Carnegie Mellon

School of Computer Science

Deep Reinforcement Learning and Control

iLQR

Katerina Fragkiadaki

Optimal Control (Open Loop)

- The optimal control problem:

$$\begin{aligned}\min_{x,u} \sum_{t=0}^{T} c_t(x_t, u_t) \\ \text{s.t. } x_0 = \bar{x}_0 \\ x_{t+1} = f(x_t, u_t) \quad t = 0, ..., T-1\end{aligned}$$

Optimal Control (Open Loop)

- The optimal control problem:

$$\min_{x,u} \sum_{t=0}^{T} c_t(x_t, u_t)$$
$$\text{s.t.} \;\; x_0 = \bar{x}_0$$
$$x_{t+1} = f(x_t, u_t) \quad t = 0, ..., T-1$$

- Solution:
  - Sequence of controls u and resulting state sequence x
- In general non-convex optimization problem, can be solved with sequential convex programming (SCP): https://stanford.edu/class/ee364b/lectures/seq_slides.pdf

Optimal Control (Closed Loop a.k.a. MPC)

Given: $\bar{x}_0$

For $t = 0, 1, 2, ..., T$

- Solve

$$\begin{aligned} \min_{x,u} & \sum_{k=t}^{T} c_k(x_k, u_k) \\ \text{s.t. } & x_{k+1} = f(x_k, u_k), \quad \forall k \in \{t, t+1, ..., T-1\} \\ & x_t = \bar{x}_t \end{aligned}$$

- Execute u_t
- Observe resulting state, $\bar{x}_{t+1}$
- Initialize with solution from $t-1$ to solve fast at time t

Shooting methods vs collocation methods

Collocation Method: optimize over actions and states, with constraints

$$\min_{u_1,\ldots,u_T,x_1,\ldots,x_T} \sum_{t=1}^{T} c(x_t, u_t) \ \text{ s.t } x_t = f(x_{t-1}, u_{t-1})$$

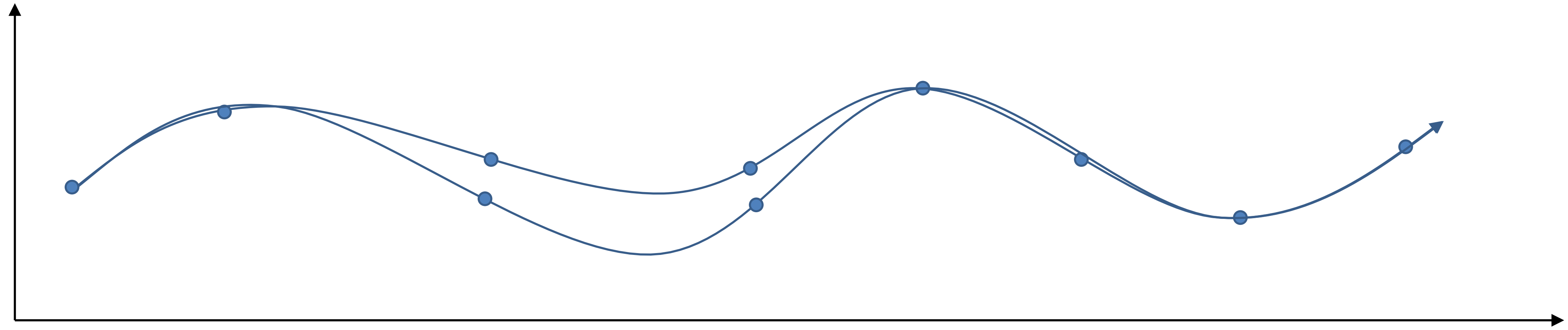
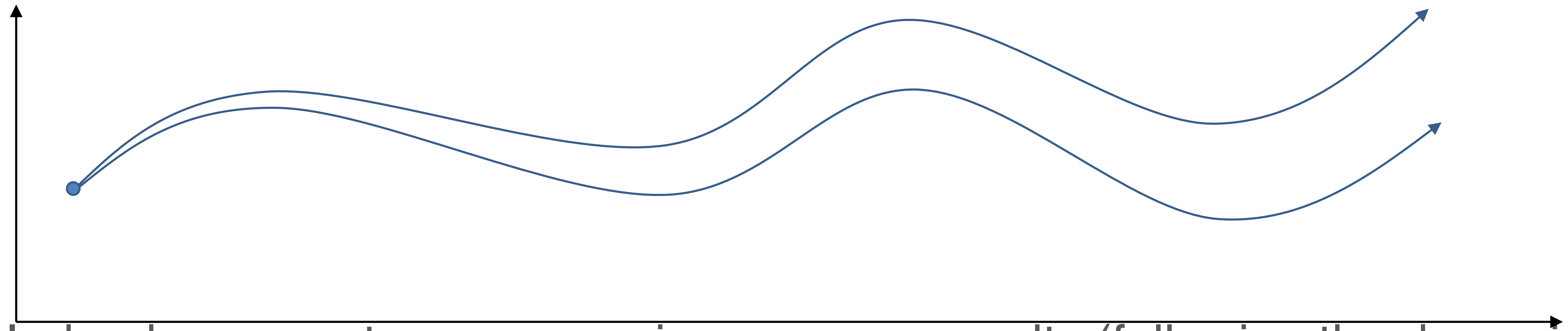


Diagram: Sergey Levine

Shooting methods vs collocation methods

Shooting Method: optimize over actions only

$$\min_{u_1,\ldots,u_T} c(x_1,u_1) + c(f(x_1,u_1),u_2) + \cdots + c(f(f(\ldots)\ldots),u_T)$$



Indeed, x are not necessary since every u results (following the dynamics) in a state sequence x, for which in turn the cost can be computed

- Not clear how to initialize in a way that the resulting state is close to a goal state

Diagram: Sergey Levine

Bellman's Curse of Dimensionality

- n-dimensional state space
- Number of states grows exponentially in n (for fixed number of discretization levels per coordinate)
- In practice
  - Discretization is considered only computationally feasible up to 5 or 6 dimensional state spaces even when using
    - Variable resolution discretization
    - Highly optimized implementations

Linear case: LQR

- Very special case: Optimal Control for Linear Dynamic Systems and Quadratic Cost (a.k.a. LQ setting)
- Can solve continuous state-space optimal control problem exactly
- Running time: $O(Tn^3)$

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + c(f(f(\ldots)\ldots),\mathbf{u}_T)$$

$$\underbrace{f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} + \mathbf{f}_t}_{\text{linear}} \qquad \underbrace{c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{C}_t \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} + \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{c}_t}_{\text{quadratic}}$$

Linear dynamics: Newtonian Dynamics

- $x_{t+1} = x_t + \Delta t \dot{x}_t + \Delta t^2 F_x$
- $y_{t+1} = y_t + \Delta t \dot{y}_t + \Delta t^2 F_y$
- $\dot{x}_{t+1} = \dot{x}_t + \Delta t F_x$
- $\dot{y}_{t+1} = \dot{y}_t + \Delta t F_y$

What is the state x?

