# System Dynamics Homework 4

October 16, 2023

Gabe Morris

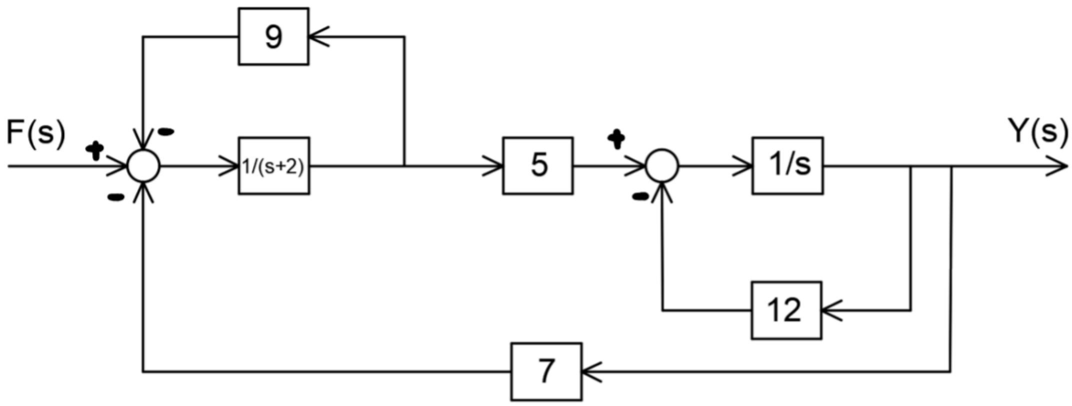
[1]:
```python
import sympy as sp
import control as ct
import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import odeint

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

# Contents

# 1 Problem 1

## 1.1 Given




## 1.2 Find

Find the transfer function $\frac{Y(s)}{F(s)}$ for the block diagram.

## 1.3 Solution

The solution can be determined using two different methods. The first is an algebraic solution where B is the expression after the first block seen above. The second can be determined using the feedback and series functions.

[2]:
```python
# Using algebra
from sympy.abc import F, Y, B, s

eq1 = sp.Eq((F - 9*B - 7*Y)*1/(s + 2), B)
eq2 = sp.Eq((B*5 - 12*Y)*1/s, Y)
display(eq1, eq2)
```

$$\frac{-9B + F - 7Y}{s + 2} = B$$

$$\frac{5B - 12Y}{s} = Y$$

[3]:
```python
sol = sp.solve([eq1, eq2], [Y, B], dict=True)[0]
sol[Y]/F
```

[3]:

$$\frac{5}{s^2 + 23s + 167}$$

[4]:

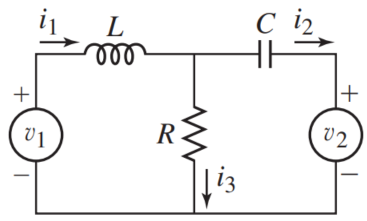
```python
sys1 = ct.feedback(ct.tf(1, [1, 0]), 12)
sys2 = ct.series(5, sys1)
sys3 = ct.feedback(ct.tf(1, [1, 2]), 9)
sys4 = ct.series(sys3, sys2)
sys5 = ct.feedback(sys4, 7)
sys5
```

[4]:

$$\frac{5}{s^2 + 23s + 167}$$

# 2 Problem 2

## 2.1 Given




$$L = 500\,mH,\ C = 100\,\mu F,\ R = 300\,\Omega$$

$$v_1 = 5e^{-t}\sin(6t)\,V,\ v_2 = 10\sin(t)\,V$$

All initial conditions are zero.

## 2.2 Find

a. The system of ODE's (should be two equations if using mesh currents).
b. Solve the system for $i_1$, $i_2$, and $i_3$ as seen the figure above. Use any method to find the result and plot up to 6 seconds.

## 2.3 Solution

### 2.3.1 Part A

[5]:
```
L_, C_, R_ = 500e-3, 100e-6, 300

