PS 2

Keyi Zhang

Issued: Tue 09/25/2025

Due: Thu 100/07/2025

Problem 1:

In class, we have often come across derivatives of the form shown below:

$$\frac{\partial}{\partial x}(x^T G x + v^T x)$$

For a constant symmetric matrix **G** and a constant vector v, simplify explicitly the above expression containing a variable vector x. How would your simplified answer change if **G** is not symmetric?

[Hint: Start with a vector x with 2, or maybe 3, components, (and appropriate G and v) to do the explicit matrix algebra.]

Answer 1:

Know:

x: variable vector

G: constant symmetric matrix

v: constant vector

In the derivatives,

Quadratic term:

$$x^T G x$$

Linear term:

$$v^T x$$

Assume That:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad G = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Quadratic term:

$$x^T G x = [x_1, x_2] \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = a x_1^2 + (b+c) x_1 x_2 + d x_2^2$$

Linear term:

$$v^T x = v_1 x_1 + v_2 x_2$$

Compute the partial derivatives with respect to x_1 and $x2$ separately:

Partial derivative w.r.t x_1 :

$$\frac{\partial}{\partial x_1}(x^T G x + v^T x) = 2a x_1 + (b+c) x_2 + v_1$$

Partial derivative w.r.t x_2 :

$$\frac{\partial}{\partial x_2}(x^T G x + v^T x) = 2d x_2 + (b+c) x_1 + v_2$$

Gradient vector:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 2a x_1 + (b+c) x_2 + v_1 \\ 2d x_2 + (b+c) x_1 + v_2 \end{bmatrix}$$

As:

$$G = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad G^T = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

$$(G + G^T)x + v = \begin{bmatrix} 2a x_1 + (b+c) x_2 + v_1 \\ (b+c) x_1 + 2d x_2 + v_2 \end{bmatrix} = \begin{bmatrix} 2a x_1 + (b+c) x_2 + v_1 \\ 2d x_2 + (b+c) x_1 + v_2 \end{bmatrix} = \frac{\partial f}{\partial x}$$

$$\frac{\partial f}{\partial x} = (G + G^T)x + v$$

If G is symmetric,

$G = G^t$

Then:

$$G + G^T = 2G$$
$$\frac{\partial f}{\partial x} = 2Gx + v$$

In sum:

G is symmetric	$\frac{\partial f}{\partial x} = 2Gx + v$
G is not symmetric	$\frac{\partial f}{\partial x} = (G + G^T)x + v$

Problem 2:

Consider the dynamic equations that we have examined for a two degree-of-freedom planar manipulator in the vertical plane with two rotational joints:

$$H(q)\ddot{}q + C(q, \dot{}q)\dot{}q + g(q) = \tau$$

where:

$$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}, H = \begin{bmatrix} H_{11} & H_{12} \\ H_{12} & H_{22} \end{bmatrix}, C(q, \dot{q})\dot{q} = \begin{bmatrix} -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2 \\ h\dot{q}_2^1 \end{bmatrix}$$

$$\begin{aligned} H_{11} &= m_1 l_2 c_1 + I_1 + m_2(l_1^2 + l_2 c_2 + 2 l_1 l c_2 \cos q_2) + I_2 \\ H_{22} &= m_2 l_2 c_2 + I_2 \\ H_{12} &= m_2 l_1 l c_2 \cos q_2 + m_2 l_2 c_2 + I_2 \\ h &= m_2 l_1 l c_2 \sin q_2 \end{aligned}$$

(a) The Coriolis and centripetal torque vector C(q, ˙q) ˙q is a uniquely defined physical quantity. Show that, however, given C(q, ˙q) ˙q alone, a unique solution cannot be obtained for the matrix C(q, ˙q). Identify two possible solutions of C(q, ˙q).
(b) You have been given the Coriolis and centripetal torque vector C(q, ˙q) ˙q. In addition, now use the condition that ˙H −2C is a skew symmetric matrix to solve for the unique value of C(q, ˙q).
(c) Verify that C(q, ˙q) that you found above satisfies the following component-wise definition for the case of a n-link manipulator:

$$C_{ij} = \frac{1}{2}\dot{H}_{ij} + \frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i}\right)\dot{q}_k$$

Answer:

(a)

Know:

$$H(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \tau$$

$$C(q, \dot{q})\dot{q} = \begin{bmatrix} -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2 \\ h\dot{q}_1^2 \end{bmatrix}, \quad q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}, \quad H = \begin{bmatrix} H_{11} & H_{12} \\ H_{12} & H_{22} \end{bmatrix}$$

