Optimal Control Theory and the Linear Quadratic Regulator

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2023

Today

- Feedback from last lecture
- Recap
- General optimal control problem
- The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2.

Today

- ✓ Feedback from last lecture
- Recap
- General optimal control problem
- The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

Recap

Bellman Consistency and the Bellman Equations

- **Theorem:** Every policy π satisfies the Bellman consistency conditions:
  - $V^\pi(s) = r(s, \pi(s)) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,\pi(s))}[V^\pi(s')]$

- A function $V : S \to R$ satisfies the Bellman equations if

$$V(s) = \max_a \left\{ r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[V(s') \right] \right\}, \forall s$$

- **Theorem:**
  - V satisfies the Bellman equations if and only if $V = V^\star$.

Value Iteration Algorithm:

1. Initialization: $V^0(s) = 0, \ \forall s$
2. For $t = 0, \ldots T-1$

   $$V^{t+1}(s) = \max_a \left\{ r(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^t(s') \right\}, \ \forall s$$

3. Return: $V^T(s)$

   $$\pi(s) = \arg\max_a \left\{ r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} V^T(s') \right\}$$

- For $V \in \mathbb{R}^{|S|}$, define $\mathcal{T} : \mathbb{R}^{|S|} \mapsto \mathbb{R}^{|S|}$, where

  $$\left(\mathcal{T} V\right)(s) := \max_a \left[r(s,a) + \gamma \mathbb{E}_{s' \sim P(s,a)} V(s') \right]$$

- Bellman equations: $V = \mathcal{T} V$
- Value iteration: $V^{t+1} \leftarrow \mathcal{T} V^t$

Convergence of Value Iteration:

- The "infinity norm": For any vector $x \in R^d$, define $|x|_\infty = \max_i |x_i|$

- Theorem: Given any V, V', we have: $\|\mathcal{T}V - \mathcal{T}V'\|_\infty \leq \gamma \|V - V'\|_\infty$

- Corollary: If we set $T = \frac{1}{1-\gamma} \ln\left(\frac{1}{\epsilon(1-\gamma)}\right)$ iterations,

  VI will return a value V^T s.t. $\|V^T - V^\star\|_\infty \leq \epsilon$.

  - VI then has computational complexity $O\left(|S|^2 |A| T\right)$.

Policy Iteration (PI)

- Initialization: choose a policy $\pi^0 : S \mapsto A$
- For $t = 0,1,\ldots T-1$
  1. **Policy Evaluation**: given π^t, compute $Q^{\pi^t}(s,a)$:
  2. **Policy Improvement**: set $\pi^{t+1}(s) := \arg\max_a Q^{\pi^t}(s,a)$

Policy Iteration (PI)

- Initialization: choose a policy $\pi^0 : S \mapsto A$
- For $t = 0,1,\ldots T-1$
  1. **Policy Evaluation**: given π^t, compute $Q^{\pi^t}(s,a)$:
  2. **Policy Improvement**: set $\pi^{t+1}(s) := \arg\max_a Q^{\pi^t}(s,a)$

- Computing Q^{π^t}
  - Computing V^{π^t}: O(S^3) with linear system solving
  - Computing Q^{π^t} with V^{π^t}: O(S^2 A) using $Q^\pi(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[V^\pi(s') \right]$

Per iteration complexity: O(S^3 + S^2 A)

Convergence of Policy Iteration:

- Theorem: PI has two properties:
  - montone improvement: $V^{\pi^{t+1}}(s) \geq V^{\pi^t}(s)$
  - "contraction": $\|V^{\pi^{t+1}} - V^\star\|_\infty \leq \gamma \|V^{\pi^t} - V^\star\|_\infty$

- Corollary: If we set $T = \frac{1}{1-\gamma} \ln\left(\frac{1}{\epsilon(1-\gamma)}\right)$ iterations,

  PI will return a policy π^{t+1} s.t. $\|V^{\pi^{t+1}} - V^\star\|_\infty \leq \epsilon$

  - with total computational complexity $O\Big(\big(|S|^3 + |S|^2|A|\big)T\Big)$.

Recap

Recap

- For discrete MDPs, we covered some great algorithms for computing the optimal policy

Recap

- For discrete MDPs, we covered some great algorithms for computing the optimal policy
- But all algorithms scale polynomially in the size of the state and action spaces… what if one or both are infinite?

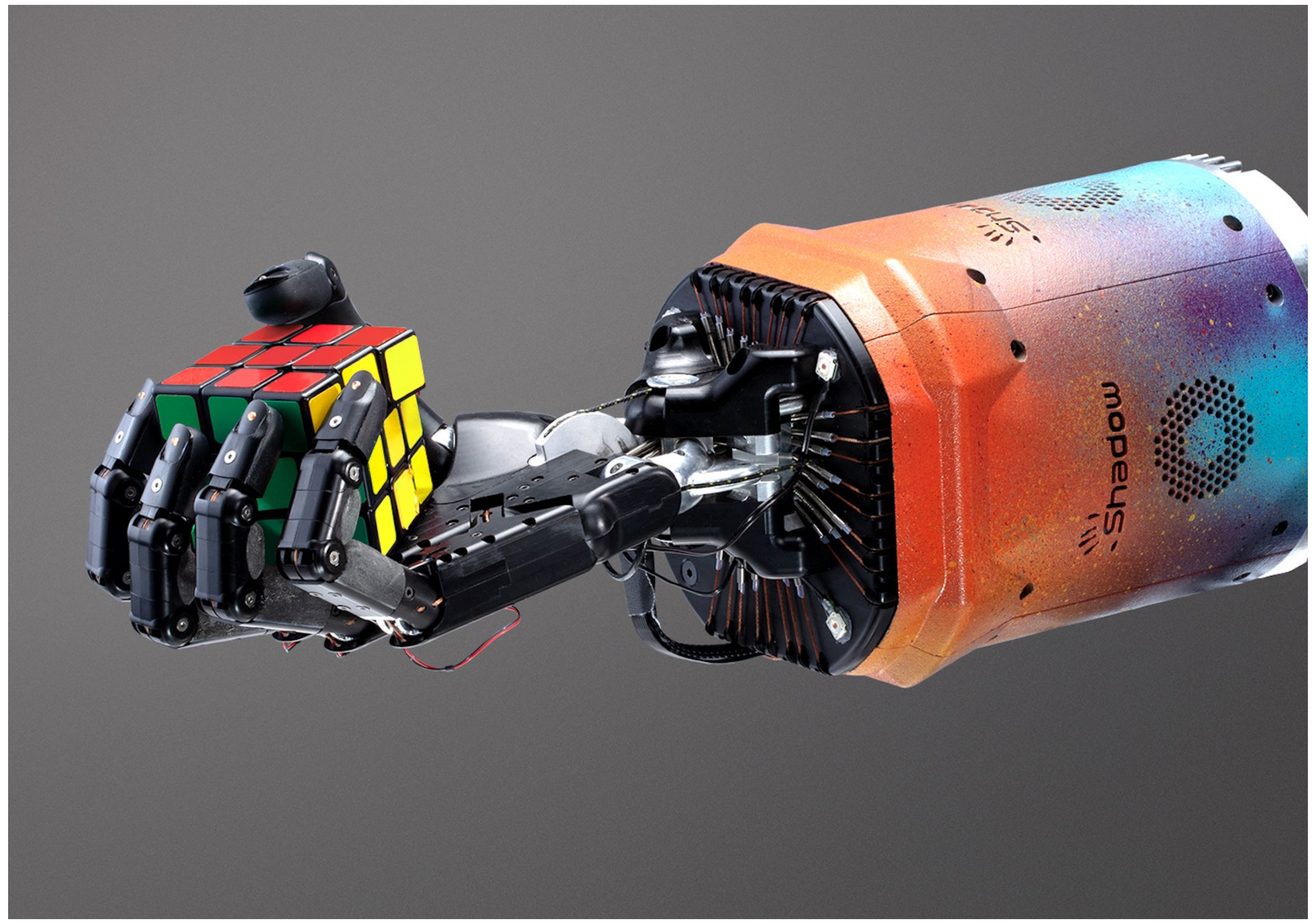
Recap

- For discrete MDPs, we covered some great algorithms for computing the optimal policy
- But all algorithms scale polynomially in the size of the state and action spaces… what if one or both are infinite?
- In this unit (next 2 lectures), we will discuss computation of good/optimal policies in continuous/infinite state and action spaces

