# Solution of Robot Learning
# Written Part

ZHANG SHUTAO 1155238521

February 17, 2026

## 1 Problem 1: Inverted Pendulum Nonlinear Dynamics (Euler–Lagrange Derivation)

### 1.1 Model Setup and some assumptions

We consider a torque-actuated inverted pendulum with mass $m$, length $l$. And its point mass $m$ at the end of a rigid, massless rod of length $l$. The inverted pendulum rotates around a fixed pivot. Let $\theta$ denote the generalized coordinate (pendulum angle), and let the control input be the applied torque $u$ at the pivot. We also include viscous damping at the joint, modeled by a torque $-b\dot{\theta}$, where $b > 0$ is the damping coefficient.

### 1.2 Kinematics

Place the pivot at the origin and describe the position of the mass in Cartesian coordinates as:

$$x = l\sin\theta, \qquad y = l\cos\theta \tag{1}$$

Taking time derivatives gives:

$$\dot{x} = l\cos\theta,\dot{\theta}, \qquad \dot{y} = -l\sin\theta\,\dot{\theta}. \tag{2}$$

Hence the squared speed is:

$$v^2 = \dot{x}^2 + \dot{y}^2 = l^2\dot{\theta}^2\left(\cos^2\theta + \sin^2\theta\right) = l^2\dot{\theta}^2. \tag{3}$$

### 1.3 Kinetic Energy and Potential Energy

The kinetic energy can be written as:

$$T = \frac{1}{2}mv^2 = \frac{1}{2}ml^2\dot{\theta}^2 \tag{4}$$

Define the (effective) moment of inertia about the pivot:

$$I = ml^2, \tag{5}$$

so that

$$T = \frac{1}{2} I \dot{\theta}^2. \tag{6}$$

The gravitational potential energy can be written as:

$$V = mgy = mgl \cos\theta, \tag{7}$$

where $g$ is the gravitational acceleration. (Any constant offset in $V$ does not affect the equations of motion.)

## 1.4 Euler–Lagrange Equation with Generalized Force

Firstly, the Lagrangian is:

$$\mathcal{L}(\theta, \dot{\theta}) = T - V = \frac{1}{2} I \dot{\theta}^2 - mgl \cos\theta. \tag{8}$$

And for generalized coordinate $\theta$, the Euler–Lagrange equation is

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}}\right) - \frac{\partial \mathcal{L}}{\partial \theta} = Q_\theta, \tag{9}$$

where $Q_\theta$ is the generalized torque applied along $\theta$. Here we include the control torque $u$ and viscous damping torque $-b\dot{\theta}$:

$$Q_\theta = u - b\dot{\theta}. \tag{10}$$

And expand the equation9, compute each term of the equation 9:

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}} = \frac{\partial}{\partial \dot{\theta}}\left(\frac{1}{2} I \dot{\theta} - mgl \cos\theta\right) = I\dot{\theta}, \tag{11}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}}\right) = \frac{d}{dt}(I\dot{\theta}) = I\ddot{\theta}, \tag{12}$$

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial}{\partial \theta}\left(\frac{1}{2} I \dot{\theta}^2 - mgl \cos\theta\right) = mgl \sin\theta. \tag{13}$$

Substituting into the Euler–Lagrange equation we can yield:

$$I\ddot{\theta} - mgl \sin\theta = u - b\dot{\theta}. \tag{14}$$

Reordering terms gives the nonlinear equation of motion:

$$I\ddot{\theta} + b\dot{\theta} + mgl \sin\theta = u, \qquad I = ml^2. \tag{15}$$

### 1.5 State-space Form

Define the state vector and the variables:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}. \tag{16}$$

$$\dot{x}_1 = x_2, \tag{17}$$

And from the equation of motion:

$$\ddot{\theta} = \frac{1}{I}\left(u - b\dot{\theta} - mgl\sin\theta\right) = \frac{1}{I}\left(u - bx_2 - mgl\sin x_1\right). \tag{18}$$

Thus the nonlinear state-space dynamics are:

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{1}{I}\left(u - bx_2 - mgl\sin x_1\right) \end{bmatrix}. \tag{19}$$

# 2 Problem 3: LQR (Optimal Control + Dynamic Programming)

## 2.1 Problem 3.1: Linearization and Discretization

### 2.1.1 Nonlinear Dynamics and State-space form

From problem 1, the inverted pendulum dynamics are given by:

$$I\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u, \qquad I = ml^2, \tag{20}$$

where $\theta$ is the pendulum angle, $u$ is the applied angle, $b$ is viscous damping and $mgl$ is the gravity term.

