EE2703 Applied Programming Lab - Assignment No 7

Name: Abishek S
Roll Number: EE18B001

March 17, 2020

1 Abstract

The goal of this assignment is the following :

- To understand and use symbolic python for analysing Laplace equations and solving them.
- To realise low pass and high pass filters using opamp circuits.
- To plot graphs and observe the output of low pass and high pass filters to different inputs.

2 Assignment

2.1 Setting up the Program

Importing the standard libraries

```
import pylab as pl
import numpy as np
import sys
import scipy.signal as sp
import matplotlib
from sympy import *
```

We initialize sympy and define utility functions for general plotting and making Bode plots..

```
init_printing()
#Complex variable of Laplace equations
s = symbols('s')

#Defining a general utility function PLOT for plotting
```

```
def PLOT(x,y,fig_no = 0,label_x = r'$\rightarrow$',
    label_y = r'$\rightarrow$',fn = pl.plot,arg3 = 'b-',
    title = "Plot",grids = True,cmap = matplotlib.cm.jet,label = ''):
    '''Plotting function to make standard plots'''
    pl.figure(fig_no)
    pl.grid(grids)
    if fn == pl.contourf:
        fn(x,y,arg3,cmap = cmap)
        pl.colorbar()
    else:
        if label == '':
            fn(x,y,arg3)
        else:
            fn(x,y,arg3,label = label)
            pl.legend()
    pl.xlabel(label_x,size = 17)
    pl.ylabel(label_y,size = 17)
    pl.title(title)

#Defining a general utility function for making BODE Plots
def bodeplot(H,fig_no = 0,title = ''):
    '''Makes Bode Plots'''
    pl.figure(fig_no)
    pl.suptitle(title)
    w,s,phi = H.bode()
    pl.subplot(2,1,1)
    pl.semilogx(w,s)
    pl.xlabel(r'$\omega$',size=17)
    pl.ylabel(r'$|H(j\omega)|-(in dB)$',size =17)
    pl.subplot(2,1,2)
    pl.semilogx(w,phi)
    pl.xlabel(r'$\omega$',size=17)
    pl.ylabel(r'$\angle(H(j\omega))$',size =17)
```

2.2 Low Pass Filter

We analyse the given circuit for Butterworth low pass filter and obtain the equations in a matrix form for solving.

$$\begin{pmatrix} 0 & 0 & 1 & -\frac{1}{G} \\ -\frac{1}{1+sR2C2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ -\frac{1}{R_1}-\frac{1}{R_2}-sC_1 & \frac{1}{R2} & 0 & sC_1 \end{pmatrix} \begin{pmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -V_i(s)/R_1 \end{pmatrix}$$

We obtain the output voltage in symolic form by solving the above matrix equation.

```
#Solving the matrix equation and getting the output Voltage for Low Pass Filter
def LowPass(R1,R2,C1,C2,G,Vi=1):
    '''Active 2nd order low pass butterworth filter using opamp'''
    A = Matrix([[0,0,1,-1/G],[-1/(1+s*R2*C2),1,0,0],
            [0,-G,G,1],[-1/R1-1/R2-s*C1,1/R2,0,s*C1]])
    b = Matrix([0,0,0,-Vi/R1])
    V = A.inv()*b
    return(A,b,V)
```

We also define two functions to obtain the transfer function in sp.lti form from the symbolic form obtained by solving the matrix equations.

```
#Extract the Transfer function and convert it to scipy lti form
def Simplify_H(V):
    '''Extracts Transfer function from the matrix inversion result'''
    Vo = V[3]  #The third element in the V column is the output voltage
    Vo = expand(simplify(Vo))  #converting to rational form
    H = SympyToScipy(Vo)
    return H

#Convert the Transfer function form sympy to scipy lti form
def SympyToScipy(Vo):
    '''Converts Transfer function in sympy to scipy'''
    v1 = fraction(Vo) #converting to numerator and denominator form
    n,d = Poly(v1[0],s),poly(v1[1],s)
    numer,denom = n.all_coeffs(), d.all_coeffs()  #extract the coefficients of 's'
    numer,denom = [float(f) for f in numer], [float(f) for f in denom]
    H = sp.lti(numer,denom)  #converting to scipy lti form
    return H
```

We get the Transfer function and make the bode plot for Low Pass Filter.

```
#Values of R1,R2,C1,C2,G chosen from given circuit
A,b,V = LowPass(10000,10000,1e-9,1e-9,1.586,1)
H_lp = Simplify_H(V)

bodeplot(H_lp,1,"Bode plot of Low Pass Filter")
pl.show()
```

