EE2703 : Applied Programming Lab

Assignment 7

Harisankar K J
EE20B043

8 April 2022

Abstract

The aim of the assignment is:

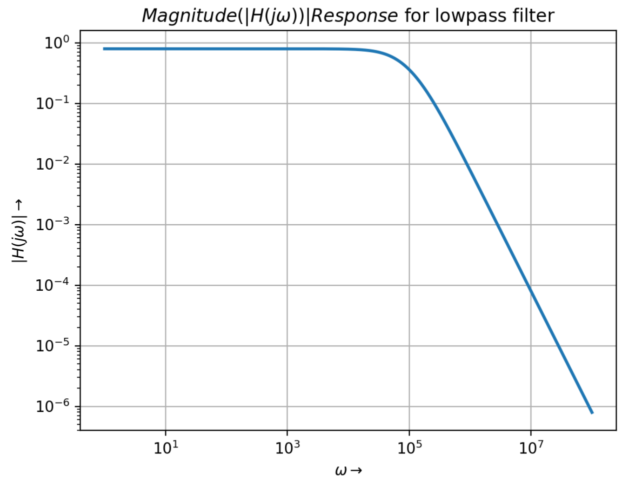
- To analyze Filters using Laplace Transform.
- To see how python can be used for symbolic Algebra.
- To plot graphs to understand high pass and low pass analog filters.

Low pass Filter

The low pass filter we use gives the following matrix after simplification of Modified Nodal Equations.

$$\begin{bmatrix} 0 & 0 & 1 & -1/G \\ \frac{-1}{sR_2C_2} & 1 & 0 & 0 \\ 0 & -G & G & 1 \\ \frac{-1}{R_1} - \frac{1}{R_2} - s*C_1 & \frac{1}{R_2} & 0 & sC_1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{-V_i(s)}{R_1} \end{bmatrix}$$

```
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 V
```

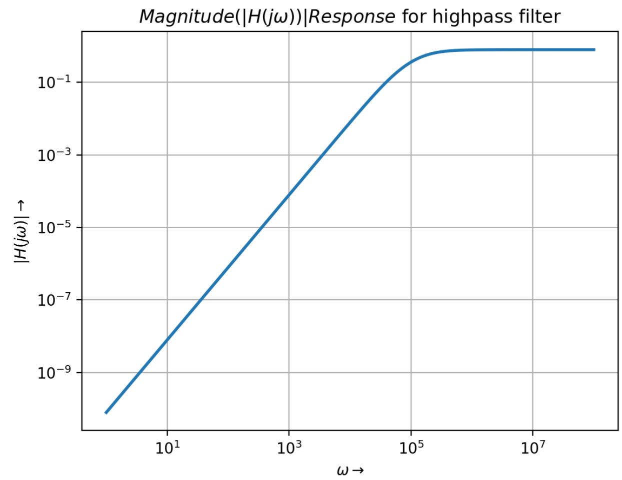


High Pass Filter

The high pass filter we use gives the following matrix after simplification of Modified Nodal Equations.

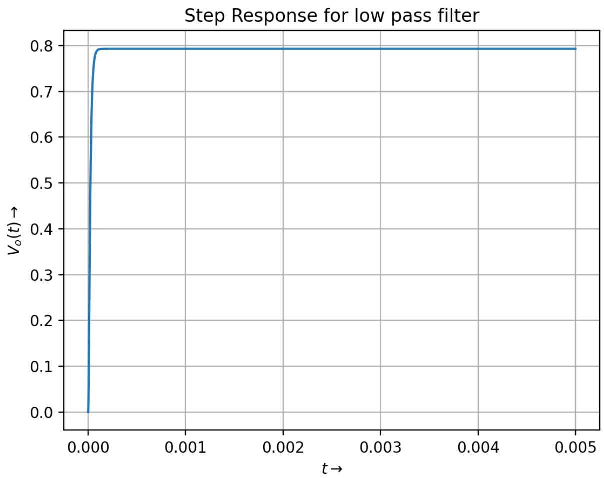
$$\begin{bmatrix} 0 & -1 & 0 & 1/G \\ \frac{s*C_2*R_3}{1+s*C_2*R_3} & 0 & -1 & 0 \\ 0 & G & -G & 1 \\ -s*C_2 - \frac{1}{R_1} - s*C_1 & 0 & s*C_2 & \frac{1}{R_1} \end{bmatrix} \begin{bmatrix} V_1 \\ V_p \\ V_m \\ V_o \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -V_i(s)*s*C_1 \end{bmatrix}$$

```
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],
                [-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 V
```



Question 1: Step Response of Low pass Filter

```
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],
                [-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 V
```

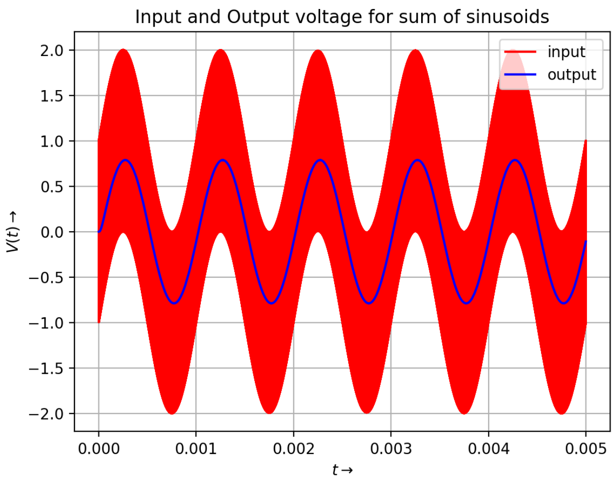


Question 2: Response for Sum of Sinusoids

When the input is a sum of sinusoids like,

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

```
v_in = sin(2000*pi*t) + cos(2e6*pi*t)
t, y, svec = sp.lsim(H, v_in, t)

# The plot for output response for sum of sinusoids of a lowpass
    filter.
figure(3)
plot(t, v_in, 'r', label='input')
plot(t, y, 'b', label='output')
title(r"Input and Output voltage for sum of sinusoids")
xlabel(r'$t\rightarrow$')
ylabel(r'$V(t)\rightarrow$')
legend()
grid()
```