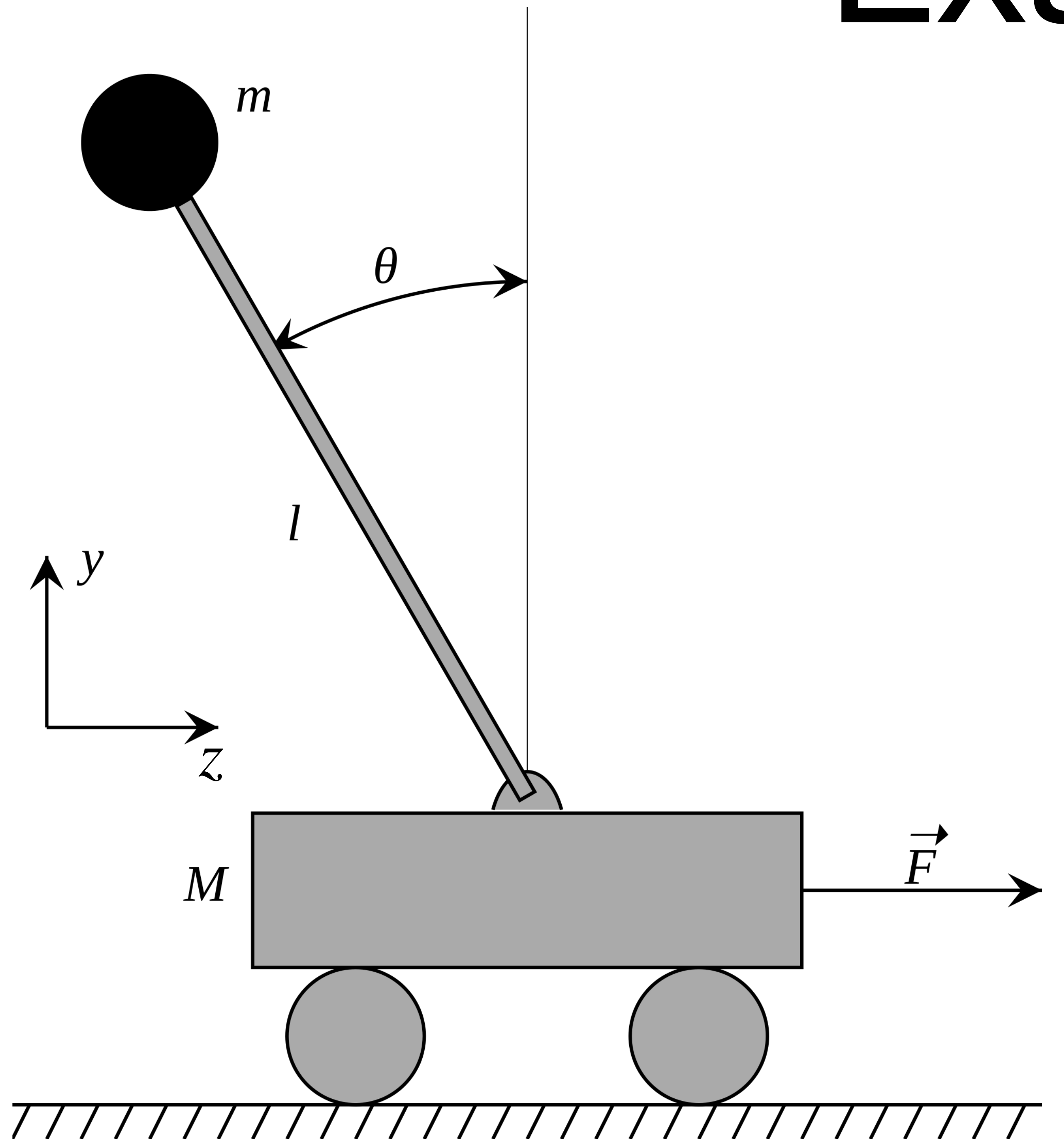
Today

- ✓ Feedback from last lecture
- ✓ Recap
- General optimal control problem
- The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