L, C, R = sp.symbols('L C R')
t = sp.Symbol('t')
V1 = sp.Function('v_1')(t)
V2 = sp.Function('v_2')(t)
iA = sp.Function('i_A')(t)  # Mesh currents
iB = sp.Function('i_B')(t)  # Mesh currents

eq1 = sp.Eq(L*iA.diff() + R*(iA - iB), V1)  # KVL around iA
eq2 = sp.Eq(1/C*iB + R*(iB.diff() - iA.diff()) + V2.diff(), 0)  # KVL around iB
display(eq1, eq2)
```

$$L\frac{d}{dt}i_A(t) + R\left(i_A(t) - i_B(t)\right) = v_1(t)$$

$$R\left(-\frac{d}{dt}i_A(t) + \frac{d}{dt}i_B(t)\right) + \frac{d}{dt}v_2(t) + \frac{i_B(t)}{C} = 0$$

### 2.3.2 Part B

The state variable solution is the easiest since we already have a system of first order ODE's.

[6]:
```
# Solving using state variables
state_sol = sp.solve([eq1, eq2], [iA.diff(), iB.diff()], dict=True)[0]
for key, value in state_sol.items(): display(sp.Eq(key, value))
```

$$\frac{d}{dt}i_A(t) = -\frac{Ri_A(t)}{L} + \frac{Ri_B(t)}{L} + \frac{v_1(t)}{L}$$

$$\frac{d}{dt}i_B(t) = -\frac{\frac{d}{dt}v_2(t)}{R} - \frac{Ri_A(t)}{L} + \frac{Ri_B(t)}{L} + \frac{v_1(t)}{L} - \frac{i_B(t)}{CR}$$

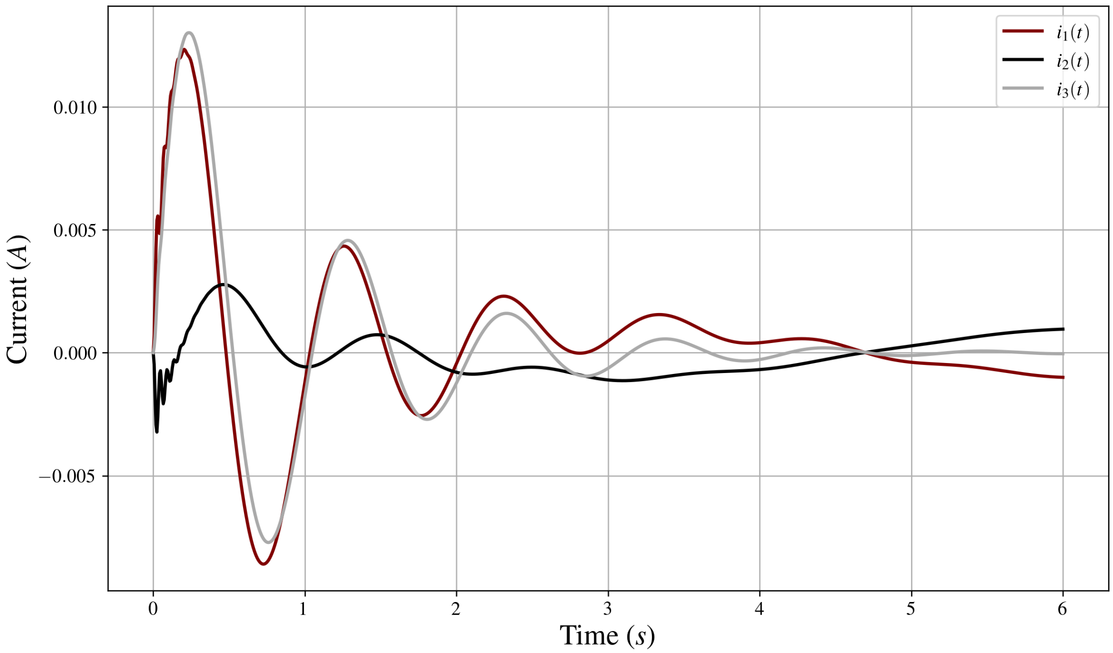
[7]:
```
v1 = lambda t_: 5*np.exp(-t_)*np.sin(6*t_)
v2_diff = lambda t_: 10*np.cos(t_)

def state_vars(i, t_):
    return [
        (v1(t_) + R_*i[1] - R_*i[0])/L_,
        -v2_diff(t_)/R_ - R_/L_*i[0] + R_/L_*i[1] + v1(t_)/L_ - i[1]/(C_*R_)
    ]



t_array = np.linspace(0, 6, 1000)
sol = odeint(state_vars, (0, 0), t_array)
iA, iB = sol[:, 0], sol[:, 1]

plt.plot(t_array, iA, label='$i_1(t)$')
plt.plot(t_array, -iB, label='$i_2(t)$')
plt.plot(t_array, iA - iB, label='$i_3(t)$')
plt.legend()
plt.xlabel('Time ($s$)')
plt.ylabel('Current ($A$)')
plt.show()
```



$i_1(t)$
$i_2(t)$
$i_3(t)$
Current ($A$)
0.010
0.005
0.000
−0.005
0
1
2
3
4
5
6
Time ($s$)