Dynamical Systems Homework 3

March 19, 2025

Gabe Morris

[1]:

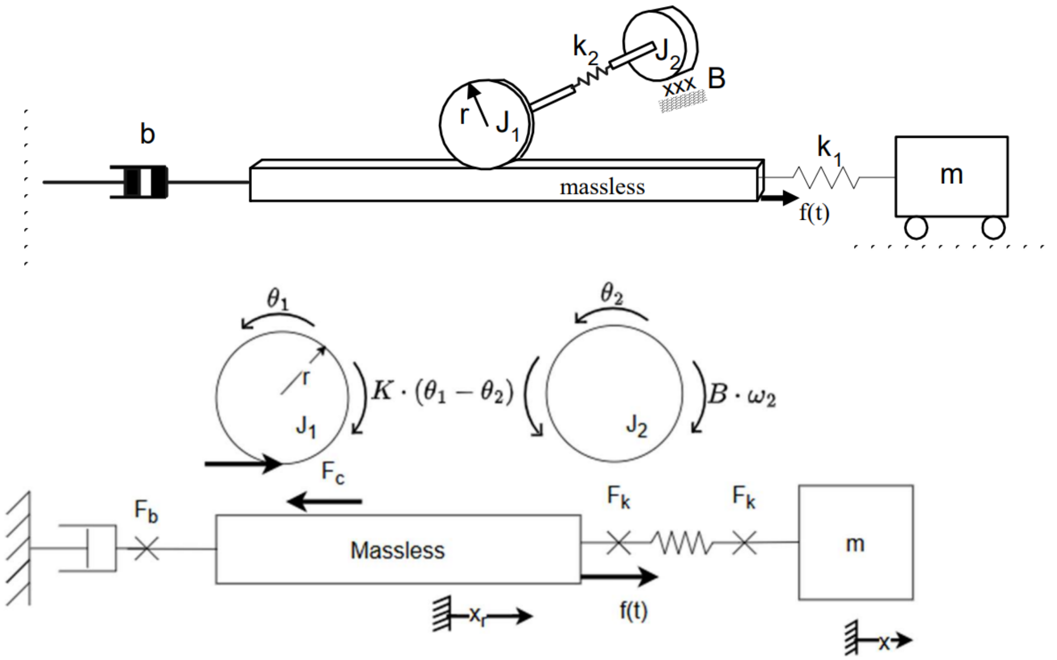
```
# toc
import sympy as sp
import matplotlib.pyplot as plt

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

Contents

Problem 1

Given



The constants from the above figure are given as follows:

$$
\begin{gathered}
r = 0.05\,m \\
k_1 = 1000\,N/m \\
k_2 = 500\,N/m \\
b = 100\,N \cdot s/m \\
B = 2.5\,N \cdot m \cdot s \\
J_1 = J_2 = 1\,kg \cdot m^2 \\
m = 1\,kg
\end{gathered}
$$

Find

With the above system,

a. Develop a set of state variable equations. Define the state variable using the general matrix form: $\dot{S} = A \cdot S + B \cdot U$, where A and B are matrices and S is a vector of state variables and U is a vector of inputs. Be careful with labeling since one of the damping coefficients is also labeled B in the figure.

b. Solve the state variable equations using a 4th order Runge-Kutta numerical method for a total simulation time of 20 seconds and assuming that all initial velocities and spring forces are zero and that $f(t) = 10\,N$. Make sure the check that the number of points used in the simulation is adequate.

c. Verify your results using energy conservation. Show that at a given time, t, the total energy that has crossed the boundary through inputs and dampers is equal to the total energy stored in the system through mass/inertias and springs.
d. Use the output matrix form $Y = C \cdot S + D \cdot U$, where Y is the chosen output and C and D are matrices. Use this form to define the contact force between the rack and the pinion using state variable simulation results. Then plot the contact force.
e. Find the input-output equation between the input force and the contact force. One approach of doing this is the use the `ss2tf` function in the `scipy.signal` module.
f. Use the laplace transform approach to solve the input-output equation and compare with results from part d. Comment on how the roots of the characteristic polynomial affect the response.

Solution

Part A

From the above figure, the massless element is denoted as x_1 and the mass denoted by m is x_2.

[2]:
```
# Define symbols
r, k1, k2, b, B, J1, J2, m, F_c, t, s = sp.symbols('r k1 k2 b B J1 J2 m F_c t␣
↪s')
th1, th2 = sp.Function('theta_1')(t), sp.Function('theta_2')(t)
x1, x2 = sp.Function('x1')(t), sp.Function('x2')(t)
f = sp.Function('f')(t)