In most robotic tasks, state is hand engineered and includes:

- Robot: position and velocities of the robotic joints
- Object: position and velocity of the object being manipulated

Those are both known: the robot knows its state and we perceive the state of the objects in the world.

In tasks where we do not even want to bother with object state, we just concatenate the robot's state across multiple time steps to implicitly infer the interaction (collision with the object)

What is the cost $c(x_t, u_t)$

- $c(x_t, u_t) = \|x_t - x^*\| + \beta \|u_t\|$

x^* is the target state

- In the final time step, you can add a term with higher weight:

Final cost $c(x_T, u_T) = 2(\|x_T - x^*\| + \beta \|u_T\|)$

- For object manipulation, x^* includes not only desired pose of the end effector but also desired pose of the objects

Linear Quadratic Regulator (LQR)

$$\min_{\mathbf{u}_1,\dots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + c(f(f(\dots)\dots),\mathbf{u}_T)$$

$$c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{C}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{c}_t$$

$$f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \mathbf{f}_t$$

Definitions:

$Q(x_t, u_t)$: optimal action value function, optimal cost-to-go at state x_t as a function of u_t assuming we act optimal past step t

$V(x_t)$: optimal state value function, optimal cost-to-go from state x_t

$$V(x_t) = \min_{u_t} Q(x_t, u_t)$$

x_0: the initial state, known and given

Principle of Optimality

An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision. (See Bellman, 1957, Chap. III.3.)

Linear Quadratic Regulator (LQR)

$$\min_{\mathbf{u}_1,\dots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + c(f(f(\dots)\dots),\mathbf{u}_T)$$

$$c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{C}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{c}_t$$

$$f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \mathbf{f}_t$$

Value iteration: backward propagation!

Start from u_T and work backwards

Linear Quadratic Regulator (LQR)

$$\min_{\mathbf{u}_1,\dots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + \underbrace{c(f(f(\dots)\dots),\mathbf{u}_T)}_{\text{only term that depends on } \mathbf{u}_T}$$

$$c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{C}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{c}_t$$

$$f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \mathbf{f}_t$$

Cost matrices

for the last time step:

$$\mathbf{C}_T = \begin{bmatrix}\mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\\ \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\end{bmatrix}$$

$$\mathbf{c}_T = \begin{bmatrix}\mathbf{c}_{\mathbf{x}_T}\\ \mathbf{c}_{\mathbf{u}_T}\end{bmatrix}$$

Value iteration: backward propagation!

Start from u_T and work backwards

$$Q(\mathbf{x}_T,\mathbf{u}_T) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{C}_T \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix} + \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{c}_T$$

Linear Quadratic Regulator (LQR)

$$\min_{\mathbf{u}_1,\dots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + \underbrace{c(f(f(\dots)\dots),\mathbf{u}_T)}_{\text{only term that depends on } \mathbf{u}_T}$$

$$c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{C}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{c}_t$$

$$f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \mathbf{f}_t$$

Value iteration: backward propagation!

Start from u_T and work backwards

$$Q(\mathbf{x}_T,\mathbf{u}_T) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{C}_T \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix} + \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{c}_T$$

Cost matrices

for the last time step:

$$\mathbf{C}_T = \begin{bmatrix}\mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\\ \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\end{bmatrix}$$

$$\mathbf{c}_T = \begin{bmatrix}\mathbf{c}_{\mathbf{x}_T}\\ \mathbf{c}_{\mathbf{u}_T}\end{bmatrix}$$

Set derivative to zero (since we have a quadratic) to find minimizer u_T:

$$\nabla_{\mathbf{u}_T} Q(\mathbf{x}_T,\mathbf{u}_T) = \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}\mathbf{x}_T + \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{u}_T + \mathbf{c}_{\mathbf{u}_T}^T = 0$$

Linear Quadratic Regulator (LQR)

$$\min_{\mathbf{u}_1,\dots,\mathbf{u}_T} c(\mathbf{x}_1,\mathbf{u}_1) + c(f(\mathbf{x}_1,\mathbf{u}_1),\mathbf{u}_2) + \cdots + \underbrace{c(f(f(\dots)\dots),\mathbf{u}_T)}_{\text{only term that depends on } \mathbf{u}_T}$$

$$c(\mathbf{x}_t,\mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{C}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \mathbf{c}_t$$

$$f(\mathbf{x}_t,\mathbf{u}_t) = \mathbf{F}_t \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \mathbf{f}_t$$

Value iteration: backward propagation!

Start from u_T and work backwards

$$Q(\mathbf{x}_T,\mathbf{u}_T) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{C}_T \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix} + \begin{bmatrix}\mathbf{x}_T\\ \mathbf{u}_T\end{bmatrix}^T \mathbf{c}_T$$

Cost matrices

for the last time step:

$$\mathbf{C}_T = \begin{bmatrix}\mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\\ \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} & \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\end{bmatrix}$$

$$\mathbf{c}_T = \begin{bmatrix}\mathbf{c}_{\mathbf{x}_T}\\ \mathbf{c}_{\mathbf{u}_T}\end{bmatrix}$$

Set derivative to zero (since we have a quadratic) to find minimizer u_T:

$$\nabla_{\mathbf{u}_T} Q(\mathbf{x}_T,\mathbf{u}_T) = \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}\mathbf{x}_T + \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{u}_T + \mathbf{c}_{\mathbf{u}_T}^T = 0$$

$$\mathbf{u}_T = -\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}^{-1}\left(\mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}\mathbf{x}_T + \mathbf{c}_{\mathbf{u}_T}\right)$$

$$\mathbf{u}_T = \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T$$

$$\mathbf{K}_T = -\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}^{-1}\mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}$$

$$\mathbf{k}_T = -\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}^{-1}\mathbf{c}_{\mathbf{u}_T}$$

Linear Quadratic Regulator (LQR)

Remember: $V(x_t) = \min_{u_t} Q(x_t, u_t)$

Substituting the minimizer u_T into $Q(x_T, u_T)$ gives us $V(x_T)$!