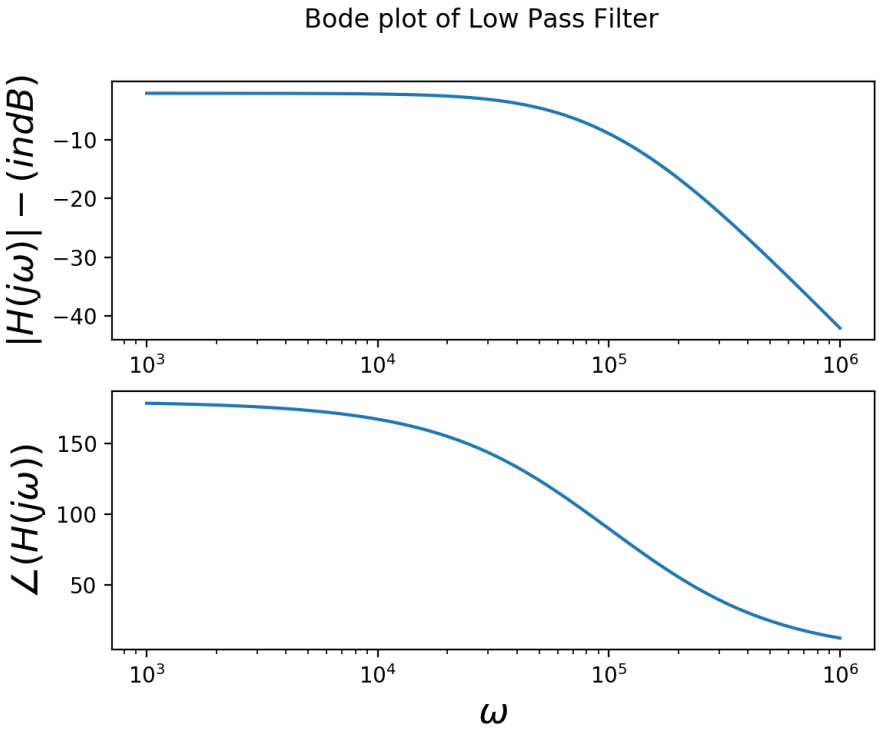

Figure 1: Bode Plot for Low Pass Filter

2.3 High Pass Filter

We analyse the given circuit for high pass filter and obtain the equations in a matrix form for solving.

$$\begin{pmatrix} 0 & 0 & 1 & -\frac{1}{G} \\ -\frac{-sR_3C_2}{1+sR_3C_2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ -1-(sR_1C_1)-(sR_3C_2)) & sC_2R_1 & 0 & 1 \end{pmatrix} \begin{pmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -V_i(s)sR_1C_1 \end{pmatrix}$$

We obtain the output voltage in symolic by solving the above matrix equation.

```
#Solving the matrix equation and getting the output Voltage for High Pass Filter
def HighPass(R1,R2,C1,C2,G,Vi = 1):
    '''Active 2nd order high pass filter using opamp'''
    A = Matrix([[0,0,1,-1/G],[-1/(1+1/(s*R2*C2)),1,0,0],
            [0,-G,G,1],[-s*C1-s*C2-1/R1,s*C2,0,1/R1]])
    b = Matrix([0,0,0,-Vi*s*C1])
    V = A.inv()*b
    return(A,b,V)
```

We get the Transfer function and make the bode plot for High Pass Filter.

```
#Values of R1,R2,C1,C2,G chosen from given circuit
A,b,V = HighPass(1e4,1e4,1e-9,1e-9,1.586,1)
H_hp = Simplify_H(V)

bodeplot(H_hp,2,"Bode plot of High Pass Filter")
pl.show()
```

