From LQR to Nonlinear Control

Lucas Janson and Sham Kakade

CS/Stat 184: Introduction to Reinforcement Learning

Fall 2022

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Feedback from feedback forms

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

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Recap: LQR

Problem Statement (finite horizon, time homogeneous):

$$\arg \min_{\pi_0,\ldots,\pi_{T-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[s_T^\top Q s_T + \sum_{t=0}^{T-1} (s_t^\top Q s_t + a_t^\top R a_t) \right]$$

such that $s_{t+1} = As_t + Ba_t + w_t\,, \quad s_0 \sim \mu_0\,, \quad a_t = \pi_t(s_t)\,, \quad w_t \sim N(0,\sigma^2 I)$

Recap: LQR

Problem Statement (finite horizon, time homogeneous):

$$\arg \min_{\pi_0,\ldots,\pi_{T-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[s_T^\top Q s_T + \sum_{t=0}^{T-1} (s_t^\top Q s_t + a_t^\top R a_t) \right]$$

such that $s_{t+1} = As_t + Ba_t + w_t\,, \quad s_0 \sim \mu_0\,, \quad a_t = \pi_t(s_t)\,, \quad w_t \sim N(0,\sigma^2 I)$

- States $s_t \in \mathbb{R}^d$
- Actions/controls $a_t \in \mathbb{R}^k$
- Additive noise $w_t \sim \mathcal{N}(0,\sigma^2 I)$
- Dynamics linear with state coefficient matrix $A \in \mathbb{R}^{d\times d}$ and action coefficient matrix $B \in \mathbb{R}^{d\times k}$
- Cost function quadratic with positive semidefinite state coefficient matrix $Q \in \mathbb{R}^{d\times d}$ and positive semidefinite action coefficient matrix $R \in \mathbb{R}^{k\times k}$

Recap: LQR Optimal Control

Recap: LQR Optimal Control

$V_T^\star(s) = s^\top Q s,$ define $P_T = Q, p_T = 0,$

Recap: LQR Optimal Control

$$V_T^\star(s) = s^\top Q s, \quad \text{define } P_T = Q, p_T = 0,$$

We showed that $V_t^\star(s) = s^\top P_t s + p_t$, where:

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

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

Recap: LQR Optimal Control

$$V_T^\star(s) = s^\top Q s, \quad \text{define } P_T = Q, p_T = 0,$$

We showed that $V_t^\star(s) = s^\top P_t s + p_t$, where:

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

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

Along the way, we also showed that $\pi_t^\star(s) = -K_t s$, where:

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

Recap: LQR Optimal Control

$$V^\star_T(s) = s^\top Q s, \quad \text{define } P_T = Q, p_T = 0,$$

We showed that $V^\star_t(s) = s^\top P_t s + p_t$, where:

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

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

Along the way, we also showed that $\pi^\star_t(s) = -K_t s$, where:

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

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

Beyond LQR

Beyond LQR

We saw a number of extensions to LQR that essentially reduced to the same problem

Beyond LQR

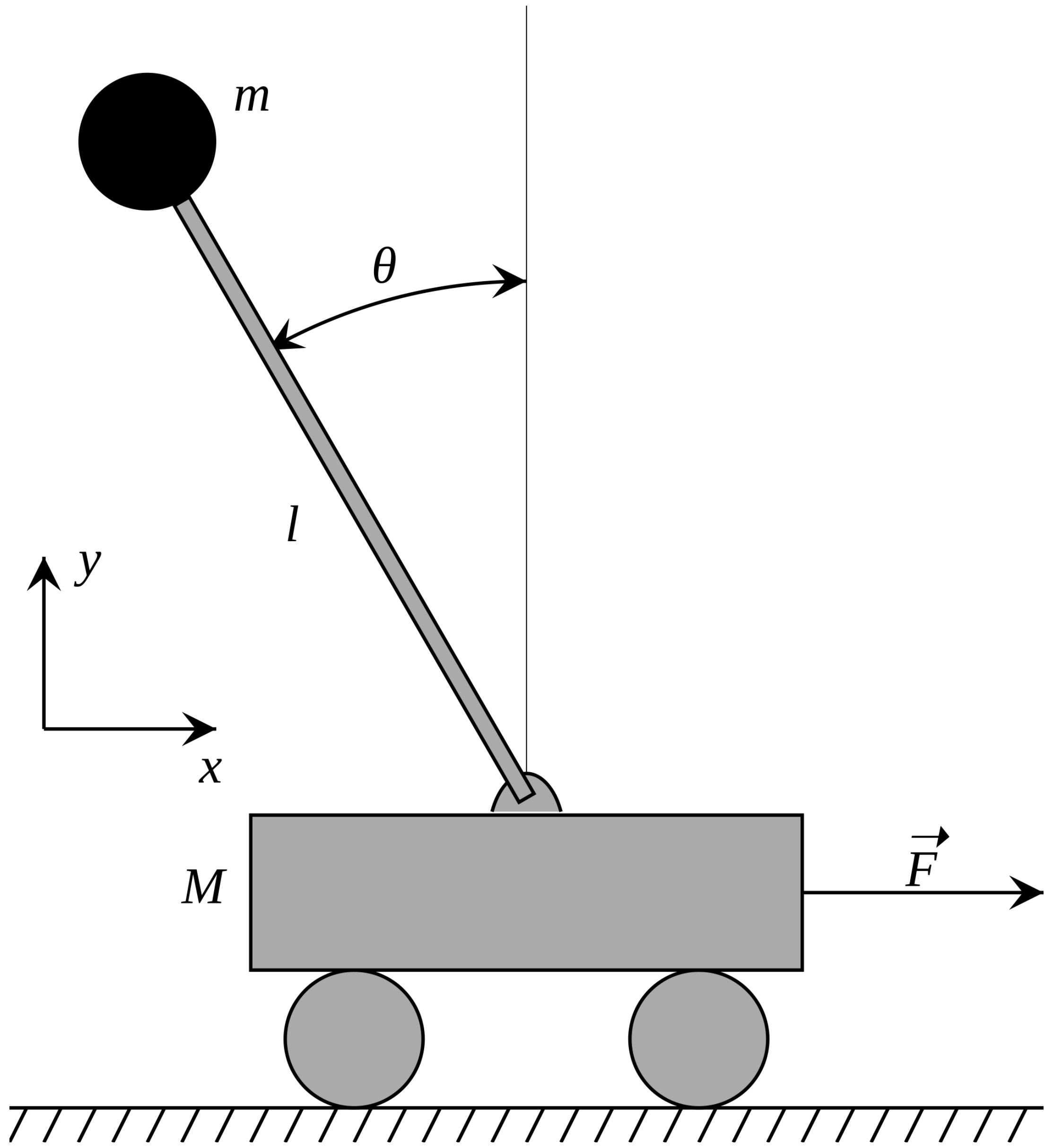
We saw a number of extensions to LQR that essentially reduced to the same problem

But what about problems with nonlinear dynamics and/or nonquadratic costs?

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

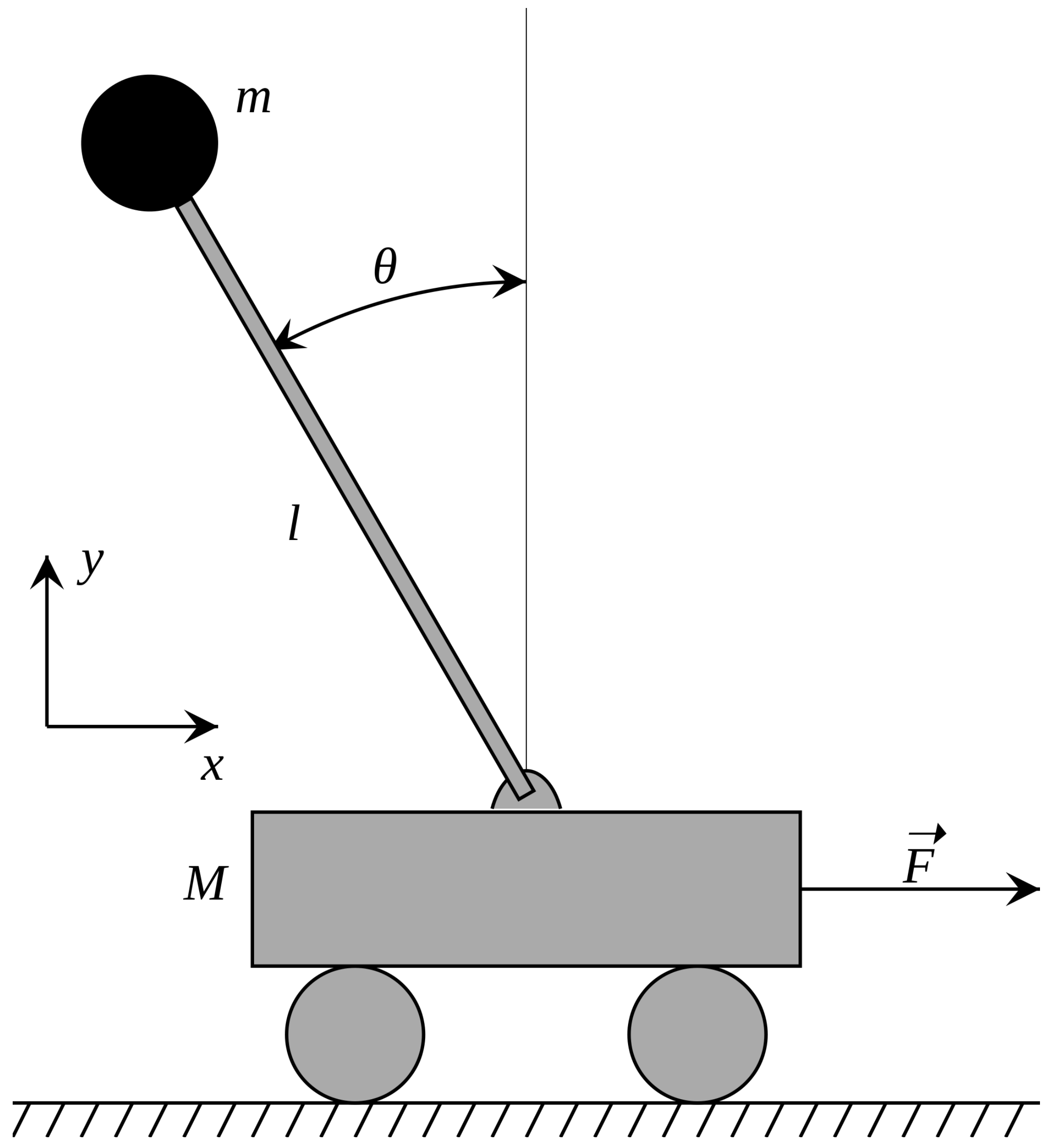
Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

