EE2703: Assignment 8

Yogesh Agarwala
EE19B130

April 25, 2021

[1]:
```
import numpy as np
from sympy import *
import scipy.signal as sp
import pylab
```

[2]:
```
"""
Function to plot graphs
"""
def display_plot(i,x,y,title,xlabel='t',ylabel='x'):
    pylab.figure(i)
    pylab.plot(x,y,'-r',label=r'$V_{o}$')
    pylab.title(title)
    pylab.xlabel(xlabel,fontsize=15)
    pylab.ylabel(ylabel,fontsize=15)
    pylab.legend(loc ='upper right')
    pylab.grid(True)
    pylab.show()
```

0.1 1. Lowpass Filter

[3]:
```
'''
Lowpass filter
'''
def lowpass(R1,R2,C1,C2,G,Vi):
    s=symbols('s')
    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)
```

[4]:
```
"""
Convert Sympy transfer function polynomial to Scipy LTI
"""
def sympyToLTI(xpr, s=symbols('s')):
    num, den = simplify(xpr).as_numer_denom()  # returns the expressions
```

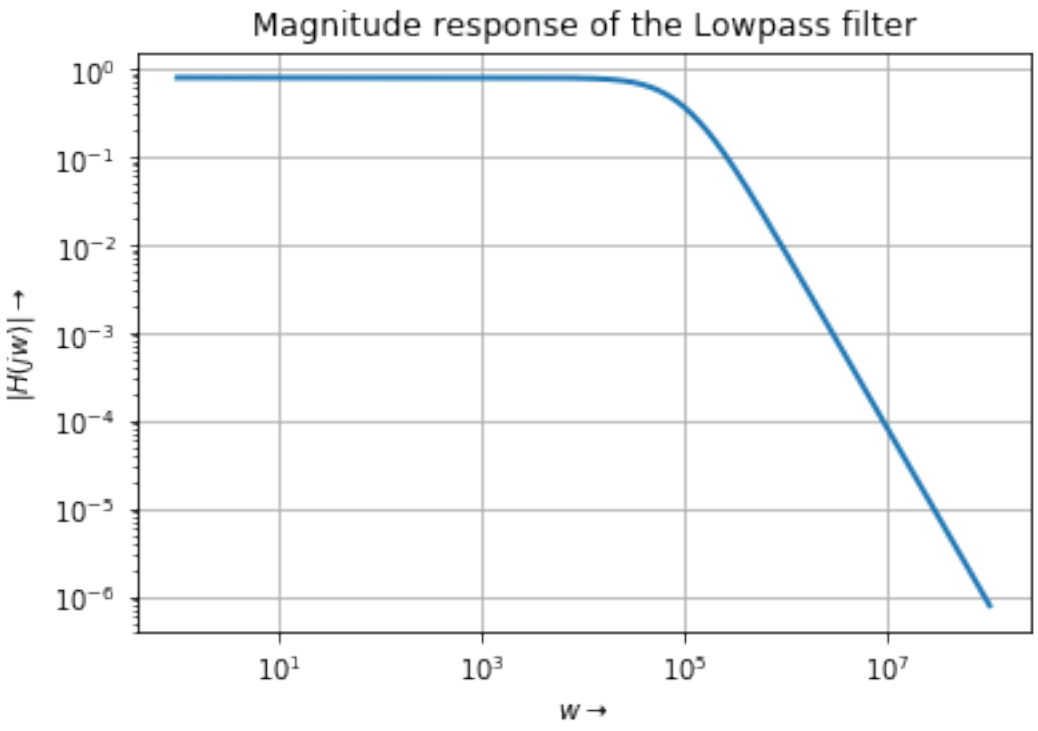
```
    p_num_den = poly(num, s), poly(den, s)
    c_num_den = [expand(p).all_coeffs() for p in p_num_den]  # returns the␣
 ↪coefficients
    l_num, l_den = [lambdify((), c)() for c in c_num_den]  # convert to floats
    return sp.lti(l_num, l_den)
```

