EE2703: Assignment 7

Sagar, EE20B115

April 8, 2022

1 Aim

In this assignment we aim to analyse circuits using the Symbolic algebra methods of **Sympy**.

2 Procedure and Observations

2.1 Low-Pass Filter

In this section, we analyse the circuit given in the tutorial.Writing KCL equations (in s-domain) of the nodes marked on the figure, we get the following matrix:

$$\begin{pmatrix} 0 & 0 & 1 & \frac{-1}{G} \\ \frac{-1}{1+sR_2C_2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ -\frac{1}{R_1} - \frac{1}{R_2} - sC_1 & \frac{1}{R_2} & 0 & sC_1 \end{pmatrix} \begin{pmatrix} V_1(s) \\ V_p(s) \\ V_m(s) \\ V_o(s) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{V_i(s)}{R_1} \end{pmatrix} \tag{1}$$

`Sympy` allows us to create matrices with symbolic entries, and also perform mathematical operations on them, as if they were `numpy` arrays. In the above circuit, the values of R_1, R_2, C_1, C_2 are $10k\Omega$, $10k\Omega$, 10pF, 10pF respectively.

Solving for $V_o(s)$, (with the above given values) we get:

$$V_o(s) = \frac{-0.0001586 \cdot V_i(s)}{2 \times 10^{-14}s^2 + 4.414 \times 10^{-9}s + 0.0002} \tag{2}$$

From (2), we get the step response of the circuit as:

$$V_o(s) = \frac{-0.0001586 \cdot \frac{1}{s}}{2 \times 10^{-14}s^2 + 4.414 \times 10^{-9}s + 0.0002} \tag{3}$$

Sympy allows us to convert a symbolic expression into functions that are compatible with other packages like **numpy**, **scipy** etc. This can be accomplished by converting the expression into a *lambda* function.

However, since we are required to use the `scipy.signal` toolbox, we have to convert the above the symbolic expression to a format with which we can easily create a `signal.lti` object. For that, we extract the coefficients of the numerator and denominator polynomials of $V_o(s)$ and create a `signal.lti` object using the same.

2.1.1 Code:

```
def SYMtoLTI(SYMFunc):
num, den = SYMFunc.as_numer_denom()
num = sym.Poly(num, s)
den = sym.Poly(den, s)
numCoeffs = num.all_coeffs()
```

```
denCoeffs = den.all_coeffs()
for i in range(len(numCoeffs)):
x = float(numCoeffs[i])
numCoeffs[i] = x
for j in range(len(denCoeffs)):
x = float(denCoeffs[j])
denCoeffs[j] = x
return numCoeffs, denCoeffs

R1 = 1e4
R2 = 1e4
C1 = 1e-9
C2 = 1e-9
G = 1.58
Vi = 1

s = sym.symbols('s')
A = sym.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 = sym.Matrix([0,0,0,-Vi/R1])
V = A.inv()*b

voNum, voDen = SYMtoLTI(V[3])
circuit1 = sig.lti(voNum, voDen)

w=np.linspace(1, 1e6, int(1e6))
w, mag, phase = sig.bode(circuit1, w)
fig1=plt.figure(1)
fig1.suptitle('Bode Plot of Transfer Function of Lowpass Filter')
plt.subplot(2,1,1)
plt.semilogx(w, mag)
plt.ylabel('$20log(\|H(j\omega)\|)$')
plt.subplot(2,1,2)
plt.semilogx(w, phase)
plt.xlabel(r'$\omega \ \to$')
plt.ylabel(r'$\angle H(j\omega)$')
plt.savefig("Figure_1")
plt.show()

t = np.linspace(0, 0.1, int(1e6))
time, voStep = sig.step(circuit1, None, t)
plt.figure(2)
plt.plot(time, voStep)
plt.title('Step Response of Lowpass Filter')
plt.xlabel('$t\ \to$')
plt.ylabel('$V_o(t)\ \to$')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_2")
plt.show()

Vi = np.heaviside(t, 1)*(np.sin(2e3*np.pi*t)+np.cos(2e6*np.pi*t))
```

```
plt.figure(3)
plt.plot(t, Vi)
plt.title('$V_i(t)=(sin(2x10^3\pi t)+cos(2x10^6\pi t))u(t)$ to Lowpass Filter'
plt.xlabel('t')
plt.ylabel('Vi(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_3")
plt.show()

time, vOut, rest = sig.lsim(circuit1, Vi, t)
plt.figure(4)
plt.plot(time, vOut)
plt.title('$V_o(t)$ for $V_i(t)=(sin(2x10^3\pi t)+cos(2x10^6\pi t))u(t)$ for L
plt.xlabel('t')
plt.ylabel('Vo(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_4")
plt.show()
```