Setting for Local Linearization Approach:



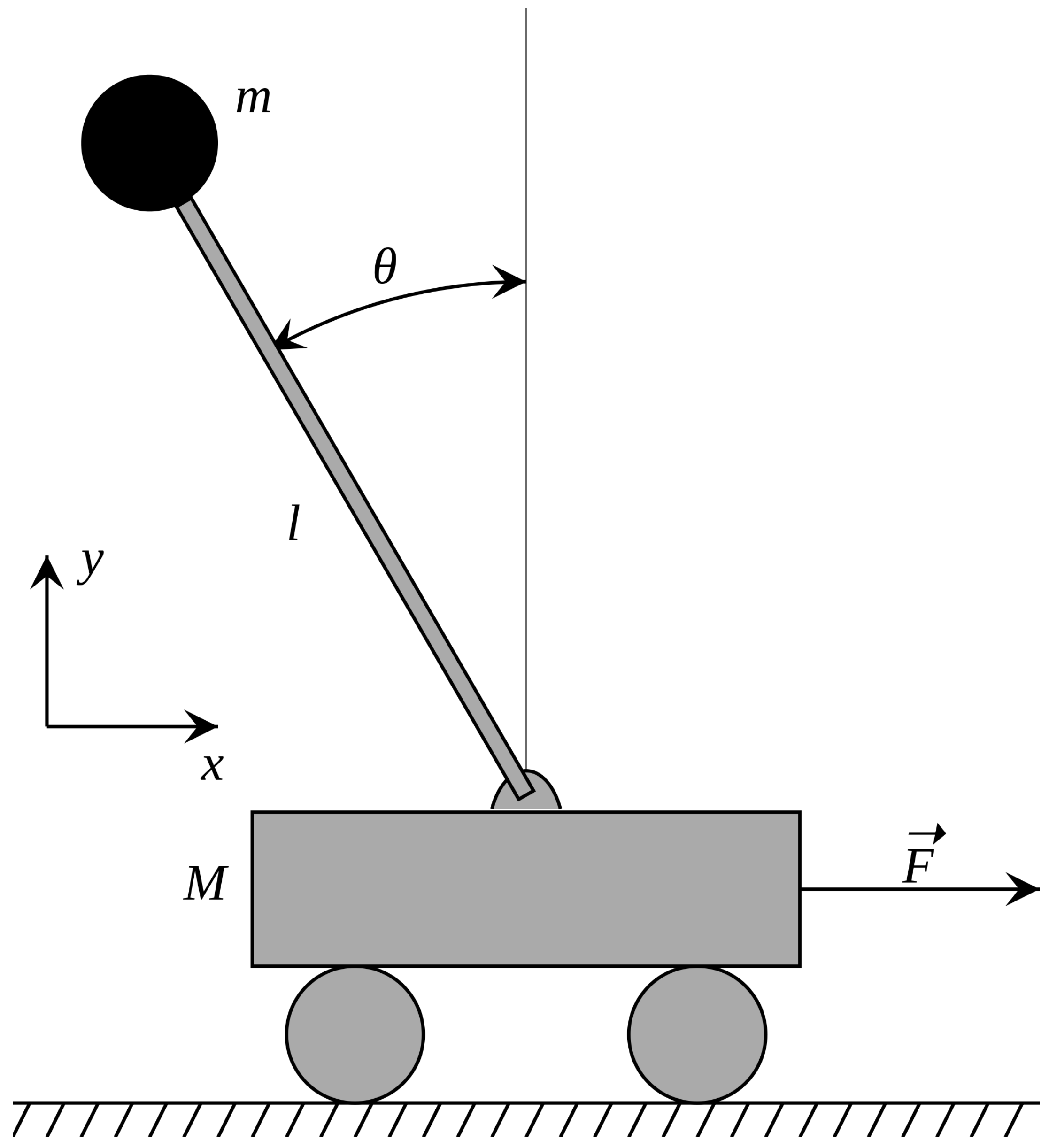
Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi \left[\sum_{t=0}^{T-1} c(s_t, a_t) \right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

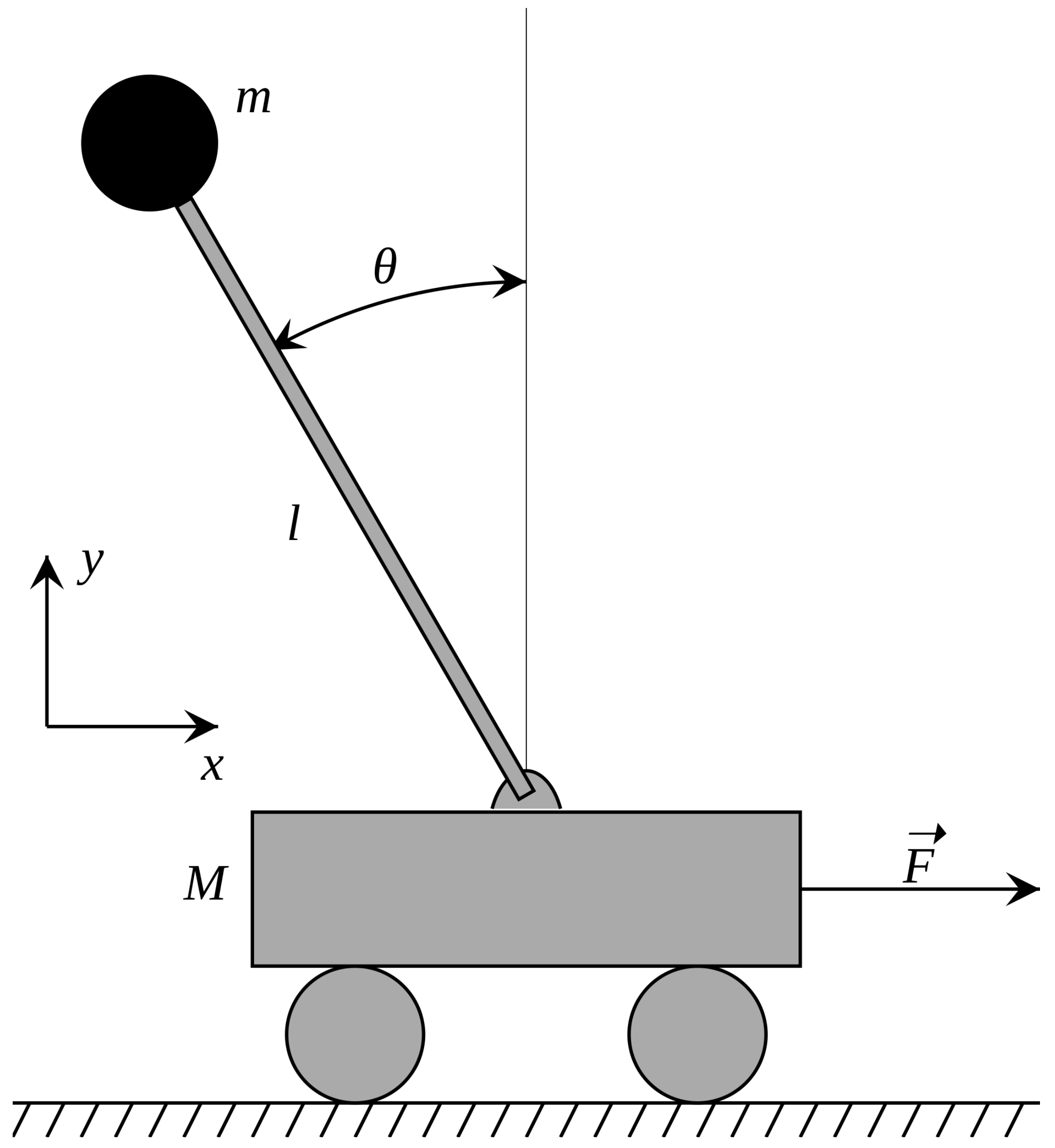
$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi \left[\sum_{t=0}^{T-1} c(s_t, a_t) \right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

No noise!