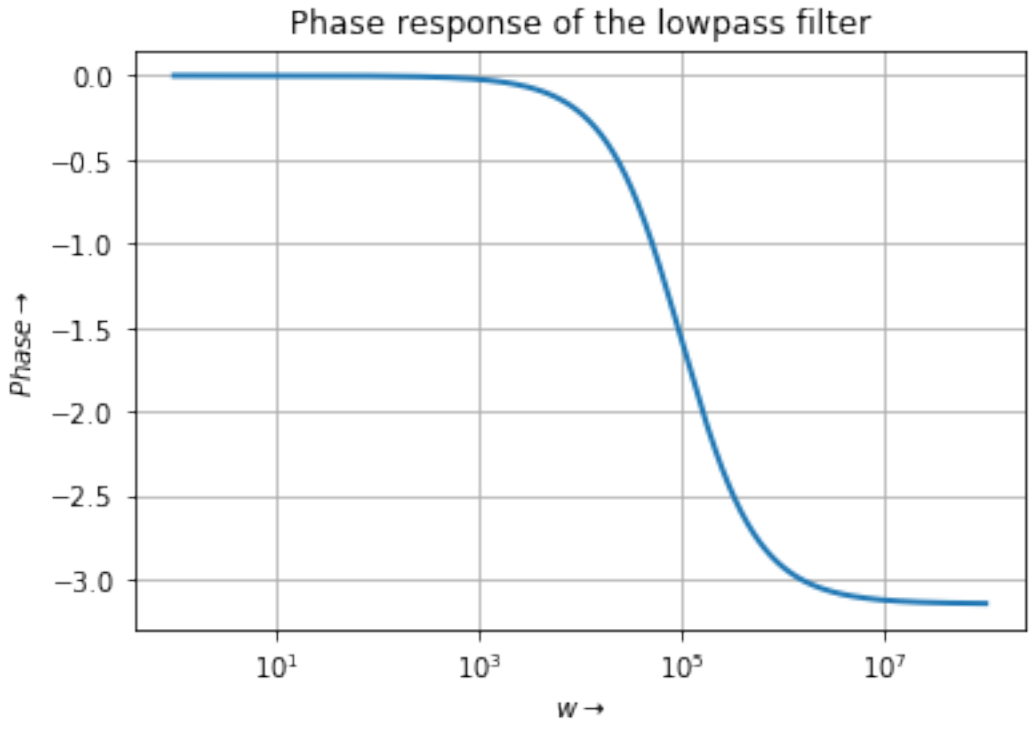
[5]:
```
s = symbols('s')
A,b,V=lowpass(10000,10000,1e-9,1e-9,1.586,1)
Vo = V[3]
H = sympyToLTI(Vo)
```

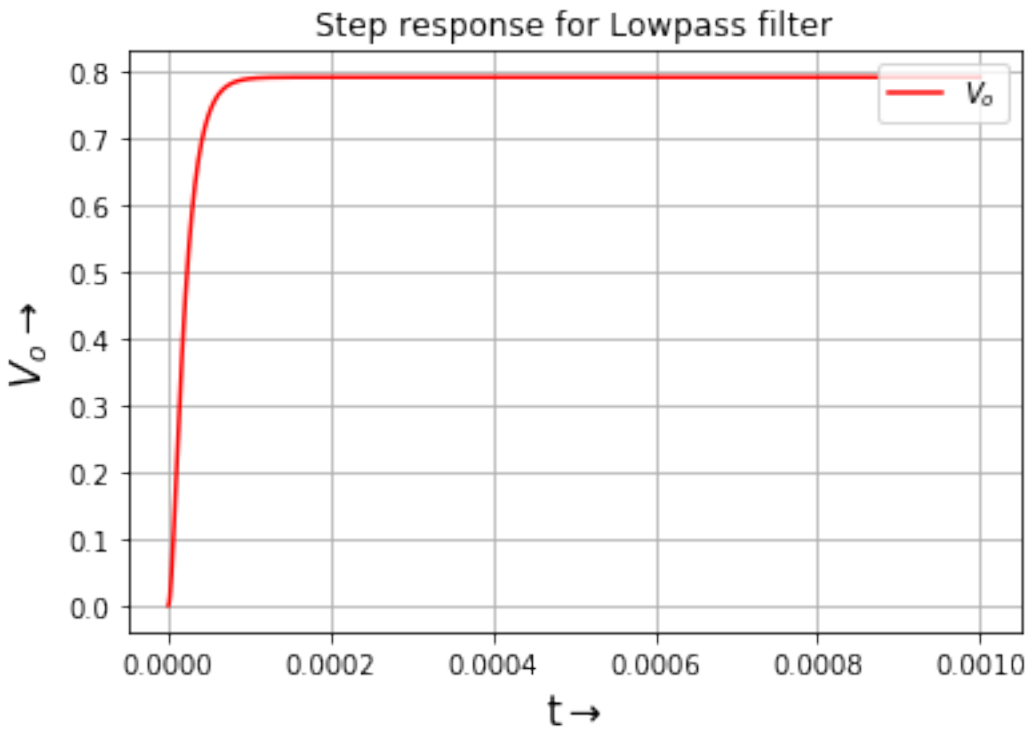
[6]:
```
"""
Magnitude response
"""
w=pylab.logspace(0,8,801)
ss=1j*w
hf=lambdify(s,Vo,"numpy")
v=hf(ss)
pylab.loglog(w,abs(v),lw=2)
pylab.title('Magnitude response of the Lowpass filter')
pylab.xlabel(r'$w\rightarrow$')
pylab.ylabel(r'$|H(jw)|\rightarrow$')
pylab.grid(True)
pylab.show()

"""
Phase response
"""
pylab.semilogx(w,np.angle(v),lw=2)
pylab.title('Phase response of the lowpass filter')
pylab.xlabel(r'$w\rightarrow$')
pylab.ylabel(r'$Phase\rightarrow$')
pylab.grid(True)
pylab.show()

"""
Step response
"""
t = np.linspace(0,0.001,1000)
Vo = sp.step(H,T=t)
display_plot(0,Vo[0],Vo[1],'Step response for Lowpass␣
 ↪filter',r't$\rightarrow$',r'$V_{o}\rightarrow$')
```