Linear Quadratic Regulator (LQR)

Remember: $V(x_t) = \min_{u_t} Q(x_t, u_t)$

Substituting the minimizer u_T into $Q(x_T, u_T)$ gives us $V(x_T)$!

$$Q(\mathbf{x}_T, \mathbf{u}_T) = \text{const} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix}^T \mathbf{c}_T$$

$$\mathbf{u}_T = \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{c}_T$$

Linear Quadratic Regulator (LQR)

Remember: $V(x_t) = \min_{u_t} Q(x_t, u_t)$

Substituting the minimizer u_T into $Q(x_T, u_T)$ gives us $V(x_T)$!

$$Q(\mathbf{x}_T, \mathbf{u}_T) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{u}_T \end{bmatrix}^T \mathbf{c}_T \qquad \mathbf{u}_T = \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{c}_T$$

$$\begin{aligned} V(\mathbf{x}_T) =& \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T}\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{K}_T\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{K}_T\mathbf{x}_T + \\ & \mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{k}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{k}_T + \mathbf{x}_T^T\mathbf{c}_{\mathbf{x}_T} + \mathbf{x}_T^T\mathbf{K}_T^T\mathbf{c}_{\mathbf{u}_T} + \text{const} \end{aligned}$$

Linear Quadratic Regulator (LQR)

Remember: $V(x_t) = \min_{u_t} Q(x_t, u_t)$

Substituting the minimizer u_T into $Q(x_T, u_T)$ gives us $V(x_T)$!

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T\mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{c}_T$$

$$\begin{aligned} V(\mathbf{x}_T) = & \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T}\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{K}_T\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T}\mathbf{x}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{K}_T\mathbf{x}_T + \\ & \mathbf{x}_T^T\mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{k}_T + \frac{1}{2}\mathbf{x}_T^T\mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{k}_T + \mathbf{x}_T^T\mathbf{c}_{\mathbf{x}_T} + \mathbf{x}_T^T\mathbf{K}_T^T\mathbf{c}_{\mathbf{u}_T} + \text{const} \end{aligned}$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T\mathbf{V}_T\mathbf{x}_T + \mathbf{x}_T^T\mathbf{v}_T$$

optimal cost-to-go as a function of the final state

$$\mathbf{V}_T = \mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} + \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{K}_T + \mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} + \mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{K}_T$$

$$\mathbf{v}_T = \mathbf{c}_{\mathbf{x}_T} + \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T}\mathbf{k}_T + \mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T} + \mathbf{K}_T^T\mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T}\mathbf{k}_T$$

Linear Quadratic Regulator (LQR)

Remember: $V(x_t) = \min_{u_t} Q(x_t, u_t)$

Substituting the minimizer u_T into $Q(x_T, u_T)$ gives us $V(x_T)$!

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{C}_T \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix} + \begin{bmatrix} \mathbf{x}_T \\ \mathbf{K}_T \mathbf{x}_T + \mathbf{k}_T \end{bmatrix}^T \mathbf{c}_T$$

$$V(\mathbf{x}_T) = \frac{1}{2} \mathbf{x}_T^T \mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} \mathbf{x}_T + \frac{1}{2} \mathbf{x}_T^T \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T} \mathbf{K}_T \mathbf{x}_T + \frac{1}{2} \mathbf{x}_T^T \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} \mathbf{x}_T + \frac{1}{2} \mathbf{x}_T^T \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T} \mathbf{K}_T \mathbf{x}_T +$$
$$\mathbf{x}_T^T \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T} \mathbf{k}_T + \frac{1}{2} \mathbf{x}_T^T \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T} \mathbf{k}_T + \mathbf{x}_T^T \mathbf{c}_{\mathbf{x}_T} + \mathbf{x}_T^T \mathbf{K}_T^T \mathbf{c}_{\mathbf{u}_T} + \text{const}$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2} \mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T$$

$$\mathbf{V}_T = \mathbf{C}_{\mathbf{x}_T,\mathbf{x}_T} + \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T} \mathbf{K}_T + \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{x}_T} + \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T} \mathbf{K}_T$$

$$\mathbf{v}_T = \mathbf{c}_{\mathbf{x}_T} + \mathbf{C}_{\mathbf{x}_T,\mathbf{u}_T} \mathbf{k}_T + \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T} + \mathbf{K}_T^T \mathbf{C}_{\mathbf{u}_T,\mathbf{u}_T} \mathbf{k}_T$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

cost at T-1

best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \underbrace{\frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1}}_{\text{cost at T-1}} + \underbrace{V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))}_{\text{best cost-to-go}}$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

cost at T-1

best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\left[\begin{array}{c}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{array}\right]^T \mathbf{C}_{T-1}\left[\begin{array}{c}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{array}\right] + \left[\begin{array}{c}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{array}\right]^T \mathbf{c}_{T-1} + \underbrace{V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))}_{V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T}$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

cost at T-1 best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{C}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{c}_{T-1} + \underbrace{V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))}_{V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T}$$

$$q_*(s, a) = r(s, a) + \gamma \sum_{s' \in S} T(s'|s, a) \mathrm{v}_*(s')$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost

best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{C}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{c}_{T-1} + V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T$$

$$q_*(s, a) = r(s, a) + \gamma \sum_{s' \in S} T(s'|s, a) \mathrm{v}_*(s')$$

We can eliminate x_T by writing only in terms of quantities of T-1!