2.1.2 Plots:

Now, we can easily create a `signal.lti` object with the coefficients we got, and use `signal.bode` to obtain the Bode magnitude and phase plots, which are shown below.

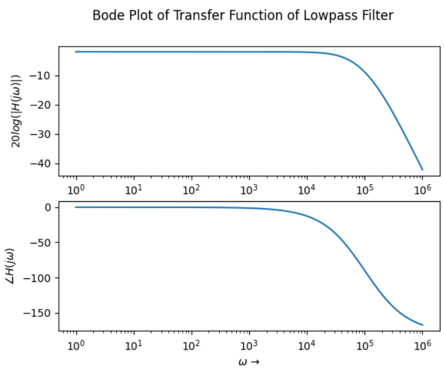


Figure 1: Bode Magnitude and Phase plots of step response of LPF

To see the step response of the system, we can use `signal.step`. The step response of the system is shown below.

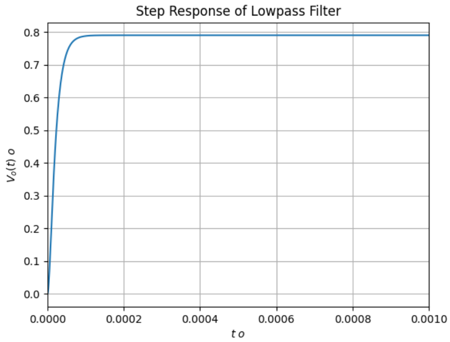


Figure 2: Step response of LPF

Now, we shall see that the circuit is indeed a low-pass filter by plotting the output for a mixed-frequency input, which has both high frequency and low frequency components.

We shall give the following input to the filter:

$$v_i(t) = (sin(2\pi \times 10^3 t) + cos(2\pi \times 10^6 t))u(t) \tag{4}$$

We shall use `signal.lsim` to calculate the time-domain response of the system.