Define the state vector:

$$x \triangleq \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{21}$$

Then the nonlinear dynamics system can be written as:

$$\dot{x} = f(x,u) = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{1}{I}\left(u - bx_2 - mgl\sin x_1\right) \end{bmatrix}. \tag{22}$$

### 2.1.2 Equilibrium point for the upright position

We linearize around the upright equilibrium in a specified state:

$$\theta^\star = \pi, \qquad \dot{\theta}^\star = 0, \qquad u^\star = 0. \tag{23}$$

This is an equilibrium because substituting $(\theta^\star, \dot{\theta}^\star, u^\star)$ into (20) yields

$$I\cdot 0 + b\cdot 0 + mgl\sin(\pi) = 0 \quad \Rightarrow \quad 0 = 0. \tag{24}$$

### 2.1.3 Deviation (error) coordinates

For LQR, it is standard to work with deviation variables (small perturbations around the equilibrium):

$$\delta\theta \triangleq \theta - \pi, \qquad \delta\dot{\theta} \triangleq \dot{\theta} - 0 = \dot{\theta}, \qquad \delta u \triangleq u - 0 = u. \tag{25}$$

Collect them into:

$$\delta x \triangleq x - x^\star = \begin{bmatrix} \delta\theta \\ \delta\dot{\theta} \end{bmatrix}. \tag{26}$$

In the remainder, we abuse notation and simply write $x$ to mean the deviation state $\delta x$, since the linearized dynamics are expressed in deviation coordinates.

### 2.1.4 Linearization method A (using the $\sin(\pi + \delta\theta) \approx -\delta\theta$)

The hint suggests using:

$$\sin(\pi + \delta\theta) \approx -\delta\theta \quad \text{for small } \delta\theta. \tag{27}$$

Substitute $\theta = \pi + \delta\theta$ into (20):

$$I\ddot{\theta} + b\dot{\theta} + mgl\sin(\pi + \delta\theta) = u. \tag{28}$$

Using (27), we get

$$I\ddot{\theta} + b\dot{\theta} - mgl\,\delta\theta = u. \tag{29}$$

Because $\theta = \pi + \delta\theta$, we have $\dot{\theta} = \delta\dot{\theta}$ and $\ddot{\theta} = \delta\ddot{\theta}$, hence

$$I\,\delta\ddot{\theta} + b\,\delta\dot{\theta} - mgl\,\delta\theta = \delta u. \tag{30}$$

Solve for $\delta\ddot{\theta}$:

$$\delta\ddot{\theta} = \frac{mgl}{I}\delta\theta - \frac{b}{I}\delta\dot{\theta} + \frac{1}{I}\delta u. \tag{31}$$

Now write the first-order state-space form with $x = [\delta\theta,\, \delta\dot{\theta}]^\top$:

$$\dot{x} = \begin{bmatrix} \delta\dot{\theta} \\ \delta\ddot{\theta} \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{mgl}{I}x_1 - \frac{b}{I}x_2 + \frac{1}{I}u \end{bmatrix}. \tag{32}$$

Therefore the continuous-time linearized dynamics are:

$$\dot{x} = Ax + Bu, \tag{33}$$

with

$$A = \begin{bmatrix} 0 & 1 \\ \frac{mgl}{I} & -\frac{b}{I} \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \frac{1}{I} \end{bmatrix}. \tag{34}$$

### 2.1.5 Linearization method B (Jacobian Method)

As a consistency check, linearization can also be obtained via Jacobian:

$$\delta\dot{x} \approx \left.\frac{\partial f}{\partial x}\right|_{(x^\star,u^\star)} \delta x + \left.\frac{\partial f}{\partial u}\right|_{(x^\star,u^\star)} \delta u. \tag{35}$$

From (22), we have:

$$f(x,u) = \begin{bmatrix} x_2 \\ \frac{1}{I}\left(u - bx_2 - mgl\sin x_1\right) \end{bmatrix}. \tag{36}$$