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost

best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1} + \underbrace{V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))}_{V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T}$$

$$q_*(s, a) = r(s, a) + \gamma \sum_{s' \in S} T(s'|s, a) \mathrm{v}_*(s')$$

We can eliminate x_T by writing only in terms of quantities of T-1!

$$f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \mathbf{x}_T = \mathbf{F}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \mathbf{f}_{T-1}$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{C}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{c}_{T-1} + \underbrace{V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))}_{V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T \mathbf{V}_T \mathbf{x}_T + \mathbf{x}_T^T \mathbf{v}_T}$$

$$q_*(s,a) = r(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) v_*(s')$$

We can eliminate x_T by writing only in terms of quantities of T-1!

$$f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \mathbf{x}_T = \mathbf{F}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \mathbf{f}_{T-1}$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{F}_{T-1}}_{\text{quadratic}} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{f}_{T-1}}_{\text{linear}} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T \mathbf{v}_T}_{\text{linear}}$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

$$Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \text{const} + \overbrace{\frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1}}^{\text{Immediate cost}} + \underbrace{\overbrace{V(f(\mathbf{x}_{T-1},\mathbf{u}_{T-1}))}^{\text{best cost-to-go}}}_{V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\mathbf{x}_T^T\mathbf{V}_T\mathbf{x}_T + \mathbf{x}_T^T\mathbf{v}_T}$$

$$q_*(s,a) = r(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) \mathrm{v}_*(s')$$

We can eliminate x_T by writing only in terms of quantities of T-1!

$$f(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \mathbf{x}_T = \mathbf{F}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \mathbf{f}_{T-1}$$

$$V(\mathbf{x}_T) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T\mathbf{V}_T\mathbf{F}_{T-1}}_{\text{quadratic}}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T\mathbf{V}_T\mathbf{f}_{T-1}}_{\text{linear}} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \underbrace{\mathbf{F}_{T-1}^T\mathbf{v}}_{\text{linear}}$$

We have written $V(x_T)$ only in terms of x_{T-1}, u_{T-1} !

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost

best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1} + V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))$$

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{Q}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{q}_{T-1}$$

$$\mathbf{Q}_{T-1} = \mathbf{C}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{F}_{T-1}$$

$$\mathbf{q}_{T-1} = \mathbf{c}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{f}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{v}_T$$

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

$$Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \text{const} + \overbrace{\frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1}}^{\text{Immediate cost}} + \overbrace{V(f(\mathbf{x}_{T-1},\mathbf{u}_{T-1}))}^{\text{best cost-to-go}}$$

$$\Downarrow$$

$$Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{Q}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{q}_{T-1}$$

$$\mathbf{Q}_{T-1} = \mathbf{C}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{F}_{T-1}$$

$$\mathbf{q}_{T-1} = \mathbf{c}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{f}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{v}_T$$

We have written optimal action value function $Q(x_{T-1}, u_{T-1})$ only in terms of x_{T-1}, u_{T-1}!

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost best cost-to-go

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{C}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{c}_{T-1} + V(f(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}))$$

$$\Downarrow$$

$$Q(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{Q}_{T-1} \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix} + \begin{bmatrix} \mathbf{x}_{T-1} \\ \mathbf{u}_{T-1} \end{bmatrix}^T \mathbf{q}_{T-1}$$

$$\mathbf{Q}_{T-1} = \mathbf{C}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{F}_{T-1}$$

$$\mathbf{q}_{T-1} = \mathbf{c}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{f}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{v}_T$$

We have written optimal action value function $Q(x_{T-1}, u_{T-1})$ only in terms of x_{T-1}, u_{T-1}!

Let's take derivative to find the minimizing u_{T-1}!

Linear Quadratic Regulator (LQR)

We propagate the optimal value function backwards!!

Immediate cost

best cost-to-go

$$Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \text{const} + \underbrace{\frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{C}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{c}_{T-1}} + \underbrace{V(f(\mathbf{x}_{T-1},\mathbf{u}_{T-1}))}$$

$$\Downarrow$$

$$Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \text{const} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{Q}_{T-1}\begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix} + \begin{bmatrix}\mathbf{x}_{T-1}\\ \mathbf{u}_{T-1}\end{bmatrix}^T \mathbf{q}_{T-1}$$

$$\mathbf{Q}_{T-1} = \mathbf{C}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{F}_{T-1}$$

$$\mathbf{q}_{T-1} = \mathbf{c}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{V}_T \mathbf{f}_{T-1} + \mathbf{F}_{T-1}^T \mathbf{v}_T$$

We have written optimal action value function $Q(x_{T-1}, u_{T-1})$ only in terms of x_{T-1}, u_{T-1}!

Let's take derivative to find the minimizer u_{T-1}:

$$\nabla_{\mathbf{u}_{T-1}} Q(\mathbf{x}_{T-1},\mathbf{u}_{T-1}) = \mathbf{Q}_{\mathbf{u}_{T-1},\mathbf{x}_{T-1}}\mathbf{x}_{T-1} + \mathbf{Q}_{\mathbf{u}_{T-1},\mathbf{u}_{T-1}}\mathbf{u}_{T-1} + \mathbf{q}_{\mathbf{u}_{T-1}}^T = 0$$

$$\mathbf{u}_{T-1} = \mathbf{K}_{T-1}\mathbf{x}_{T-1} + \mathbf{k}_{T-1} \qquad \mathbf{K}_{T-1} = -\mathbf{Q}_{\mathbf{u}_{T-1},\mathbf{u}_{T-1}}^{-1}\mathbf{Q}_{\mathbf{u}_{T-1},\mathbf{x}_{T-1}} \qquad \mathbf{k}_{T-1} = -\mathbf{Q}_{\mathbf{u}_{T-1},\mathbf{u}_{T-1}}^{-1}\mathbf{q}_{\mathbf{u}_{T-1}}$$

