Op Amp with Filter

October 17, 2023

[1]:
```
import sympy as sp
import control as ct
import matplotlib.pyplot as plt
import numpy as np

plt.style.use('maroon_ipynb.mplstyle')
```

Example from lecture 13.

[2]:
```
Rs, Rf, R, C, s = sp.symbols('R_s R_f R C s')
Vo, Vi, V1, V2 = sp.Function('V_o')(s), sp.Function('V_i')(s), sp.
 ↪Function('V1')(s), sp.Function('V2')(s)

eq1 = sp.Eq(V1, -Rf/Rs*Vi)
eq2 = sp.Eq(V2, -V1)
eq3 = sp.Eq((V2 - Vo)/R - Vo/(1/(C*s)), 0)
display(eq1, eq2, eq3)
```

$$V_1(s) = -\frac{R_f V_i(s)}{R_s}$$

$$V_2(s) = -V_1(s)$$

$$-CsV_o(s) + \frac{V_2(s) - V_o(s)}{R} = 0$$

[3]:
```
sol = sp.solve([eq1, eq2, eq3], [Vo, V1, V2], dict=True)[0]
tf = sol[Vo]/Vi
tf
```

[3]:

$$\frac{R_f}{CRR_s s + R_s}$$

[4]:
```
values = [
    (Rf, 10_000),
    (Rs, 1_000),
    (R, 100),
    (C, sp.S('100e-6'))
]

tf_subs = tf.subs(values)
```

```
tf_subs
```

[4]:

$$\frac{10000}{10.0s + 1000}$$

[5]:
```
sys1 = ct.tf(10_000, [10, 1000])
sys1
```

[5]:

$$\frac{1 \times 10^4}{10s + 1000}$$

[6]:
```
t_array = np.linspace(0, 0.1, 5_000)
noise = np.random.uniform(-5/1000, 5/1000, t_array.shape)
vi = np.full(t_array.shape, 10/1000) + noise
_, response = ct.forced_response(sys1, T=t_array, U=vi)

fig, ax = plt.subplots()

ax.plot(t_array, response, label='$v_o(t)$')
ax.plot(t_array, vi, color='darkgrey', label='$v_i(t)$')
ax.set_ylabel('Voltage ($V$)')
ax.set_xlabel('Time ($s$)')
ax.legend()
plt.show()
```