Question 4: Response to Damped Sinusoids

In this case we assign the input voltage as a damped sinusoid like,
Low frequency,

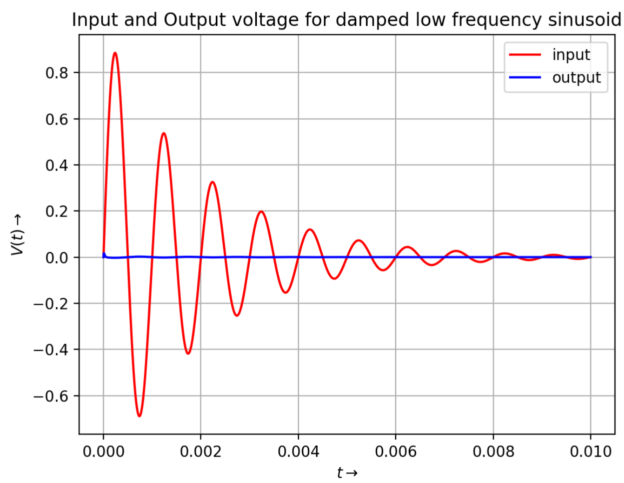
$$V_i(t) = e^{-500t}(\ \sin(2000\pi t)\ )u_o(t)\ Volts$$

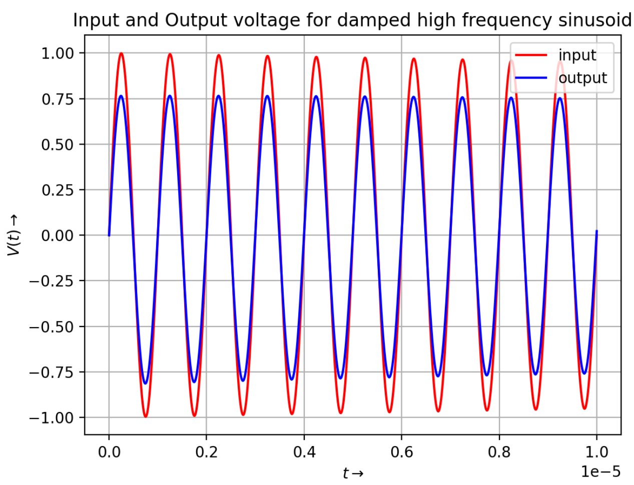
High frequency,

$$V_i(t) = e^{-5000t}(\ \sin(2 * 10^6\pi t)\ )u_o(t)\ Volts$$

```
# High frequency

t = linspace(0, 1e-5, 100000)
v_in = exp(-5000*t)*sin(2e6*pi*t)
t, y, svec = sp.lsim(H, v_in, t)
figure(6)
plot(t, v_in, 'r', label='input')
plot(t, y, 'b', label='output')
title(r"Input and Output voltage for damped high frequency 
    sinusoid")
xlabel(r'$t\rightarrow$')
ylabel(r'$V(t)\rightarrow$')
legend()
grid()

# Low frequency

t = linspace(0, 1e-2, 10000)
v_in = exp(-500*t)*sin(2e3*pi*t)*heaviside(0, 1)
```

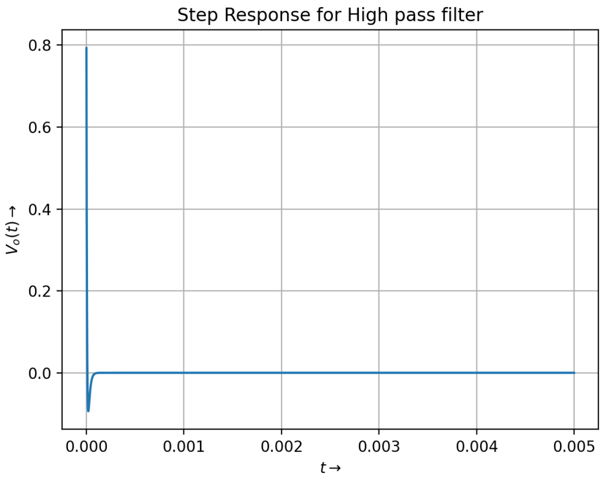
```
t, y, svec = sp.lsim(H, v_in, t)
figure(7)
plot(t, v_in, 'r', label='input')
plot(t, y, 'b', label='output')
title(r"Input and Output voltage for damped low frequency 
    sinusoid")
xlabel(r'$t\rightarrow$')
ylabel(r'$V(t)\rightarrow$')
legend()
grid()
show()
```





Question 5: Step Response of high pass Filter

```
V = highpass(10000, 10000, 1e-9, 1e-9, 1.586, 1/s)

Vo = V[3].simplify()
H1 = sympytoscipy(Vo)

t, y1 = sp.impulse(H1, None, linspace(0, 5e-3, 100000))
figure(5)
plot(t, y1)
title(r"Step Response for High pass filter")
xlabel(r'$t\rightarrow$')
ylabel(r'$V_o(t)\rightarrow$')
grid()
```



0.1 Conclusion

- Sympy module has allowed us to analyse quite complicated circuits by analytically solving their node equations.
- We We interpreted the solutions by plotting time domain responses using the signals toolbox.