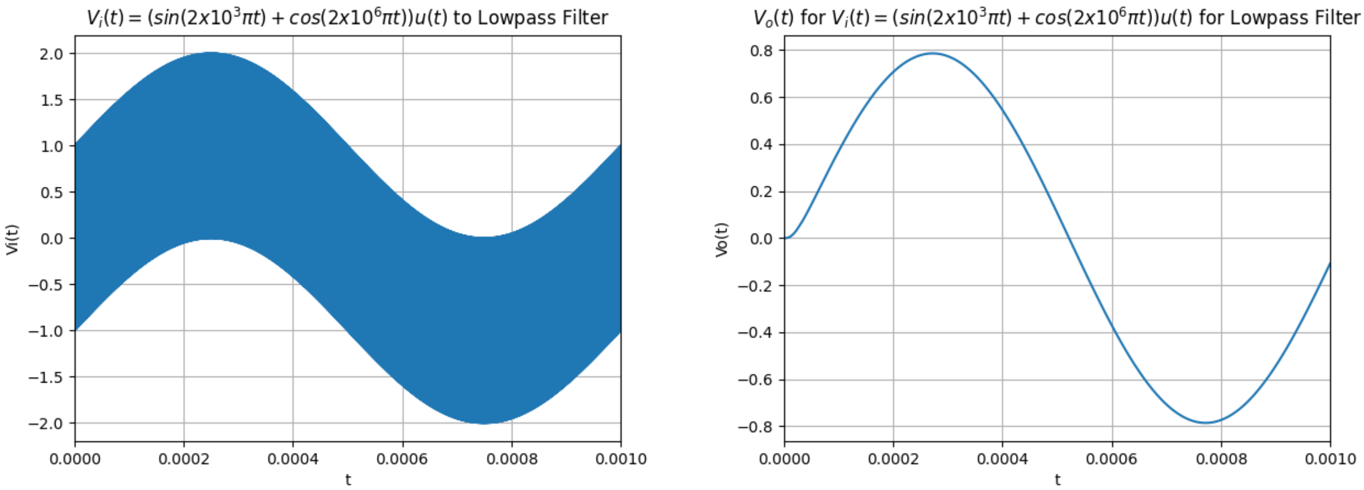


Figure 3: *Left*: Mixed frequency input*Right*: Filtered Output

2.1.3 Observations:

We can see that the output contains only the low frequency component of the input (1 KHz sinusoid). Thus, the circuit is a low-pass filter. It's cut-off frequency for the values of R_1, R_2, C_1, C_2 used is $\frac{1}{2\pi}$ MHz.

2.2 High-Pass Filter

We shall now look at a slightly modified version of the above circuit.Performing a similar procedure like before, we get the KCL matrix as:

$$\begin{pmatrix} 0 & 0 & 1 & \frac{-1}{G} \\ \frac{sR_3C_2}{1+sR_3C_2} & 0 & -1 & 0 \\ 0 & -G & G & 1 \\ -\frac{1}{R_1} - sC_2 - sC_1 & 0 & sC_2 & \frac{1}{R_1} \end{pmatrix} \begin{pmatrix} V_1(s) \\ V_p(s) \\ V_m(s) \\ V_o(s) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -sC_1V_i(s) \end{pmatrix} \tag{5}$$

Solving it for $V_o(s)$, we get,

$$V_o(s) = \frac{1.586 \times 10^{-14} s^2 \cdot V_i(s)}{2 \times 10^{-14} s^2 + 4.414 \times 10^{-9} s + 0.0002} \tag{6}$$

2.2.1 Code:

```
    def HighPass(R1, R3, C1, C2, G, Vi):
A = sym.Matrix([[0, -1, 0, 1/G],
                [s*C2*R3/(s*C2*R3+1), 0, -1, 0],
                [0, G, -G, 1],
                [-s*C2-1/R1-s*C1, 0, s*C2, 1/R1]])

b = sym.Matrix([0,
                0,
                0,
                -Vi*s*C1])
return (A.inv()*b)[3]

Vo = HighPass(10000, 10000, 1e-9, 1e-9, 1.58, 1)
voNum, voDen = SYMtoLTI(Vo)
circuit2 = sig.lti(voNum, voDen)

w, mag, phase = sig.bode(circuit2, w)
fig5=plt.figure(5)
fig5.suptitle('Bode Plot of Transfer Function of HighPass Filter')
plt.subplot(2,1,1)
plt.semilogx(w, mag)
plt.ylabel('$20log(\|H(j\omega)\|)$')
plt.subplot(2,1,2)
plt.semilogx(w, phase)
plt.xlabel(r'$\omega \ \to$')
plt.ylabel(r'$\angle H(j\omega)$')
plt.savefig("Figure_5")
plt.show()

vi = np.heaviside(t, 1)*(np.sin(2e3*np.pi*t)+np.cos(2e6*np.pi*t))

plt.figure(6)
plt.plot(t, vi)
plt.title('$V_i(t)=(sin(2x10^3\pi t)+cos(2x10^6\pi t))u(t)$ to HighPass Filter
plt.xlabel('t')
plt.ylabel('Vi(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_6")
plt.show()

time, vOut, rest = sig.lsim(circuit2, vi, t)
plt.figure(7)
plt.plot(time, vOut)
plt.title('$V_o(t)$ for $V_i(t)=(sin(2x10^3\pi t)+cos(2x10^6\pi t))u(t)$ for H
plt.xlabel('t')
plt.ylabel('Vo(t)')
```

```
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_7")
plt.show()

ViDampedHighFreq = np.heaviside(t, 1)*(np.sin(2*np.pi*t))*np.exp(-t)
ViDampedLowFreq = np.heaviside(t, 1)*(np.sin(2e5*np.pi*t))*np.exp(-t)

time, VoutDampedLowFreq, rest = sig.lsim(circuit2, ViDampedHighFreq, t)
plt.figure(8)
plt.plot(time, VoutDampedLowFreq)
plt.title('$V_o(t)$ for $V_i(t)=sin(2\pi t)e^{-t}u(t)$ for HighPass Filter')
plt.xlabel('t')
plt.ylabel('Vo(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_8")
plt.show()

time, VoutDampedHighFreq, rest = sig.lsim(circuit2, ViDampedLowFreq, t)
plt.figure(9)
plt.plot(time, VoutDampedHighFreq)
plt.title('$V_o(t)$ for $V_i(t)=sin(2x10^5\pi t)e^{-t}u(t)$ for HighPass filte
plt.xlabel('t')
plt.ylabel('Vo(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_9")
plt.show()

time, voStep = sig.step(circuit2, None, t)
plt.figure(10)
plt.plot(time, voStep)
plt.title('Step Response of HighPass Filter')
plt.xlabel('t')
plt.ylabel('Vo(t)')
plt.xlim(0, 1e-3)
plt.grid(True)
plt.savefig("Figure_10")
plt.show()
```

