ASSIGNMENT 7: CIRCUIT ANALYSIS USING SYMPY

Hariharan P EE20B042

April 4, 2022

1 Brief Up:

The tasks of this assignment is the following:

- Analyze Filters using Laplace Transform.
- Solving symbolic Algebra using sympy.
- Plot graphs to analyze high pass and low pass analog filters.

2 Low Pass Filter:

The low pass filter given has the following matrix equation after simplification of the modified nodal equations.

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

The python code is as shown below:

```
s = symbols('s')

def lowpass(R1,R2,C1,C2,G,Vi):

        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
```

```
A,b,V=lowpass(10000,10000,1e-9,1e-9,1.586,1)
Vout=V[3]
H = sympyToTrFn(Vout)
ww=plt.logspace(0,8,801)
ss=1j*ww
hf=lambdify(s,Vout,'numpy')
v=hf(ss)
```

The plot for the magnitude of the transfer function is as shown below:

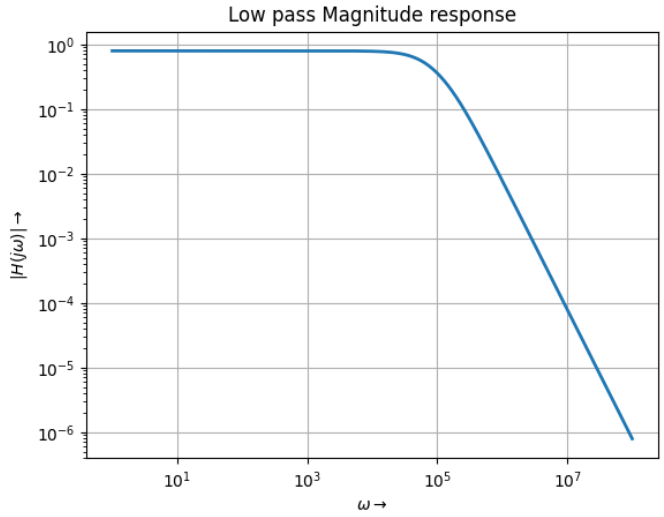

Figure 1: Magnitude bode plot of the low pass filter

3 High Pass Filter:

The high pass filter given has the following matrix equations after simplification of the modified nodal equations

$$\begin{pmatrix} 0 & 0 & 1 & -\frac{1}{G} \\ -\frac{-sR_3C_2}{1+sR_3C_2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ -1-(sR_1C_1)-(sR_1C_2)) & sC_2R_1 & 0 & 1 \end{pmatrix} \begin{pmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -V_i(s)sR_1C_1 \end{pmatrix}$$

The python code snippet is as shown below:

```
def highpass(R1,R3,C1,C2,G,Vi):

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

A,b,V2=highpass(10000,10000,1e-9,1e-9,1.586,1)
Vout2=V2[3]
H2 = sympyToTrFn(Vout2)
ww=plt.logspace(0,8,801)
ss=1j*ww
hf=lambdify(s,Vout2,"numpy")
v=hf(ss)

plt.title("High pass Magnitude response")
plt.xlabel(r'$\omega\rightarrow$')
plt.ylabel(r'$|H(j\omega)|\rightarrow$')
plt.loglog(ww,abs(v),lw=2)
plt.grid(True)
plt.show()
```

The plot for the magnitude of the transfer function is as shown below:

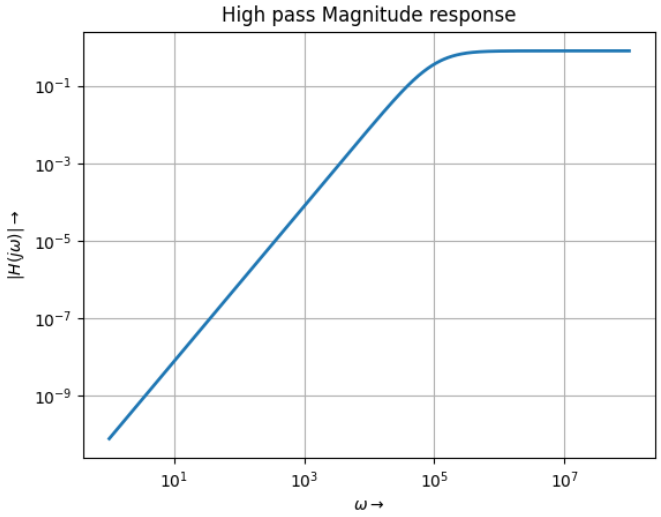

Figure 2: Magnitude bode plot of the high pass filter

4 Sympy to transfer function:

The sympy functionswith parameter as 's', must be converted to another form that is understood by sp.signal. This is done using the *sympyToTrFn()* function.
The python code snippet is as shown:

```
def sympyToTrFn(Y):

    Y = expand(simplify(Y))

    num,den = fraction(Y)
    num, den = [[float(i) for i in Poly(j, s).all_coeffs()] for j in Y.as

    H = sp.lti(num,den)
    return H
```

5 Step Response Of the Low Pass Filter:

Lets find the step response of the low pass filter by assigning $Vi(s) = 1/s$.
The python code below explains it.

```
As1,Bs1,Vs1=lowpass(10000,10000,1e-9,1e-9,1.586,1/s)
Vout1=Vs1[3]
H1 = sympyToTrFn(Vout1)
t,y1 = sp.impulse(H1,None,plt.linspace(0,5e-3,10000))
plt.plot(t,y1)
plt.title("Step Response for low pass filter")
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$V_o(t)\rightarrow$')
plt.grid(True)
plt.show()
```

The plot for the step response of the low pass circuit is as shown:

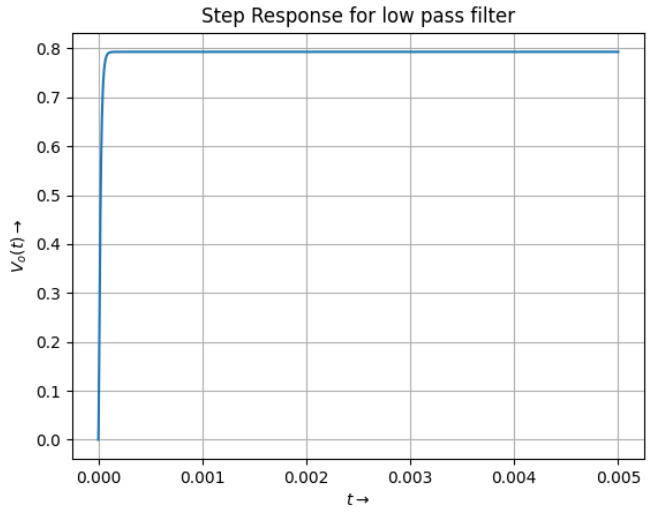

Figure 3: Step response of low pass filter.

6 Response to Sum Of Sinusoids:

For the input being sum of sinusoids, like

$$V_i(t) = (\ \sin(2000\pi t) + \cos\left(2 * 10^6 \pi t\right)\ ) u_o(t)\ Volts$$

Then the output response for the lowpass filter and high pass filter are found as shown in the code below.

```
t=plt.linspace(0,1e-2,100000)
def inp1(t):
    return (plt.sin(2000*plt.pi*t)+plt.cos(2e6*plt.pi*t))

def inp_response(Y,title,inp,tmax=5e-3):
    H= sympyToTrFn(Y)
    t = plt.linspace(0,tmax,100000)
    t,y,_ = sp.lsim(H,inp(t),t)
    plt.title(title)
    plt.xlabel("t")
    plt.ylabel("Vo")
    plt.plot(t,y)
    plt.show()
    return


inp_response(Vout,"Response of Low pass Filter to sum of sinusoids"
,inp1)
```

```
inp_response(Vout2,"Response of High pass Filter to sum of sinusoids"
,inp1,1e-2)
```

The output response to the sum of sinusoids for the low pass filter and high pass filter are as shown:

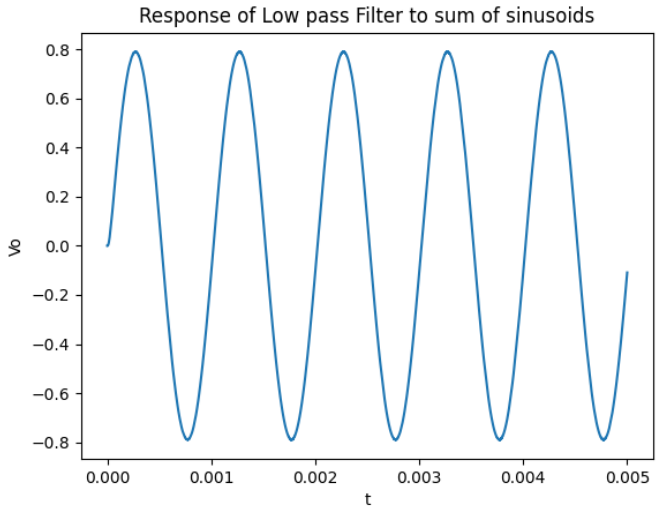

Figure 4: Output response to the sum of sinusoids by low pass filter

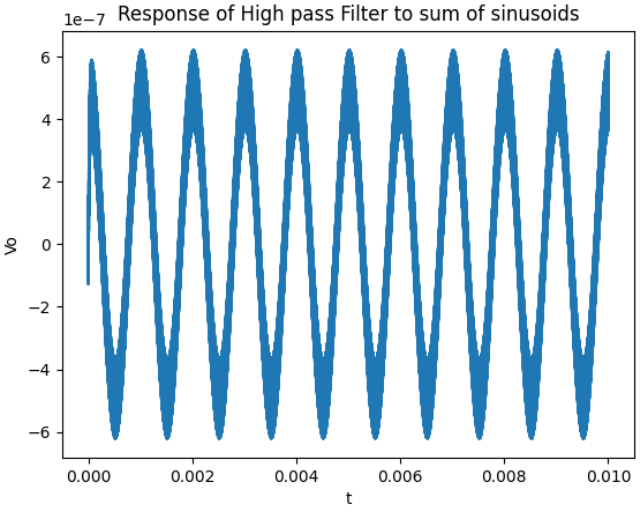

Figure 5: Output response to the sum of sinusoids by high pass filter

7 Response to Damped Sinusoids:

Lets assign the input voltage as a damped sinusoid like,
Low frequency,

$$V_i(t) = e^{-1000t}(\ \sin(2000\pi t)\ )u_o(t)\ Volts$$

High frequency,

$$V_i(t) = e^{-1000t}(\ \sin\left(2 * 10^6 \pi t\right)\ )u_o(t)\ Volts$$

The high frequency and low frequency plots of the input damped sinusoids to a high pass filter are as shown: The python code snippet to execute the

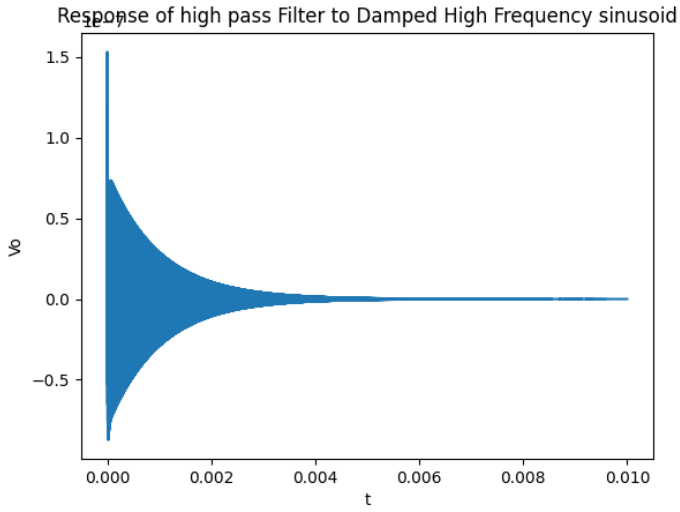

Figure 6: High frequency damped sinusoid

above is as shown:

```
def inp2(t):
    return plt.sin(1e7*t)*plt.exp(-1000*t)
def inp3(t):
    return plt.sin(1e3*t)*plt.exp(-1000*t)
def inp_response(Y, title ,inp ,tmax=5e-3):
    H= sympyToTrFn(Y)
    t = plt.linspace(0,tmax,100000)
    t,y,_ = sp.lsim(H,inp(t),t)
    plt.title(title)
    plt.xlabel("t")
    plt.ylabel("Vo")
    plt.plot(t,y)
```

