemented this by the following code:

```
freq=arange(1.4,1.6,0.05)
freq=list(freq)
a=0.05
col=["y","b","g","orange","m"]
H_3=sp.lti(array([1]),array([1,0,2.25]))
figure(3)
for i in freq:
t_3,x_3,svec=sp.lsim(H_3,input(i,t),t)
plot(t,x_3,label="\u03C9 = "+str(i),color=col[freq.index(i)])#
```

3.4 Solving the Problem 4:

In this we, have to a coupled spring problem, whose system is defined as follows:

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

where the initial condition is $x(0) = 1, \dot{x}(0) = y(0) = \dot{y}(0) = 0$. The above equations translaet to:

$$s^2X(s) - s + X(s) - Y(s) = 0s^2Y(s) + 2(Y(s) - X(s)) = 0$$

We need to solve for its time evolution, and from it obtain x(t) and y(t) for $0t20$
So, initially I'm defining the transfer functions for $x(t)$ and $y(t)$, by asserting the proper boundary conditions, and then solving for the individual impulse responses by using the function $.impulse()$.
So, after solving the transfer functions are as follows:

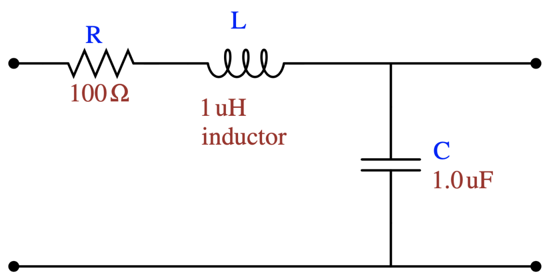
$$X(s) = \frac{s^3 + 2s}{(s^2+2)(s^2+2) - 2} Y(s) = \frac{2(s^3+2s)}{((s^2+2)(s^2+2)-2)(s^2+2)}$$

And then solving for the impulse responses.

```
################4 ~~~~~~~~~~~~~
t=linspace(0,20,20001)
num_x=array([1,0,2,0])
den_x=polyadd( polymul(array([1,0,2]),array([1,0,1])) , array([-2]))
X=sp.lti(num_x,den_x)
num_y=polymul(num_x,array([2]))
den_y=polymul(den_x,array([1,0,2]))
Y=sp.lti(num_y,den_y)
t_x,x=sp.impulse(X,None,t)
t_y,y=sp.impulse(Y,None,t)
```

3.5 Solving the Problem 5:

In this problem, we have to obtain the magnitude and phase response of the Steady State Transfer function of the following two-port network:



So, the transfer function for the above circuit is as follows:

$$\frac{V_{out}}{V_{in}} = \frac{\frac{1}{sC}}{R + sL + \frac{1}{sC}}$$

By substituting the values of R, L, C and the solving it by the $.impulse()$ function, we get the t-domain for the above circuit.
And this is implemented in python as follows:

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

num=array([1])
den=array([L*C,R*C,1])
H_5=sp.lti(num,den)
w_5,S_5,phi_5=sp.bode(H_5) # w=r
```

3.6 Solving the Problem 6:

This problem is nothing but repeating the fifth, with a defined input:

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

and the transfer function is mentioned above (in problem 5).
Let the impulse response of the system - $H(s)$ is $h(t)$, and for finding the output in t-domain, we need to convolute the functions $h(t)$ and $v_i(t)$

$$v_o(t) = h(t) * v_i(t)$$

Thus this can be implemented by the pre-defined function in signal toolbox, $.lsim()$

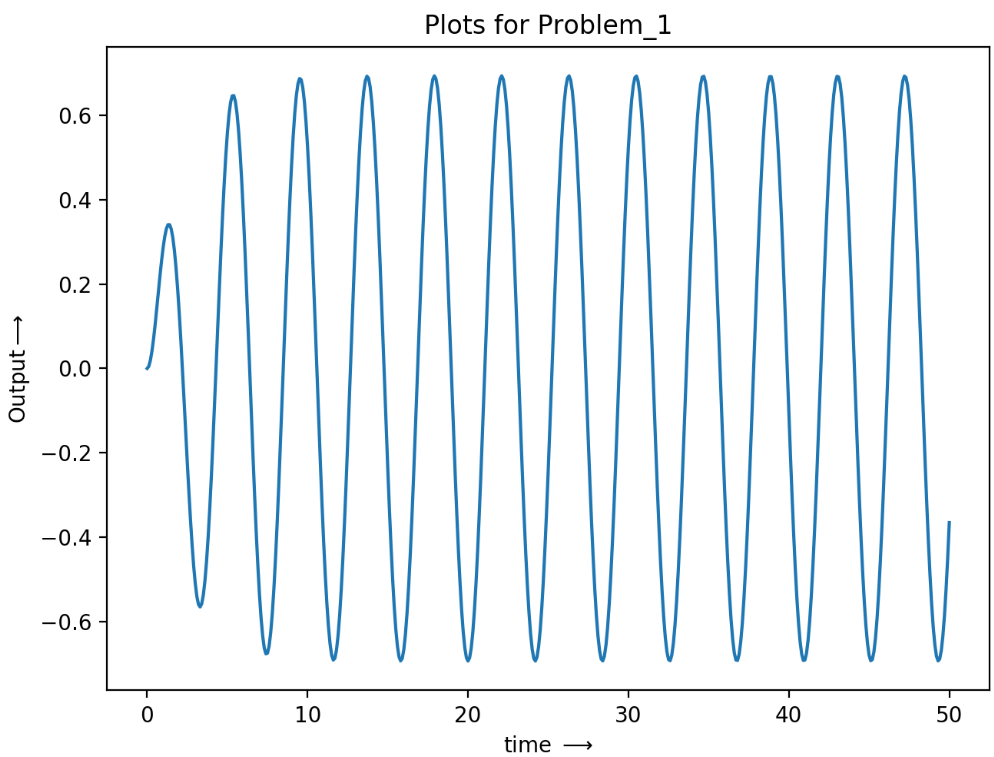
```
t=linspace(0,10e-3,20001)
u=cos((1e3)*t)-cos((1e6)*t)
t,y,svec=sp.lsim(H_5,u,t)
t,y,svec=sp.lsim(H_5,u,t)
```

3.7 Plots of Functions and Coefficients:

```
t=linspace(0,50,500)
```

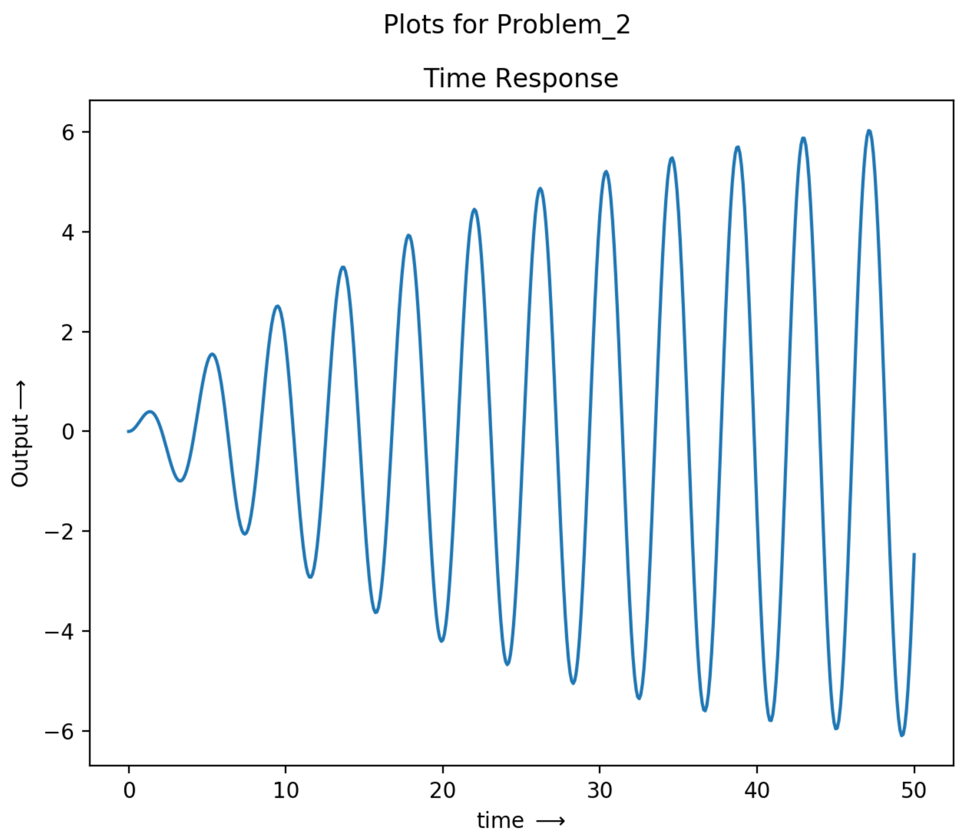
3.7.1 Plots for Problem 1:

```
subplots(3,1)
subplot(3,1,1).semilogx(w_1,S_1,color='orange')
title("Magnitude Plot")
subplot(3,1,2).semilogx(w_1,phi_1,color='green')
title("Phase Plot")
t_1,x_1=sp.impulse(H_1,None,t)
subplot(3,1,3).plot(t,x_1)
title("Time Response")
suptitle("Plots for Problem_1")
```



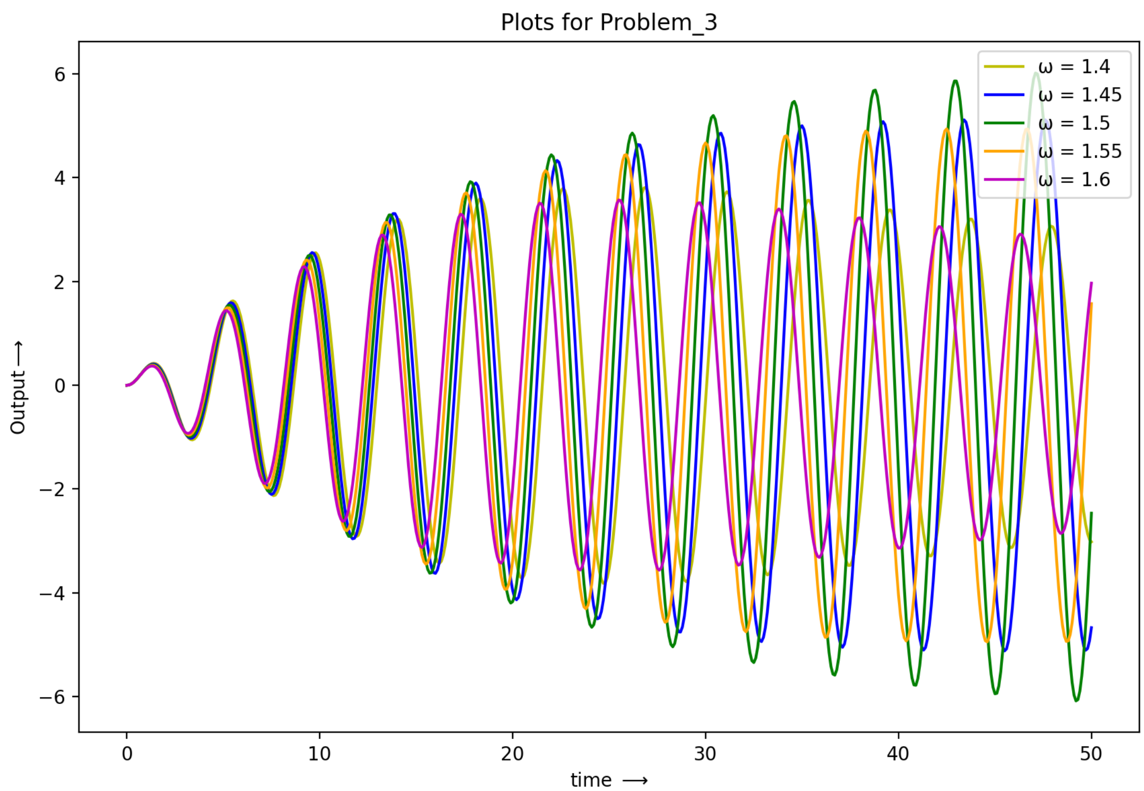
3.7.2 Plots for Problem 2:

```
subplots(3,1)
subplot(3,1,1).semilogx(w_2,S_2,color='orange')
title("Magnitude Plot")
subplot(3,1,2).semilogx(w_2,phi_2,color='g')
title("Phase Plot")
t_2,x_2=sp.impulse(H_2,None,t)
subplot(3,1,3).plot(t,x_2)
title("Time Response")
suptitle("Plots for Problem_2")
```



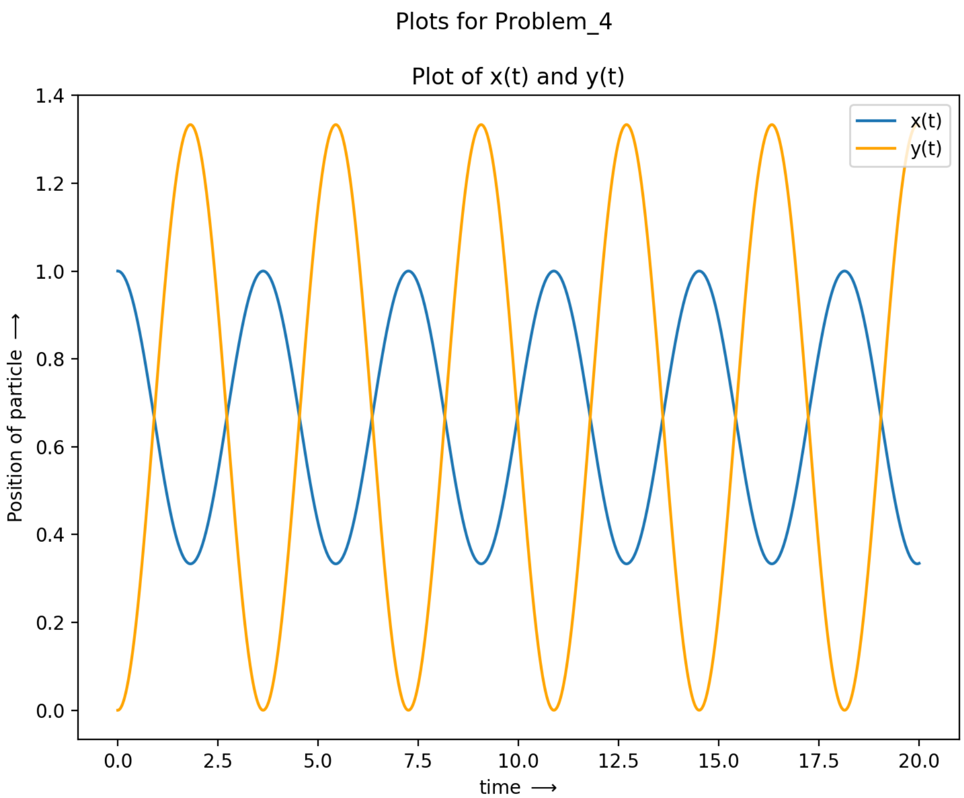
3.7.3 Plots for Problem 3:

```
plot(t,x_3,label="\u03C9 = "+str(i),color=col[freq.index(i)])#
title("Plots for Problem_3")
legend(loc="upper right")
xlabel(r'time ' '$\longrightarrow$')
ylabel(r'Output' '$\longrightarrow$')
```



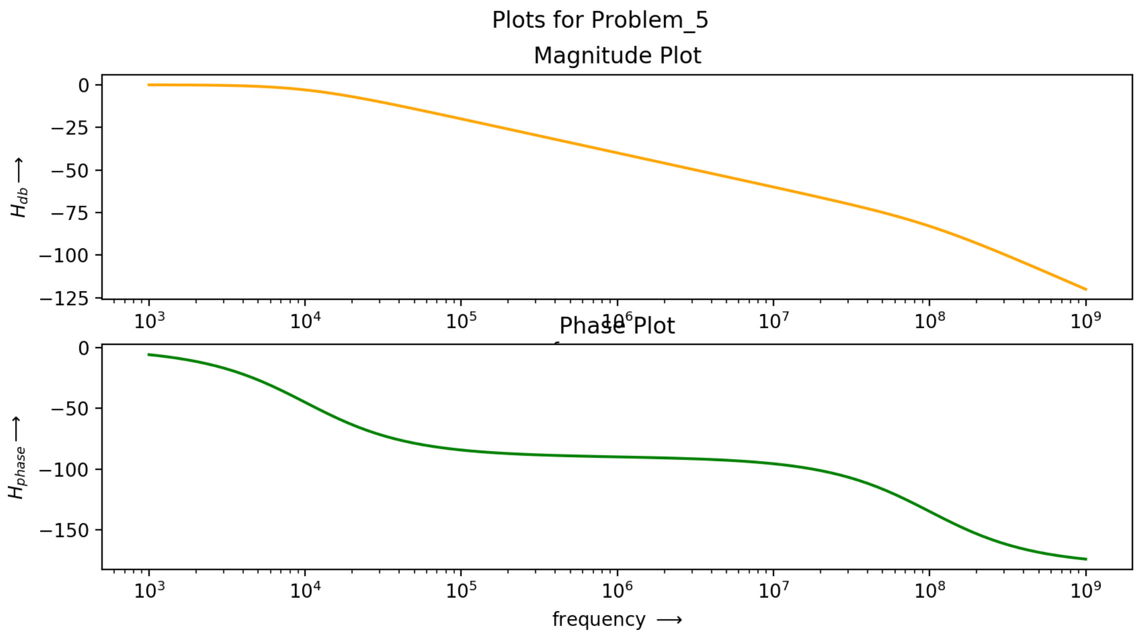
3.7.4 Plots for Problem 4:

```
figure(4)
plot(t,x,label="x(t)")
plot(t,y,label="y(t)",color='orange')
title('Plot of x(t) and y(t)')
legend(loc="upper right")
suptitle("Plots for Problem_4")
```



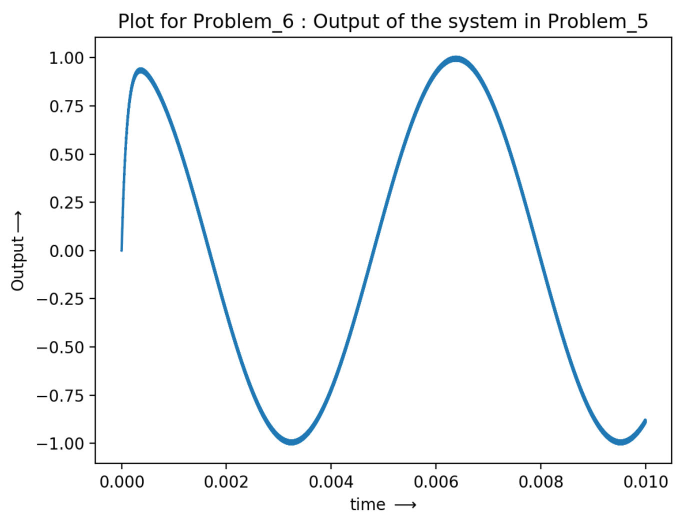
3.7.5 Plots for Problem 5:

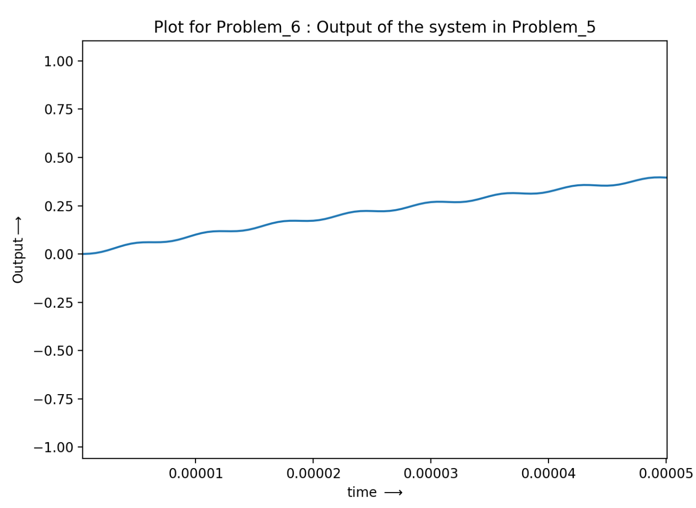
```
subplots(2,1)
subplot(2,1,1).semilogx(w_5,S_5,color='orange')
title("Magnitude Plot")
subplot(2,1,2).semilogx(w_5,phi_5,color='g')
title("Phase Plot")
suptitle("Plots for Problem_5")
```



3.7.6 Plots for Problem 6:

```
figure(6)
t=linspace(0,10e-3,20001)
u=cos((1e3)*t)-cos((1e6)*t)
t,y,svec=sp.lsim(H_5,u,t)
title("Plot for Problem_6 : Output of the system in Problem_5")
plot(t,y)
```





4. Conclusion and Reasoning:

4.1 In Problem 3, when varying the Frequency:

As the characteristic response of the system is,

$$H(s) = \frac{1}{s^2 + (1.5)^2}$$

So, it has a natural frequency of $\omega = 1.5 rad sec^{-1}$, thus -the poles are on the imaginary axes, thus there is no damping in the system.
So, when we provide a signal of frequency $1.5 rad sec^{-1}$, it is going to resonate with the system, hence the maximum amplitude at that frequency.

4.2 In Problem 6, in the first 30μ seconds

As, initially the transient responses do not die, and also the natural frequency of the system is higher when compared to that of the input's frequency.
So, as the natural response do not die for the first few moments, thus the natural response's is going to dominate.
As time progresses, the natural response gets attenuated enough, so that the forced response dominates over the natural response.
The small ripples in the output are due to the attenuated natural response of the system.