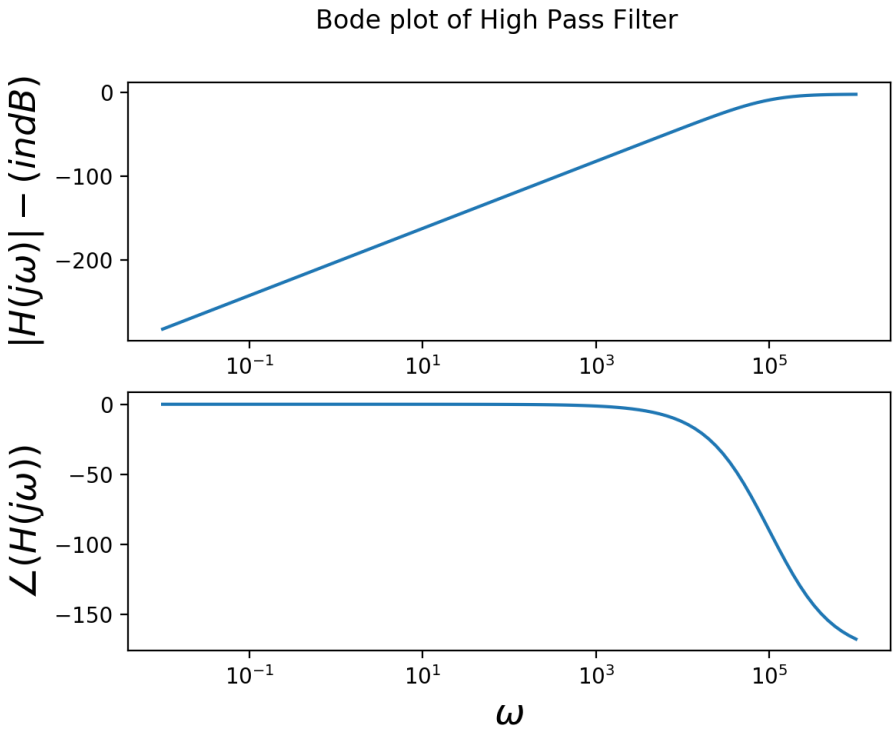

Figure 2: Bode Plot for High Pass Filter

We observe that the Transfer function of the **low pass filter drops off towards higher frequencies** and that of the **high pass filter increases towards higher frequencies**, hence their names.

3 Assignment Questions

3.1 Question 1

We plot the **step response** of Low Pass Filter, (i.e) output of low pass filter for input : $v_i(t) = u(t)$ which implies $V_i(S) = 1/S$.

```
t = np.arange(0,1e-2,1e-7)  #Time scale

A,b,V = LowPass(10000,10000,1e-9,1e-9,1.586,1/s)
_,vtd = sp.impulse(Simplify_H(V),None,t)
PLOT(t,vtd,3,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Step response of Low Pass Filter')
pl.show()
```

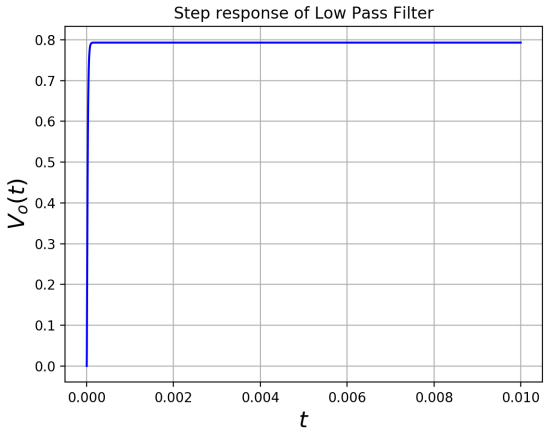

Figure 3: Response of Low pass Filter to unit step function

We observe that the step response of the low pass filter **starts from zero initially** when the input suddenly changes at t = 0 as it doesn't allow the high frequency.
Then it settles to a final value of 0.79 as it **allows DC input intact** due to it's low frequency.
No overshoot in low pass filter case as it is a Butterworth filter.

3.2 Question 2

We plot the input voltage which is **sum of two sinusoids** of different frequencies. We also plot the **response of Low pass Filter** to the input. The input voltage is given as :

$$v_i(t) = (\sin(2000\pi t) + cos(2 \times 10^6 \pi t))u(t)$$

```
t = np.arange(0,5e-3,1e-7)  #Time scale

inp = np.sin(2000*np.pi*t)+np.cos(2*(10**6)*np.pi*t)
PLOT(t,inp,5,r'$t$',r'$V_i(t)$',title = 'Sum of 2 sinusoids Input')
pl.show()
```

