Symbolic Computing using SymPy

Assignment 7

EE2703 - Applied Programming Lab

Abhigyan Chattopadhyay
EE19B146

20th April 2021

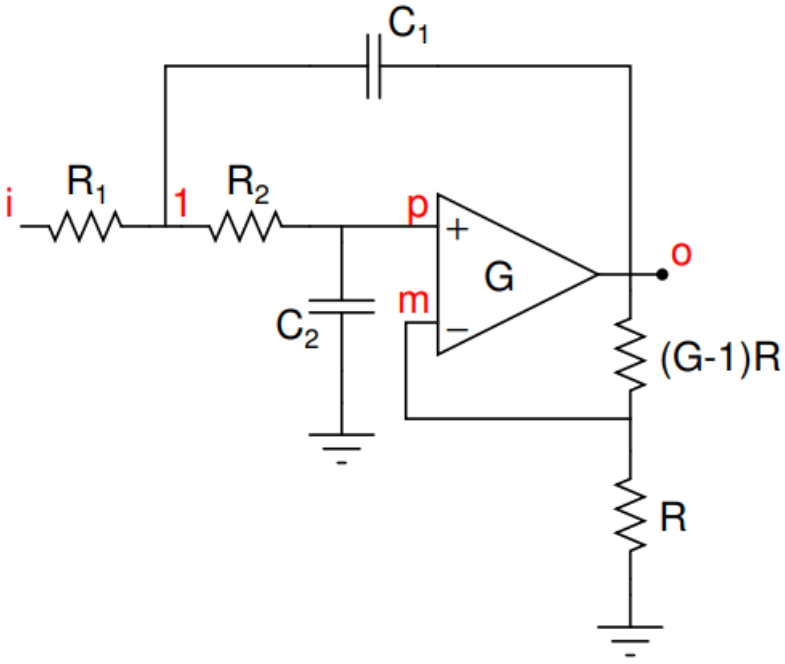
Contents

1 The Problem at Hand

We will be using Symbolic computing to make it easier to calculate transfer functions of Active filters and work with them much more easily.

2 Active Filters

Active filters make use of Operational Amplifiers to improve the filtering quality and allow for controllable bandwidth.



The above circuit behaves as an active lowpass filter. We will now analyze the circuit in the Laplace domain manually first:

$$V_m = \frac{V_o}{G} \tag{1}$$

$$V_1\left(\frac{1}{R_1} + \frac{1}{R_2} + sC_1\right) - \frac{V_i}{R_1} - sC_1V_o - \frac{V_p}{R_2} = 0 \tag{2}$$

$$V_p = \frac{V_1}{1 + sC_2R_2} \tag{3}$$

$$V_o = G(V_p - V_m) \tag{4}$$

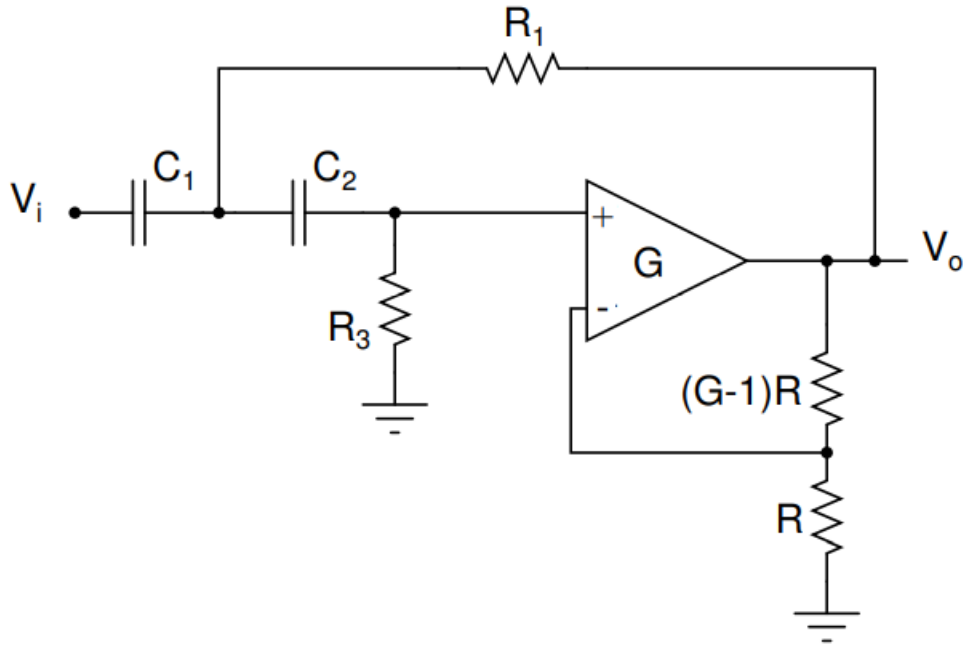
We can convert these linear equations into a Matrix equation to simplify the calculation:

$$\begin{bmatrix} 0 & 0 & 1 & -\frac{1}{G} \\ -\frac{1}{1+sC_2R_2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ \frac{1}{R_2} + \frac{1}{R_1} + sC_1 & -\frac{1}{R_2} & 0 & -sC_1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{V_i}{R_1} \end{bmatrix} \tag{5}$$

Or: $Av = b$

Now, we can solve the matrix equation for any given input V_i to obtain v.

Later on, we will be using a highpass filter which is essentially same as solving the lowpass filter, but we will have different equations for it.



Equations for the highpass filter are:

$$V_m = \frac{V_o}{G} \tag{6}$$

$$V_1\left(sC_2 + \frac{1}{R_1} + sC_1\right) - sC_1V_i - \frac{V_o}{R_1} - sC_2V_p = 0 \tag{7}$$

$$V_p = \frac{V_1}{1 + \frac{1}{sC_2R_3}} \tag{8}$$

$$V_o = G(V_p - V_m) \tag{9}$$

$$\begin{bmatrix} 0 & 0 & 1 & -\frac{1}{G} \\ -\frac{1}{1+\frac{1}{sC_2R_3}} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ \frac{1}{R_1} + sC_1 + sC_2 & -sC_2 & 0 & -\frac{1}{R_1} \end{bmatrix} \begin{bmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ sC_1V_i \end{bmatrix} \tag{10}$$

Again, this is also expressible in the form $Av = b$.

Now, we will set up these problems using SciPy and SymPy to work through them and get their responses to various inputs.

3 Solutions to the Given Problems

3.1 Pre-requisite functions

We will create a function to convert from the SymPy format to the SciPy TransferFunctionContinuous class so that we can plot the values easily.

Code:

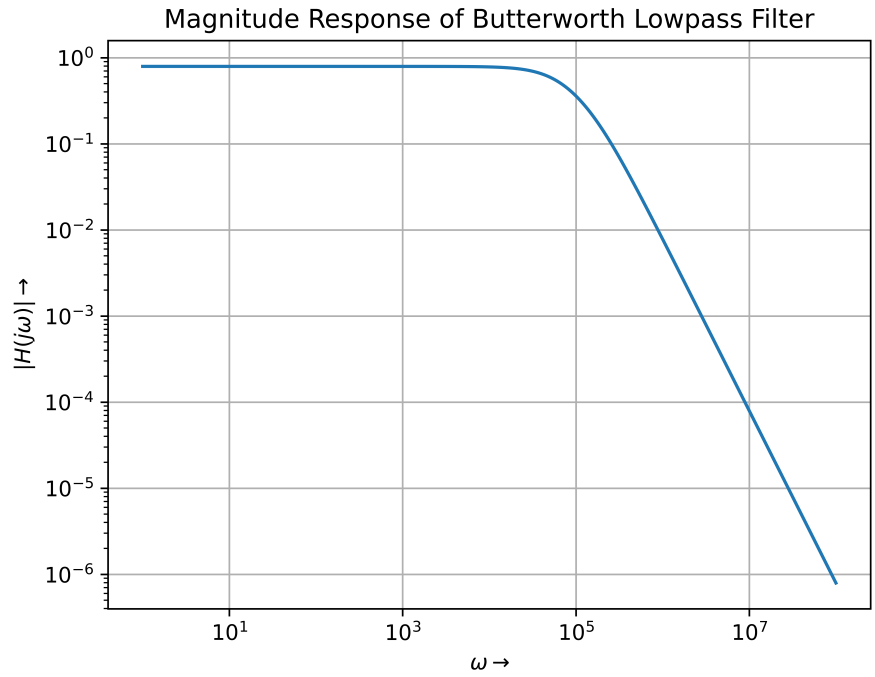
```
def sympy_to_lti(symp, s=symbols('s')):
    """Convert Sympy transfer function polynomial to Scipy
    ↪ TransferFunctionContinuous"""
    n, d = fraction(symp)  # expressions of numerator and denominator
    p_num_den = poly(n, s), poly(d, s)
    num, den = p_num_den[0].all_coeffs(), p_num_den[1].all_coeffs()
    return sp.lti(np.array(num,dtype=float), np.array(den,dtype=float))
```

Next we will create the Low Pass Butterworth Filter and plot its frequency response using the lambdify command:

```
# Setting up the Low Pass Butterworth Filter
s = symbols('s')
G = 1.586
R1 = R2 = 1e4
C1 = C2 = 1e-9
V1 = 1
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,-V1/R1])
V = A.inv()*b

# Plotting its Frequency Response
Vo = V[3]
H = sympy_to_lti(Vo)
omega = np.logspace(0,8,801)
ss = 1j*omega
hf = lambdify(s, Vo, 'numpy')
v = hf(ss)
plt.figure(0)
plt.loglog(omega,abs(v))
plt.xlabel(r"$\omega \rightarrow$")
plt.ylabel(r"$|H(j \omega)|\rightarrow$")
plt.grid(True)
plt.title("Magnitude Response of Butterworth Lowpass Filter")
plt.savefig("images/fig0",dpi=1000)
plt.show()
```

Graph obtained:



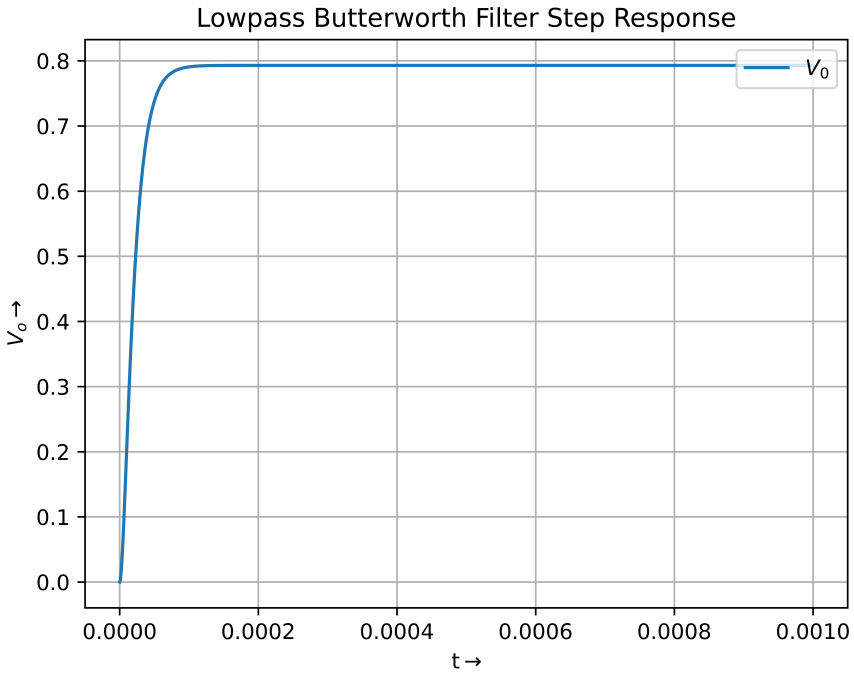
3.2 Question 1

Code:

```
t = np.linspace(0,0.001,1000)
t1,Vo1 = sp.step(H,T=t)

plt.figure(1)
plt.plot(t1,Vo1,"-",label=r"$V_0$")
plt.xlabel(r't$\rightarrow$')
plt.ylabel(r'$V_{o}\rightarrow$')
plt.title('Lowpass Butterworth Filter Step Response')
plt.legend(loc='upper right')
plt.grid(True)
plt.savefig("images/fig1",dpi=1000)
plt.show()
```

Graph obtained:

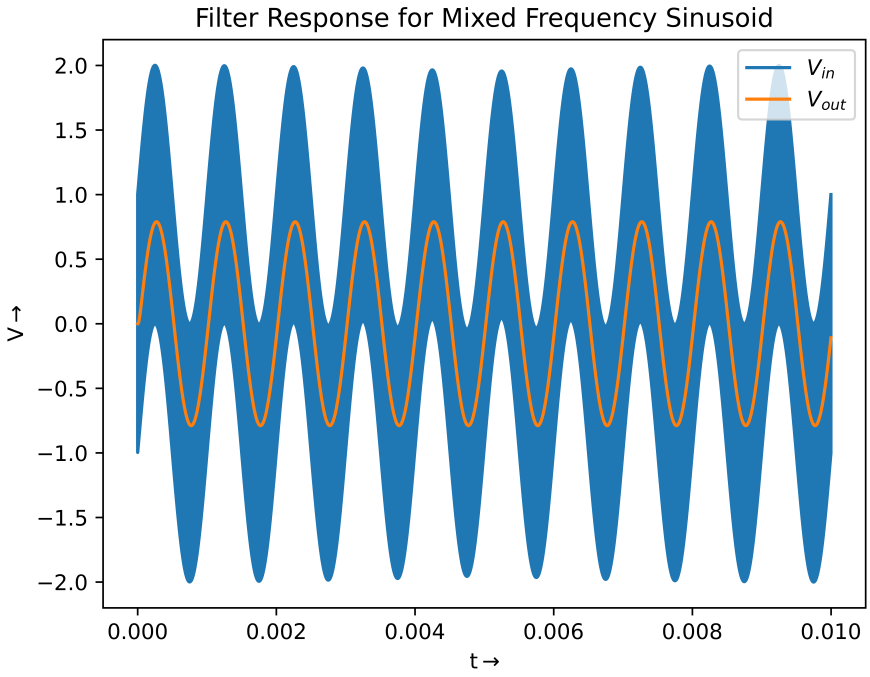


3.3 Question 2

Now we will plot the response of Lowpass Butterworth filter to a mixed frequency sinusoid:

```
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)
plt.figure(2)
plt.plot(Vo[0],Vi,label=r'$V_{in}$')
plt.plot(Vo[0],Vo[1],label=r'$V_{out}$')
plt.xlabel(r't$\rightarrow$')
plt.ylabel(r'V$\rightarrow$')
plt.title("Filter Response for Mixed Frequency Sinusoid")
plt.legend(loc="upper right")
plt.savefig("images/fig2", dpi=1000)
plt.show()
```

Graph obtained:



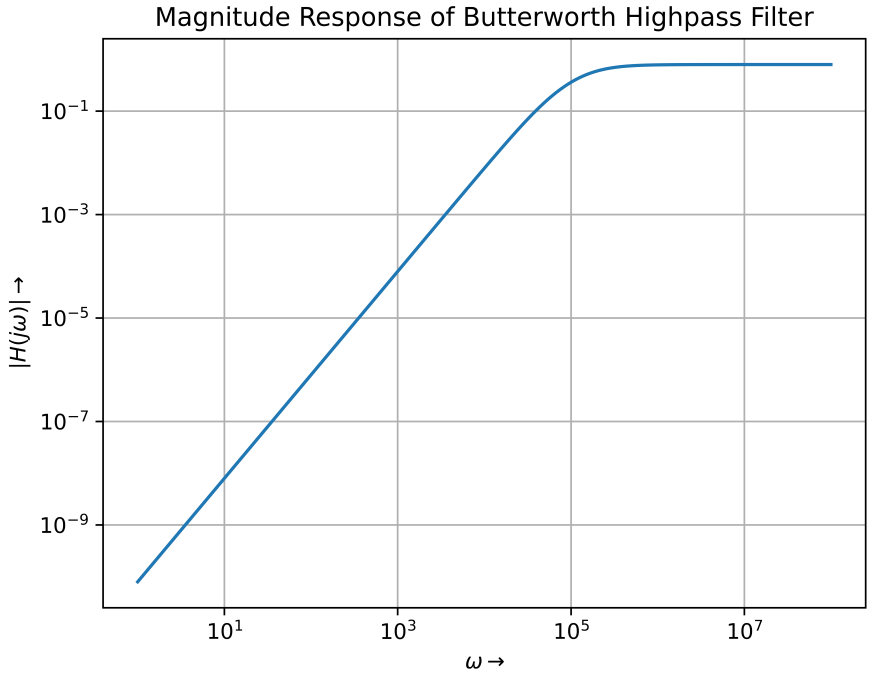
3.4 Question 3

Now we will set up the High Pass Butterworth Filter and plot its magnitude response:

```
G = 1.586
R1 = R3 = 1e4
C1 = C2 = 1e-9
Vi = 1
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

# Question 3: Magnitude Response of the Filter
Vo = V[3]
H = sympy_to_lti(Vo)
omega = np.logspace(0,8,801)
ss = 1j*omega
hf = lambdify(s, Vo, 'numpy')
v = hf(ss)
plt.figure(3)
plt.loglog(omega,abs(v))
plt.xlabel(r"$\omega \rightarrow$")
plt.ylabel(r"$|H(j \omega)|\rightarrow$")
plt.grid(True)
plt.title("Magnitude Response of Butterworth Highpass Filter")
plt.savefig("images/fig3",dpi=1000)
plt.show()
```

Graph obtained:



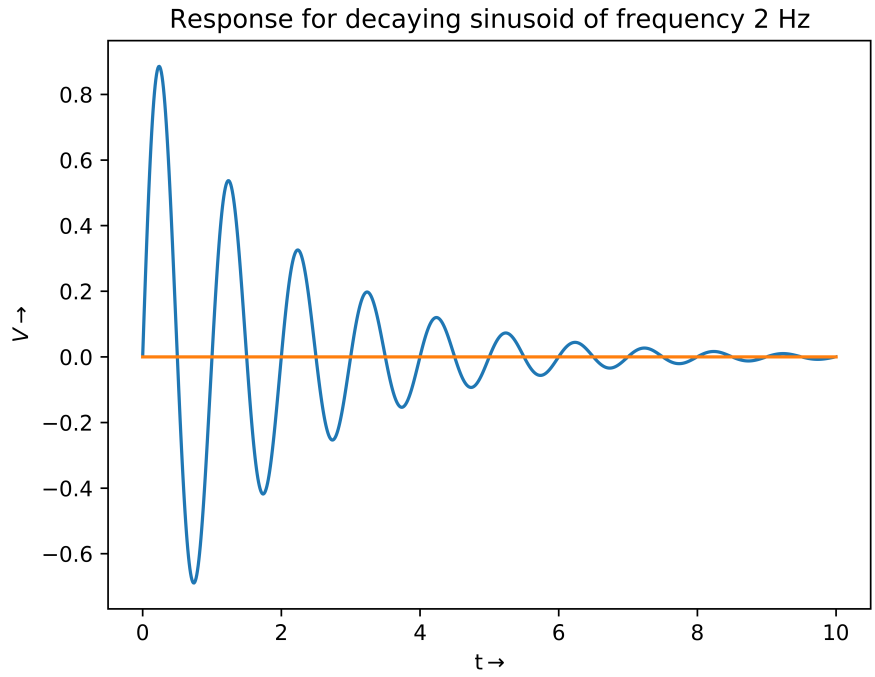
3.5 Question 4

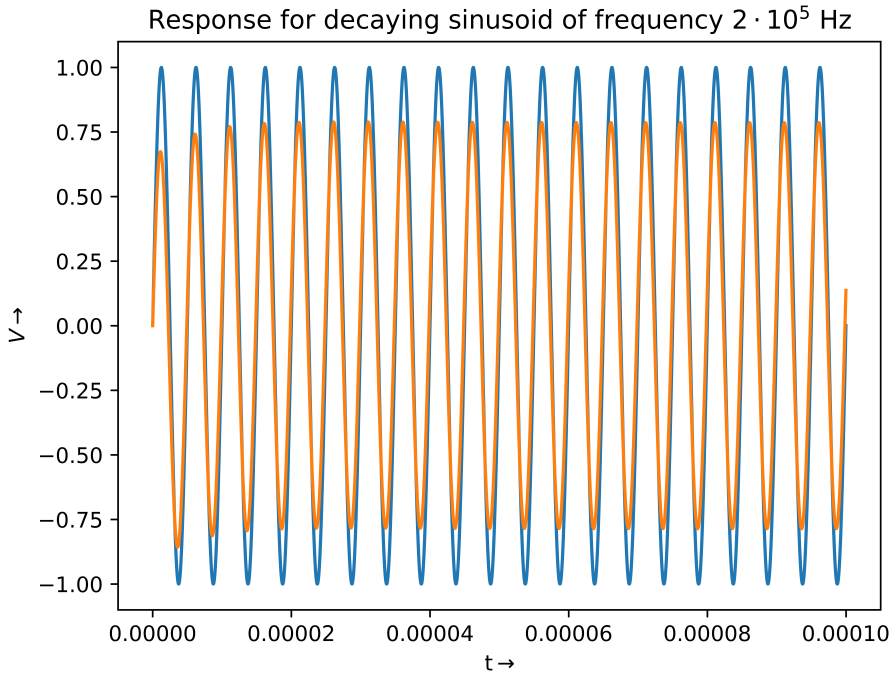
Now, we will find the response of the Highpass filter to two different damped sinusoids:

```
t = np.linspace(0,10,1000)
Vi = np.multiply(np.multiply(np.exp(-0.5*t),np.sin(2*np.pi*t)),np.heaviside(t
    ↪ ,0.5))
Vo = sp.lsim(H,Vi,T=t)
plt.figure(4)
plt.plot(Vo[0],Vi,label=r'$V_{in}$')
plt.plot(Vo[0],Vo[1],label=r'$V_{out}$')
plt.xlabel(r't$\rightarrow$')
plt.ylabel(r'$V\rightarrow$')
plt.title(r"Response for decaying sinusoid of frequency $2$ Hz")
plt.savefig("images/fig4",dpi=1000)
plt.show()

t = np.linspace(0,0.0001,10000)
Vi = np.multiply(np.multiply(np.exp(-0.5*t),np.sin(2*np.pi*200000*t)),np.
    ↪ heaviside(t,0.5))
Vo = sp.lsim(H,Vi,T=t)
plt.figure(5)
plt.plot(Vo[0],Vi,label=r'$V_{in}$')
plt.plot(Vo[0],Vo[1],label=r'$V_{out}$')
plt.xlabel(r't$\rightarrow$')
plt.ylabel(r'$V\rightarrow$')
plt.title(r"Response for decaying sinusoid of frequency $2 \cdot 10^5$ Hz")
plt.savefig("images/fig5",dpi=1000)
plt.show()
```