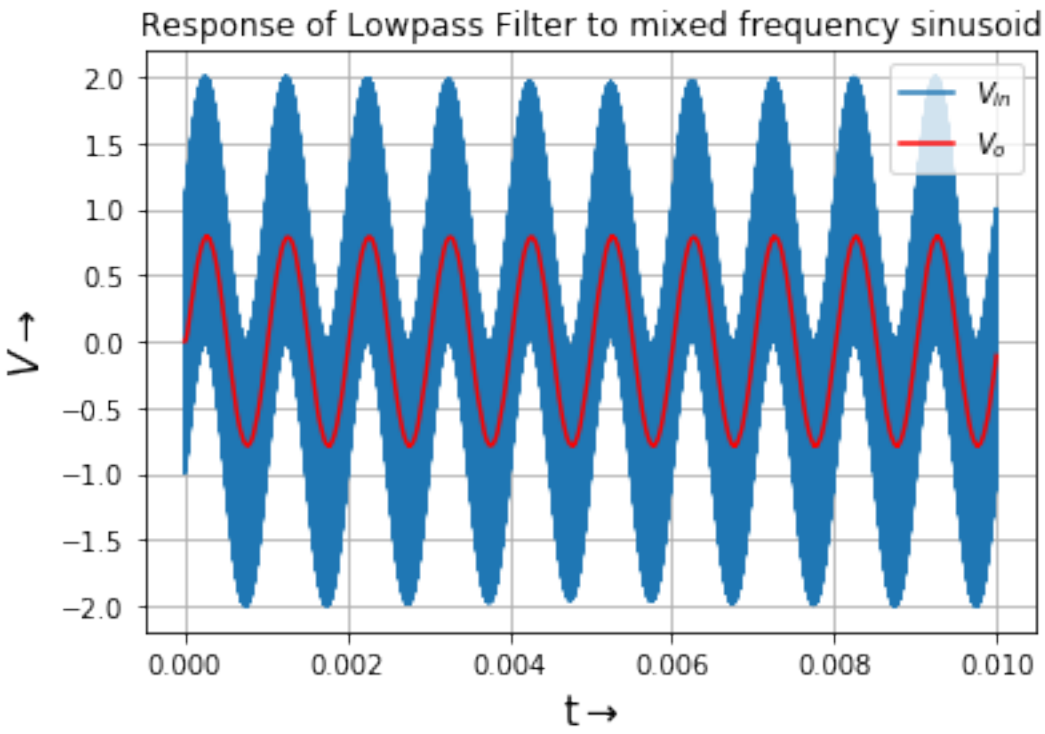

Magnitude response of the Lowpass filter
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
$|H(jw)| \rightarrow$
10^{1}
10^{3}
10^{5}
10^{7}
$w \rightarrow$