Let:

$$C(q,\dot{q}) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Then:

$$C(q,\dot{q})\dot{q} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2 \\ h\dot{q}_1^2 \end{bmatrix}$$

By multiplying the 2×2 Coriolis matrix C with the joint velocity vector $\dot{q}$, we can write the two resulting **2 scalar equations**:

$$C_{11}\dot{q}_1 + C_{12}\dot{q}_2 = -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2$$

$$C_{21}\dot{q}_1 + C_{22}\dot{q}_2 = h\dot{q}_1^2$$

Analyze the number of equations versus unknowns,

There are **4 unknowns** in total: $C11, C12, C21, C22$

The number of equations is less than the number of unknowns, so given C(q, ˙q) ˙q alone, a unique solution cannot be obtained for the matrix C(q, ˙q).

(b)

know:

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

$$C(q,\dot{q})\dot{q} = \begin{bmatrix} -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2 \\ h\dot{q}_1^2 \end{bmatrix}, \quad q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}, \quad H = \begin{bmatrix} H_{11} & H_{12} \\ H_{12} & H_{22} \end{bmatrix}$$

$$\begin{aligned} H_{11} &= m_1 l_2 c_1 + I_1 + m_2(l_1^2 + l_2 c_2 + 2 l_1 l c_2 \cos q_2) + I_2 \\ H_{22} &= m_2 l_2 c_2 + I_2 \\ H_{12} &= m_2 l_1 l c_2 \cos q_2 + m_2 l_2 c_2 + I_2 \\ h &= m_2 l_1 l c_2 \sin q_2 \end{aligned}$$

Then:

$$h(q_2) = m_2 l_1 l_{c2} \sin q_2, \qquad \dot{q} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}.$$

Take the partial derivative:

$$\frac{\partial H_{11}}{\partial q_2} = -2h,$$
$$\frac{\partial H_{12}}{\partial q_2} = -h,$$
$$\frac{\partial H_{22}}{\partial q_2} = 0$$

Thus, we can solve the $\dot{H}$:

$$\dot{H} = \begin{bmatrix} -2h\,\dot{q}_2 & -h\,\dot{q}_2 \\ -h\,\dot{q}_2 & 0 \end{bmatrix}$$

Because $\dot{H} - 2C$ is skew symmetric, so:

$$C + C^T = \dot{H}$$

$$C(q, \dot{q}) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Thus, we can solve two terms in $C(q, \dot{q})$:

$$2C_{11} = -2h\,\dot{q}_2 \quad \Rightarrow \quad C_{11} = -h\,\dot{q}_2,$$
$$C_{22} = 0 \quad \Rightarrow \quad C_{22} = 0,$$
$$C_{12} + C_{21} = -h\,\dot{q}_2$$

$$C(q, \dot{q}) = \begin{bmatrix} -h\,\dot{q}_2 & C_{12} \\ C_{21} & 0 \end{bmatrix}$$

Known:

$$C(q, \dot{q})\dot{q} = \begin{bmatrix} -2h\dot{q}_1\dot{q}_2 - h\dot{q}_2^2 \\ h\dot{q}_1^2 \end{bmatrix},$$

Then, we can have the following equations:

$$C_{11}\dot{q}_1 + C_{12}\dot{q}_2 = -2h\,\dot{q}_1\dot{q}_2 - h\,\dot{q}_2^2$$
$$C_{21}\dot{q}_1 + C_{22}\dot{q}_2 = h\,\dot{q}_1^2$$

1. for the first term, with $C_{11} = -2h\dot{q}$, we can have:

   $((-h\,\dot{q}_2)\,\dot{q}_1 + C_{12}\,\dot{q}_2 = -2h\,\dot{q}_1\dot{q}_2 - h\,\dot{q}_2^2)$

$C_{12}\,\dot{q}_2 = -2h\,\dot{q}_1\dot{q}_2 - h\,\dot{q}_2^2 + h\,\dot{q}_1\dot{q}_2 = -h\,\dot{q}_2(\dot{q}_1 + \dot{q}_2)$

$C_{12} = -h(\dot{q}_1 + \dot{q}_2)$

2. for the first term, with $C_{22} = 0$, we can have:

   $C_{21}\,\dot{q}_1 = h\,\dot{q}_1^2$

   $C_{21} = h\,\dot{q}_1$

We can have the result we had before by adding C_12 and $C21$:

$$C_{12} + C_{21} = -h(\dot{q}_1 + \dot{q}_2) + h\,\dot{q}_1 = -h\,\dot{q}_2$$

Thus, we can solve each term in $C(q, \dot{q})$:

$$C(q,\dot{q}) = \begin{bmatrix} -h\,\dot{q}_2 & -h(\dot{q}_1 + \dot{q}_2) \\ h\,\dot{q}_1 & 0 \end{bmatrix}$$

(c)

Known:

$$C_{ij} = \frac{1}{2}\dot{H}_{ij} + \frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i}\right)\dot{q}_k$$

$$\dot{H} = \begin{bmatrix} -2h\,\dot{q}_2 & -h\,\dot{q}_2 \\ -h\,\dot{q}_2 & 0 \end{bmatrix}$$

$$C(q,\dot{q}) = \begin{bmatrix} -h\,\dot{q}_2 & -h(\dot{q}_1 + \dot{q}_2) \\ h\,\dot{q}_1 & 0 \end{bmatrix}$$

Based on:

$$\begin{aligned} H_{11} &= m_1 l_2 c_1 + I_1 + m_2(l_1^2 + l_2 c_2 + 2 l_1 l c_2 \cos q_2) + I_2 \\ H_{22} &= m_2 l_2 c_2 + I_2 \\ H_{12} &= m_2 l_1 l c_2 \cos q_2 + m_2 l_2 c_2 + I_2 \end{aligned}$$

We can know that:

H_{11} depends on q2, but does not depend on q1

H_{22} does not depend on q1 or q2

H_{12} depends on q2, but does not depend on q1

Then we can proof each element in $C(q, \dot{q})$:

For C_{11}

$$C_{11} = \frac{1}{2}\dot{H}_{11} + \frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{1k}}{\partial q_1} - \frac{\partial H_{k1}}{\partial q_1}\right)\dot{q}_k$$

1. first term: $\frac{1}{2}\dot{H}_{11} = -h\,\dot{q}_2$
2. second term: $\frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{1k}}{\partial q_1} - \frac{\partial H_{k1}}{\partial q_1}\right)\dot{q}_k = 0 \quad (\text{because } H_{11} \text{ does not depend on } q_1)$

$C_{11} = -h\,\dot{q}_2$

For C_{12}

$$C_{12} = \frac{1}{2}\dot{H}_{12} + \frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{1k}}{\partial q_2} - \frac{\partial H_{k2}}{\partial q_1}\right)\dot{q}_k$$

1. first term: $\frac{1}{2}\dot{H}_{12} = -\frac{1}{2}h\dot{q}_2$$
2. second term: $\frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{1k}}{\partial q_2} - \frac{\partial H_{k2}}{\partial q_1}\right)\dot{q}k = \frac{1}{2}((-2h)\dot{q}_1 + (-h)\dot{q}_2)$

$C_{12} = -h(\dot{q}_1 + \dot{q}_2)$

For C_{21}

$$C_{21} = \frac{1}{2}\dot{H}_{21} + \frac{1}{2}\sum_{k=1}^{2}\ \left(\frac{\partial H_{2k}}{\partial q_1} - \frac{\partial H_{k1}}{\partial q_2}\right)\ \dot{q}_k$$

1. first term: $\frac{1}{2}\dot{H}_{21} = -\frac{1}{2}h\dot{q}_2$
2. second term: $\frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{2k}}{\partial q_2} - \frac{\partial H_{k1}}{\partial q_1}\right)\dot{q}k = h\dot{q}_1 + \frac{1}{2}h\dot{q}_2$

$C_{21} = h\dot{q}_1$

For C_{22}

$$C_{22} = \frac{1}{2}\dot{H}_{22} + \frac{1}{2}\sum_{k=1}^{2}\ \left(\frac{\partial H_{2k}}{\partial q_2} - \frac{\partial H_{k2}}{\partial q_2}\right)\dot{q}_k$$

1. first term: $\frac{1}{2}\dot{H}_{22}$ = 0 (H_{22} does not depend on q1 or q2)
2. second term: $\frac{1}{2}\sum_{k=1}^{2}\left(\frac{\partial H_{2k}}{\partial q_2} - \frac{\partial H_{k2}}{\partial q_2}\right)\dot{q}_k$ = 0 (H_{22} does not depend on q1 or q2)

$C_2 2$ = 0

In sum:

$$C(q, \dot{q}) = \begin{bmatrix} -h\,\dot{q}_2 & -h(\dot{q}_1 + \dot{q}_2) \\ h\,\dot{q}_1 & 0 \end{bmatrix}$$

Problem 3:

(a) Consider a 2-link manipulator, in the horizontal plane. Assume that the manipulator is subject to a unit force at its endpoint, pointing towards its "shoulder".
Compute and plot the joint torques required so that the manipulator does not move, as a function of configuration.
(b) Same question as (a), but in the vertical plane. (You may need some extra numerical assumptions, please make them simple and explicit.)