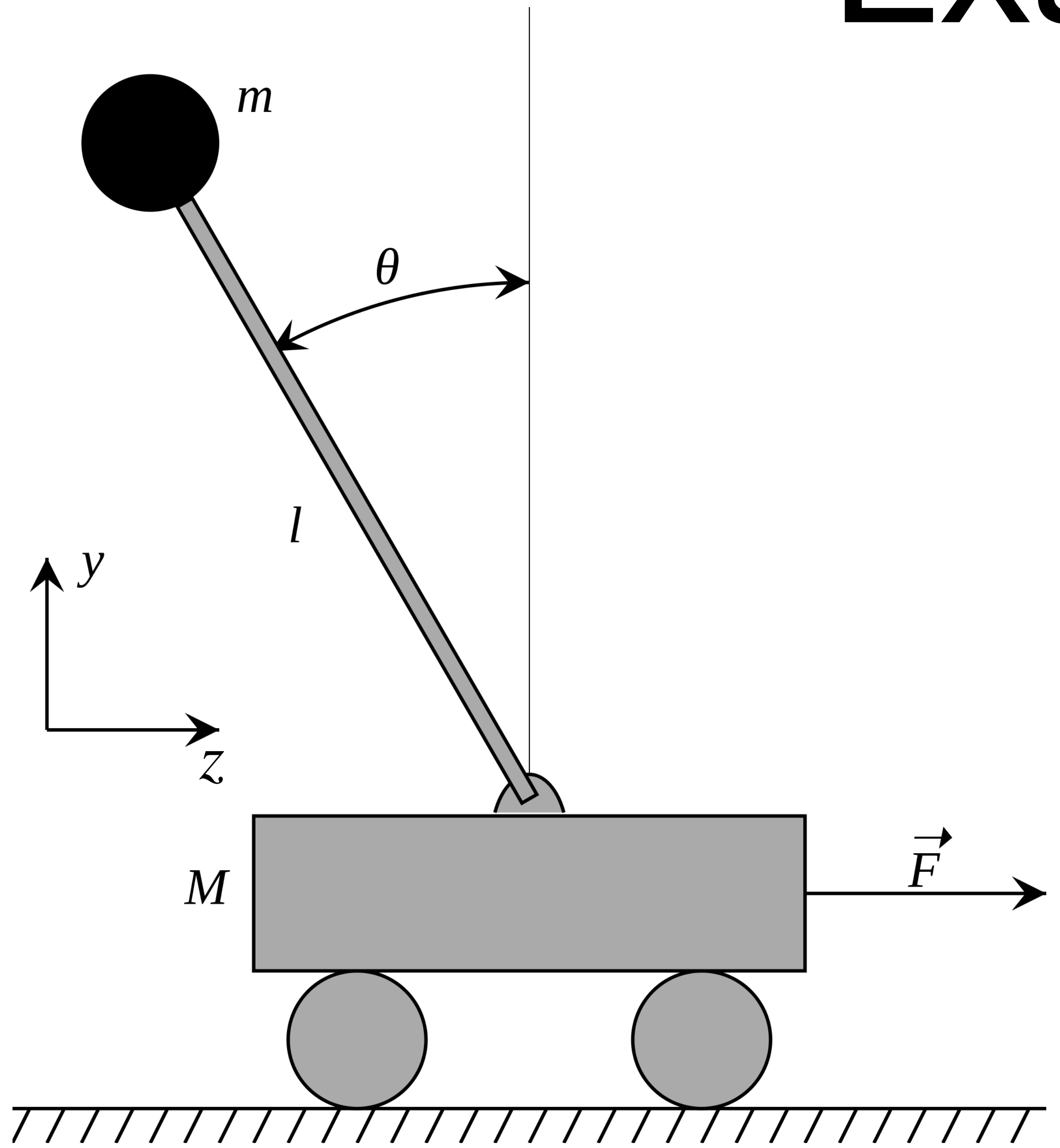
Robotics and Controls



Example: CartPole

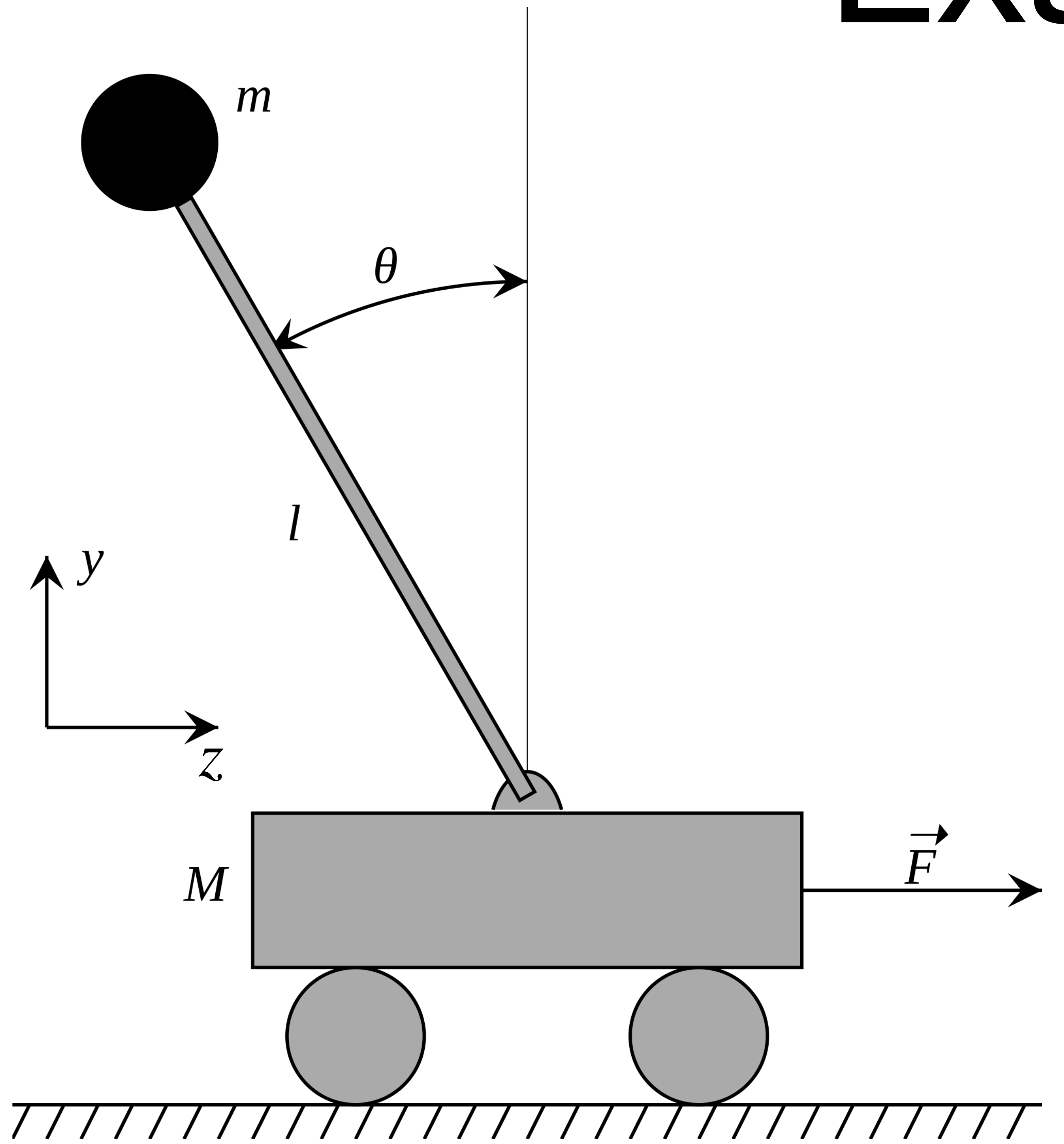


Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

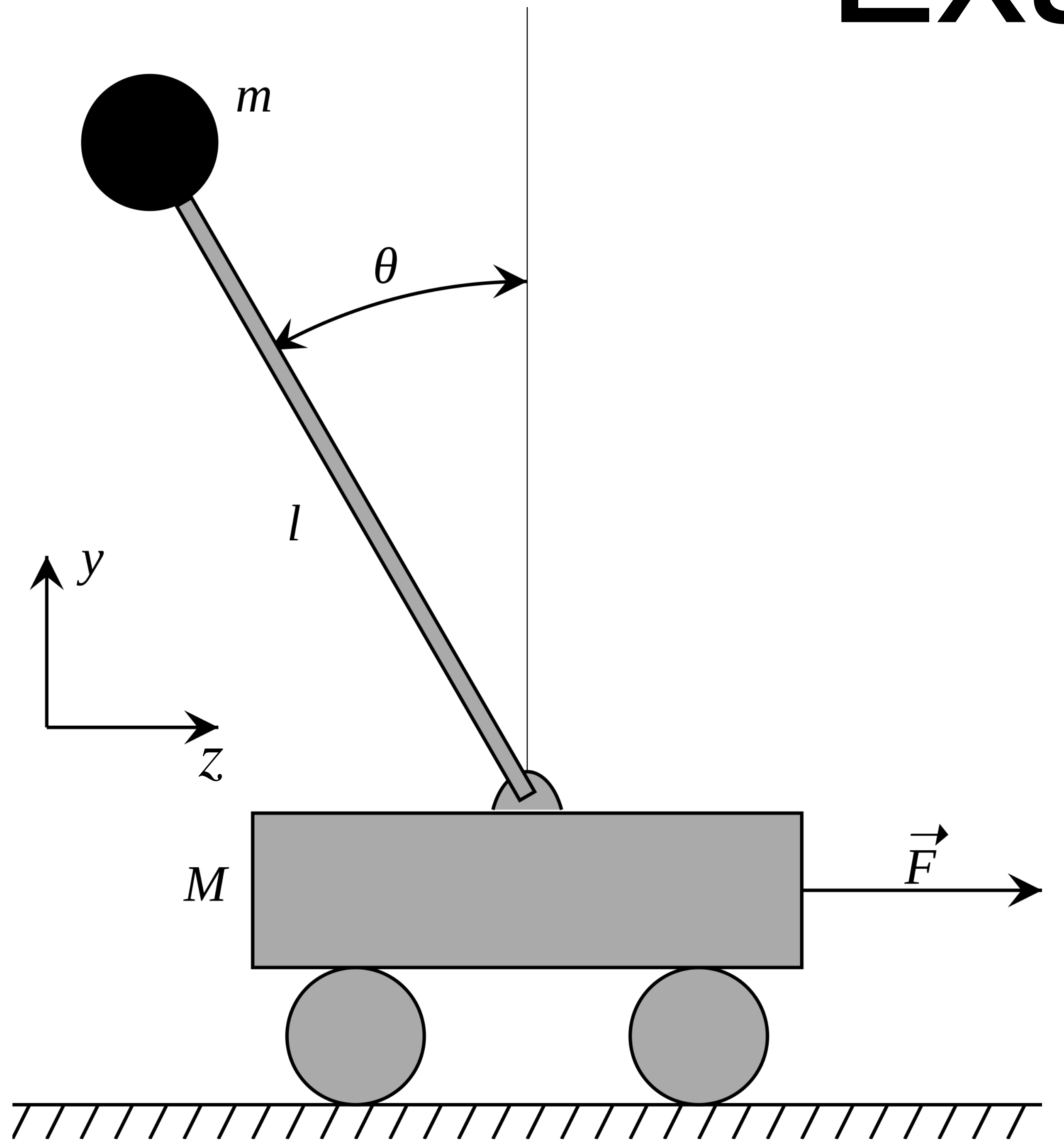
Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