After which, we can compute the partial derivatives:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial x_2}{\partial x_1} & \frac{\partial x_2}{\partial x_2} \\ \frac{\partial}{\partial x_1}\left(\frac{u-bx_2-mgl\sin x_1}{I}\right) & \frac{\partial}{\partial x_2}\left(\frac{u-bx_2-mgl\sin x_1}{I}\right) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{mgl}{I}\cos x_1 & -\frac{b}{I} \end{bmatrix}. \tag{37}$$

At $x_1 = \theta^\star = \pi$, $\cos\pi = -1$, so

$$\left.\frac{\partial f}{\partial x}\right|_{(x^\star,u^\star)} = \begin{bmatrix} 0 & 1 \\ \frac{mgl}{I} & -\frac{b}{I} \end{bmatrix} = A,$$

and

$$\left.\frac{\partial f}{\partial u}\right|_{(x^\star,u^\star)} = \begin{bmatrix} 0 \\ \frac{1}{I} \end{bmatrix} = B.$$

This matches (34).

### 2.1.6 Euler discretization

Euler discretization is:

$$x_{k+1} = x_k + \Delta t\left(Ax_k + Bu_k\right). \tag{38}$$

Rearranging:

$$x_{k+1} = \left(I + \Delta tA\right)x_k + \left(\Delta tB\right)u_k. \tag{39}$$

Thus the discrete-time matrices are

$$A_d = I + \Delta t\, A, \qquad B_d = \Delta t\, B. \tag{40}$$

Substituting (34), we obtain explicit forms:

$$A_d = \begin{bmatrix} 1 & \Delta t \\ \Delta t\frac{mgl}{I} & 1 - \Delta t\frac{b}{I} \end{bmatrix}, \qquad B_d = \begin{bmatrix} 0 \\ \frac{\Delta t}{I} \end{bmatrix}. \tag{41}$$

## 2.2 Problem 3.2: Finite-horizon Discrete-time LQR via Dynamic Programming

### 2.2.1 Problem statement

Consider the discrete-time linear system

$$x_{k+1} = A_d x_k + B_d u_k, \qquad k = 0, 1, \ldots, N-1, \tag{42}$$

and the finite-horizon quadratic cost

$$J = \sum_{k=0}^{N-1} \left( x_k^\top Q x_k + u_k^\top R u_k \right) + x_N^\top Q_f x_N, \tag{43}$$

where $Q \succeq 0$, $R \succ 0$, and $Q_f \succeq 0$. The goal is to find an optimal sequence $\{u_k^\star\}_{k=0}^{N-1}$ minimizing $J$.

### 2.2.2 Cost-to-go and Bellman optimality

Define the optimal cost-to-go at time $k$:

$$J_k^\star(x) \triangleq \min_{\{u_i\}_{i=k}^{N-1}} \left[ \sum_{i=k}^{N-1} \left( x_i^\top Q x_i + u_i^\top R u_i \right) + x_N^\top Q_f x_N \right], \tag{44}$$

with terminal condition

$$J_N^\star(x) = x^\top Q_f x. \tag{45}$$

Bellman's optimality principle yields the recursion

$$J_k^\star(x) = \min_u \left( x^\top Q x + u^\top R u + J_{k+1}^\star(x^+) \right), \qquad x^+ = A_d x + B_d u, \tag{46}$$

where $x$ denotes the current state $x_k$ and $x^+$ denotes the next state $x_{k+1}$.

### 2.2.3 Induction hypothesis: quadratic form of the value function

We claim $J_k^\star(x)$ is a quadratic form:

$$J_k^\star(x) = x^\top P_k x, \tag{47}$$

for some symmetric matrix $P_k \succeq 0$.

**Base case.** At $k = N$, (45) implies $P_N = Q_f$.

**Inductive step.** Assume (47) holds for $k+1$, i.e.,

$$J_{k+1}^\star(x) = x^\top P_{k+1} x. \tag{48}$$

We show it then holds for $k$ and derive the recursion for $P_k$.

### 2.2.4 Substitute the quadratic hypothesis into Bellman

Substitute (48) into (46):

$$J_k^\star(x) = \min_u \left( x^\top Q x + u^\top R u + (A_d x + B_d u)^\top P_{k+1} (A_d x + B_d u) \right). \tag{49}$$

### 2.2.5 Expand the quadratic expression

Expand the last term:

$$\begin{aligned}(A_d x + B_d u)^\top P_{k+1} (A_d x + B_d u) &= x^\top A_d^\top P_{k+1} A_d x + x^\top A_d^\top P_{k+1} B_d u + u^\top B_d^\top P_{k+1} A_d x + u^\top B_d^\top P_{k+1} B_d u \\ &= x^\top A_d^\top P_{k+1} A_d x + 2\, u^\top B_d^\top P_{k+1} A_d x + u^\top B_d^\top P_{k+1} B_d u,\end{aligned} \tag{50}$$

where the middle two terms are transposes of each other (scalars) and thus sum to twice one of them.