Phase response of the lowpass filter
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
Phase →
10^{1}
10^{3}
10^{5}
10^{7}
$w \rightarrow$



0.2 2. Response of Lowpass Filter to mixed frequency sinusoid

[7]:
```
t = np.linspace(0,0.01,100000)
Vi = np.multiply((np.sin(2000*np.pi*t)+np.cos(2000000*np.pi*t)),np.heaviside(t,0.
 ↪5))
Vo = sp.lsim(H,Vi,T=t)
pylab.figure(1)
pylab.plot(Vo[0],Vi,label=r'$V_{in}$')
display_plot(1,Vo[0],Vo[1],'Response of Lowpass Filter to mixed frequency␣
 ↪sinusoid',r't$\rightarrow$',r'$V\rightarrow$')
```



0.3 3. Highpass Filter

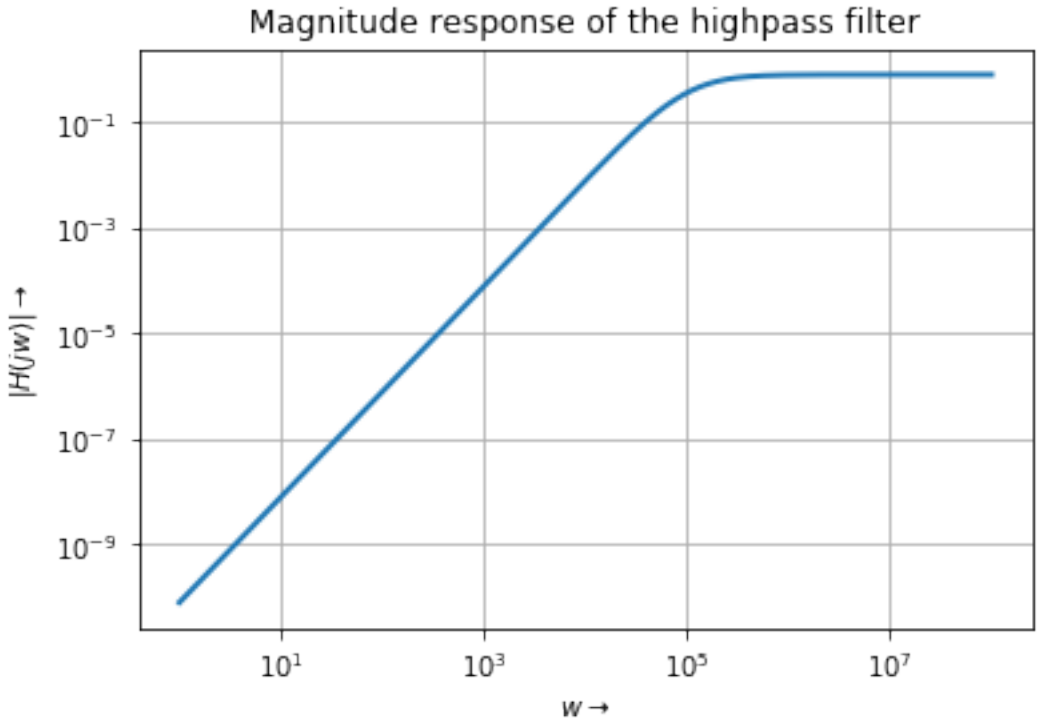
[8]:
```
'''
High pass filter
'''
def highpass(R1,R3,C1,C2,G,Vi):
    s=symbols('s')
    A=Matrix([[0,0,1,-1/G],[-1/(1+1/(s*R3*C2)),1,0,0],[0,-G,G,1],[-s*C1-s*C2-1/
 ↪R1,s*C2,0,1/R1]])
    b=Matrix([0,0,0,-Vi*s*C1])
    V = A.inv()*b
    return (A,b,V)
```