2.2.2 Plots:

The Bode magnitude and phase plots of the transfer function as:

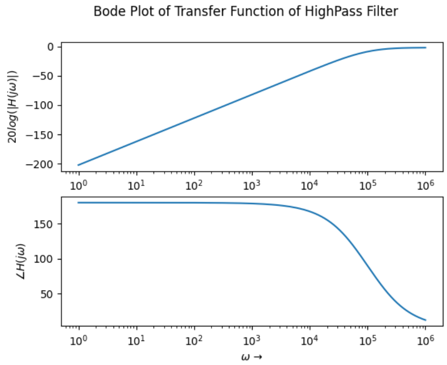


Figure 4: Bode Magnitude and Phase plots of step response of HPF

The step response of the circuit was obtained as:

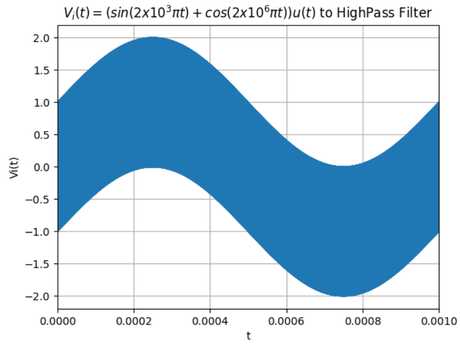


Figure 5: Step Response of HPF

The response to the mixed frequency input in Eq (4) is:

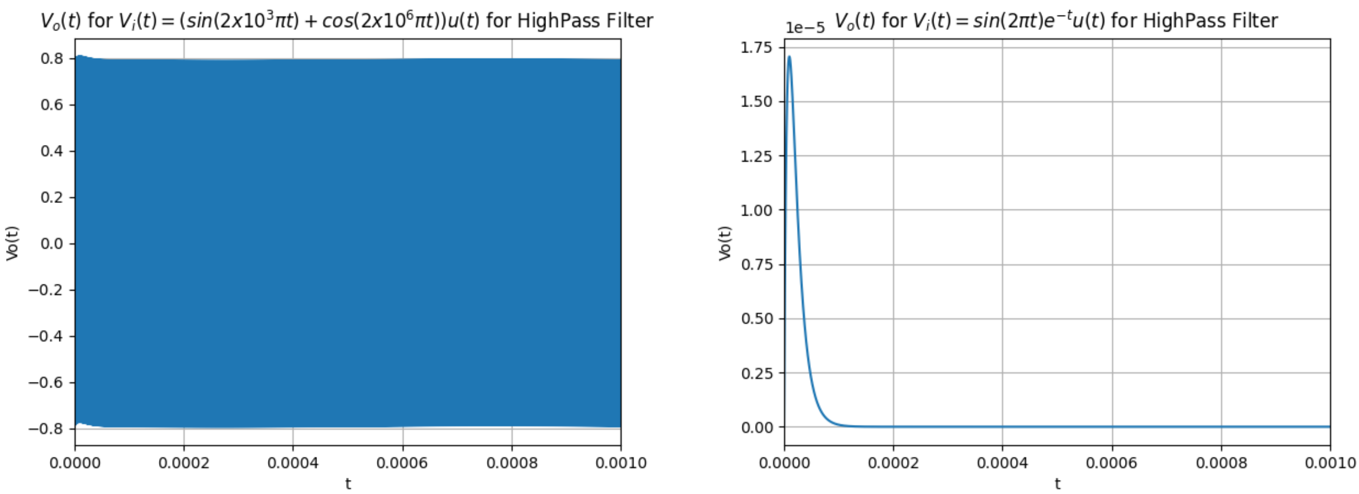
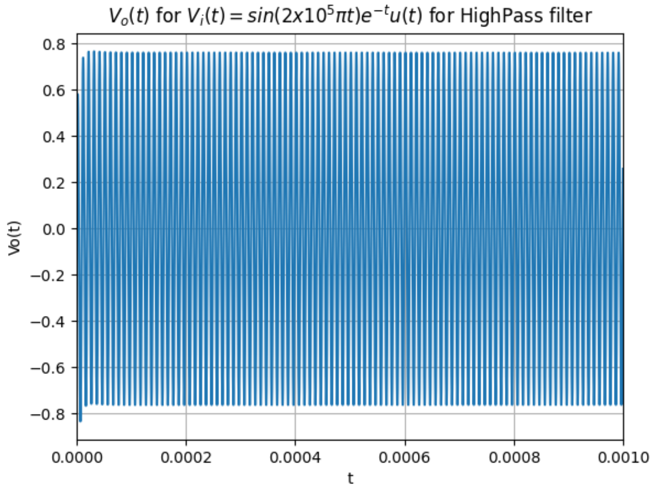


Figure 6: *Left*: Mixed frequency input*Right*: Filtered Output

We shall look at the response of the system to a damped sinusoid. We shall consider two of them - one of high frequency (0.1 MHz) and the other of low frequency (1 Hz).



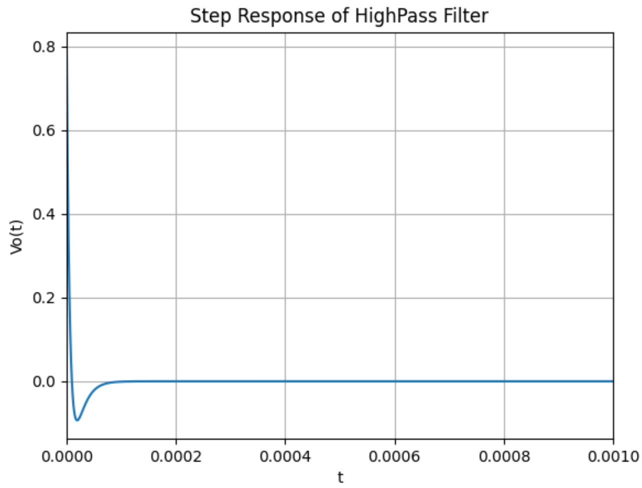


Figure 7: *Left*: Low frequency input*Right*:High frequency input

2.2.3 Observations:

We see that the output of the filter if the input is a low frequency damped sinusoid is 0, except for initial transients. This is expected due to the inherent nature of the circuit to act as a high-pass filter. This is why we can see that it allows the high-frequency input to pass through.

3 Conclusions

1. **Sympy** provides a way to analyse LTI systems using their Laplace transforms. The toolbox was used to study the behaviour of a low pass filter,implemented using an op-amp of gain G. For a mixed frequency sinusoid as input, it was found that the filter suppressed the high frequencies while allowing the low frequency components.

2. Similarly, a high pass filter was implemented using an op-amp with the same gain. The magnitude response of the filter was plotted and its output was analysed for damped sinusoids.The step response of the filter was found to have a non-zero peak at $t = 0$,due to the sudden change in the input voltage.