Plug (50) into (49):

$$J_k^\star(x) = \min_u \Big[ x^\top (Q + A_d^\top P_{k+1} A_d) x + 2\, u^\top (B_d^\top P_{k+1} A_d) x + u^\top (R + B_d^\top P_{k+1} B_d) u \Big]. \tag{51}$$

For convenience, define

$$S_k \triangleq R + B_d^\top P_{k+1} B_d, \qquad G_k \triangleq B_d^\top P_{k+1} A_d, \qquad H_k \triangleq Q + A_d^\top P_{k+1} A_d. \tag{52}$$

Then the minimized expression becomes

$$\Phi(u; x) = x^\top H_k x + 2 u^\top G_k x + u^\top S_k u, \qquad J_k^\star(x) = \min_u \Phi(u; x). \tag{53}$$

### 2.2.6 Minimize with respect to $u$ (first-order optimality condition)

Because $R \succ 0$ and $B_d^\top P_{k+1} B_d \succeq 0$, we have

$$S_k = R + B_d^\top P_{k+1} B_d \succ 0, \tag{54}$$

so $\Phi(u; x)$ is a strictly convex quadratic function of $u$, hence the minimizer is unique and can be found by setting the gradient to zero.

Compute the gradient w.r.t. $u$:

$$\nabla_u (u^\top S_k u) = 2 S_k u, \qquad \nabla_u (2 u^\top G_k x) = 2 G_k x, \qquad \nabla_u (x^\top H_k x) = 0.$$

Thus:

$$\nabla_u \Phi(u; x) = 2 S_k u + 2 G_k x = 0 \quad \Longrightarrow \quad S_k u^\star = -G_k x. \tag{55}$$

Therefore,

$$u^\star = -S_k^{-1} G_k x. \tag{56}$$

Define the time-varying feedback gain

$$K_k \triangleq S_k^{-1} G_k = \left( R + B_d^\top P_{k+1} B_d \right)^{-1} B_d^\top P_{k+1} A_d, \tag{57}$$

so that the optimal control law is

$$u_k^\star = -K_k x_k. \tag{58}$$

#### 2.2.7 Substitute the optimal $u^\star$ back to obtain the Riccati recursion

To obtain $P_k$, we compute the minimized value $\min_u \Phi(u;x)$. A clean way is to "complete the square":

$$\begin{aligned}\Phi(u;x) &= x^\top H_k x + 2u^\top G_k x + u^\top S_k u \\ &= x^\top H_k x + (u + S_k^{-1}G_k x)^\top S_k (u + S_k^{-1}G_k x) - x^\top G_k^\top S_k^{-1} G_k x. \end{aligned} \tag{59}$$

The middle term is always $\geq 0$ and is minimized to 0 at $u^\star = -S_k^{-1}G_k x$. Hence,

$$J_k^\star(x) = \min_u \Phi(u;x) = x^\top \left(H_k - G_k^\top S_k^{-1} G_k\right) x. \tag{60}$$

Therefore $J_k^\star(x)$ is quadratic with

$$P_k = H_k - G_k^\top S_k^{-1} G_k. \tag{61}$$

Substitute definitions (52) into (61):

$$P_k = Q + A_d^\top P_{k+1} A_d - A_d^\top P_{k+1} B_d \left(R + B_d^\top P_{k+1} B_d\right)^{-1} B_d^\top P_{k+1} A_d. \tag{62}$$

Together with the terminal condition

$$P_N = Q_f, \tag{63}$$

we obtain the finite-horizon backward Riccati recursion.

#### 2.2.8 Final boxed result (what we implement)

The finite-horizon discrete-time LQR solution is:

$$\boxed{\begin{aligned} &P_N = Q_f, \\ &K_k = \left(R + B_d^\top P_{k+1} B_d\right)^{-1} B_d^\top P_{k+1} A_d, \\ &u_k^\star = -K_k x_k, \\ &P_k = Q + A_d^\top P_{k+1} A_d - A_d^\top P_{k+1} B_d \left(R + B_d^\top P_{k+1} B_d\right)^{-1} B_d^\top P_{k+1} A_d, \qquad k = N-1, \ldots, 0. \end{aligned}} \tag{64}$$

# 3 Problem 3.4 Conceptual Questions

## 3.1 Increasing $Q$ typically:

**Answer: (a)**

Explanation: Increasing $Q$ penalizes state error more, leading to stronger control actions and faster convergence.