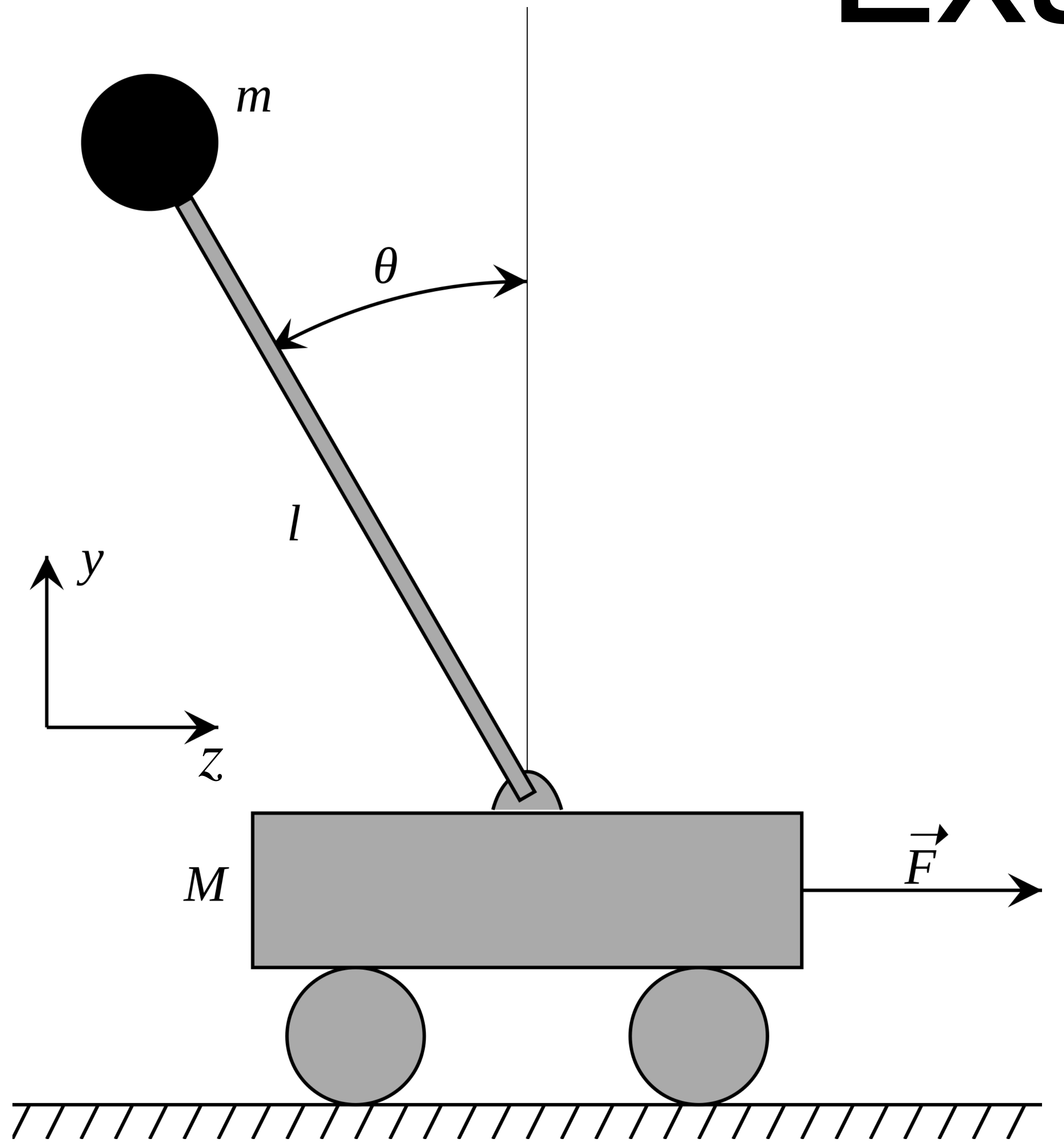
WARNING!

Notation change for controls lectures only:

States are x (instead of s)

Actions are called "controls" and are u (instead of a)

Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

WARNING!

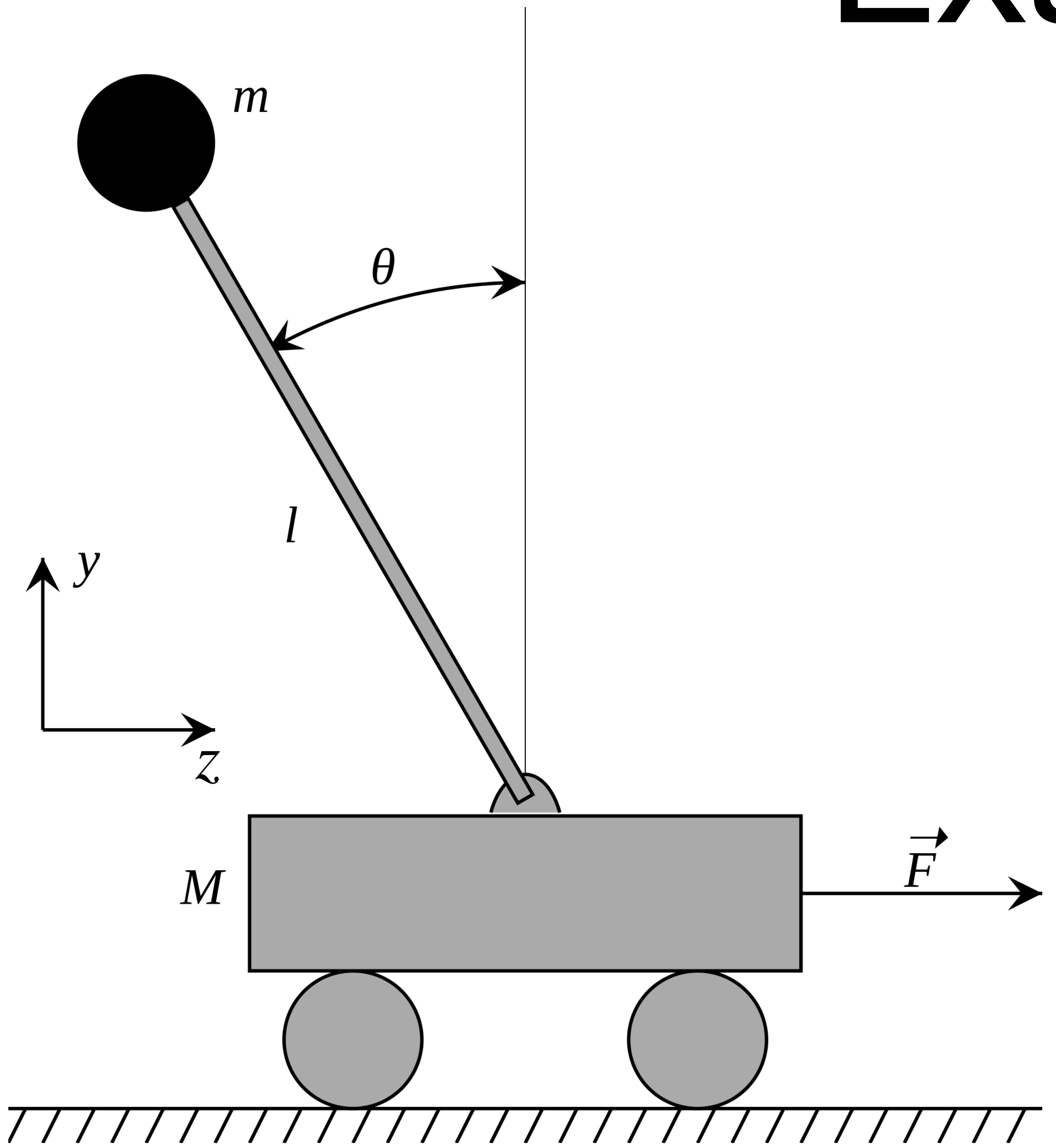
Notation change <u>for controls lectures only</u>:

States are x (instead of s)

Actions are called "controls" and are u (instead of a)

Goal: stabilizing around the point $(x = x^{\star}, u = 0)$

Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

WARNING!