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

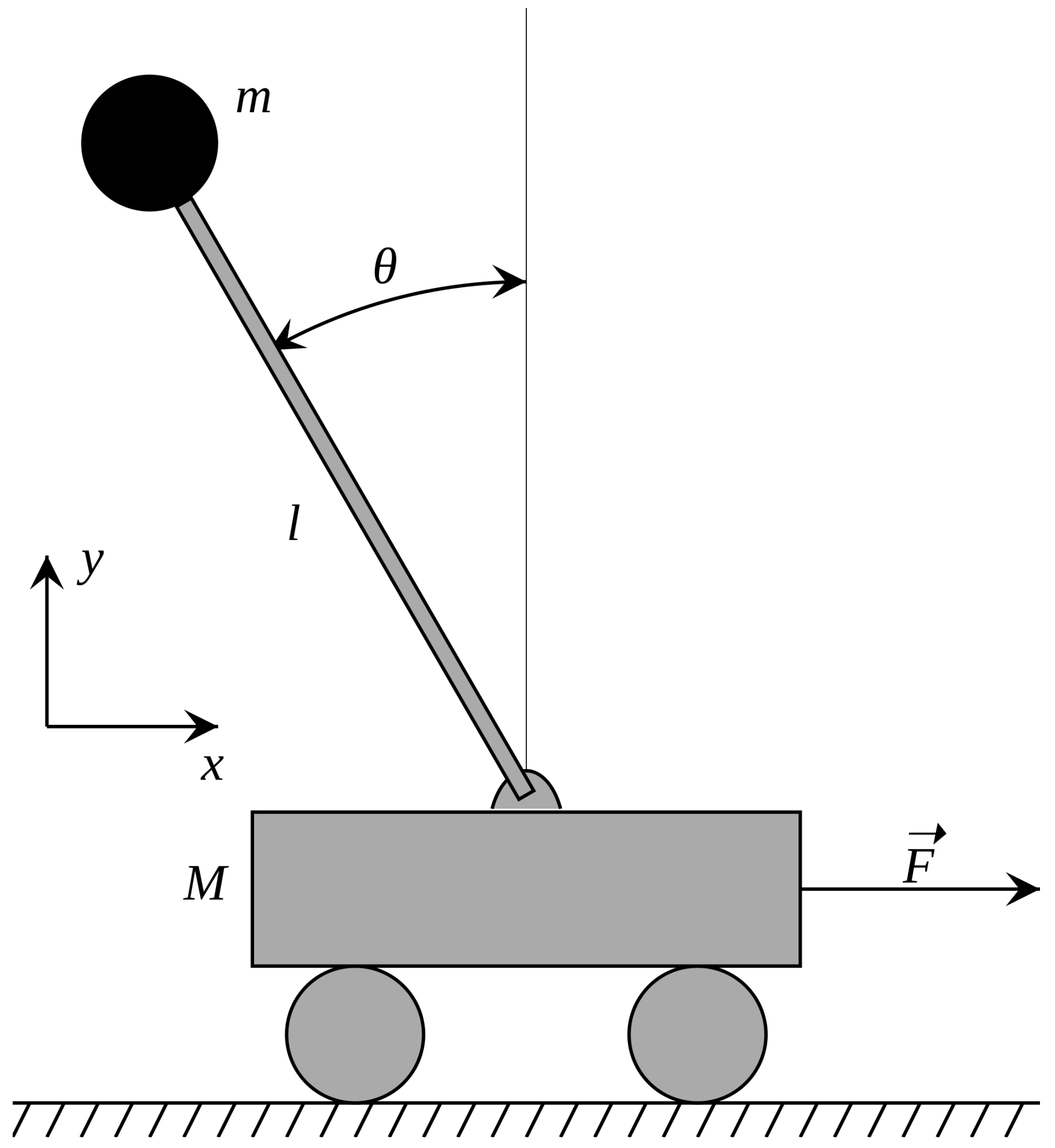
$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi\left[\sum_{t=0}^{T-1} c(s_t, a_t)\right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

No noise! No terminal cost $c_t(s_T)$!

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi \left[\sum_{t=0}^{T-1} c(s_t, a_t) \right]$

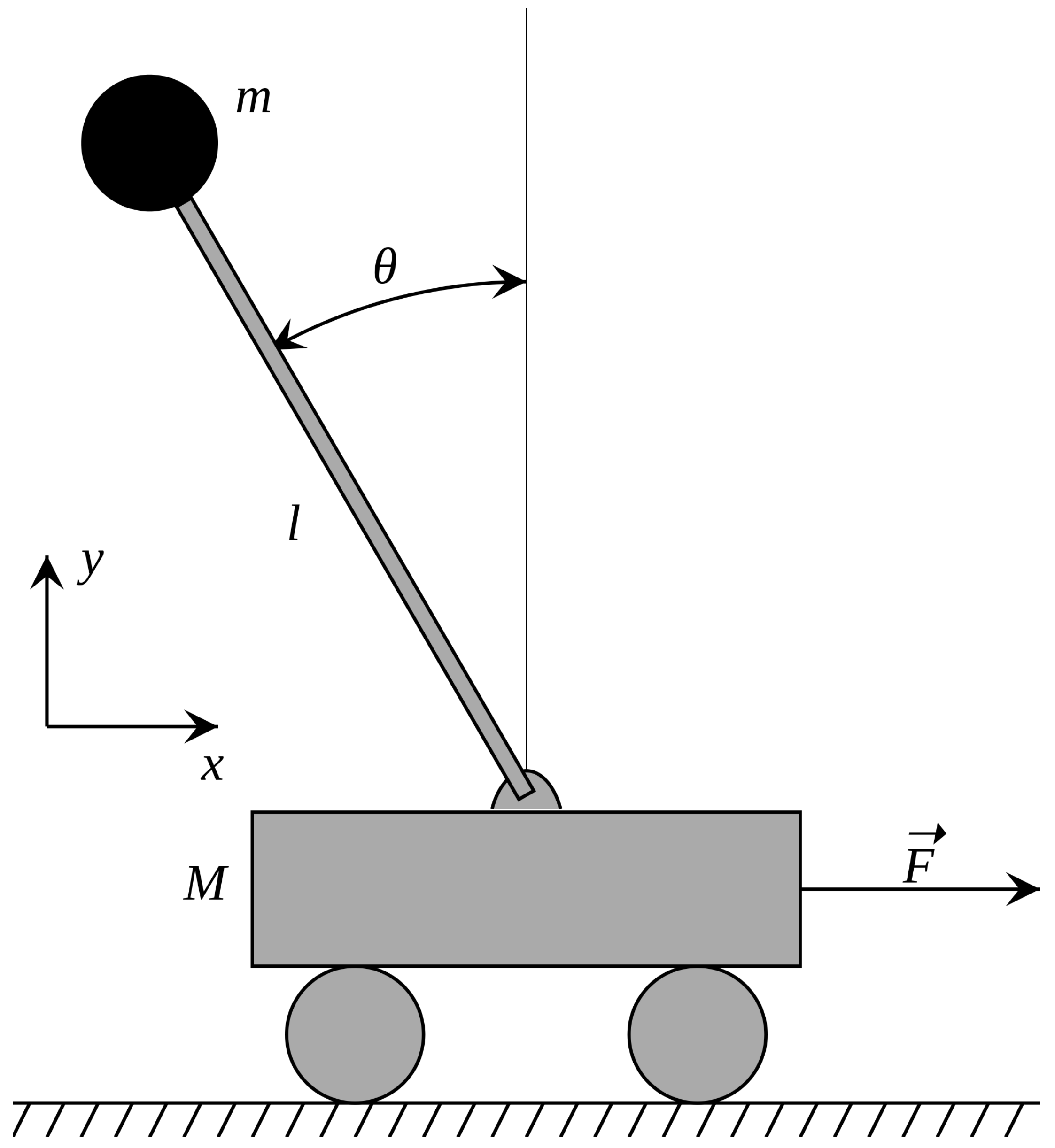
s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

No noise! No terminal cost $c_t(s_T)$!