### 3.2 Increasing $R$ typically:

**Answer: (b)**

Explanation: Increasing $R$ penalizes control effort more, resulting in smaller torques and slower system response.

### 3.3 Increasing $Q_f$ typically:

**Answer: (a)**

Explanation: Increasing $Q_f$ places more weight on the final state, emphasizing terminal accuracy.

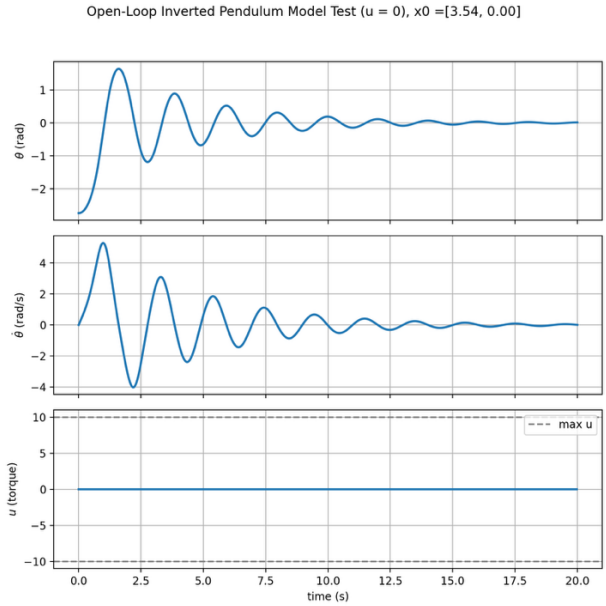
ROSE 5760 Robot Learning Assignment 1

Coding Results Part

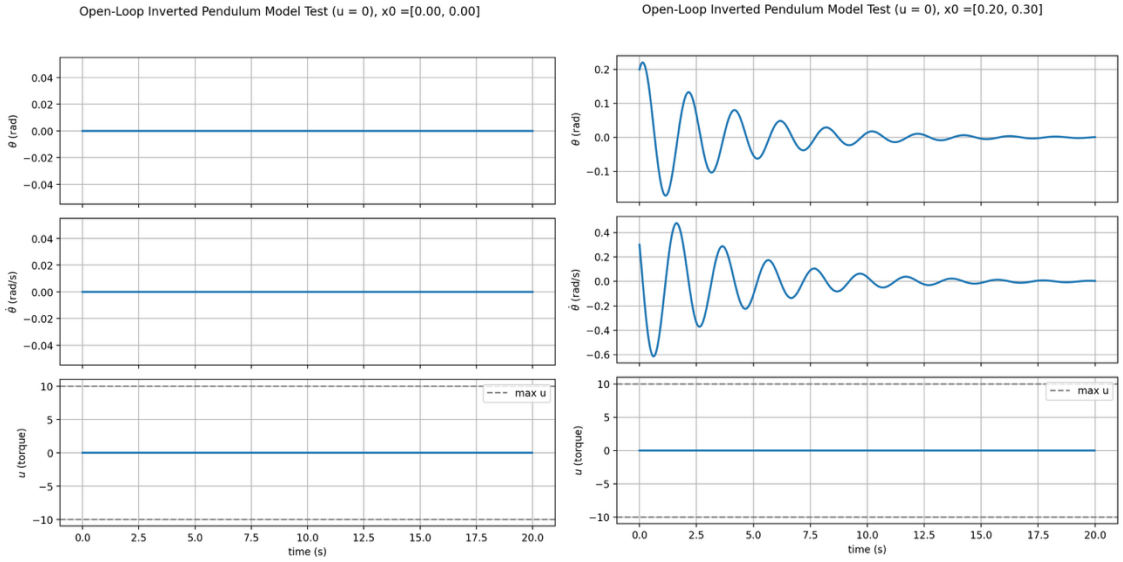