(a)

Define:

joint angles: q_1, q_2

link lengths: l_1, $l2$

The x, y are:

$$x = \cos\theta_1 + \cos(\theta_1 + \theta_2)$$
$$y = \sin\theta_1 + \sin(\theta_1 + \theta_2)$$

The planar Jacobian is:

$$J(q) = \begin{bmatrix} -l_1 \sin q_1 - l_2 \sin(q_1 + q_2) & -l_2 \sin(q_1 + q_2) \\ l_1 \cos q_1 + l_2 \cos(q_1 + q_2) & l_2 \cos(q_1 + q_2) \end{bmatrix}$$

End position:

$$p(q) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{bmatrix}.$$

The unit force at the endpoint pointing towards the shoulder is:

$$\mathbf{F} = -\frac{\mathbf{p}}{\|\mathbf{p}\|} = -\frac{(x,\, y)}{\sqrt{x^2 + y^2}}$$

The required joint torques to resist the external force are:

$$\boldsymbol{\tau} = J(q)^\top \mathbf{F}$$

The calculation of torques are:

$$\tau_1 = J_{11}F_x + J_{21}F_y$$
$$\tau_2 = J_{12}F_x + J_{22}F_y$$

For $torque1$, we replace items with: $x = \cos\theta_1 + \cos(\theta_1+\theta_2), y = \sin\theta_1 + \sin(\theta_1+\theta_2)$

$$\begin{gathered}
\tau_1 = [-\sin\theta_1 - \sin(\theta_1+\theta_2)]\left(\frac{-x}{x^2+y^2}\right) + [\cos\theta_1 + \cos(\theta_1+\theta_2)]\left(\frac{-y}{x^2+y^2}\right) \\
= \frac{1}{x^2+y^2}\Big[(-\sin\theta_1 - \sin(\theta_1+\theta_2))(-x) + (\cos\theta_1 + \cos(\theta_1+\theta_2))(-y)\Big] \\
= \frac{1}{x^2+y^2}\Big[(\sin\theta_1 + \sin(\theta_1+\theta_2))x - (\cos\theta_1 + \cos(\theta_1+\theta_2))y\Big] \\
= \frac{1}{x^2+y^2}\Big[(\sin\theta_1 + \sin(\theta_1+\theta_2))(\cos\theta_1 + \cos(\theta_1+\theta_2)) \\
= \frac{1}{x^2+y^2}\Big[0\Big] \\
= 0
\end{gathered}$$

For $torque2$, we replace items with: $x = \cos\theta_1 + \cos(\theta_1+\theta_2), y = \sin\theta_1 + \sin(\theta_1+\theta_2)$

$$\begin{gathered}
\tau_2 = [-\sin(\theta_1+\theta_2)]\left(\frac{-x}{x^2+y^2}\right) + [\cos(\theta_1+\theta_2)]\left(\frac{-y}{x^2+y^2}\right) \\
= \tau_2 = \frac{1}{x^2+y^2}\Big[\sin(\theta_1+\theta_2)(\cos\theta_1 + \cos(\theta_1+\theta_2)) - \cos(\theta_1+\theta_2)(\sin\theta_1 + \sin(\theta_1+\theta_2))\Big] \\
= \frac{1}{x^2+y^2}\Big[\sin(\theta_1+\theta_2)\cos\theta_1 - \cos(\theta_1+\theta_2)\sin\theta_1\Big] \\
= \frac{1}{x^2+y^2}\,\sin\big[(\theta_1+\theta_2) - \theta_1\big] \\
= \frac{1}{x^2+y^2}\,\sin\theta_2
\end{gathered}$$

Simplify x^2+y^2:

$$\begin{gathered}
x^2+y^2 = [\cos\theta_1 + \cos(\theta_1+\theta_2)]^2 + [\sin\theta_1 + \sin(\theta_1+\theta_2)]^2 \\
= \cos^2\theta_1 + 2\cos\theta_1\cos(\theta_1+\theta_2) + \cos^2(\theta_1+\theta_2) + \sin^2\theta_1 + 2\sin\theta_1\sin(\theta_1+\theta_2) + \sin^2(\theta_1+\theta_2) \\
= (\cos^2\theta_1 + \sin^2\theta_1) + [\cos^2(\theta_1+\theta_2) + \sin^2(\theta_1+\theta_2)] + 2[\cos\theta_1\cos(\theta_1+\theta_2) + \sin\theta_1\sin(\theta_1+\theta_2)] \\
= 1 + 1 + 2\cos[(\theta_1+\theta_2) - \theta_1] \\
= 2 + 2\cos\theta_2
\end{gathered}$$