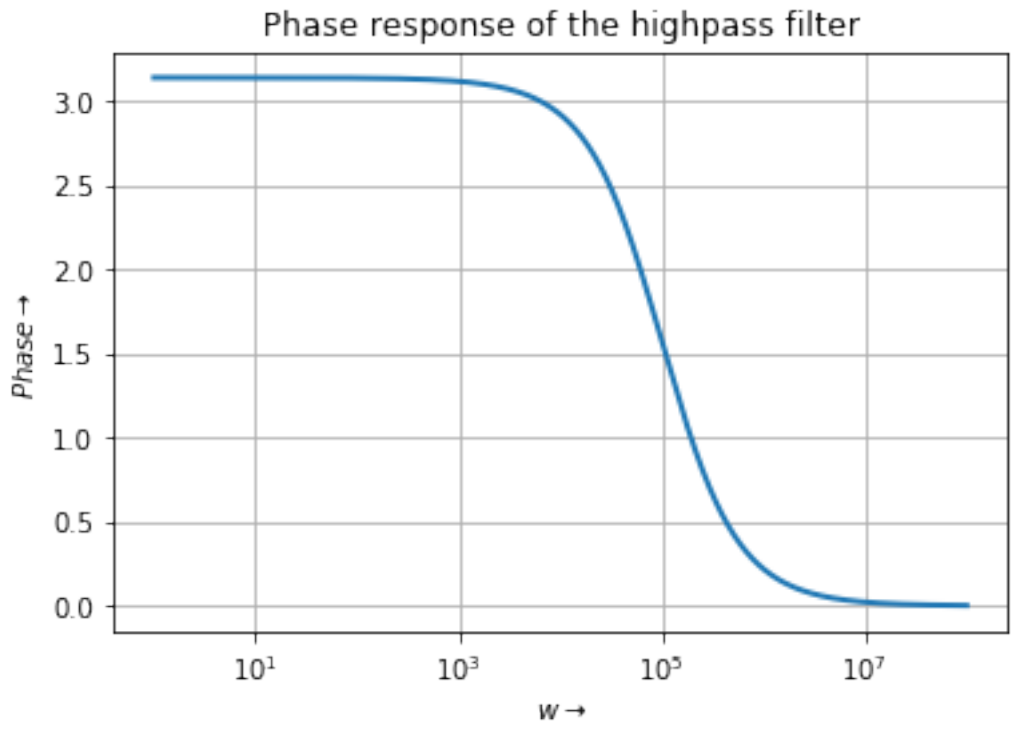
[9]:
```
A,b,V = highpass(10000,10000,1e-9,1e-9,1.586,1)
Vo = V[3]
H = sympyToLTI(Vo)
```

[10]:
```
"""
Magnitude response
"""
w=pylab.logspace(0,8,801)
ss=1j*w
```

```
hf=lambdify(s,Vo,'numpy')
v=hf(ss)
pylab.loglog(w,abs(v),lw=2)
pylab.title('Magnitude response of the highpass filter')
pylab.xlabel(r'$w\rightarrow$')
pylab.ylabel(r'$|H(jw)|\rightarrow$')
pylab.grid(True)
pylab.show()

"""
Phase response
"""
pylab.semilogx(w,np.angle(v),lw=2)
pylab.title('Phase response of the highpass filter')
pylab.xlabel(r'$w\rightarrow$')
pylab.ylabel(r'$Phase\rightarrow$')
pylab.grid(True)
pylab.show()
```