ZHANG SHUTAO 1155238521

## Problem 1.3: Free Dynamics Tests

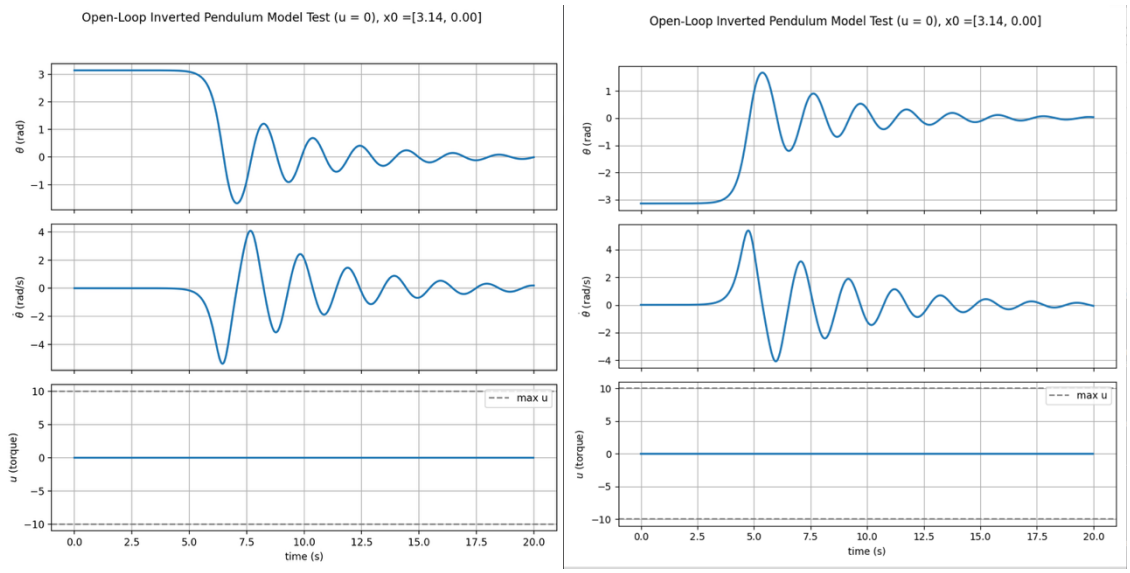
1. Default arguments




2. Initial condition: $[\theta_0, \dot{\theta}_0] = [0, 0]$ and $[0 + 0.2, 0.3]$

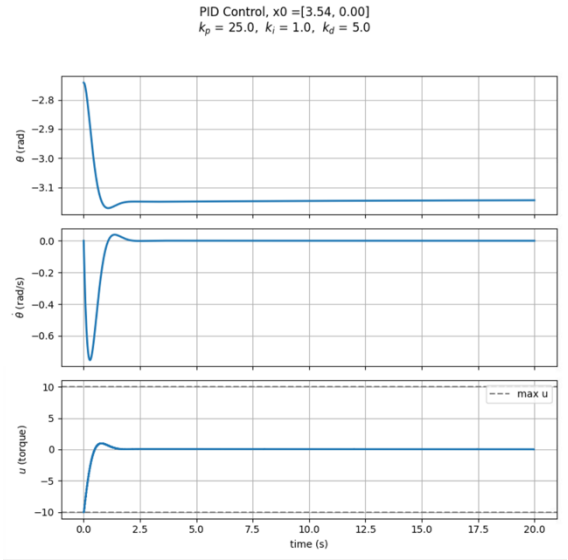



3. Initial condition: $[\theta_0, \dot{\theta}_0] = [\pi, 0]$ and $[\pi, +0.001, 0]$



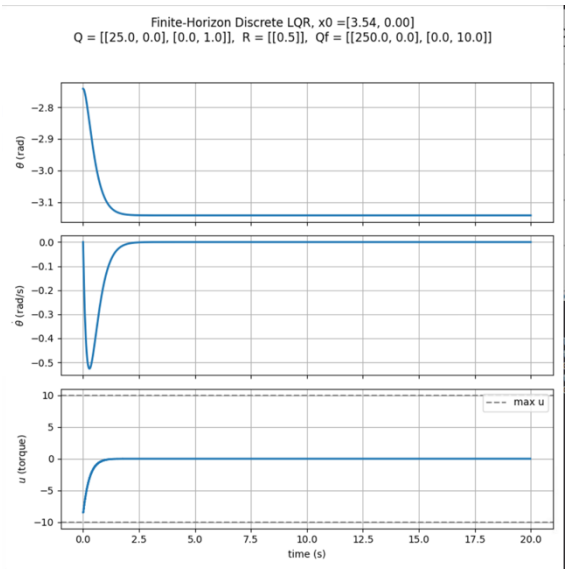

Problem 2: PID Control

1. PID Evaluation




Problem 3: Linear Quadratic Regulator (LQR)