Notation change <u>for controls lectures only</u>:

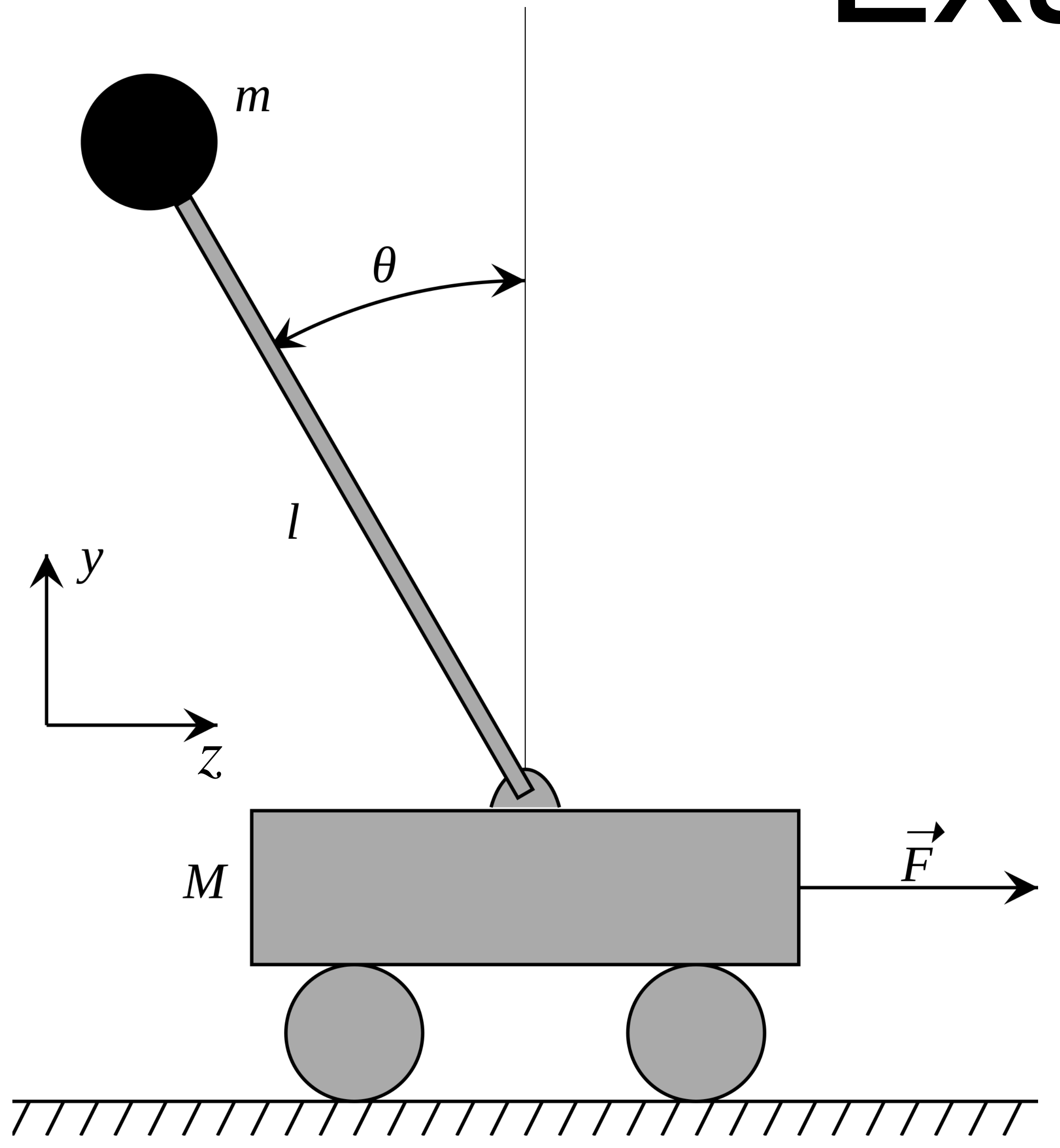
States are x (instead of s)

Actions are called "controls" and are u (instead of a)

Goal: stabilizing around the point $(x = x^\star, u = 0)$

$$c(x_h, u_h) = u_h^\top R u_h + (x_h - x^\star)^\top Q (x_h - x^\star)$$

Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

WARNING!

Notation change for controls lectures only:

States are x (instead of s)

Actions are called "controls" and are u (instead of a)

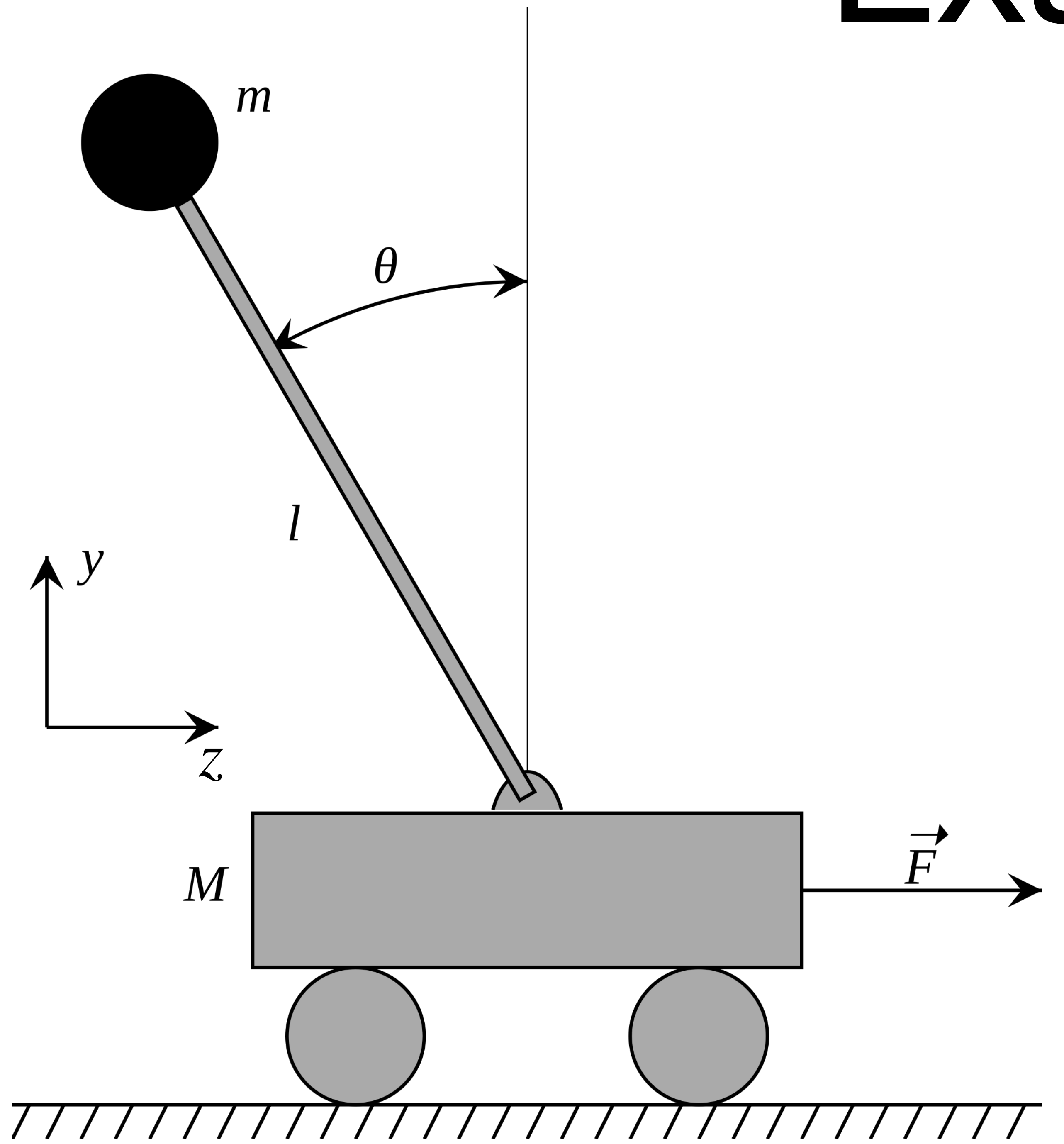
Goal: stabilizing around the point $(x = x^\star, u = 0)$

$$c(x_h, u_h) = u_h^\top R u_h + (x_h - x^\star)^\top Q (x_h - x^\star)$$

Optimal control:

$$\min_{\pi_0, \ldots, \pi_{H-1}: X \to U} \mathbb{E}\left[\sum_{h=0}^{H-1} c(x_h, u_h)\right] \quad \text{s.t. } x_{h+1} = f(x_h, u_h), \; x_0 \sim \mu_0$$

Example: CartPole



State: position and velocity of the cart, angle and angular velocity of the pole

Control=action: force on the cart

WARNING!