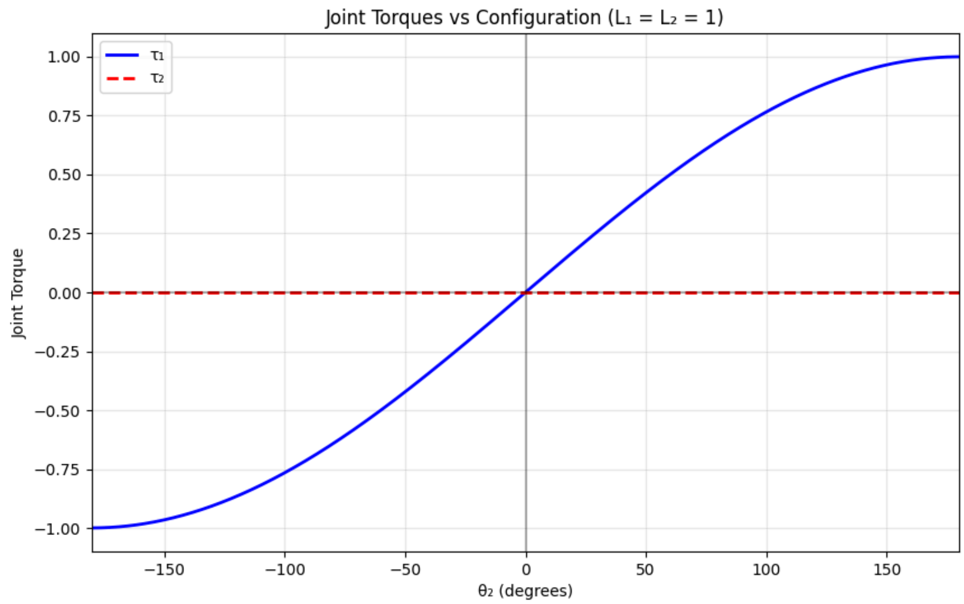
$$\begin{gathered}
\tau_2 = \frac{\sin\theta_2}{\sqrt{2+2\cos\theta_2}} \\
= \sin\left(\frac{\theta_2}{2}\right)
\end{gathered}$$

Thus:

$$\tau_1 = 0$$
$$\tau_2 = \sin\left(\frac{\theta_2}{2}\right)$$

Compute and plot the joint torques:

```
import numpy as np
import matplotlib.pyplot as plt

#design parameters
L1, L2 = 1.0, 1.0
theta2 = np.linspace(-np.pi, np.pi, 1000)

theta2 = theta2[theta2 != 0]

d = np.sqrt(L1**2 + L2**2 + 2*L1*L2*np.cos(theta2))
tau1 = (L1 * L2 * np.sin(theta2)) / d
tau2 = np.zeros_like(tau1)

plt.figure(figsize=(10, 6))
plt.plot(np.degrees(theta2), tau1, 'b-', linewidth=2, label='τ₁')
plt.plot(np.degrees(theta2), tau2, 'r--', linewidth=2, label='τ₂')
plt.xlabel('θ₂ (degrees)')
plt.ylabel('Joint Torque')
plt.title('Joint Torques vs Configuration (L₁ = L₂ = 1)')
plt.grid(True, alpha=0.3)
plt.legend()
plt.axhline(y=0, color='k', linestyle='-', alpha=0.3)
plt.axvline(x=0, color='k', linestyle='-', alpha=0.3)
plt.xlim(-180, 180)
plt.show()
```



(b)

Known:

$$\boldsymbol{\tau} = J(q)^{\top}\mathbf{F}$$

$$\tau_1 = 0$$
$$\tau_2 = \sin\left(\frac{\theta_2}{2}\right)$$

Design Parameters：

Link lengths: L1=L2=1 m

Link masses: m1=m2=1 kg

Gravitational acceleration: g=9.81 m/s²

Unit force magnitude: F=1 N

The total torque is:

$$\tau_{\text{total}} = \tau_{\text{force}} + \tau_{\text{gravity}}$$

Compute the gravity torque:

1. The center of the mass:

$$\mathbf{l}_{c1} = \left(\frac{L_1}{2}\cos\theta_1, \frac{L_1}{2}\sin\theta_1\right)$$

$$\mathbf{l}_{c2} = \left(L_1 \cos\theta_1 + \frac{L_2}{2}\cos(\theta_1+\theta_2), L_1 \sin\theta_1 + \frac{L_2}{2}\sin(\theta_1+\theta_2) \right)$$

2. Mass

$$\mathbf{F}_{g1} = (0, -m_1 g)$$

$$\mathbf{F}_{g2} = (0, -m_2 g)$$

3. Jacobian matrices for the centers of mass

$$J_{c1} = \begin{bmatrix} -\frac{L_1}{2}\sin\theta_1 & 0 \\ \frac{L_1}{2}\cos\theta_1 & 0 \end{bmatrix}$$

$$J_{c2} = \begin{bmatrix} -L_1\sin\theta_1 - \frac{L_2}{2}\sin(\theta_1+\theta_2) & -\frac{L_2}{2}\sin(\theta_1+\theta_2) \\ L_1\cos\theta_1 + \frac{L_2}{2}\cos(\theta_1+\theta_2) & \frac{L_2}{2}\cos(\theta_1+\theta_2) \end{bmatrix}$$

4. The gravity torque:

$$\boldsymbol{\tau}_{\text{gravity}} = J_{c1}^T \mathbf{F}_{g1} + J_{c2}^T \mathbf{F}_{g2}$$

$$\begin{aligned} \tau_{\text{gravity},1} &= -9.81 \left(\frac{1}{2}\cos\theta_1 + \cos\theta_1 + \frac{1}{2}\cos(\theta_1+\theta_2) \right) \\ &= -9.81 \left(\frac{3}{2}\cos\theta_1 + \frac{1}{2}\cos(\theta_1+\theta_2) \right) \end{aligned}$$

$$\tau_{\text{gravity},2} = -9.81 \left(\frac{1}{2}\cos(\theta_1+\theta_2) \right)$$

Compute the total torque is:

$$\tau_{\text{total}} = \tau_{\text{force}} + \tau_{\text{gravity}}$$

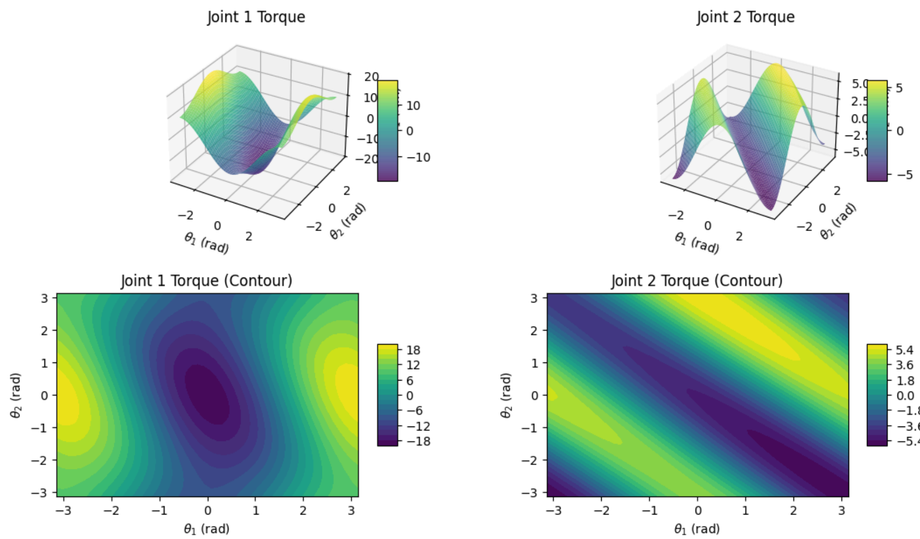
$$\tau_1 = -9.81 \left(\frac{3}{2}\cos\theta_1 + \frac{1}{2}\cos(\theta_1+\theta_2) \right)$$

$$\tau_2 = \sin\left(\frac{\theta_2}{2}\right) - 9.81 \left(\frac{1}{2}\cos(\theta_1+\theta_2) \right)$$

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Design Parameters
L1, L2 = 1.0, 1.0
m1, m2 = 1.0, 1.0
g = 9.81
```