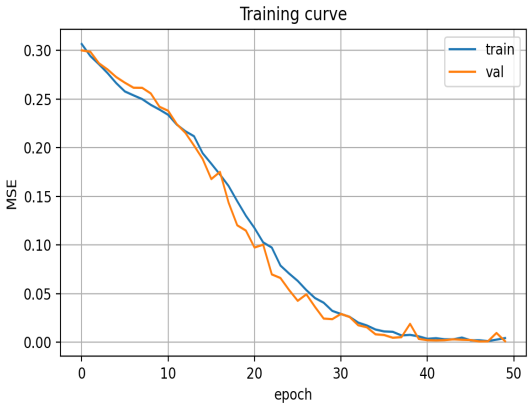
1. LQR Implementation

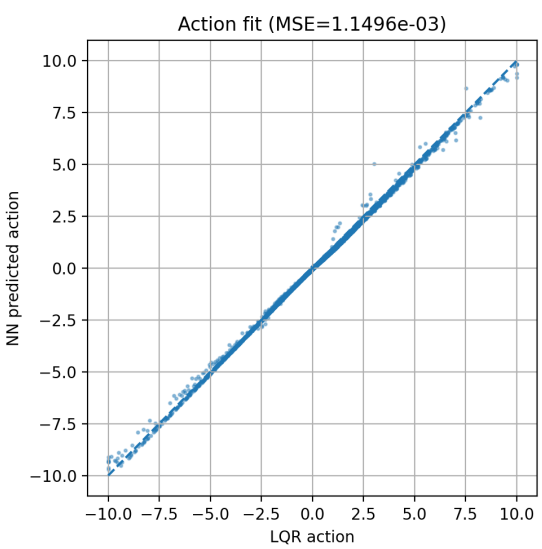


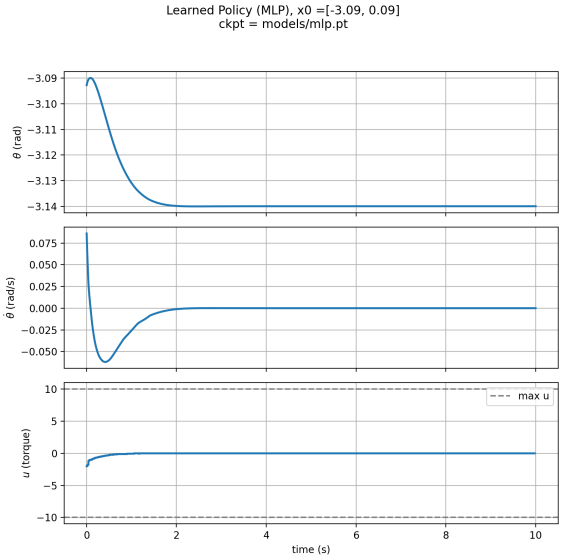

## Problem 4: Supervised Learning from LQR

1. MLP Policy

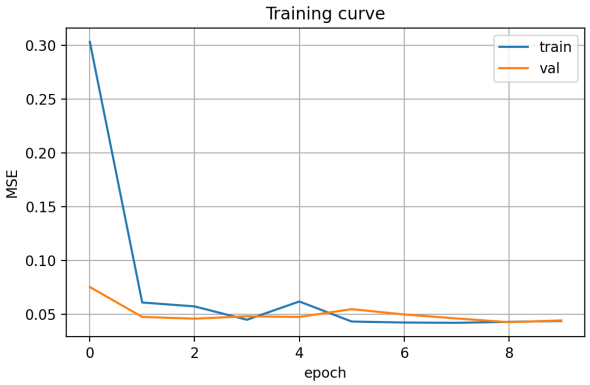
```
(robole) zhang@ZHANG:~/zst/robot_learning$ python train_mlp.py
Epoch 0001 | train 0.307837 | val 0.303708
Epoch 0005 | train 0.260436 | val 0.276873
Epoch 0010 | train 0.234905 | val 0.252749
Epoch 0015 | train 0.180592 | val 0.194910
Epoch 0020 | train 0.105138 | val 0.140903
Epoch 0025 | train 0.056354 | val 0.063179
Epoch 0030 | train 0.024322 | val 0.024593
Epoch 0035 | train 0.010686 | val 0.014245
Epoch 0040 | train 0.004305 | val 0.002952
Epoch 0045 | train 0.001779 | val 0.002353
Epoch 0050 | train 0.005509 | val 0.001207
Saved: models/mlp.pt
```