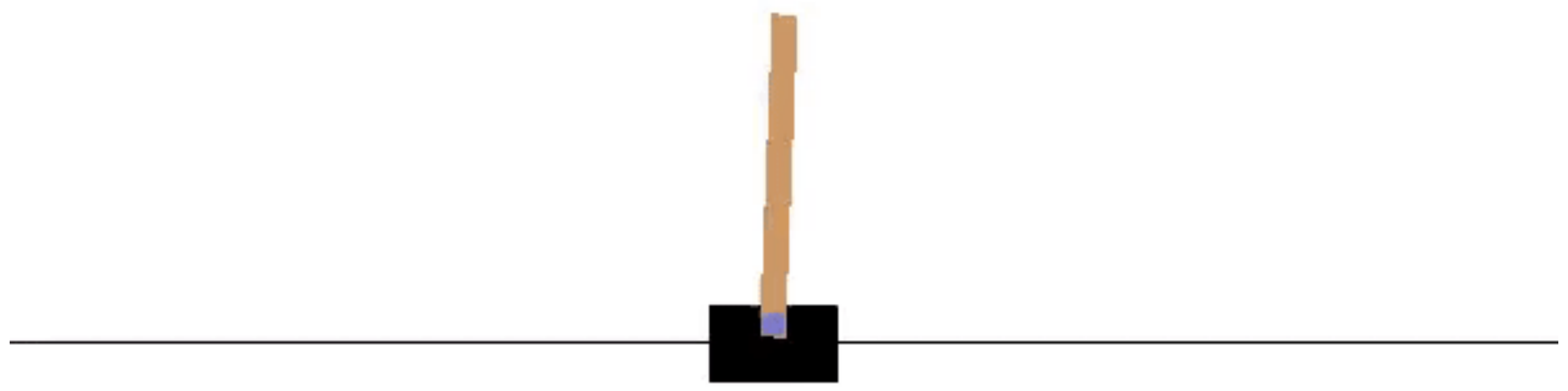
Notation change <u>for controls lectures only</u>:

States are x (instead of s)

Actions are called "controls" and are u (instead of a)

Goal: stabilizing around the point $(x = x^\star, u = 0)$

$$c(x_h, u_h) = u_h^\top R u_h + (x_h - x^\star)^\top Q (x_h - x^\star)$$



Optimal control:

$$\min_{\pi_0,\ldots,\pi_{H-1}:X\to U} \mathbb{E}\left[\sum_{h=0}^{H-1} c(x_h, u_h)\right] \quad \text{s.t. } x_{h+1} = f(x_h, u_h),\ x_0 \sim \mu_0$$

More Generally: Optimal Control

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,
- f_h is a function (the dynamics) that determines the next state $x_{h+1} \in \mathbb{R}^d$

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,
- f_h is a function (the dynamics) that determines the next state $x_{h+1} \in \mathbb{R}^d$

Objective is to find control policy π_h which minimizes the total cost (horizon H),

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,
- f_h is a function (the dynamics) that determines the next state $x_{h+1} \in \mathbb{R}^d$

Objective is to find control policy π_h which minimizes the total cost (horizon H),

$$\text{minimize } \mathbb{E}\left[c_H(x_H) + \sum_{h=0}^{H-1} c_h(x_h, u_h)\right]$$

$$\text{s.t. } x_{h+1} = f_h(x_h, u_h, w_h),\ u_h = \pi_h(x_h),\ x_0 \sim \mu_0$$

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,
- f_h is a function (the dynamics) that determines the next state $x_{h+1} \in \mathbb{R}^d$

Objective is to find control policy π_h which minimizes the total cost (horizon H),

$$\text{minimize } \mathbb{E}\left[c_H(x_H) + \sum_{h=0}^{H-1} c_h(x_h, u_h)\right]$$

$$\text{s.t. } x_{h+1} = f_h(x_h, u_h, w_h),\ u_h = \pi_h(x_h),\ x_0 \sim \mu_0$$

- Randomness (in the dynamics) enters via w_h, e.g., $w_h \sim \mathcal{N}(0,\Sigma)$

More Generally: Optimal Control

General dynamical system is described as $x_{h+1} = f_h(x_h, u_h, w_h)$, where

- $x_h \in \mathbb{R}^d$ is the state which starts at initial value $x_0 \sim \mu_0$,
- $u_h \in \mathbb{R}^k$ is the control (action),
- w_h is the noise/disturbance,
- f_h is a function (the dynamics) that determines the next state $x_{h+1} \in \mathbb{R}^d$

Objective is to find control policy π_h which minimizes the total cost (horizon H),

$$\text{minimize } \mathbb{E}\left[c_H(x_H) + \sum_{h=0}^{H-1} c_h(x_h, u_h)\right]$$

$$\text{s.t. } x_{h+1} = f_h(x_h, u_h, w_h),\ u_h = \pi_h(x_h),\ x_0 \sim \mu_0$$

- Randomness (in the dynamics) enters via w_h, e.g., $w_h \sim \mathcal{N}(0,\Sigma)$
- Note c_H separated out because by convention there is no u_H

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Recall: VI/PI computation times scaled polynomially in $|S|$ and $|A|$

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Recall: VI/PI computation times scaled polynomially in $|S|$ and $|A|$

But curse of dimensionality means $|S|$ and $|A|$ will scale like $(1/\epsilon)^d$

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Recall: VI/PI computation times scaled polynomially in $|S|$ and $|A|$

But curse of dimensionality means $|S|$ and $|A|$ will scale like $(1/\epsilon)^d$

E.g., $\epsilon = 0.01$, $d = k = 10$ gives $|S|^2|A|$ on the order of 10^{60}…

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Recall: VI/PI computation times scaled polynomially in $|S|$ and $|A|$

But curse of dimensionality means $|S|$ and $|A|$ will scale like $(1/\epsilon)^d$

E.g., $\epsilon = 0.01, d = k = 10$ gives $|S|^2|A|$ on the order of 10^{60}…

Even the idea of discretizing relies on continuity (i.e., rounding nearby values to the same grid point only works if system treats them nearly the same),

Discretize to finite state/action spaces?

$$x \in \mathbb{R}^d, u \in \mathbb{R}^k$$

Idea: Round states and controls onto an ϵ-grid of their spaces; then use tools from finite MDPs

E.g., if $\epsilon = 0.01$, round x and u to 2 decimal places

Assuming state/control spaces are bounded, this makes both finite

Recall: VI/PI computation times scaled polynomially in $|S|$ and $|A|$

But curse of dimensionality means $|S|$ and $|A|$ will scale like $(1/\epsilon)^d$

E.g., $\epsilon = 0.01$, $d = k = 10$ gives $|S|^2|A|$ on the order of 10^{60}…

Even the idea of discretizing relies on continuity (i.e., rounding nearby values to the same grid point only works if system treats them nearly the same),

So why not rely on this more formally by assuming smoothness/structure on the dynamics f and cost c?

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ General optimal control problem
- The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

The Linear Quadratic Regulator (LQR)

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

Gaussian noise: $w_h \sim \mathcal{N}(0, \Sigma)$

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

Gaussian noise: $w_h \sim \mathcal{N}(0, \Sigma)$

- Why not linear for c? Want it bounded below so we can minimize it

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

Gaussian noise: $w_h \sim \mathcal{N}(0, \Sigma)$

- Why not linear for c? Want it bounded below so we can minimize it
- $Q \in \mathbb{R}^{d \times d}$ and $R \in \mathbb{R}^{k \times k}$ are positive definite matrices

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

Gaussian noise: $w_h \sim \mathcal{N}(0, \Sigma)$

- Why not linear for c? Want it bounded below so we can minimize it
- $Q \in \mathbb{R}^{d \times d}$ and $R \in \mathbb{R}^{k \times k}$ are positive definite matrices
- $A \in \mathbb{R}^{d \times d}$, $B \in \mathbb{R}^{d \times k}$, $\Sigma \in \mathbb{R}^{d \times d}$ determine the dynamics

The Linear Quadratic Regulator (LQR)

Linear dynamics: $x_{h+1} = f(x_h, u_h, w_h) = Ax_h + Bu_h + w_h$

Quadratic cost function: $c(x_h, u_h) = x_h^\top Q x_h + u_h^\top R u_h, \quad c_H(x_H) = x_H^\top Q x_H$

Gaussian noise: $w_h \sim \mathcal{N}(0, \Sigma)$

- Why not linear for c? Want it bounded below so we can minimize it
- $Q \in \mathbb{R}^{d \times d}$ and $R \in \mathbb{R}^{k \times k}$ are positive definite matrices
- $A \in \mathbb{R}^{d \times d}$, $B \in \mathbb{R}^{d \times k}$, $\Sigma \in \mathbb{R}^{d \times d}$ determine the dynamics
- Note lack of subscripts on c (except at H) and f: time-homogeneous

Is LQR useful?

Is LQR useful?

Surprisingly yes, despite its simplicity!

Is LQR useful?

Surprisingly yes, despite its simplicity!

Any smooth dynamics function is locally approximately linear, and any smooth function with a minimum is locally approximately quadratic near its minimum

Is LQR useful?