```
theta1 = np.linspace(-np.pi, np.pi, 50)
theta2 = np.linspace(-np.pi, np.pi, 50)
Theta1, Theta2 = np.meshgrid(theta1, theta2)

tau_force_1 = np.zeros_like(Theta1)
tau_force_2 = np.sin(Theta2/2)

tau_gravity_1 = -g * (1.5 * np.cos(Theta1) + 0.5 * np.cos(Theta1 + Theta2))
tau_gravity_2 = -g * (0.5 * np.cos(Theta1 + Theta2))

tau_total_1 = tau_force_1 + tau_gravity_1
tau_total_2 = tau_force_2 + tau_gravity_2

fig = plt.figure(figsize=(15, 12))

ax1 = fig.add_subplot(2, 2, 1, projection='3d')
surf1 = ax1.plot_surface(Theta1, Theta2, tau_total_1, cmap='viridis', alpha=0.8)
ax1.set_xlabel(r'$\theta_1$ (rad)')
ax1.set_ylabel(r'$\theta_2$ (rad)')
ax1.set_zlabel(r'$\tau_1$ (Nm)')
ax1.set_title('Joint 1 Torque')
fig.colorbar(surf1, ax=ax1, shrink=0.5, aspect=5)

ax2 = fig.add_subplot(2, 2, 2, projection='3d')
surf2 = ax2.plot_surface(Theta1, Theta2, tau_total_2, cmap='viridis', alpha=0.8)
ax2.set_xlabel(r'$\theta_1$ (rad)')
ax2.set_ylabel(r'$\theta_2$ (rad)')
ax2.set_zlabel(r'$\tau_2$ (Nm)')
ax2.set_title('Joint 2 Torque')
fig.colorbar(surf2, ax=ax2, shrink=0.5, aspect=5)

ax3 = fig.add_subplot(2, 2, 3)
contour1 = ax3.contourf(Theta1, Theta2, tau_total_1, 20, cmap='viridis')
ax3.set_xlabel(r'$\theta_1$ (rad)')
ax3.set_ylabel(r'$\theta_2$ (rad)')
ax3.set_title('Joint 1 Torque (Contour)')
fig.colorbar(contour1, ax=ax3, shrink=0.5, aspect=5)

ax4 = fig.add_subplot(2, 2, 4)
contour2 = ax4.contourf(Theta1, Theta2, tau_total_2, 20, cmap='viridis')
ax4.set_xlabel(r'$\theta_1$ (rad)')
ax4.set_ylabel(r'$\theta_2$ (rad)')
ax4.set_title('Joint 2 Torque (Contour)')
fig.colorbar(contour2, ax=ax4, shrink=0.5, aspect=5)
```

```
plt.subplots_adjust(hspace=0.3, wspace=0.3)
plt.show()
```



Problem 4:

Consider a 2-link manipulator in the vertical plane. Choosing arbitrary initial conditions, simulate the dynamics of the manipulator. Here, we assume no joint torque, no friction, only gravity, so that you can imagine this manipulator as a 2-link free pendulum.

Know:

1. The center of the mass:

$$\mathbf{l}_{c1} = \left(\frac{L_1}{2}\cos\theta_1, \frac{L_1}{2}\sin\theta_1\right)$$

$$\mathbf{l}_{c2} = \left(L_1\cos\theta_1 + \frac{L_2}{2}\cos(\theta_1+\theta_2), L_1\sin\theta_1 + \frac{L_2}{2}\sin(\theta_1+\theta_2)\right)$$

2. Compute the torque and the $\ddot{q}$

$$\tau = D(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) = 0$$
$$\ddot{q} = D^{-1}(q)\,\big(-C(q,\dot{q})\,\dot{q} - G(q)\big)$$

3. Compute the gravity

$$G_1 = (m_1 l_{c1} + m_2 l_1) g\cos(q_1) + m_2 l_{c2} g\cos(q_1+q_2)$$
$$G_2 = m_2 l_{c2} g\cos(q_1+q_2)$$

4. Compute the Inertia matrix

$$H(q) = \begin{bmatrix} H_{11}(q) & H_{12}(q) \\ H_{21}(q) & H_{22}(q) \end{bmatrix}$$

$$\begin{gathered} H_{11}(q) = I_1 + I_2 + m_1 l_{c1}^2 + m_2 \left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} \cos q_2 \right) \\ H_{12}(q) = I_2 + m_2 \left(l_{c2}^2 + l_1 l_{c2} \cos q_2 \right) \\ H_{21}(q) = I_2 + m_2 \left(l_{c2}^2 + l_1 l_{c2} \cos q_2 \right) \\ H_{22}(q) = I_2 + m_2 l_{c2}^2 \end{gathered}$$

5. Compute the Coriolis C:

$$C(q, \dot{q})\dot{q} = \begin{bmatrix} -m_2 l_1 l_{c2} \sin q_2 \left(2\dot{q}_1 \dot{q}_2 + \dot{q}_2^2 \right) \\ m_2 l_1 l_{c2} \sin q_2 \, \dot{q}_1^2 \end{bmatrix}$$

6. Design Parameters:

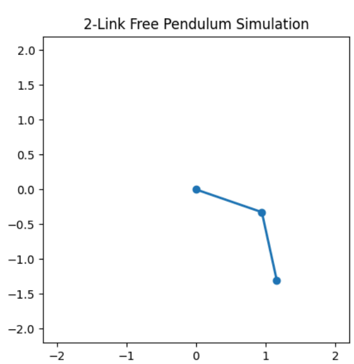
$$\begin{gathered} m1, m2 = 1.0, 1.0 \\ l1, l2 = 1.0, 1.0 \\ lc1, lc2 = 0.5, 0.5 \\ I1, I2 = 0.1, 0.1 \\ g = 9.81 \end{gathered}$$

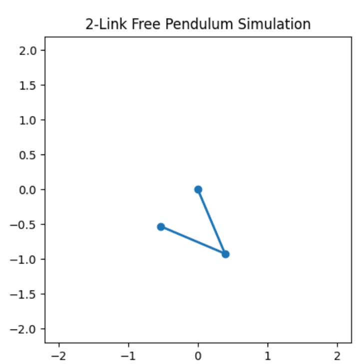
7. Simulation

The simulation computes the motion of the two-link manipulator by first expressing the centers of mass, then deriving the inertia matrix $H(q)$, Coriolis term $C(q, \dot{q})\dot{q}$, and gravity vector $G(q)$. With no external torque ($\tau = 0$), the joint accelerations are calculated as

$$\ddot{q} = H^{-1}(q)\big(- C(q, \dot{q})\dot{q} - G(q) \big)$$

and numerically integrated over time to obtain the joint angles $q_1(t), q_2(t)$ and the corresponding positions of the links.