Assumptions:

1. **We have black-box access to f & c:**

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi \left[\sum_{t=0}^{T-1} c(s_t, a_t) \right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

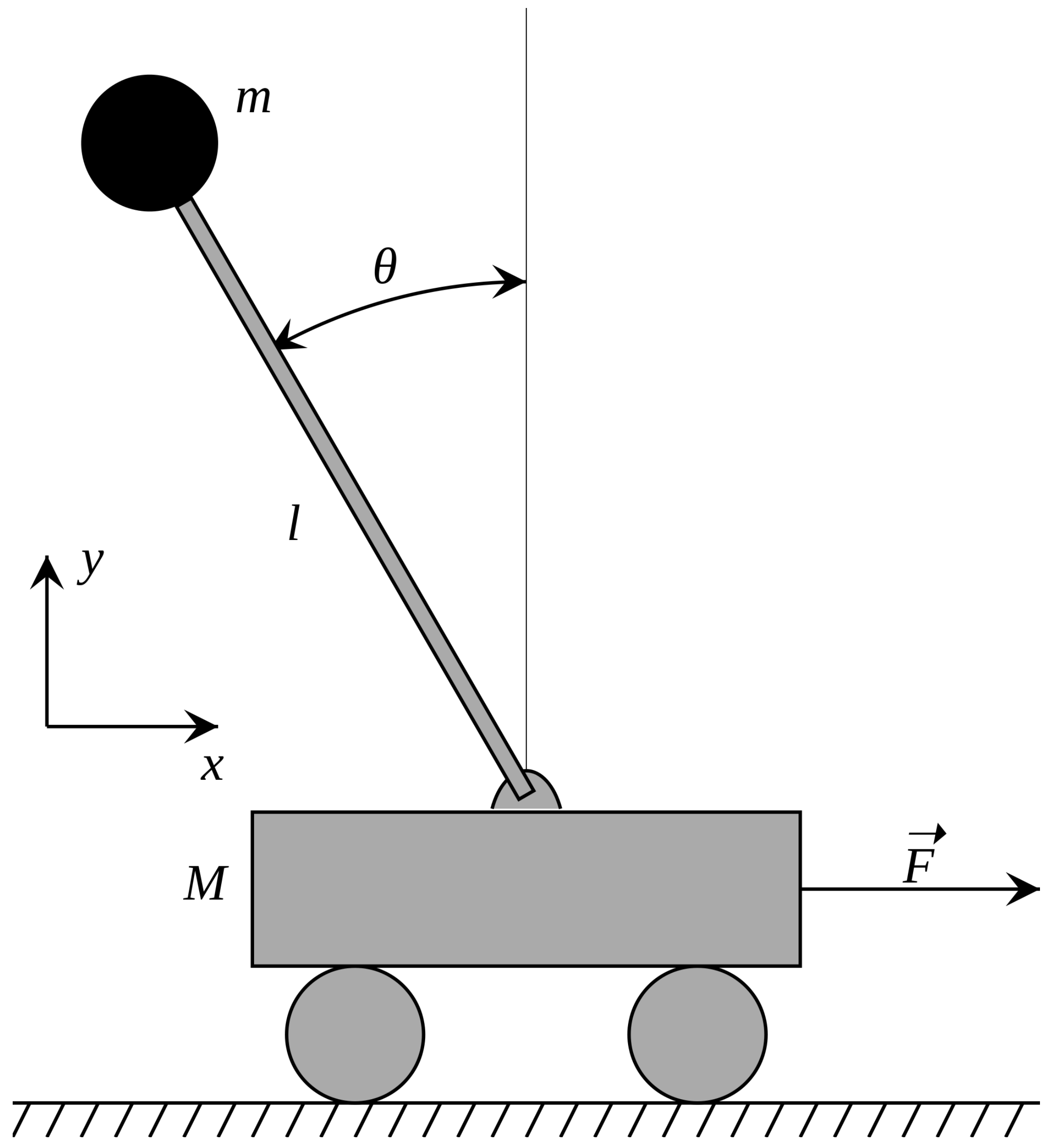
No noise! No terminal cost $c_t(s_T)$!

Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form but can be queried at any (s, a) to give s', c, where $s' = f(s, a), c = c(s, a)$

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi\left[\sum_{t=0}^{T-1} c(s_t, a_t)\right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

No noise! No terminal cost $c_t(s_T)$!

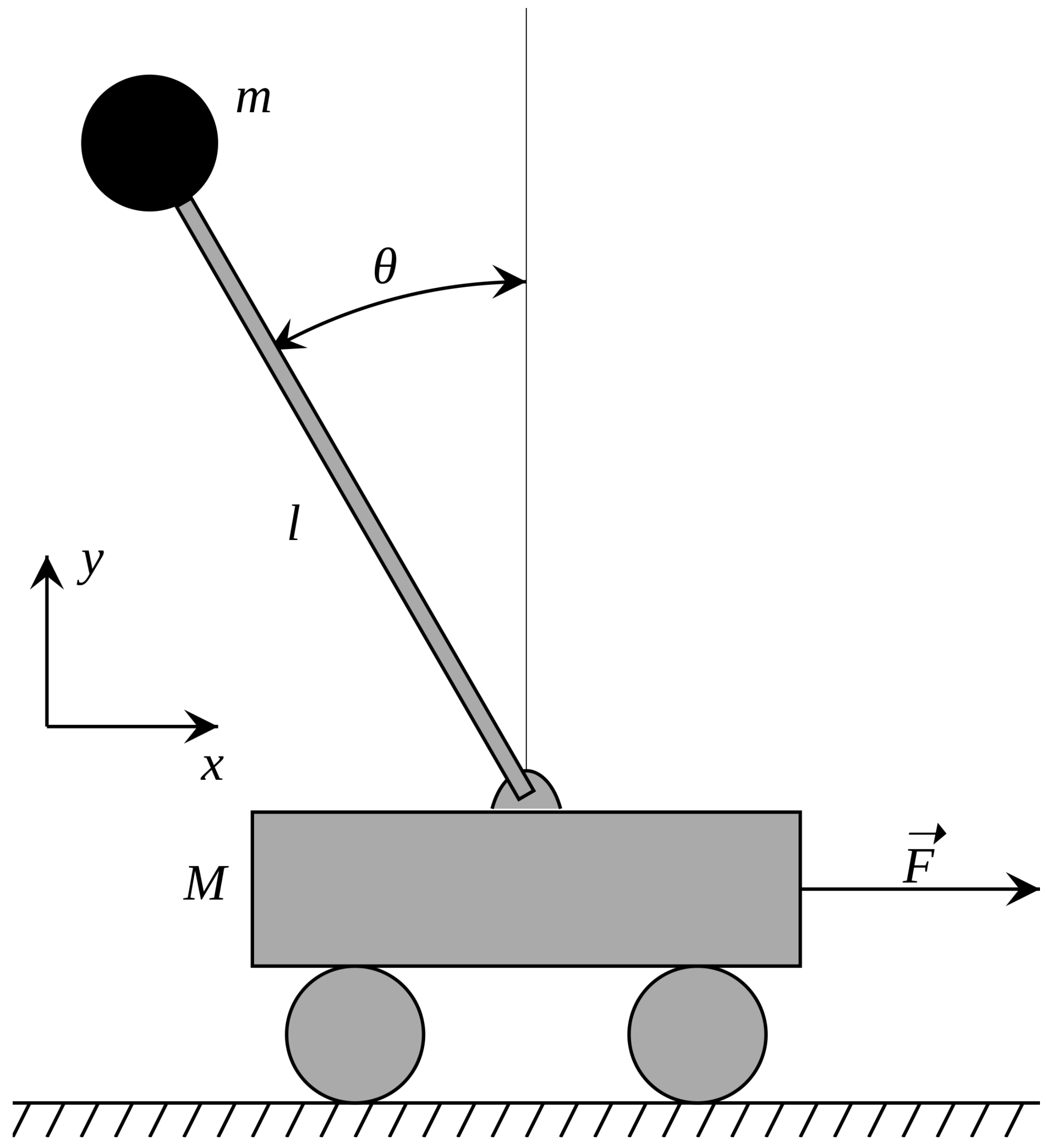
Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form but can be queried at any (s, a) to give s', c, where $s' = f(s, a), c = c(s, a)$

2. f is differentiable and c is twice differentiable

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(s = s^\star, a = a^\star)$

$$c(s_t, a_t) = d(a_t, a^\star) + d(s_t, s^\star)$$

minimize $\mathbb{E}_\pi\left[\sum_{t=0}^{T-1} c(s_t, a_t)\right]$

s.t. $s_{t+1} = f(s_t, a_t), \quad a_t = \pi(s_t), \quad s_0 \sim \mu_0$

No noise! No terminal cost $c_t(s_T)$!

Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form
but can be queried at any (s, a) to give s', c,
where $s' = f(s, a), c = c(s, a)$

2. f is differentiable and c is twice differentiable

$$\nabla_s f(s, a), \nabla_a f(s, a), \nabla_s c(s, a), \nabla_a c(s, a),$$
$$\nabla_s^2 c(s, a), \nabla_a^2 c(s, a), \nabla_{s,a}^2 c(s, a)$$

Local Linearization of Dynamics

Local Linearization of Dynamics

Assume that all possible initial states s_0 are close to $s^\star$ and can be kept there with actions close to $a^\star$

Local Linearization of Dynamics

Assume that all possible initial states s_0 are close to $s^\star$ and can be kept there with actions close to $a^\star$

We can approximate $f(s, a)$ locally with a first-order Taylor expansion:

$$f(s, a) \approx f(s^\star, a^\star) + \nabla_s f(s^\star, a^\star)\left(s - s^\star\right) + \nabla_a f(s^\star, a^\star)(a - a^\star)$$

Local Linearization of Dynamics

Assume that all possible initial states s_0 are close to $s^\star$ and can be kept there with actions close to $a^\star$

We can approximate $f(s, a)$ locally with a first-order Taylor expansion:

$$f(s,a) \approx f(s^\star, a^\star) + \nabla_s f(s^\star, a^\star)\left(s - s^\star\right) + \nabla_a f(s^\star, a^\star)(a - a^\star)$$

where:

$$\nabla_s f(s,a) \in \mathbb{R}^{d\times d}, \nabla_s f(s,a)[i,j] = \frac{\partial f[i]}{\partial s[j]}(s,a), \quad \nabla_u f(s,a) \in \mathbb{R}^{d\times k}, \nabla_a f(s,a)[i,j] = \frac{\partial f[i]}{\partial a[j]}(s,a)$$

Local Linearization of Cost Function

Local Linearization of Cost Function

We can approximate $c(s, a)$ locally at $(s^\star, a^\star)$ with second-order Taylor expansion:

Local Linearization of Cost Function

We can approximate $c(s, a)$ locally at $(s^\star, a^\star)$ with second-order Taylor expansion:

$$c(s,a) \approx c(s^\star, a^\star) + \nabla_s c(s^\star, a^\star)^\top (s - s^\star) + \nabla_a c(s^\star, a^\star)^\top (a - a^\star)$$
$$+\frac{1}{2}(s - s^\star)^\top \nabla_s^2 c(s^\star, a^\star)(s - s^\star) + \frac{1}{2}(a - a^\star)^\top \nabla_a^2 c(s^\star, a^\star)(a - a^\star) + (a - a^\star)^\top \nabla_{a,s}^2 c(s, a)(s - s^\star)$$

Local Linearization of Cost Function

We can approximate $c(s, a)$ locally at $(s^\star, a^\star)$ with second-order Taylor expansion:

$$c(s, a) \approx c(s^\star, a^\star) + \nabla_s c(s^\star, a^\star)^\top (s - s^\star) + \nabla_a c(s^\star, a^\star)^\top (a - a^\star)$$

$$+\frac{1}{2}(s - s^\star)^\top \nabla_s^2 c(s^\star, a^\star)(s - s^\star) + \frac{1}{2}(a - a^\star)^\top \nabla_a^2 c(s^\star, a^\star)(a - a^\star) + (a - a^\star)^\top \nabla_{a,s}^2 c(s, a)(s - s^\star)$$

$$\nabla_s c(s, a) \in \mathbb{R}^d, \quad \nabla_s c(s, a)[i] = \frac{\partial c}{\partial s[i]}(s, a),$$

$$\nabla_a c(s, a) \in \mathbb{R}^k, \quad \nabla_a c(s, a)[i] = \frac{\partial c}{\partial a[i]}(s, a),$$

$$\nabla_s^2 c(s, a) \in \mathbb{R}^{d \times d}, \quad \nabla_s^2 c(s, a)[i, j] = \frac{\partial^2 c}{\partial s[i] \partial s[j]}(s, a),$$

$$\nabla_{a,s}^2 c(s, a) \in \mathbb{R}^{k \times d}, \quad \nabla_{a,s}^2 c(s, a)[i, j] = \frac{\partial^2 c}{\partial a[i] \partial s[j]}(s, a)$$

Local Linearization: Putting it all Together

$$c(s,a) \approx c(s^\star, a^\star) + \nabla_s c(s^\star, a^\star)^\top (s - s^\star) + \nabla_a c(s^\star, a^\star)^\top (a - a^\star)$$
$$+\frac{1}{2}(s - s^\star)^\top \nabla_s^2 c(s^\star, a^\star)(s - s^\star) + \frac{1}{2}(a - a^\star)^\top \nabla_a^2 c(s^\star, a^\star)(a - a^\star) + (a - a^\star)^\top \nabla_{a,s}^2 c(s,a)(s - s^\star)$$

$$f(s,a) \approx f(s^\star, a^\star) + \nabla_s f(s^\star, a^\star)\left(s - s^\star\right) + \nabla_a f(s^\star, a^\star)(a - a^\star)$$

Local Linearization: Putting it all Together

$$c(s,a) \approx c(s^\star,a^\star) + \nabla_s c(s^\star,a^\star)^\top (s - s^\star) + \nabla_a c(s^\star,a^\star)^\top (a - a^\star)$$
$$+\frac{1}{2}(s-s^\star)^\top \nabla_s^2 c(s^\star,a^\star)(s-s^\star) + \frac{1}{2}(a-a^\star)^\top \nabla_a^2 c(s^\star,a^\star)(a-a^\star) + (a-a^\star)^\top \nabla_{a,s}^2 c(s,a)(s-s^\star)$$

$$f(s,a) \approx f(s^\star,a^\star) + \nabla_s f(s^\star,a^\star)\left(s-s^\star\right) + \nabla_a f(s^\star,a^\star)(a-a^\star)$$

Rearranging terms, we get back to the following formulation:

$$\arg \min_{\pi_0,\ldots,\pi_{T-1}:\mathbb{R}^d\to\mathbb{R}^k} \quad \mathbb{E}\left[\sum_{t=0}^{T-1}(s_t{}^\top Q s_t + a_t{}^\top R a_t + a_t{}^\top M s_t + s_t{}^\top q + a_t{}^\top r + c)\right]$$

such that $\quad s_{t+1} = As_t + Ba_t + v\,, \quad s_0 \sim \mu_0\,, \quad a_t = \pi_t(s_t)$

(HW3 problem)

Summary So far:

For tasks such as balancing near goal state $(s^\star, a^\star)$,
we can perform **first order Taylor expansion on** $f(s, a)$,
and **second order Taylor expansion on** $c(s, a)$ around the balancing point $(s^\star, a^\star)$

$$\arg \min_{\pi_0, \ldots, \pi_{T-1} : \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E} \left[\sum_{t=0}^{T-1} (s_t^\top Q s_t + a_t^\top R a_t + a_t^\top M s_t + s_t^\top q + a_t^\top r + c) \right]$$

such that $\quad s_{t+1} = A s_t + B a_t + v \,, \quad s_0 \sim \mu_0 \,, \quad a_t = \pi_t(s_t)$

Summary So far:

For tasks such as balancing near goal state $(s^\star, a^\star)$,
we can perform **first order Taylor expansion on** $f(s, a)$,
and **second order Taylor expansion on** $c(s, a)$ around the balancing point $(s^\star, a^\star)$

$$\arg \min_{\pi_0, \ldots, \pi_{T-1}: \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E} \left[\sum_{t=0}^{T-1} (s_t^\top Q s_t + a_t^\top R a_t + a_t^\top M s_t + s_t^\top q + a_t^\top r + c) \right]$$

such that $s_{t+1} = A s_t + B a_t + v\,, \quad s_0 \sim \mu_0\,, \quad a_t = \pi_t(s_t)$

Last step: checking some practical issues

Locally Convexifying the Cost Function

Locally Convexifying the Cost Function

Note that $c(s, a)$ might not even be convex;

So, $\nabla_s^2 c(s^\star, a^\star)$ & $\nabla_a^2 c(s^\star, a^\star)$ may not be positive definite

Locally Convexifying the Cost Function

Note that $c(s, a)$ might not even be convex;

So, $\nabla_s^2 c(s^\star, a^\star)$ & $\nabla_a^2 c(s^\star, a^\star)$ may not be positive definite

In practice, we force them to be positive definite:

Locally Convexifying the Cost Function

Note that $c(s, a)$ might not even be convex;

So, $\nabla_s^2 c(s^\star, a^\star)$ & $\nabla_a^2 c(s^\star, a^\star)$ may not be positive definite

In practice, we force them to be positive definite:

Given a *symmetric matrix* $H \in \mathbb{R}^{d \times d}$,

we compute the eigen-decomposition $H = \sum_{i=1}^{d} \sigma_i u_i u_i^\top$, and we approximate H as

$$H \approx \sum_{i=1}^{d} \mathbf{1}(\sigma_i > 0)\sigma_i u_i u_i^\top + \lambda I,$$

for some small $\lambda > 0$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (s, a), the black boxes outputs s', c, where

$$s' = f(s, a), c = c(s, a)$$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (s, a), the black boxes outputs s', c, where

$$s' = f(s, a), c = c(s, a)$$

Compute gradient using finite differencing:

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (s, a), the black boxes outputs s', c, where

$$s' = f(s, a), c = c(s, a)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial s[j]}(s, a) \approx \frac{f(s + \delta_j, a)[i] - f(s - \delta_j, a)[i]}{2\delta}, \text{ where } \delta_j = [0,\ldots,0, \underbrace{\delta}_{j'th \text{ entry}}, 0,\ldots 0]^\top$$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (s, a), the black boxes outputs s', c, where

$$s' = f(s, a), c = c(s, a)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial s[j]}(s, a) \approx \frac{f(s + \delta_j, a)[i] - f(s - \delta_j, a)[i]}{2\delta}, \text{ where } \delta_j = [0, \ldots, 0, \underbrace{\delta}_{j'th \text{ entry}}, 0, \ldots 0]^\top$$

To compute second derivative, e.g., $\frac{\partial^2 c}{\partial a[i] \partial s[j]}(s, a)$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (s, a), the black boxes outputs s', c, where

$$s' = f(s, a), c = c(s, a)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial s[j]}(s, a) \approx \frac{f(s + \delta_j, a)[i] - f(s - \delta_j, a)[i]}{2\delta}, \text{ where } \delta_j = [0, \ldots, 0, \underbrace{\delta}_{j'th \text{ entry}}, 0, \ldots 0]^\top$$

To compute second derivative, e.g., $\frac{\partial^2 c}{\partial a[i] \partial s[j]}(s, a)$

First implement finite differencing procedure for $\partial c / \partial a[i]$, and then perform another finite differencing with respect to $s[j]$ on top of the first finite differencing procedure for $\partial c / \partial a[i]$

Summary for local linearization approach

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(s, a)$
and second order Taylor expansion on $c(s, a)$, both around the balancing point $(s^\star, a^\star)$

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(s, a)$
and second order Taylor expansion on $c(s, a)$, both around the balancing point $(s^\star, a^\star)$

2. Force Hessians $\nabla_s^2 c(s, a)$ & $\nabla_a^2 c(s, a)$ to be positive definite

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(s, a)$
and second order Taylor expansion on $c(s, a)$, both around the balancing point $(s^\star, a^\star)$

2. Force Hessians $\nabla_s^2 c(s, a)$ & $\nabla_a^2 c(s, a)$ to be positive definite

3. Leverage finite differences to approximate gradients and Hessians

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(s, a)$
and second order Taylor expansion on $c(s, a)$, both around the balancing point $(s^\star, a^\star)$

2. Force Hessians $\nabla_s^2 c(s, a)$ & $\nabla_a^2 c(s, a)$ to be positive definite

3. Leverage finite differences to approximate gradients and Hessians

4. The approximation is an LQR, so we know how to compute the optimal policy

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ Locally linearization
- Iterative LQR

Limits of Local Linearization

Limits of Local Linearization

Local linearization can work if s_0 is very close to $s^\star$ and stays there with near-optimal (i.e., near-$a^\star$) actions

Limits of Local Linearization

Local linearization can work if s_0 is very close to $s^\star$ and stays there with near-optimal (i.e., near-$a^\star$) actions

But when s_t is far away from $s^\star$ or a_t needs to be far from $a^\star$ for any t, first/second-order Taylor expansion is not accurate anymore

Idea of Iterative LQR

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(s^\star, a^\star)$, linearize/quadratize around some other $(\bar{s}, \bar{a})$

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(s^\star, a^\star)$, linearize/quadratize around some other $(\bar{s}, \bar{a})$

In fact, we can even linearize/quadratize around different points $(\bar{s}_t, \bar{a}_t)$ at each t

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(s^\star, a^\star)$, linearize/quadratize around some other $(\bar{s}, \bar{a})$

In fact, we can even linearize/quadratize around different points $(\bar{s}_t, \bar{a}_t)$ at each t

After linearization and quadratization at time t around waypoint $(\bar{s}_t, \bar{a}_t)$, $\forall t$, re-arranging terms gives:

$$\arg \min_{\pi_0, \ldots, \pi_{T-1}: \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{t=0}^{T-1} (s_t^\top Q_t s_t + a_t^\top R_t a_t + a_t^\top M_t s_t + s_t^\top q_t + a_t^\top r_t + c_t)\right]$$

such that $s_{t+1} = A_t s_t + B_t a_t + v_t, \quad s_0 \sim \mu_0, \quad a_t = \pi_t(s_t)$

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(s^\star, a^\star)$, linearize/quadratize around some other $(\bar{s}, \bar{a})$

In fact, we can even linearize/quadratize around different points $(\bar{s}_t, \bar{a}_t)$ at each t

After linearization and quadratization at time t around waypoint $(\bar{s}_t, \bar{a}_t)$, $\forall t$, re-arranging terms gives:

$$\arg\min_{\pi_0,\ldots,\pi_{T-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{t=0}^{T-1} (s_t^\top Q_t s_t + a_t^\top R_t a_t + a_t^\top M_t s_t + s_t^\top q_t + a_t^\top r_t + c_t)\right]$$

such that $s_{t+1} = A_t s_t + B_t a_t + v_t\,, \quad s_0 \sim \mu_0\,, \quad a_t = \pi_t(s_t)$

Time-dependent LQR problem: we know the solution

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(s^\star, a^\star)$, linearize/quadratize around some other $(\bar{s}, \bar{a})$

In fact, we can even linearize/quadratize around different points $(\bar{s}_t, \bar{a}_t)$ at each t

After linearization and quadratization at time t around waypoint $(\bar{s}_t, \bar{a}_t)$, $\forall t$, re-arranging terms gives:

$$\arg\min_{\pi_0,\ldots,\pi_{T-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{t=0}^{T-1} (s_t^\top Q_t s_t + a_t^\top R_t a_t + a_t^\top M_t s_t + s_t^\top q_t + a_t^\top r_t + c_t)\right]$$

such that $s_{t+1} = A_t s_t + B_t a_t + v_t, \quad s_0 \sim \mu_0, \quad a_t = \pi_t(s_t)$

Time-dependent LQR problem: we know the solution

Question: how to choose the waypoints $(\bar{s}_t, \bar{a}_t)$ to get the best approximation/solution?

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

For each t, linearize $f(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s, a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_t is not

For each t, linearize $f(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s,a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_t is not

For each t, linearize $f(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s, a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

For each t, quadratize $c_t(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$:

$$c_t(s,a) \approx \frac{1}{2} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_{s,a}^2 c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_{a,s}^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_a^2 c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix} + \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_a c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} + c(\bar{s}_t^i, \bar{a}_t^i)$$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_t is not

For each t, linearize $f(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s, a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

For each t, quadratize $c_t(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$:

$$c_t(s, a) \approx \frac{1}{2} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_{s,a}^2 c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_{a,s}^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_a^2 c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix} + \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_a c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} + c(\bar{s}_t^i, \bar{a}_t^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{T-1}^i$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0, 1, \ldots$

Note that although true f is stationary, its approximation f_t is not

For each t, linearize $f(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s, a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

For each t, quadratize $c_t(s, a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$:

$$c_t(s, a) \approx \frac{1}{2} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_{s,a}^2 c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_{a,s}^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_a^2 c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix} + \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_a c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} + c(\bar{s}_t^i, \bar{a}_t^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{T-1}^i$

Set new nominal trajectory: $\bar{s}_0^{i+1} = \bar{s}_0, \;\; \bar{a}_t^{i+1} = \pi_t^i(\bar{s}_t^{i+1}),$ and $\bar{s}_{t+1}^{i+1} = f(\bar{s}_t^{i+1}, \bar{a}_t^{i+1})$

Iterative LQR

Recall $s_0 \sim \mu_0$; denote $\mathbb{E}_{s_0 \sim \mu_0}[s_0] = \bar{s}_0$

Initialize $\bar{a}_0^0, \ldots, \bar{a}_{T-1}^0$, (e.g., by local linearization)

Generate nominal trajectory: $\bar{s}_0^0 = \bar{s}_0, \bar{a}_0^0, \ldots, \bar{a}_t^0, \bar{s}_{t+1}^0 = f(\bar{s}_t^0, \bar{a}_t^0), \ldots, \bar{s}_{T-1}^0, \bar{a}_{T-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_t is not

For each t, linearize $f(s,a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$: $f_t(s,a) \approx f(\bar{s}_t^i, \bar{a}_t^i) + \nabla_s f(\bar{s}_t^i, \bar{a}_t^i)(s - \bar{s}_t^i) + \nabla_a f(\bar{s}_t^i, \bar{a}_t^i)(a - \bar{a}_t^i)$

For each t, quadratize $c_t(s,a)$ at $(\bar{s}_t^i, \bar{a}_t^i)$:

$$c_t(s,a) \approx \frac{1}{2}\begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_{s,a}^2 c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_{a,s}^2 c(\bar{s}_t^i, \bar{a}_t^i) & \nabla_a^2 c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix} + \begin{bmatrix} s - \bar{s}_t^i \\ a - \bar{a}_t^i \end{bmatrix}^\top \begin{bmatrix} \nabla_s c(\bar{s}_t^i, \bar{a}_t^i) \\ \nabla_a c(\bar{s}_t^i, \bar{a}_t^i) \end{bmatrix} + c(\bar{s}_t^i, \bar{a}_t^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{T-1}^i$