Surprisingly yes, despite its simplicity!

Any smooth dynamics function is locally approximately linear, and any smooth function with a minimum is locally approximately quadratic near its minimum

E.g., think of heating/cooling a room: if done right, temperature should rarely deviate much from a fixed value, and shouldn’t have to do too much heating or cooling, i.e., states and controls stay local to some fixed points!

Is LQR useful?

Surprisingly yes, despite its simplicity!

Any smooth dynamics function is locally approximately linear, and any smooth function with a minimum is locally approximately quadratic near its minimum

E.g., think of heating/cooling a room: if done right, temperature should rarely deviate much from a fixed value, and shouldn't have to do too much heating or cooling, i.e., states and controls stay local to some fixed points!

In fact, because the LQR model is so well-studied in control theory, many human-engineered systems are designed to be approximately linear where possible

Is LQR useful?

Surprisingly yes, despite its simplicity!

Any smooth dynamics function is locally approximately linear, and any smooth function with a minimum is locally approximately quadratic near its minimum

E.g., think of heating/cooling a room: if done right, temperature should rarely deviate much from a fixed value, and shouldn't have to do too much heating or cooling, i.e., states and controls stay local to some fixed points!

In fact, because the LQR model is so well-studied in control theory, many human-engineered systems are designed to be approximately linear where possible

That said, it is indeed far too simple for many more complex (nonlinear) systems, though next lecture we will see how to extend it to some nonlinear systems to get surprisingly good solutions

Example: 1-d Vehicle

Robot moving in 1-d by choosing to apply force u_h left (negative) or right (positive)

Example: 1-d Vehicle

Robot moving in 1-d by choosing to apply force u_h left (negative) or right (positive)

Newton: Force = mass × acceleration, so if vehicle mass = m, acceleration = $\frac{u_h}{m}$

Example: 1-d Vehicle

Robot moving in 1-d by choosing to apply force u_h left (negative) or right (positive)

Newton: Force = mass × acceleration, so if vehicle mass = m, acceleration = $\frac{u_h}{m}$

If time steps are separated by δ (small), then we can approximate acceleration (derivative of velocity) by finite difference of velocities v_h:

$$\text{acceleration}_h = \frac{v_h - v_{h-1}}{\delta} = \frac{u_h}{m}$$

Example: 1-d Vehicle

Robot moving in 1-d by choosing to apply force u_h left (negative) or right (positive)

Newton: Force = mass × acceleration, so if vehicle mass = m, acceleration = $\frac{u_h}{m}$

If time steps are separated by δ (small), then we can approximate acceleration (derivative of velocity) by finite difference of velocities v_h:

$$\text{acceleration}_h = \frac{v_h - v_{h-1}}{\delta} = \frac{u_h}{m}$$

Same trick to approximate velocity (derivative of position) via positions p_h:

$$v_h = \frac{p_h - p_{h-1}}{\delta}$$

Example: 1-d Vehicle

Robot moving in 1-d by choosing to apply force u_h left (negative) or right (positive)

Newton: Force = mass × acceleration, so if vehicle mass = m, acceleration = $\frac{u_h}{m}$

If time steps are separated by δ (small), then we can approximate acceleration (derivative of velocity) by finite difference of velocities v_h:

$$\text{acceleration}_h = \frac{v_h - v_{h-1}}{\delta} = \frac{u_h}{m}$$

Same trick to approximate velocity (derivative of position) via positions p_h:

$$v_h = \frac{p_h - p_{h-1}}{\delta}$$

So if state $x_h = (p_h, v_h)$, we basically get linear dynamics!

LQR Value and Q functions

LQR Value and Q functions

Given a policy $\pi = (\pi_0, \ldots, \pi_{h-1})$, define the value function $V_h^\pi : \mathbb{R}^d \to \mathbb{R}$ as:

$$V_h^\pi(x) = \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \,\middle|\, u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

LQR Value and Q functions

Given a policy $\pi = (\pi_0, \ldots, \pi_{h-1})$, define the value function $V_h^\pi : \mathbb{R}^d \to \mathbb{R}$ as:

$$V_h^\pi(x) = \mathbb{E}\Big[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \,\Big|\, u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\Big]$$

and the Q function $Q_h^\pi : \mathbb{R}^d \times \mathbb{R}^k \to \mathbb{R}$ as:

$$Q_h^\pi(x, u) = \mathbb{E}\Big[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \,\Big|\, u_h = u,\ u_i = \pi_i(x_i)\ \forall i > h,\ x_h = x\Big]$$

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ General optimal control problem
- ✓ The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

LQR Optimal Control

LQR Optimal Control

$$V_h^\star(x) = \min_\pi V_h^\pi(x) = \min_{\pi_h, \pi_{h+1}, \ldots, \pi_{H-1}} \mathbb{E}\Big[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \;\Big|\; u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\Big]$$

LQR Optimal Control

$$V_h^\star(x) = \min_\pi V_h^\pi(x) = \min_{\pi_h, \pi_{h+1}, \ldots, \pi_{H-1}} \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \,\middle|\, u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

Theorem:

1. $V_h^\star$ is a quadratic function, i.e., $V_h^\star(x) = x^\top P_h x + p_h$ for some $P_h \in \mathbb{R}^{d\times d}$ and $p_h \in \mathbb{R}^d$
2. The optimal policy $\pi_h^\star$ is linear, i.e., $\pi_h^\star(x) = -K_h x$ for some $K_h \in \mathbb{R}^{k\times d}$
3. P_h, p_h, and K_h can be computed exactly

LQR Optimal Control

$$V_h^\star(x) = \min_\pi V_h^\pi(x) = \min_{\pi_h, \pi_{h+1}, \ldots, \pi_{H-1}} \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \,\middle|\, u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x \right]$$

Theorem:

1. $V_h^\star$ is a quadratic function, i.e., $V_h^\star(x) = x^\top P_h x + p_h$ for some $P_h \in \mathbb{R}^{d\times d}$ and $p_h \in \mathbb{R}^d$
2. The optimal policy $\pi_h^\star$ is linear, i.e., $\pi_h^\star(x) = -K_h x$ for some $K_h \in \mathbb{R}^{k\times d}$
3. P_h, p_h, and K_h can be computed exactly

We will cover the steps of the proof the theorem and derive the optimal policy along the way via dynamic programming

Key Steps in the Proof

Dynamic programming (finite-horizon), stepping backwards in time from H to 0

Key Steps in the Proof

Dynamic programming (finite-horizon), stepping backwards in time from H to 0

1. **Base case:** Show that $V_H^\star(x)$ is quadratic

Key Steps in the Proof

Dynamic programming (finite-horizon), stepping backwards in time from H to 0

1. **Base case:** Show that $V_H^\star(x)$ is quadratic

2. **Inductive hypothesis:** Assuming $V_{h+1}^\star(x)$ is quadratic,
   a) Show that $Q_h^\star(x, u)$ is quadratic (in both x and u)
   b) Derive the optimal policy $\pi_h^\star(x) = \arg\min_u Q_h^\star(x, u)$, and show that it's linear
   c) Show $V_h^\star(x)$ is quadratic

Key Steps in the Proof

Dynamic programming (finite-horizon), stepping backwards in time from H to 0

1. **Base case:** Show that $V_H^\star(x)$ is quadratic

2. **Inductive hypothesis:** Assuming $V_{h+1}^\star(x)$ is quadratic,
    a) Show that $Q_h^\star(x, u)$ is quadratic (in both x and u)
    b) Derive the optimal policy $\pi_h^\star(x) = \arg\min_u Q_h^\star(x, u)$, and show that it's linear
    c) Show $V_h^\star(x)$ is quadratic

3. **Conclusion:** $V_h^\star(x)$ is quadratic and $\pi_h^\star(x)$ is linear and we'll have their formulas

Base case at H

Base case at H

Recall the value function at a given h is:

$$V_h^\pi(x) = \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \;\middle|\; u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

Base case at H

Recall the value function at a given h is:

$$V_h^\pi(x) = \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \;\middle|\; u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

For V_H^π, everything disappears except first term $x_H^\top Q x_H = x^\top Q x$:

$$V_H^\star(x) = x^\top Q x$$

Base case at H

Recall the value function at a given h is:

$$V_h^\pi(x) = \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \;\middle|\; u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