Linear case: LQR

Backward recursion:

```
for $t = T$ to 1:
    $\mathbf{Q}_t = \mathbf{C}_t + \mathbf{F}_t^T \mathbf{V}_{t+1} \mathbf{F}_t$
    $\mathbf{q}_t = \mathbf{c}_t + \mathbf{F}_t^T \mathbf{V}_{t+1} \mathbf{f}_t + \mathbf{F}_t^T \mathbf{v}_{t+1}$
    $Q(\mathbf{x}_t, \mathbf{u}_t) = \text{const} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{Q}_t \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} + \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{q}_t$
    $\mathbf{u}_t \leftarrow \arg\min_{\mathbf{u}_t} Q(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{K}_t \mathbf{x}_t + \mathbf{k}_t$
    $\mathbf{K}_t = -\mathbf{Q}_{\mathbf{u}_t, \mathbf{u}_t}^{-1} \mathbf{Q}_{\mathbf{u}_t, \mathbf{x}_t}$
    $\mathbf{k}_t = -\mathbf{Q}_{\mathbf{u}_t, \mathbf{u}_t}^{-1} \mathbf{q}_{\mathbf{u}_t}$
    $\mathbf{V}_t = \mathbf{Q}_{\mathbf{x}_t, \mathbf{x}_t} + \mathbf{Q}_{\mathbf{x}_t, \mathbf{u}_t} \mathbf{K}_t + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t, \mathbf{x}_t} + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t, \mathbf{u}_t} \mathbf{K}_t$
    $\mathbf{v}_t = \mathbf{q}_{\mathbf{x}_t} + \mathbf{Q}_{\mathbf{x}_t, \mathbf{u}_t} \mathbf{k}_t + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t} + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t, \mathbf{u}_t} \mathbf{k}_t$
    $V(\mathbf{x}_t) = \text{const} + \frac{1}{2} \mathbf{x}_t^T \mathbf{V}_t \mathbf{x}_t + \mathbf{x}_t^T \mathbf{v}_t$
```

Diagram: Sergey Levine

Linear case: LQR

Backward recursion:

```
for $t = T$ to $1$:
    $\mathbf{Q}_t = \mathbf{C}_t + \mathbf{F}_t^T \mathbf{V}_{t+1} \mathbf{F}_t$
    $\mathbf{q}_t = \mathbf{c}_t + \mathbf{F}_t^T \mathbf{V}_{t+1} \mathbf{f}_t + \mathbf{F}_t^T \mathbf{v}_{t+1}$
    $Q(\mathbf{x}_t, \mathbf{u}_t) = \text{const} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{Q}_t \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} + \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{q}_t$
    $\mathbf{u}_t \leftarrow \arg\min_{\mathbf{u}_t} Q(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{K}_t \mathbf{x}_t + \mathbf{k}_t$
    $\mathbf{K}_t = -\mathbf{Q}_{\mathbf{u}_t,\mathbf{u}_t}^{-1} \mathbf{Q}_{\mathbf{u}_t,\mathbf{x}_t}$
    $\mathbf{k}_t = -\mathbf{Q}_{\mathbf{u}_t,\mathbf{u}_t}^{-1} \mathbf{q}_{\mathbf{u}_t}$
    $\mathbf{V}_t = \mathbf{Q}_{\mathbf{x}_t,\mathbf{x}_t} + \mathbf{Q}_{\mathbf{x}_t,\mathbf{u}_t} \mathbf{K}_t + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t,\mathbf{x}_t} + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t,\mathbf{u}_t} \mathbf{K}_t$
    $\mathbf{v}_t = \mathbf{q}_{\mathbf{x}_t} + \mathbf{Q}_{\mathbf{x}_t,\mathbf{u}_t} \mathbf{k}_t + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t} + \mathbf{K}_t^T \mathbf{Q}_{\mathbf{u}_t,\mathbf{u}_t} \mathbf{k}_t$
    $V(\mathbf{x}_t) = \text{const} + \frac{1}{2} \mathbf{x}_t^T \mathbf{V}_t \mathbf{x}_t + \mathbf{x}_t^T \mathbf{v}_t$
```

Forward recursion:

```
for $t = 1$ to $T$:
    $\mathbf{u}_t = \mathbf{K}_t \mathbf{x}_t + \mathbf{k}_t$
    $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$
```

We know x_0

Non-linear case:Use iterative approximations!

First order Taylor expansion for the dynamics around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$f(\mathbf{x}_t, \mathbf{u}_t) \approx f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \left[\begin{array}{c} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{array} \right]$$

Second order Taylor expansion for the cost around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$c(\mathbf{x}_t, \mathbf{u}_t) \approx c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \left[\begin{array}{c} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{array} \right] + \frac{1}{2} \left[\begin{array}{c} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{array} \right]^T \nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \left[\begin{array}{c} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{array} \right]$$

Non-linear case:Use iterative approximations!

First order Taylor expansion for the dynamics around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$f(\mathbf{x}_t, \mathbf{u}_t) \approx f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}$$

Second order Taylor expansion for the cost around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$c(\mathbf{x}_t, \mathbf{u}_t) \approx c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}^T \nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}$$

Non-linear case:Use iterative approximations!

First order Taylor expansion for the dynamics around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$f(\mathbf{x}_t, \mathbf{u}_t) \approx f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}$$

Second order Taylor expansion for the cost around a trajectory $\hat{x}_t, \hat{u}_t, t = 1 \cdots T$:

$$c(\mathbf{x}_t, \mathbf{u}_t) \approx c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}^T \nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \begin{bmatrix} \mathbf{x}_t - \hat{\mathbf{x}}_t \\ \mathbf{u}_t - \hat{\mathbf{u}}_t \end{bmatrix}$$

$$\bar{f}(\delta \mathbf{x}_t, \delta \mathbf{u}_t) = \underbrace{\mathbf{F}_t}_{\nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t)} \begin{bmatrix} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{bmatrix}$$

$$\bar{c}(\delta \mathbf{x}_t, \delta \mathbf{u}_t) = \frac{1}{2} \begin{bmatrix} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{bmatrix}^T \underbrace{\mathbf{C}_t}_{\nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t)} \begin{bmatrix} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{bmatrix} + \begin{bmatrix} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{bmatrix}^T \underbrace{\mathbf{c}_t}_{\nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t)}$$

$$\delta \mathbf{x}_t = \mathbf{x}_t - \hat{\mathbf{x}}_t$$
$$\delta \mathbf{u}_t = \mathbf{u}_t - \hat{\mathbf{u}}_t$$

Now we can run LQR with dynamics $\bar{f}$, cost $\bar{c}$, state $\delta \mathbf{x}_t$, and action $\delta \mathbf{u}_t$