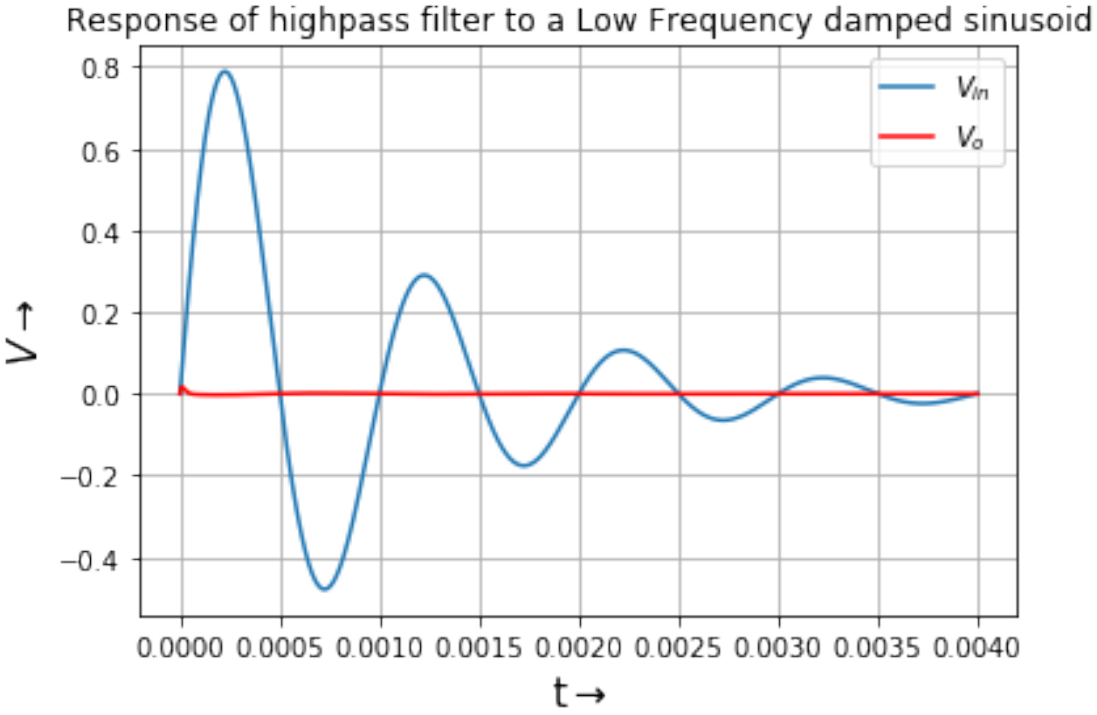
0.4 4. Response of Highpass filter to a damped sinusoid

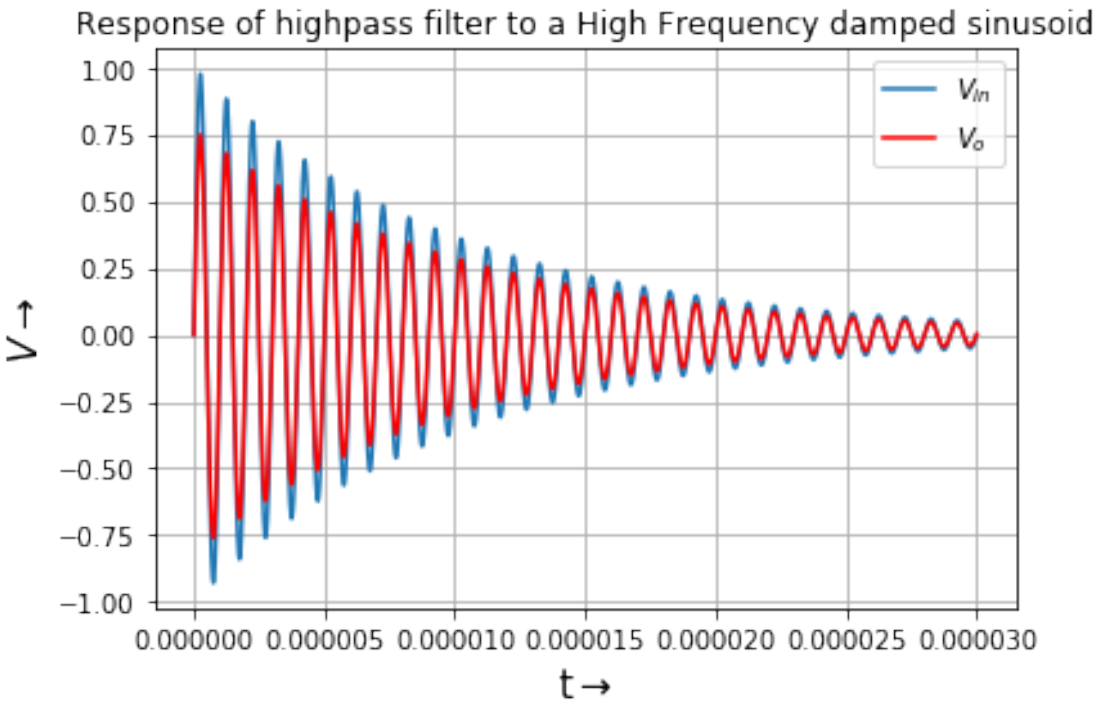
[11]:
```
import math

"""
Low Frequency damped sinusoid
"""
decay=1e1;freq=1e3
t = np.linspace(0.0,4e-3,100001)
Vi = (np.sin(2000*math.pi*t))*np.exp((-10**3)*t)
Vo = sp.lsim(H,Vi,T=t)
pylab.figure(2)
pylab.plot(Vo[0],Vi,label=r'$V_{in}$')
display_plot(2,Vo[0],Vo[1],'Response of highpass filter to a Low Frequency␣
 ↪damped sinusoid',r't$\rightarrow$',r'$V\rightarrow$')

"""
High Frequency damped sinusoid
"""
t = np.linspace(0.0,3e-5,100001)
Vi = (np.sin(2*(10**6)*math.pi*t))*np.exp((-10**5)*t)
Vo = sp.lsim(H,Vi,T=t)
pylab.figure(2)
```