For V_H^π, everything disappears except first term $x_H^\top Q x_H = x^\top Q x$:

$$V_H^\star(x) = x^\top Q x$$

Denoting $P_H := Q$ and $p_H := 0$, we get

$$V_H^\star(x) = x^\top P_H x + p_H$$

Base case at H

Recall the value function at a given h is:

$$V_h^\pi(x) = \mathbb{E}\left[x_H^\top Q x_H + \sum_{i=h}^{H-1} (x_i^\top Q x_i + u_i^\top R u_i) \;\middle|\; u_i = \pi_i(x_i)\ \forall i \geq h,\ x_h = x\right]$$

For V_H^π, everything disappears except first term $x_H^\top Q x_H = x^\top Q x$:

$$V_H^\star(x) = x^\top Q x$$

Denoting $P_H := Q$ and $p_H := 0$, we get

$$V_H^\star(x) = x^\top P_H x + p_H$$

(P_h and p_h didn't do much here, but we're going to define them recursively in the next step)

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

Induction Step

Assume $V^{\star}_{h+1}(x) = x^{\top} P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d \times d}$ and $p_{h+1} \in \mathbb{R}^{d}$

$$Q^{\star}_h(x, u) = c(x, u) + \mathbb{E}_{x' \sim f(x, u, w_{h+1})} \left[V^{\star}_{h+1}(x') \right]$$

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

$$Q^\star_h(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

$$Q^\star_h(x,u) = c(x,u) + \mathbb{E}_{x'\sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{x'\sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1}\sim \mathcal{N}(0,\sigma^2 I)}\left[V^\star_{h+1}\left(Ax + Bu + w_{h+1}\right)\right]$$

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

$$Q^\star_h(x,u) = c(x,u) + \mathbb{E}_{x'\sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{x'\sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1}\sim\mathcal{N}(0,\sigma^2 I)}\left[V^\star_{h+1}\left(Ax + Bu + w_{h+1}\right)\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1}\sim\mathcal{N}(0,\sigma^2 I)}\left[(Ax + Bu + w_{h+1})^\top P_{h+1}(Ax + Bu + w_{h+1}) + p_{h+1}\right]$$

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

$$Q^\star_h(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[V^\star_{h+1}\left(Ax + Bu + w_{h+1}\right)\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[(Ax + Bu + w_{h+1})^\top P_{h+1}(Ax + Bu + w_{h+1}) + p_{h+1}\right]$$

$$= x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[w_{h+1}^\top P_{h+1} w_{h+1}\right] + p_{h+1}$$

Induction Step

Assume $V^\star_{h+1}(x) = x^\top P_{h+1} x + p_{h+1}$, for all x, where $P_{h+1} \in \mathbb{R}^{d\times d}$ and $p_{h+1} \in \mathbb{R}^d$

$$Q^\star_h(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V^\star_{h+1}(x')\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[V^\star_{h+1}\left(Ax + Bu + w_{h+1}\right)\right]$$

$$= x^\top Q x + u^\top R u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[(Ax + Bu + w_{h+1})^\top P_{h+1}(Ax + Bu + w_{h+1}) + p_{h+1}\right]$$

$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathbb{E}_{w_{h+1} \sim \mathcal{N}(0,\sigma^2 I)}\left[w_{h+1}^\top P_{h+1} w_{h+1}\right] + p_{h+1}$$

$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Induction Step (continued)

$$
\begin{aligned}
Q_h^\star(x,u) &= c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right] \\
&= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}
\end{aligned}
$$

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$
$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$
$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Set $\nabla_u Q_h^\star(x,u) = 0$ and solve for u:

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$

$$= x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Set $\nabla_u Q_h^\star(x,u) = 0$ and solve for u:

$$\nabla_u Q_h^\star(x,u) = \nabla_u \left[u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u\right]$$

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$
$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Set $\nabla_u Q_h^\star(x,u) = 0$ and solve for u:

$$\nabla_u Q_h^\star(x,u) = \nabla_u\left[u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u\right]$$
$$= 2\left(R + B^\top P_{h+1} B\right)u + 2B^\top P_{h+1} A x$$

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$
$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Set $\nabla_u Q_h^\star(x,u) = 0$ and solve for u:

$$\nabla_u Q_h^\star(x,u) = \nabla_u\left[u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u\right]$$
$$= 2\left(R + B^\top P_{h+1} B\right)u + 2B^\top P_{h+1} A x$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h}\, x$$

Induction Step (continued)

$$Q_h^\star(x,u) = c(x,u) + \mathbb{E}_{x' \sim f(x,u,w_{h+1})}\left[V_{h+1}^\star(x')\right]$$
$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = \arg\min_u Q_h^\star(x,u)$$

Set $\nabla_u Q_h^\star(x,u) = 0$ and solve for u:

$$\nabla_u Q_h^\star(x,u) = \nabla_u\left[u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u\right]$$
$$= 2\left(R + B^\top P_{h+1} B\right)u + 2B^\top P_{h+1} A x$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h}\, x$$

$$:= -K_h x$$

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} Bu + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h} x$$

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h} x$$

$$V_h^\star(x) = Q_h^\star(x, \pi_h^\star(x))$$

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:= K_h} x$$

$$V_h^\star(x) = Q_h^\star(x, \pi_h^\star(x))$$

$$= x^\top \left(Q + A^\top P_{h+1} A\right) x + x^\top K_h^\top \left(R + B^\top P_{h+1} B\right) K_h x - 2x^\top A^\top P_{h+1} B K_h x + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top\left(Q + A^\top P_{h+1} A\right)x + u^\top\left(R + B^\top P_{h+1} B\right)u + 2x^\top A^\top P_{h+1} B u + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h}\, x$$

$$V_h^\star(x) = Q_h^\star(x, \pi_h^\star(x))$$

$$= x^\top\left(Q + A^\top P_{h+1} A\right)x + x^\top K_h^\top\left(R + B^\top P_{h+1} B\right)K_h x - 2x^\top A^\top P_{h+1} B K_h x + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Collecting the quadratic and constant terms together, $V_h^\star(x) = x^\top P_h x + p_h$, where:

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h} x$$

$$V_h^\star(x) = Q_h^\star(x, \pi_h^\star(x))$$
$$= x^\top \left(Q + A^\top P_{h+1} A\right) x + x^\top K_h^\top \left(R + B^\top P_{h+1} B\right) K_h x - 2x^\top A^\top P_{h+1} B K_h x + \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Collecting the quadratic and constant terms together, $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$
$$p_h = \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Concluding the Induction step:

$$Q_h^\star(x,u) = x^\top \left(Q + A^\top P_{h+1} A\right) x + u^\top \left(R + B^\top P_{h+1} B\right) u + 2x^\top A^\top P_{h+1} B u + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

$$\pi_h^\star(x) = -\underbrace{(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A}_{:=K_h}\, x$$

$$V_h^\star(x) = Q_h^\star(x, \pi_h^\star(x))$$

$$= x^\top \left(Q + A^\top P_{h+1} A\right) x + x^\top K_h^\top \left(R + B^\top P_{h+1} B\right) K_h x - 2x^\top A^\top P_{h+1} B K_h x + \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Collecting the quadratic and constant terms together, $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$ ← **Ricatti Equation**

$$p_h = \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Summary:

Summary:

$V_H^\star(x) = x^\top Q x,$ define $P_H = Q, p_H = 0,$

Summary:

$V_H^\star(x) = x^\top Q x$, define $P_H = Q, p_H = 0$,

We have shown that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Summary:

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

We have shown that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \operatorname{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Along the way, we also have shown that $\pi_h^\star(x) = -K_h x$, where:

$$K_h = (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

Summary:

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

We have shown that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B(R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Along the way, we also have shown that $\pi_h^\star(x) = -K_h x$, where:

$$K_h = (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

Optimal policy has nothing to do with initial distribution μ_0 or the noise σ^2!

Today

- Feedback from last lecture
- Recap
- General optimal control problem
- The linear quadratic regulator (LQR) problem
- Optimal control solution to LQR

Summary:

- Optimal control: Find optimal policy in MDP with continuous state/action spaces
- Linear quadratic regulator (LQR) is canonical problem in optimal control
    - Linear dynamics, Gaussian errors, quadratic costs
    - Optimal value and policy follow from dynamic programming

Attendance:
bit.ly/3RcTC9T

Feedback:
bit.ly/3RHtlxy