Iterative LQR (i-LQR)

Initialization: Given $\hat{x}_0$, pick a random control sequence $\hat{u}_0 \ldots \hat{u}_T$ and obtain corresponding state sequence $\hat{x}_0 \ldots \hat{x}_T$

```
until convergence:
    $\mathbf{F}_t = \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \forall t$
    $\mathbf{c}_t = \nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \forall t$
    $\mathbf{C}_t = \nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \forall t$
    Run LQR backward pass on state $\delta \mathbf{x}_t = \mathbf{x}_t - \hat{\mathbf{x}}_t$ and action $\delta \mathbf{u}_t = \mathbf{u}_t - \hat{\mathbf{u}}_t \ \forall t$
    Run forward pass with real nonlinear dynamics and $u_t = \hat{u}_t + K_t(x_t - \hat{x}_T) + k_t \ \forall t$
    Update $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{u}}_t$ based on states and actions in forward pass $\forall t$
```

Iterative LQR (i-LQR)

Initialization: Given $\hat{x}_0$, pick a random control sequence $\hat{u}_0 ... \hat{u}_T$ and obtain corresponding state sequence $\hat{x}_0 ... \hat{x}_T$

until convergence:

$$\mathbf{F}_t = \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \ \forall t$$

$$\mathbf{c}_t = \nabla_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \forall t$$

$$\mathbf{C}_t = \nabla^2_{\mathbf{x}_t, \mathbf{u}_t} c(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \ \ \forall t$$

Linear approximation around $\hat{x}, \hat{u}$

Find $\Delta u_t, t = 1...T$ so that $\hat{u}_t + \Delta u_t$ minimizes the linear approximation

Run LQR backward pass on state $\delta \mathbf{x}_t = \mathbf{x}_t - \hat{\mathbf{x}}_t$ and action $\delta \mathbf{u}_t = \mathbf{u}_t - \hat{\mathbf{u}}_t \ \forall t$

Run forward pass with real nonlinear dynamics and $u_t = \hat{u}_t + K_t(x_t - \hat{x}_T) + k_t \ \ \forall t$

Update $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{u}}_t$ based on states and actions in forward pass $\forall t$

Go to the $\hat{x}' = \hat{x} + \Delta x_t$ and $\hat{u}' = \hat{u} + \Delta u_t$

Differential Dynamic Programming

Second order approximation for the dynamics:

$$f(\mathbf{x}_t, \mathbf{u}_t) \approx f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) + \nabla_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \left[\begin{array}{c} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{array} \right] + \frac{1}{2} \left(\nabla^2_{\mathbf{x}_t, \mathbf{u}_t} f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \cdot \left[\begin{array}{c} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{array} \right] \right) \left[\begin{array}{c} \delta \mathbf{x}_t \\ \delta \mathbf{u}_t \end{array} \right]$$

The resulting method is called differential dynamic programming.

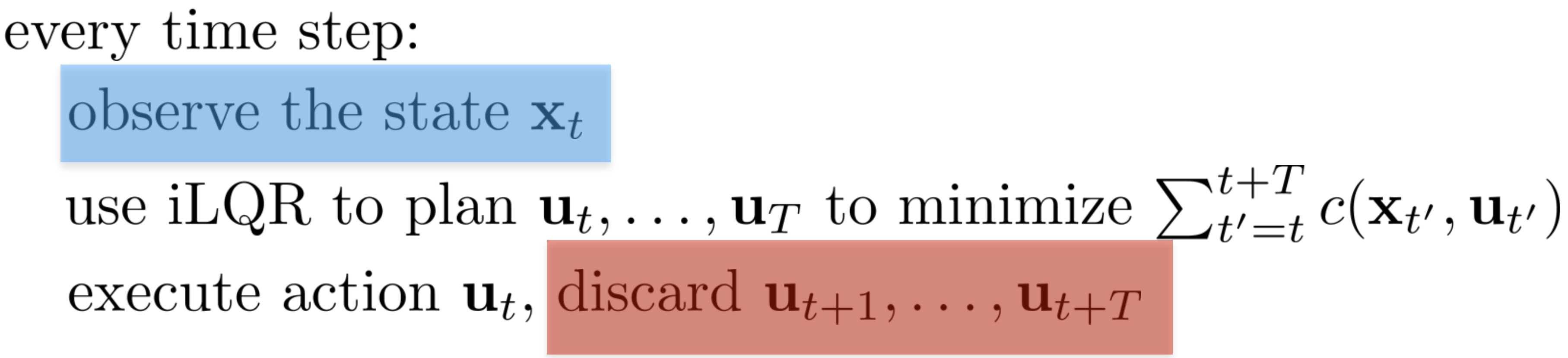
Reference: Jacobson and Mayne, “Differential dynamic programming”, 1970

Closed Loop Vs Open Loop

- So far we have been planning (e.g. 100 steps) and then we close our eyes and hope our modeling was accurate enough..
- At convergence of iLQR and DDP, we end up with linearization around the (state, input) trajectory the algorithm converged to.
- In practice: the system could not be on this trajectory due to perturbations / initial state being off / dynamics model being inaccurate
- Can we handle such noise better?

Model Predictive Control

- Yes, If we close the loop.
- Solution: at time t when asked to generate control input u_t, we could re-solve the control problem using iLQR or DDP over the time steps t through T