# Construct equations
eq1 = sp.Eq(0, f - F_c - b*x1.diff() + k1*(x2 - x1))
eq2 = sp.Eq(m*x2.diff(t, 2), k1*(x1 - x2))
eq3 = sp.Eq(J1*th1.diff(t, 2), F_c*r + k2*(th2 - th1))
eq4 = sp.Eq(J2*th2.diff(t, 2), k2*(th1 - th2) - B*th2.diff())
eq5 = sp.Eq(x1.diff(t, 2), r*th1.diff(t, 2))
eqs = [eq1, eq2, eq3, eq4, eq5]
display(*eqs)
```

$$0 = -F_c - b\frac{d}{dt}x_1(t) + k_1\left(-x_1(t) + x_2(t)\right) + f(t)$$

$$m\frac{d^2}{dt^2}x_2(t) = k_1\left(x_1(t) - x_2(t)\right)$$

$$J_1\frac{d^2}{dt^2}\theta_1(t) = F_c r + k_2\left(-\theta_1(t) + \theta_2(t)\right)$$

$$J_2\frac{d^2}{dt^2}\theta_2(t) = -B\frac{d}{dt}\theta_2(t) + k_2\left(\theta_1(t) - \theta_2(t)\right)$$

$$\frac{d^2}{dt^2}x_1(t) = r\frac{d^2}{dt^2}\theta_1(t)$$

Let's clean this up a little by considering the velocities v_1, v_2, ω_1, and ω_2. Also, let's omit considering the coordinate positions in the system and only look to the displacements of the compliance elements. I will define Δx and $\Delta\theta$ as

$$\begin{cases} \Delta x = x_2 - x_1 \\ \Delta\theta = \theta_2 - \theta_1 \end{cases}$$

[3]:
```
d_theta = sp.Function(r'\Delta\theta')(t)
d_x = sp.Function(r'\Delta x')(t)

v1, v2 = sp.Function('v1')(t), sp.Function('v2')(t)
omega1, omega2 = sp.Function(r'\omega_1')(t), sp.Function(r'\omega_2')(t)

subs = [
    (x2 - x1, d_x),
    (th2 - th1, d_theta),
    (x1.diff(), v1),
    (x2.diff(), v2),
    (th1.diff(), omega1),
    (th2.diff(), omega2)
]