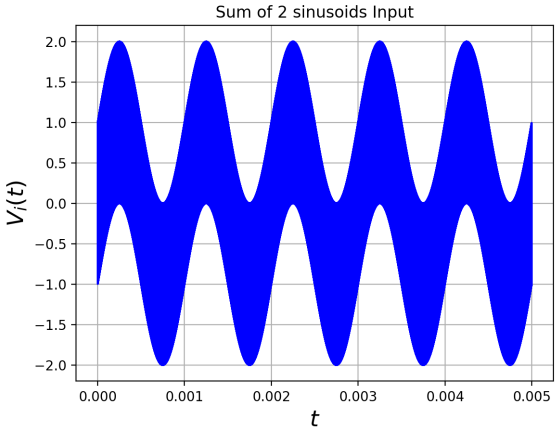

Figure 4: Input voltage comprising of sum of two sinusoids

```
t,vtd,svec = sp.lsim(H_lp,inp,t)
PLOT(t,vtd,6,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of low pass filter to sum of 2 sinusoids')
pl.show()
```

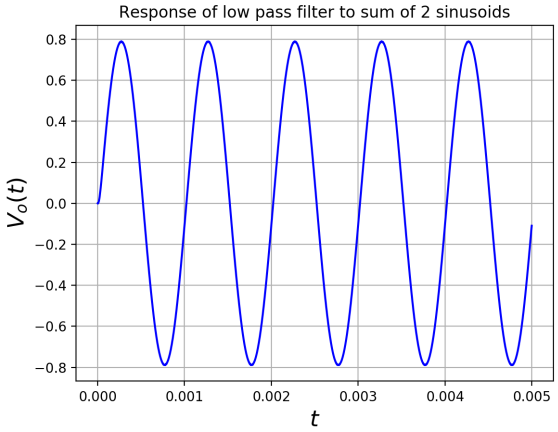

Figure 5: Response of Low pass Filter to sum of two sinusoids

As expected low pass filter **allows the low frequency sinusoid** to pass intact, and **blocks the high frequency sinusoid** almost completely.

3.3 Question 3

Refer this figure for the Bode Plot of High Pass Filter. We plot the response of the high pass filter to a **sum of two input sinusoids**.

```
t = np.arange(0,1e-5,1e-7)  #Time scale

inp = np.sin(2000*np.pi*t)+np.cos(2*(10**6)*np.pi*t)
t,vtd,svec = sp.lsim(H_hp,inp,t)
PLOT(t,vtd,7,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of high pass filter to sum of 2 sinusoids')
pl.show()
```

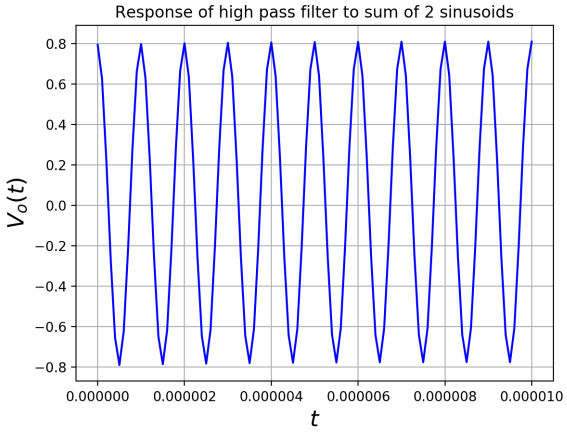

Figure 6: Response of High pass Filter to sum of two sinusoids

High pass filter **allows the high frequency sinusoid** to pass intact, and **blocks the low frequency sinusoid** almost completely.
We have used different time scales on both plots to capture the responses better.

3.4 Question 4

We **input damped sinusoidal inputs** of low and high frequencies to both the circuits and observe how they respond. We use different time scales for both frequencies and different damping to observe the variations better.

3.4.1 High Frequency Damped sinusoid

The first input voltage is given as :

$$v_i(t) = (\cos(10^6 t) \times e^{-3000t})u(t)$$

```
#Response of Low Pass and High Pass filter to high frequency damped sinusoid
damping_factor = 3000

t = np.arange(0,5e-4,1e-7)  #Time scale
```

```
inp = np.exp(-damping_factor*t)*(np.cos((10**6)*np.pi*t))
PLOT(t,inp,8,r'$t$',r'$V_i(t)$',title = 'High Frequency damped sinusoid input')
pl.show()

t,vtd,svec = sp.lsim(H_lp,inp,t)
PLOT(t,vtd,9,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of low pass filter to high frequency damped sinusoid')
pl.show()

t,vtd,svec = sp.lsim(H_hp,inp,t)
PLOT(t,vtd,10,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of high pass filter to high frequency damped sinusoid')
pl.show()
```