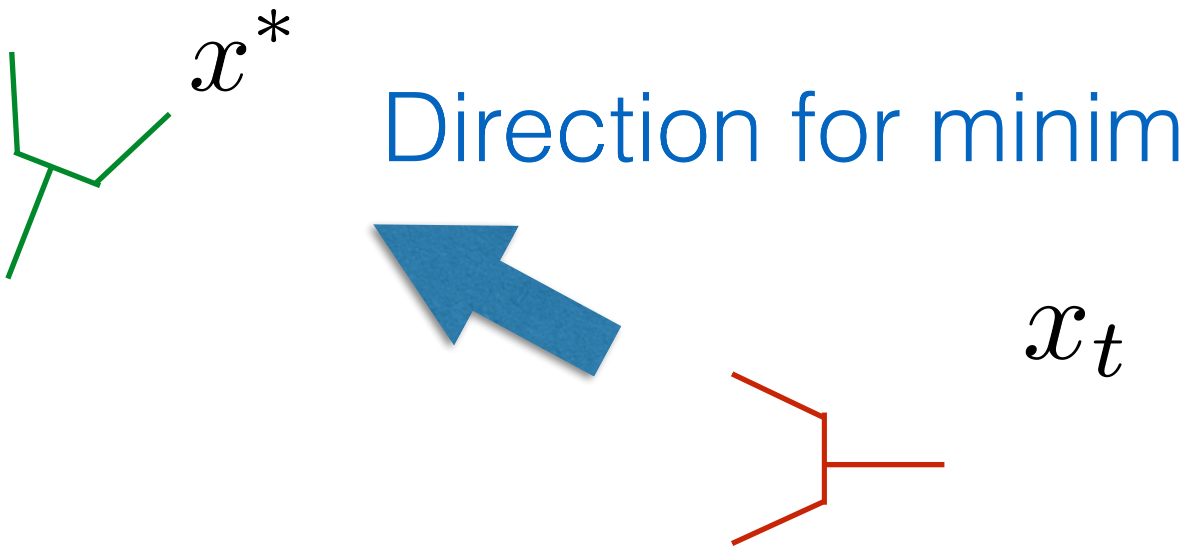


```
every time step:
    observe the state $\mathbf{x}_t$
    use iLQR to plan $\mathbf{u}_t, \ldots, \mathbf{u}_T$ to minimize $\sum_{t'=t}^{t+T} c(\mathbf{x}_{t'}, \mathbf{u}_{t'})$
    execute action $\mathbf{u}_t$, discard $\mathbf{u}_{t+1}, \ldots, \mathbf{u}_{t+T}$
```

- Re-planning entire trajectory is often impractical -> in practice: replay over horizon H (receding horizon control)

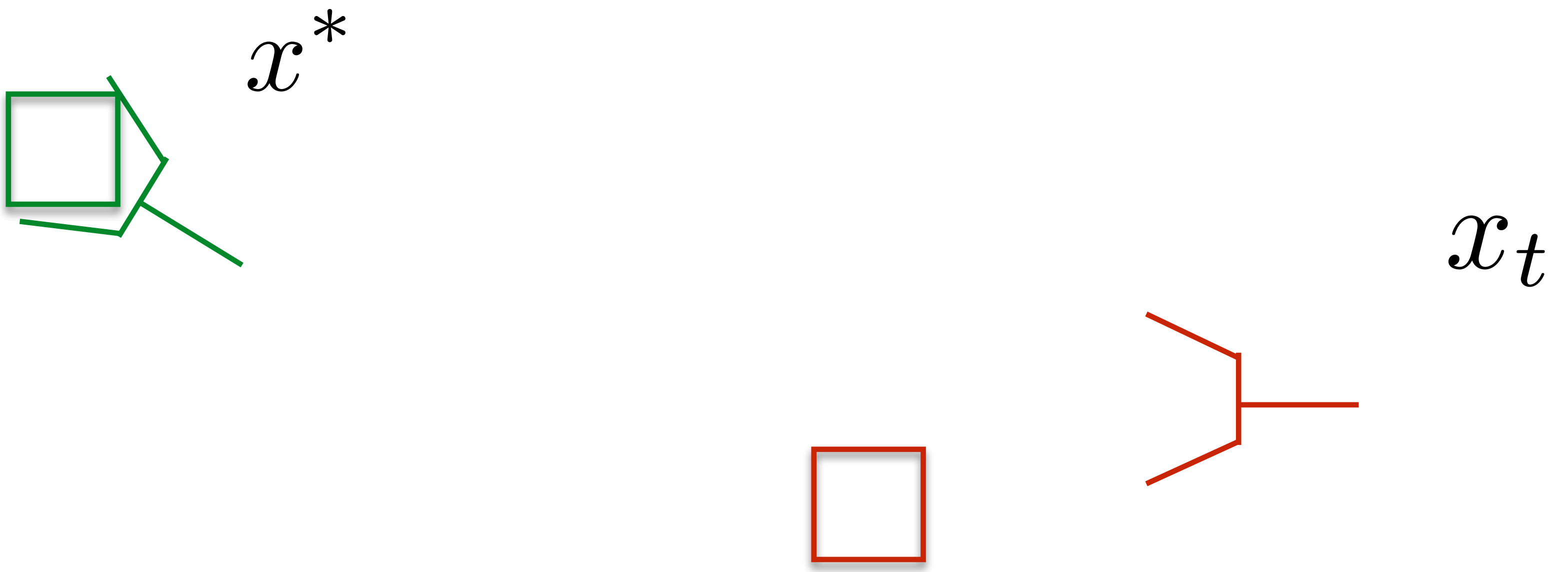
i-LQR: When it works

Cost: $\|x_t - x^*\|$



i-LQR: When it doesn't work

Cost: $\|x_t - x^*\|$



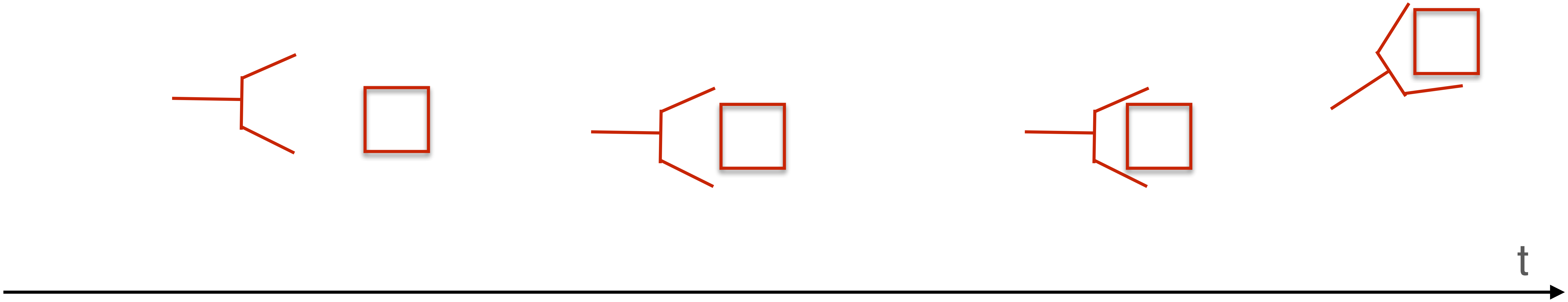
Due to discontinuities of contact, the local search can fail.