Set new nominal trajectory: $\bar{s}_0^{i+1} = \bar{s}_0, \;\; \bar{a}_t^{i+1} = \pi_t^i(\bar{s}_t^{i+1}),$ and $\bar{s}_{t+1}^{i+1} = f(\bar{s}_t^{i+1}, \bar{a}_t^{i+1})$

Note this is true f, not approximation

Practical Considerations of Iterative LQR:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{a}_0^i, \ldots, \bar{a}_{T-1}^i,$ and the latest computed controls $\bar{a}_0, \ldots, \bar{a}_{T-1}$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{a}_0^i, \ldots, \bar{a}_{T-1}^i,$ and the latest computed controls $\bar{a}_0, \ldots, \bar{a}_{T-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{a}_t^{i+1} := \alpha \, \bar{a}_t^i + (1 - \alpha)\bar{a}_t$ has the smallest cost,

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{a}^i_0, \ldots, \bar{a}^i_{T-1}$, and the latest computed controls $\bar{a}_0, \ldots, \bar{a}_{T-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{a}^{i+1}_t := \alpha\, \bar{a}^i_t + (1-\alpha)\bar{a}_t$ has the smallest cost,

$$\min_{\alpha\in[0,1]} \sum_{t=0}^{T-1} c(s_t, \bar{a}^{i+1}_t)$$

$$\text{s.t.} \quad s_{t+1} = f(s_t, \bar{a}^{i+1}_t), \quad \bar{a}^{i+1}_t = \alpha\bar{a}^i_t + (1-\alpha)\bar{a}_t, \quad s_0 = \bar{s}_0$$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{a}_0^i, \ldots, \bar{a}_{T-1}^i,$ and the latest computed controls $\bar{a}_0, \ldots, \bar{a}_{T-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{a}_t^{i+1} := \alpha\, \bar{a}_t^i + (1 - \alpha)\bar{a}_t$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{t=0}^{T-1} c(s_t, \bar{a}_t^{i+1})$$

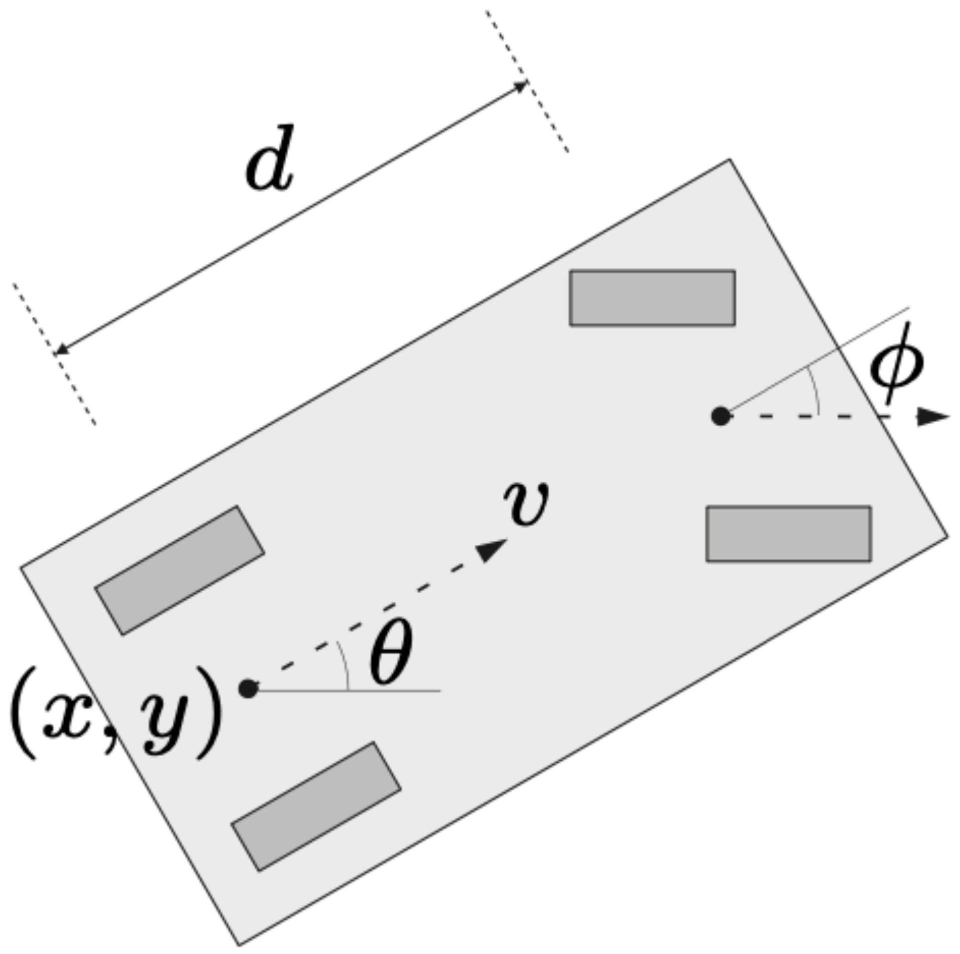
s.t. $\quad s_{t+1} = f(s_t, \bar{a}_t^{i+1}), \quad \bar{a}_t^{i+1} = \alpha \bar{a}_t^i + (1 - \alpha)\bar{a}_t, \quad s_0 = \bar{s}_0$

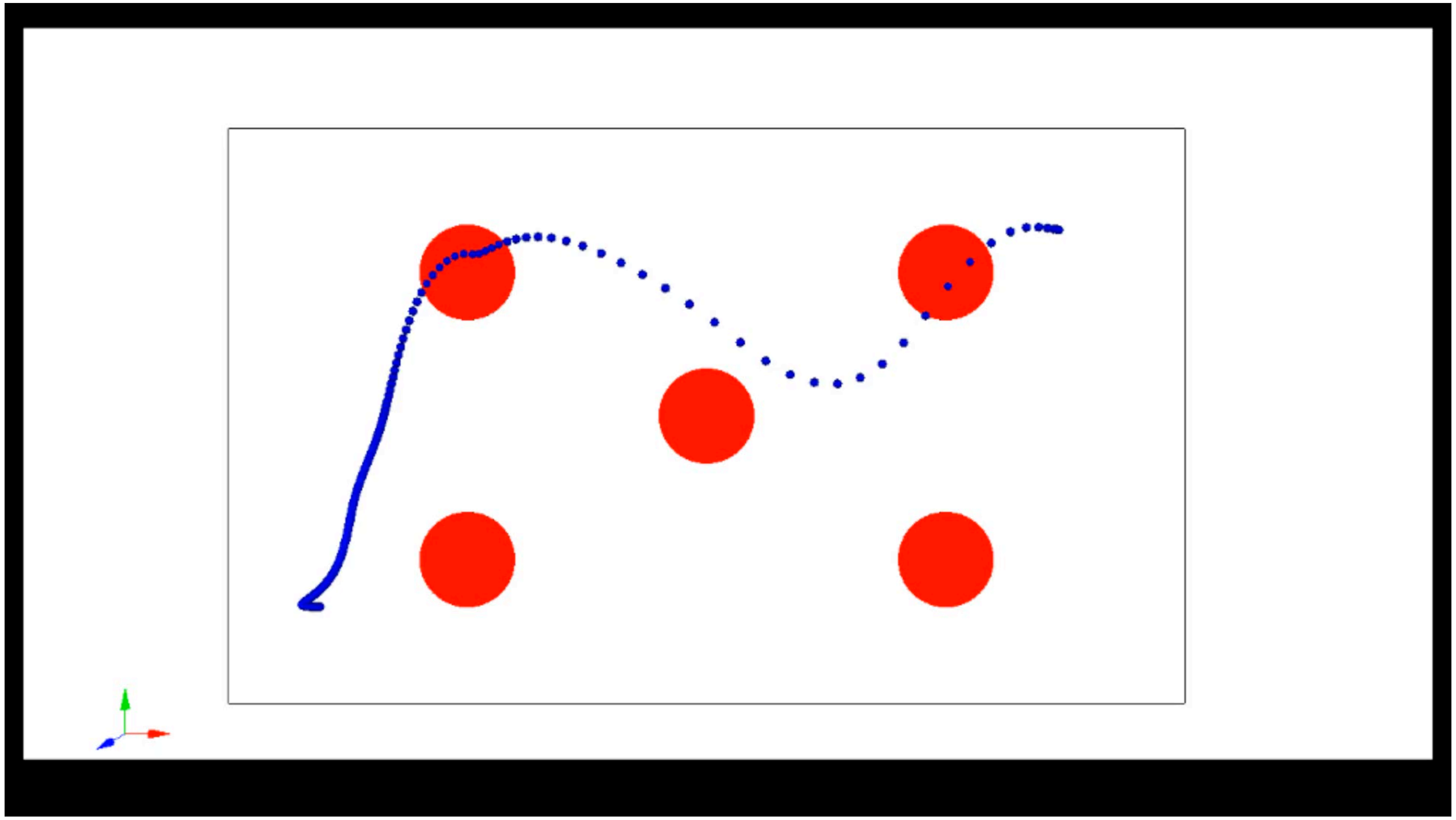
This optimization is tractable because it is 1-dimensional!