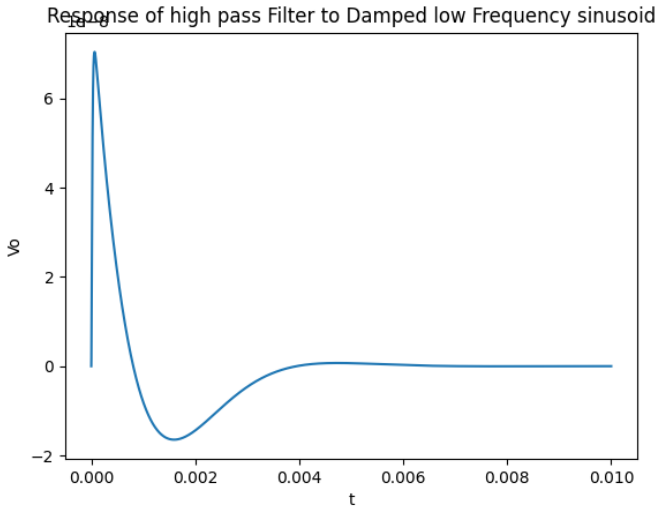

Figure 7: Low frequency damped sinusoid

```
    plt.show()
    return

inp_response(Vout2,
"Response of high pass Filter to Damped High Frequency sinusoid"
,inp2,1e-2)
inp_response(Vout2,
"Response of high pass Filter to Damped low Frequency sinusoid"
,inp3,1e-2)
```

8 Step Response of High Pass Filter:

Lets find the step response of the high pass filter by assigning $Vi(s) = 1/s$.
It's python code below .

```
As2,Bs2,Vs2=highpass(10000,10000,1e-9,1e-9,1.586,1/s)
Vout2=Vs2[3]
H3 = sympyToTrFn(Vout2)
t,y2 = sp.impulse(H3,None,plt.linspace(0,5e-3,10000))
plt.plot(t,y2)
plt.title("Step Response for high pass filter")
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$V_o(t)\rightarrow$')
plt.grid(True)
plt.show()
```

The plot of the step response for a high pass filter is as shown: The unit

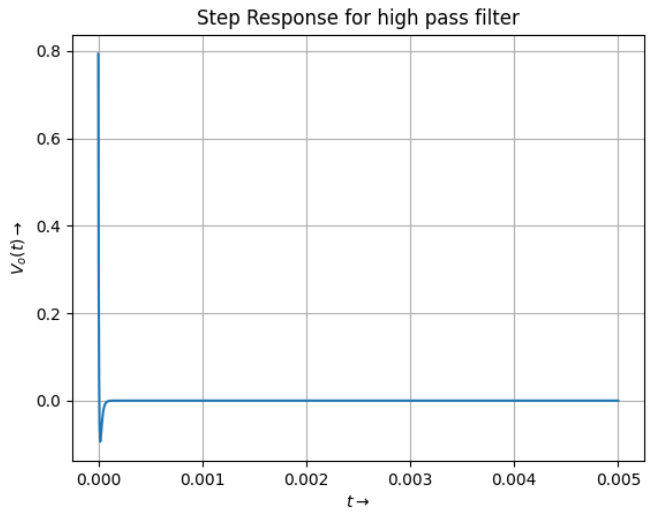

Figure 8: Step response for a high pass filter

step response is high at t=0 which is an abrupt change in the input. Since there is no other change at large time values outside the neighbourhood of 0, the Fourier transform of the unit step has high values near 0 frequency, which the high pass filter attenuates.

9 Conclusion:

The sympy module can be used to analyse complicated analog circuits by analytically solving their node equations. Then by interpreting the solutions by plotting time domain responses using the signals toolbox. This is a very useful toolbox for analyzing systems like the active filters.