```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

# Design Parameters
m1, m2 = 1.0, 1.0
l1, l2 = 1.0, 1.0
lc1, lc2 = 0.5, 0.5
I1, I2 = 0.1, 0.1
g = 9.81

def dynamics(t, y):
    q1, q2, q1dot, q2dot = y

    d11 = I1 + I2 + m1*lc1**2 + m2*(l1**2 + lc2**2 + 2*l1*lc2*np.cos(q2))
    d12 = I2 + m2*(lc2**2 + l1*lc2*np.cos(q2))
    d21 = d12
    d22 = I2 + m2*lc2**2
    D = np.array([[d11, d12], [d21, d22]])

    h = -m2*l1*lc2*np.sin(q2)
    c1 = h*q2dot*(2*q1dot + q2dot)
    c2 = h*q1dot**2
    C = np.array([c1, c2])

    g1 = (m1*lc1 + m2*l1)*g*np.cos(q1) + m2*lc2*g*np.cos(q1 + q2)
    g2 = m2*lc2*g*np.cos(q1 + q2)
    G = np.array([g1, g2])

    qddot = np.linalg.solve(D, -C - G)

    return [q1dot, q2dot, qddot[0], qddot[1]]

q1_0 = np.pi / 2
q2_0 = np.pi / 3
q1dot_0 = 0.0
q2dot_0 = 0.0
y0 = [q1_0, q2_0, q1dot_0, q2dot_0]

t_span = (0, 10)
t_eval = np.linspace(0, 10, 1000)
sol = solve_ivp(dynamics, t_span, y0, t_eval=t_eval, rtol=1e-8, atol=1e-8)

q1 = sol.y[0]
```

```
q2 = sol.y[1]
x1 = l1 * np.sin(q1)
y1 = -l1 * np.cos(q1)
x2 = x1 + l2 * np.sin(q1 + q2)
y2 = y1 - l2 * np.cos(q1 + q2)

fig, ax = plt.subplots(figsize=(8, 8))
ax.set_xlim(-2.2, 2.2)
ax.set_ylim(-2.2, 2.2)
ax.set_aspect('equal')
ax.grid(True)
ax.set_title("2-Link Free Pendulum Simulation")
line, = ax.plot([], [], 'o-', lw=2, markersize=8)

def update(frame):
    line.set_data([0, x1[frame], x2[frame]], [0, y1[frame], y2[frame]])
    return line,

ani = FuncAnimation(fig, update, frames=len(t_eval), interval=20, blit=True)
plt.show()
```