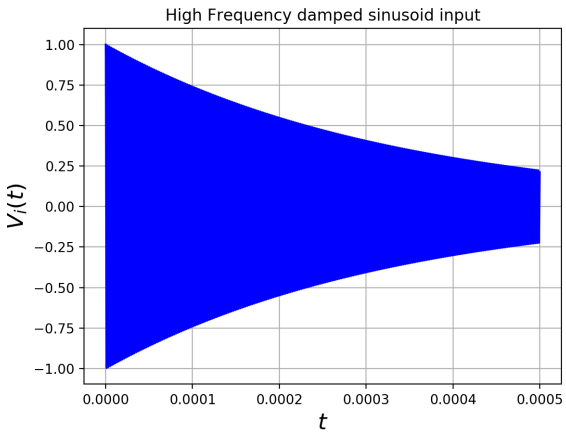

Figure 7: Input damped sinusoid voltage of high frequency

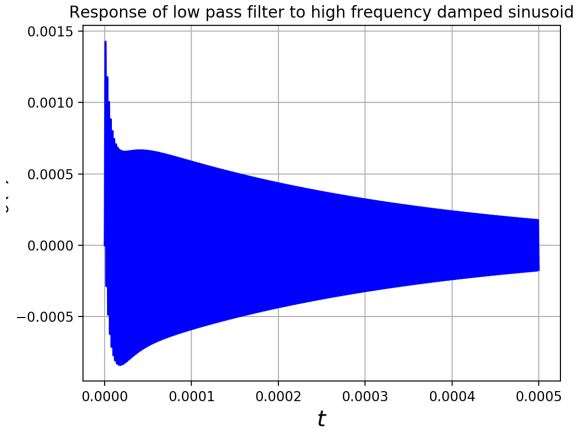

Figure 8: Response of Low pass Filter to high frequency damped sinusoid

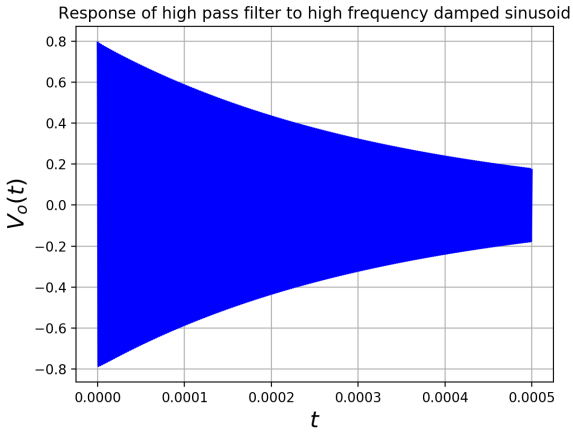

Figure 9: Response of High pass Filter to high frequency damped sinusoid

3.4.2 Low Frequency Damped sinusoid

The second input voltage is given as :

$$v_i(t) = (\sin(10^3 t) \times e^{-100t})u(t)$$

```
#Response of Low Pass and High Pass filter to low damped sinusoid
damping_factor = 100

t = np.arange(0,5e-2,1e-7)   #Time scale

inp = np.exp(-damping_factor*t)*(np.sin(1000*np.pi*t))
```

```
PLOT(t,inp,11,r'$t$',r'$V_i(t)$',title = 'Low Frequency damped sinusoid input')
pl.show()

t,vtd,svec = sp.lsim(H_lp,inp,t)
PLOT(t,vtd,12,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of low pass filter to low frequency damped sinusoid')
pl.show()

t,vtd,svec = sp.lsim(H_hp,inp,t)
PLOT(t,vtd,13,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Response of high pass filter to low frequency damped sinusoid')
pl.show()
```