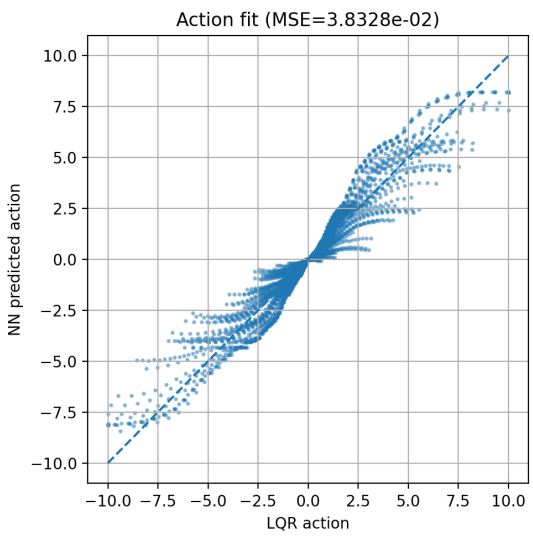


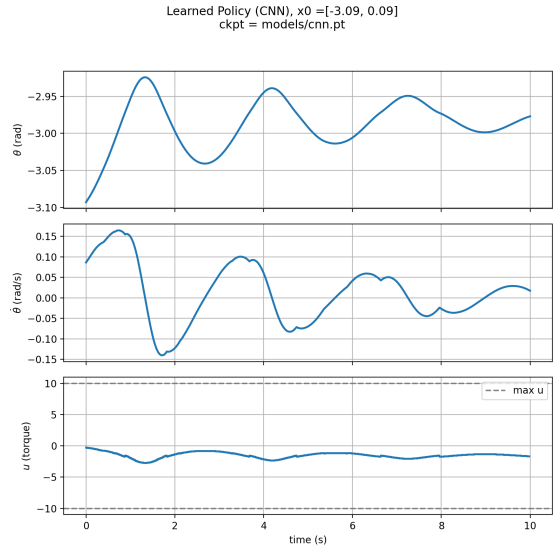


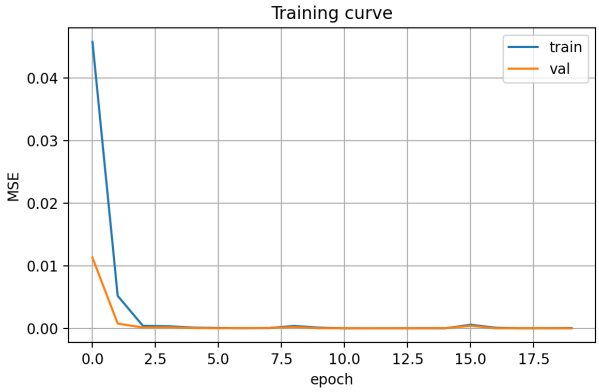

## 2. CNN Policy

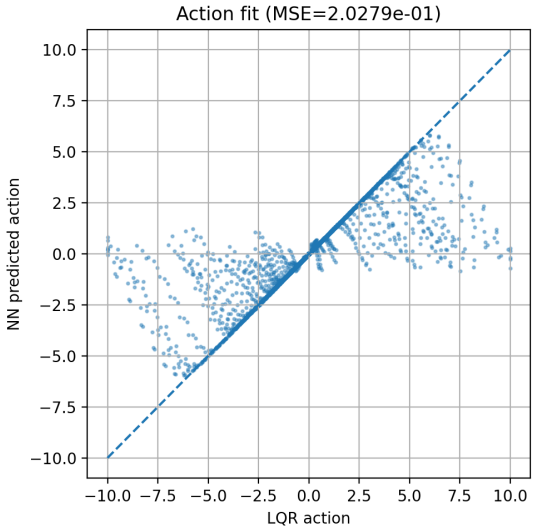



## 3. LSTM Policy




Learned Policy (LSTM), x0 =[-3.09, 0.09]
ckpt = models/lstm.pt