eqs = [eq.subs(subs) for eq in eqs]
display(*eqs)
```

$$0 = -F_c - bv_1(t) + k_1\Delta x(t) + f(t)$$

$$m\frac{d}{dt}v_2(t) = -k_1\Delta x(t)$$

$$J_1\frac{d}{dt}\omega_1(t) = F_c r + k_2\Delta\theta(t)$$

$$J_2\frac{d}{dt}\omega_2(t) = -B\omega_2(t) - k_2\Delta\theta(t)$$

$$\frac{d}{dt}v_1(t) = r\frac{d}{dt}\omega_1(t)$$

Now we can solve this system for $\dot{v}_1$, $\dot{v}_2$, $\dot{\omega}_1$ $\dot{\omega}_2$. Additionally, I will solve for F_c to be used later.

[4]:
```
state_sol = sp.solve(eqs, (v1.diff(), v2.diff(), omega1.diff(), omega2.diff(),␣
 ↪F_c), dict=True)[0]
for key, value in state_sol.items():
    display(sp.Eq(key, value))
```

$$F_c = -bv_1(t) + k_1\Delta x(t) + f(t)$$

$$\frac{d}{dt}\omega_1(t) = -\frac{brv_1(t)}{J_1} + \frac{k_1 r\Delta x(t)}{J_1} + \frac{k_2\Delta\theta(t)}{J_1} + \frac{rf(t)}{J_1}$$

$$\frac{d}{dt}\omega_2(t) = -\frac{B\omega_2(t)}{J_2} - \frac{k_2\Delta\theta(t)}{J_2}$$

$$\frac{d}{dt}v_1(t) = -\frac{br^2v_1(t)}{J_1} + \frac{k_1 r^2\Delta x(t)}{J_1} + \frac{k_2 r\Delta\theta(t)}{J_1} + \frac{r^2 f(t)}{J_1}$$

$$\frac{d}{dt}v_2(t) = -\frac{k_1 \Delta x(t)}{m}$$

Notice in the above solution that we don't have any terms that include ω_1. This is due to the direct algebraic relationship and the order in which **sympy** solved the system. Therefore, we can ignore the $\dot{\omega}_1$ equation. Also, the solution for F_c is not an ODE so it isn't directly a part of the state space solution.

[5]:
```python
# Getting the final five equations
eq1 = sp.Eq(d_x.diff(), v2 - v1)
eq2 = sp.Eq(d_theta.diff(), omega2 - v1/r)
eq3 = sp.Eq(v1.diff(), state_sol[v1.diff()])
eq4 = sp.Eq(v2.diff(), state_sol[v2.diff()])
eq5 = sp.Eq(omega2.diff(), state_sol[omega2.diff()])
eqs = [eq1, eq2, eq3, eq4, eq5]
display(*eqs)
```

$$\frac{d}{dt}\Delta x(t) = -v_1(t) + v_2(t)$$

$$\frac{d}{dt}\Delta\theta(t) = \omega_2(t) - \frac{v_1(t)}{r}$$

$$\frac{d}{dt}v_1(t) = -\frac{br^2 v_1(t)}{J_1} + \frac{k_1 r^2 \Delta x(t)}{J_1} + \frac{k_2 r \Delta\theta(t)}{J_1} + \frac{r^2 f(t)}{J_1}$$

$$\frac{d}{dt}v_2(t) = -\frac{k_1 \Delta x(t)}{m}$$

$$\frac{d}{dt}\omega_2(t) = -\frac{B\omega_2(t)}{J_2} - \frac{k_2 \Delta\theta(t)}{J_2}$$

[6]:
```python
# Putting it in the state space form
S_dot = sp.Matrix([eq.lhs for eq in eqs])
S = S_dot.integrate(t)
rhs = [eq.rhs for eq in eqs]
A, b = sp.linear_eq_to_matrix(rhs, list(S))
B = -b/f
U = sp.Matrix([f])
ans = sp.Eq(S_dot, sp.Add(sp.MatMul(A, S), sp.MatMul(B, U)))  # doing this␣
↪makes it not simplify
ans
```

Answer

$$\begin{bmatrix} \frac{d}{dt}\Delta x(t) \\ \frac{d}{dt}\Delta\theta(t) \\ \frac{d}{dt}v_1(t) \\ \frac{d}{dt}v_2(t) \\ \frac{d}{dt}\omega_2(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{r^2}{J_1} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} f(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -\frac{1}{r} & 0 & 1 \\ \frac{k_1 r^2}{J_1} & \frac{k_2 r}{J_1} & -\frac{br^2}{J_1} & 0 & 0 \\ -\frac{k_1}{m} & 0 & 0 & 0 & 0 \\ 0 & -\frac{k_2}{J_2} & 0 & 0 & -\frac{B}{J_2} \end{bmatrix} \begin{bmatrix} \Delta x(t) \\ \Delta\theta(t) \\ v_1(t) \\ v_2(t) \\ \omega_2(t) \end{bmatrix}$$