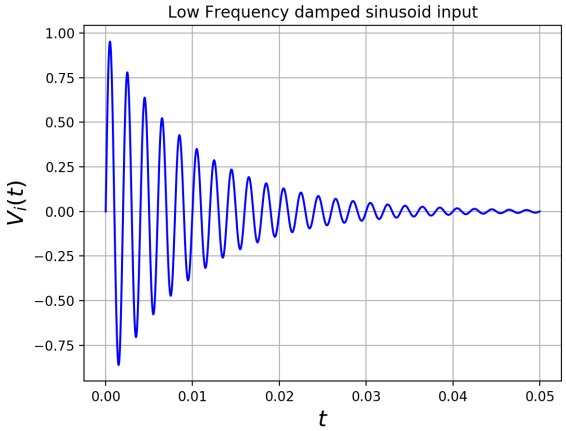

Figure 10: Input damped sinusoid voltage of low frequency

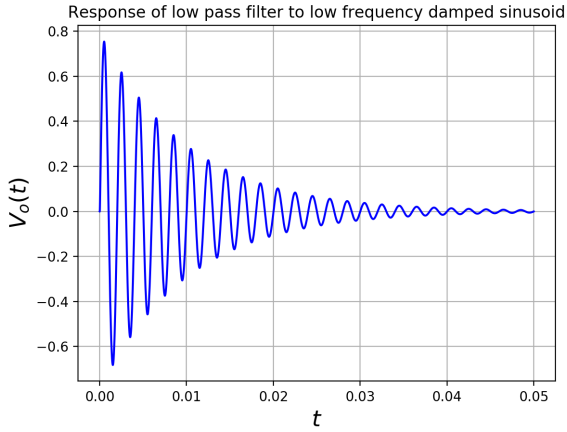

Figure 11: Response of Low pass Filter to low frequency damped sinusoid

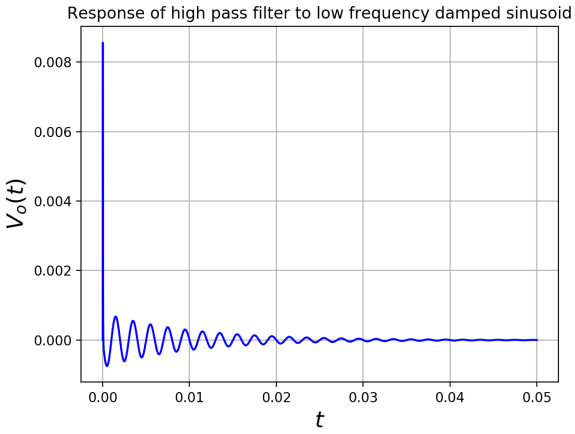

Figure 12: Response of High pass Filter to low frequency damped sinusoid

We could observe how the **low pass filter nearly blocked the high frequency damped sinusoid** input (notice the low amplitudes and faster decay), and **almost perfectly allowed low frequency damped sinusoid** to pass through.
The opposite is observed for the high pass filter, where **in case of low frequency damped sinusoid there is initially a spike** due to sudden input given at t = 0 (hence high frequency), then the input decays very fast and hence is said to be **nearly blocked**. Whereas the **high frequency damped sinusoid is almost perfectly allowed**.

3.5 Question 5

We plot the **step response of High Pass Filter**, (i.e) output of high pass filter for input : $v_i(t) = u(t)$ which implies $V_i(S) = 1/S$.

```
t = np.arange(0,1e-2,1e-7)   #Time scale

A,b,V = HighPass(1e4,1e4,1e-9,1e-9,1.586,1/s)
_,vtd = sp.impulse(Simplify_H(V),None,t)
PLOT(t,vtd,4,r"$t$",r'$V_o(t)$',pl.plot,
    title = 'Step response of High Pass Filter')
pl.show()
```

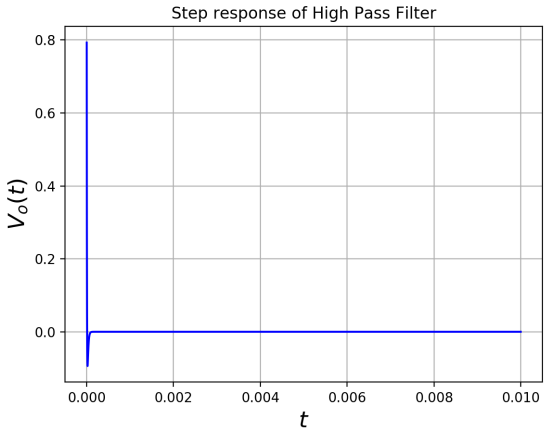

Figure 13: Response of High pass Filter to unit step function

The high pass filter **initially has a momentary high value** of 0.79 at t = 0 because of sudden input given (high frequency) which is allowed by it. It then becomes zero, showing it **doesn't allow low frequency DC** input to pass through.
Overshoot is observed in case of high pass filter.

4 Conclusions

- We solved Laplace equations in 's' using sympy in Python.
- We analysed and solved opamp circuits using sympy, and realised low pass and high pass filters using the same.
- We made Bode plots, and plotted step responses and sinusoidal responses to understand the behaviour of low pass and high pass filters.