Graphs obtained:



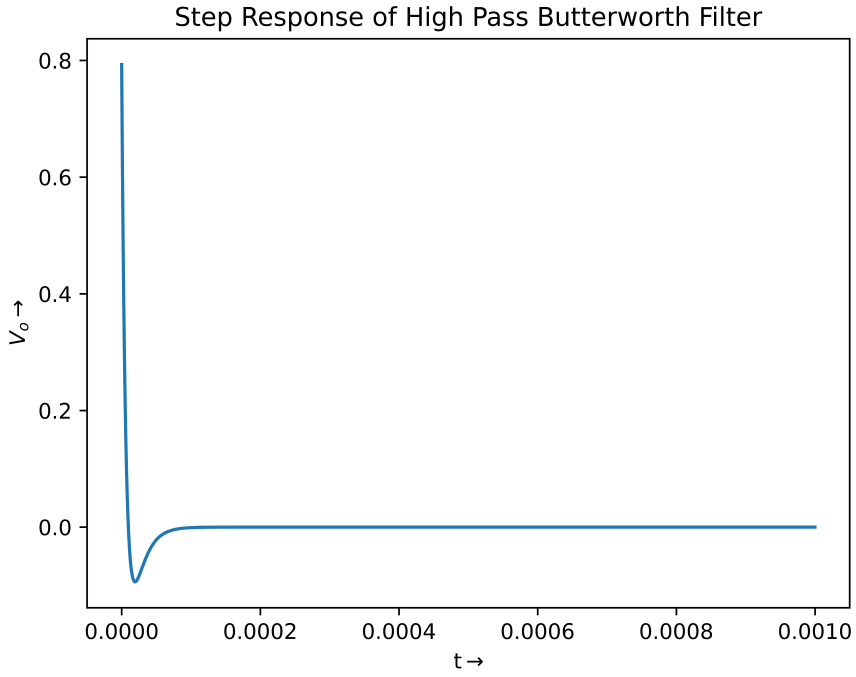


3.6 Question 5

Finally, we will plot the step response of the highpass filter:

```
plt.figure(6)
t = np.linspace(0,0.001,1000)
Vo = sp.step(H,T=t)
plt.plot(Vo[0],Vo[1])
plt.xlabel(r't$\rightarrow$')
plt.ylabel(r'$V_{o}\rightarrow$')
plt.title("Step Response of High Pass Butterworth Filter")
plt.savefig("images/fig6",dpi=1000)
plt.show()
```

Graph obtained:



In this question, it was asked whether we understand the response.

The reason for the initial peak is the initial conditions. The capacitor does not allow instantaneous changes in the voltage across it. Before $t = 0$, both capacitors have no charge, and the voltage across both is 0, and this will not change instantaneously.

Referring to the circuit diagram, the instant the 1V of the unit step is applied ($t = 0^+$), the drop across the capacitors C_1 and C_2 will still be zero. Therefore $V_m = 1V$.

$V_m = \frac{V_o}{G}$ and $V_o = G(V_p - V_m)$, therefore we get:

$$V_o = G\left(1 - \frac{V_o}{G}\right)$$

$$V_o = \frac{G}{2} = \frac{1.586}{2} = 0.793V \approx 0.8V$$

And hence, we have an initial peaking around 0.8V and then it quickly decays and settles down to the expected 0V since it is a highpass filter and DC output should hence be 0V.

4 Conclusions

SymPy provides an easy way to work with functions without losing generality and can be easily interfaced with SciPy and NumPy for scientific computing.

Two different filters were analyzed in this manner.