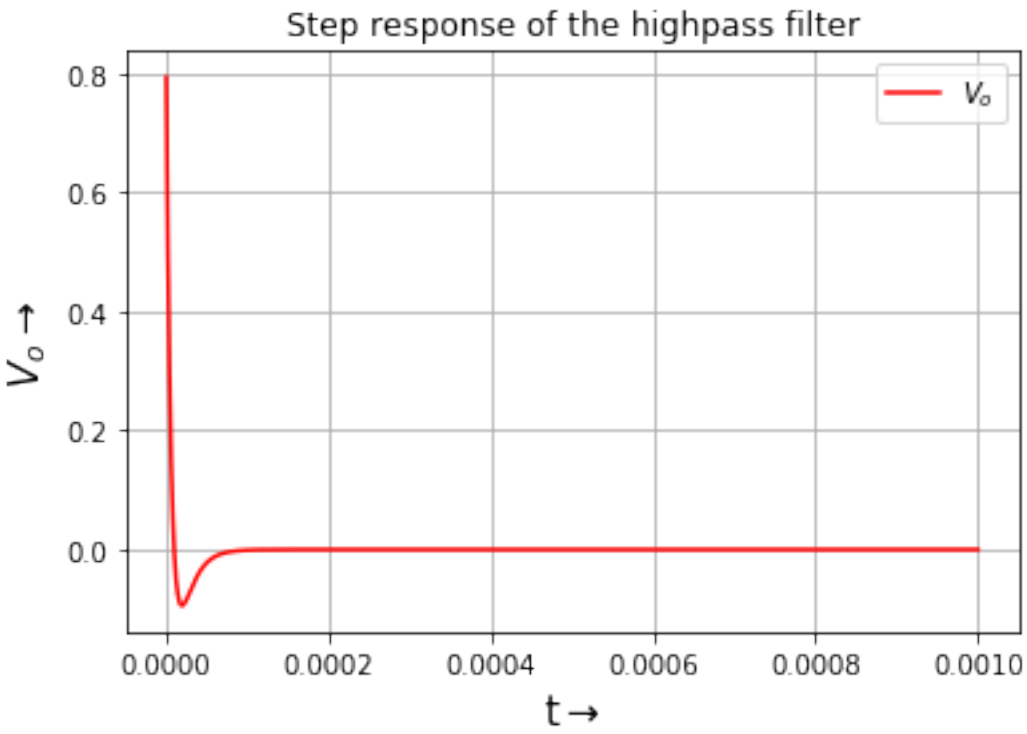
```
pylab.plot(Vo[0],Vi,label=r'$V_{in}$')
display_plot(2,Vo[0],Vo[1],'Response of highpass filter to a High Frequency␣
 ↪damped sinusoid',r't$\rightarrow$',r'$V\rightarrow$')
```





0.5 5. Response of Highpass filter to a unit step function

[12]:
```
"""
Step response
"""
t = np.linspace(0,0.001,1000)
Vo = sp.step(H,T=t)
display_plot(0,Vo[0],Vo[1],'Step response of the highpass␣
 ↪filter',r't$\rightarrow$',r'$V_{o}\rightarrow$')
```


Step response of the highpass filter
V_o
0.8
0.6
0.4
0.2
0.0
$V_o \rightarrow$
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
$t \rightarrow$