Example:

2-d car navigation

Cost function is designed such that it gets to the goal without colliding with obstacles (in red)

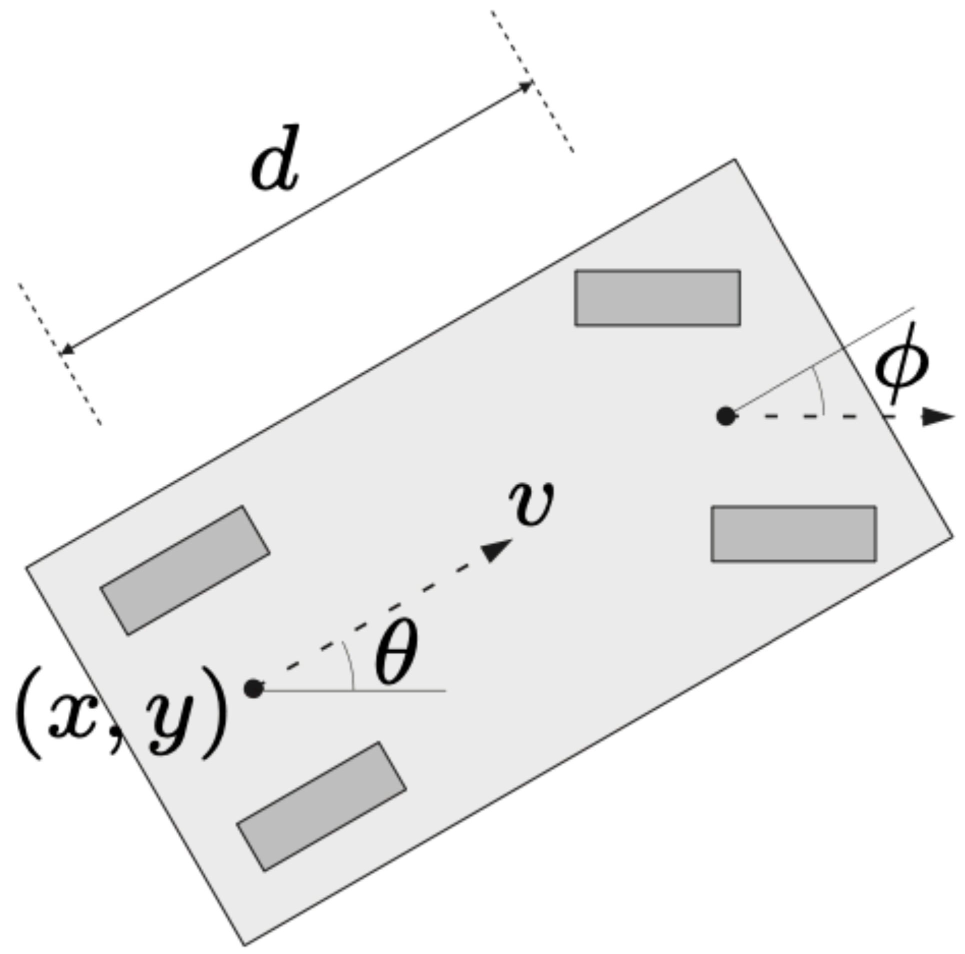


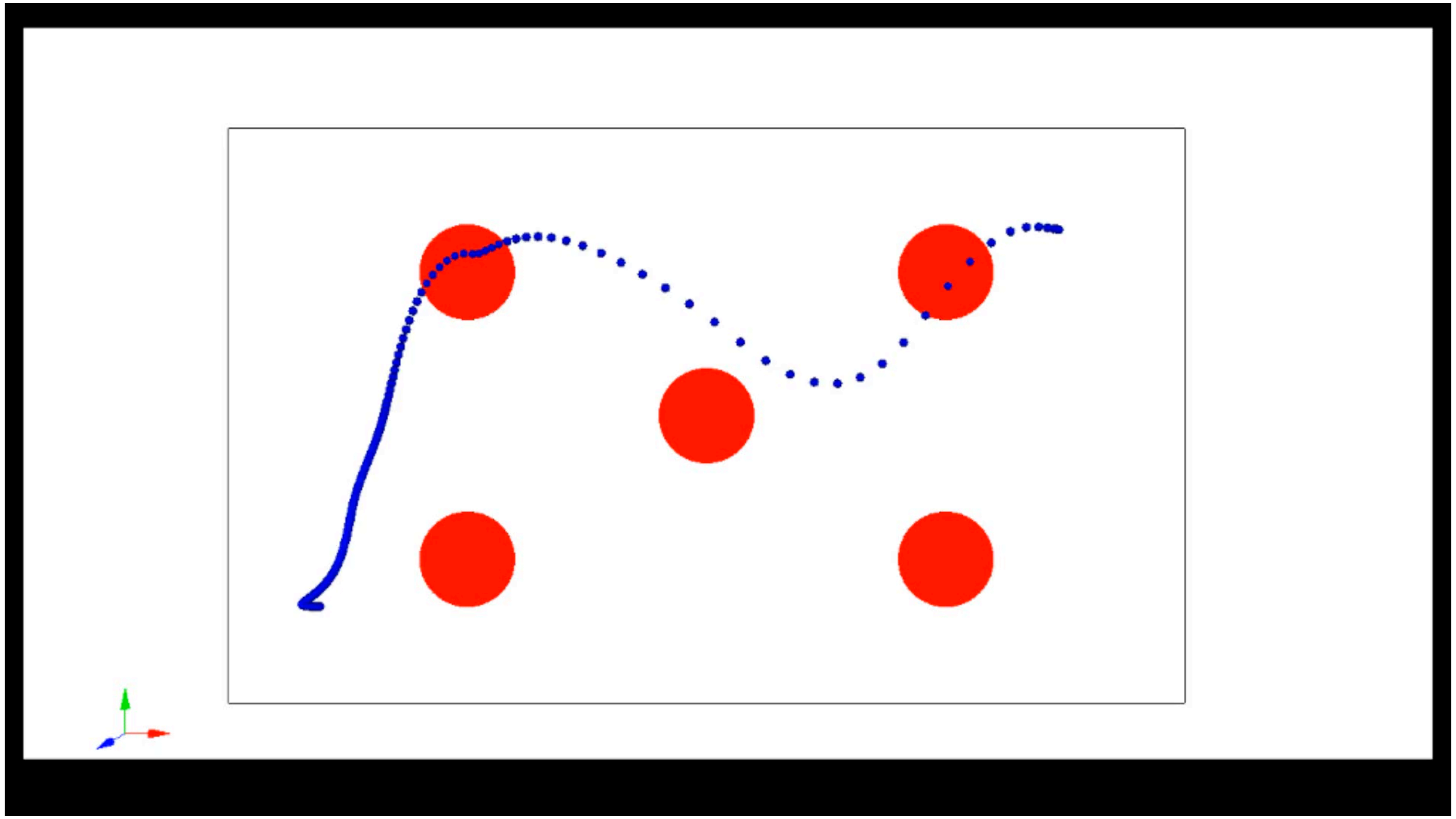


Example:

2-d car navigation

Cost function is designed such that it gets to the goal without colliding with obstacles (in red)





Summary:

Summary:

Local Linearization:

Approximate an LQR at the balance (goal) position $(s^\star, a^\star)$ and then solve the approximated LQR

Summary:

Local Linearization:

Approximate an LQR at the balance (goal) position $(s^\star, a^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Summary:

Local Linearization:

Approximate an LQR at the balance (goal) position $(s^\star, a^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

(2) computing a new nominal trajectory using the optimal policy of the LQR

Summary:

Local Linearization:

Approximate an LQR at the balance (goal) position $(s^\star, a^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

(2) computing a new nominal trajectory using the optimal policy of the LQR

Computes a locally optimal (in policy space) solution for a large class of nonlinear control problems

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Today's summary:

Today’s summary:

Local linearization

- Allows us to approximately optimally control any system near its optimum

Today’s summary:

Local linearization

- Allows us to approximately optimally control any system near its optimum

Iterative LQR

- Uses LQR approximation to find locally optimal nonlinear control solution

Today's summary:

Local linearization

- Allows us to approximately optimally control any system near its optimum

Iterative LQR

- Uses LQR approximation to find locally optimal nonlinear control solution

Next time:

- Full RL!

Today's summary:

Local linearization

- Allows us to approximately optimally control any system near its optimum

Iterative LQR

- Uses LQR approximation to find locally optimal nonlinear control solution

Next time:

- Full RL!

1-minute feedback form: https://bit.ly/3RHtlxy