As a solution we often initialize using a human demonstration instead of randomly

Learning Dexterous manipulation Policies from Experience and Imitation, Kumar, Gupta, Todorov, Levine 2016

Time varying linear dynamics

reference trajectory $\hat{x}_t, \hat{u}_t, t = 1, ..., T$



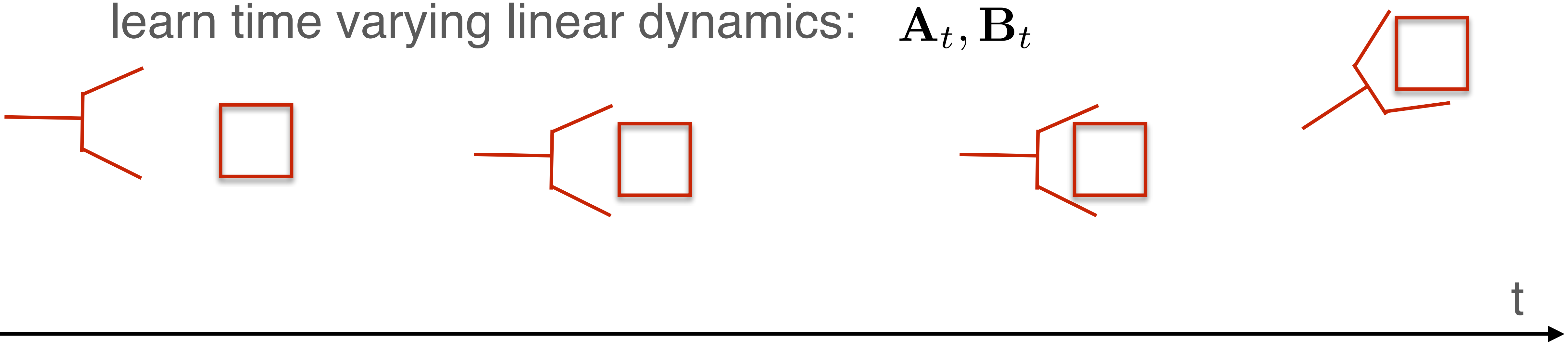
Time varying linear dynamics

$$f(x_t, u_t) \approx \mathbf{A}_t x_t + \mathbf{B}_t u_t$$

$$\mathbf{A}_t = \frac{df}{dx_t} \quad \mathbf{B}_t = \frac{df}{du_t}$$

reference trajectory $\hat{x}_t, \hat{u}_t, t = 1, ..., T$

learn time varying linear dynamics: $\mathbf{A}_t, \mathbf{B}_t$



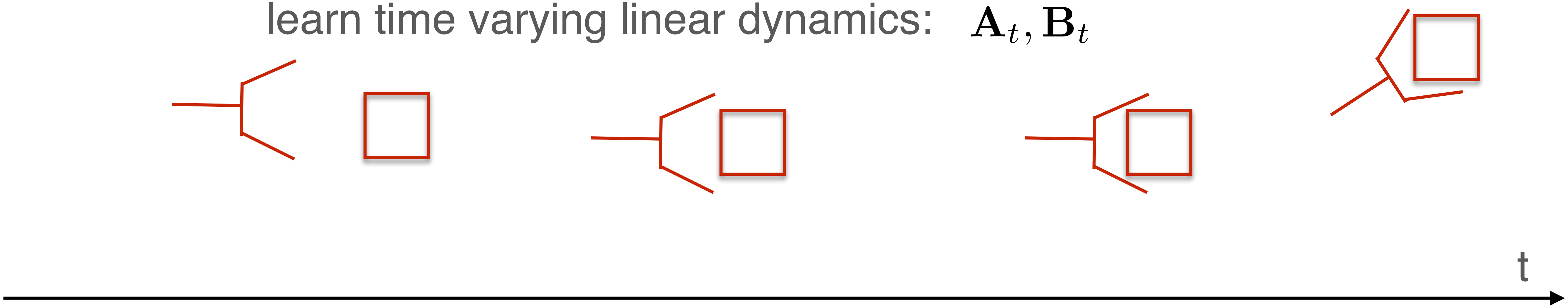
Time varying linear dynamics

$$f(x_t, u_t) \approx \mathbf{A}_t x_t + \mathbf{B}_t u_t$$

$$\mathbf{A}_t = \frac{df}{dx_t} \quad \mathbf{B}_t = \frac{df}{du_t}$$

reference trajectory $\hat{x}_t, \hat{u}_t, t = 1, ..., T$

learn time varying linear dynamics: $\mathbf{A}_t, \mathbf{B}_t$



How do I get the data to fit my linear dynamics at each time step?

We execute the controller u_t at state x_t to explore how the world works in the vicinity of the reference trajectory!

Which controller to collect samples with?

- We need a stochastic controller! Why?
- Here is a good guess: add some noise to the output of iLQR:

$$p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t(\mathbf{x}_t - \hat{\mathbf{x}}_t) + \mathbf{k}_t + \hat{\mathbf{u}}_t, \Sigma_t)$$

- It turns out that setting $\Sigma_t = \mathbf{Q}^{-1}_{\mathbf{u}_t,\mathbf{u}_t}$ solves the following maximum entropy control problem:

$$\min \sum_{t=1}^{T} E_{(\mathbf{x}_t,\mathbf{u}_t)\sim p(\mathbf{x}_t,\mathbf{u}_t)}[c(\mathbf{x}_t, \mathbf{u}_t) - \mathcal{H}(p(\mathbf{u}_t|\mathbf{x}_t))]$$

- Remember, cost to go:

$$Q(\mathbf{x}_t, \mathbf{u}_t) = \text{const} + \frac{1}{2}\begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{Q}_t \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix} + \begin{bmatrix} \mathbf{x}_t \\ \mathbf{u}_t \end{bmatrix}^T \mathbf{q}_t$$

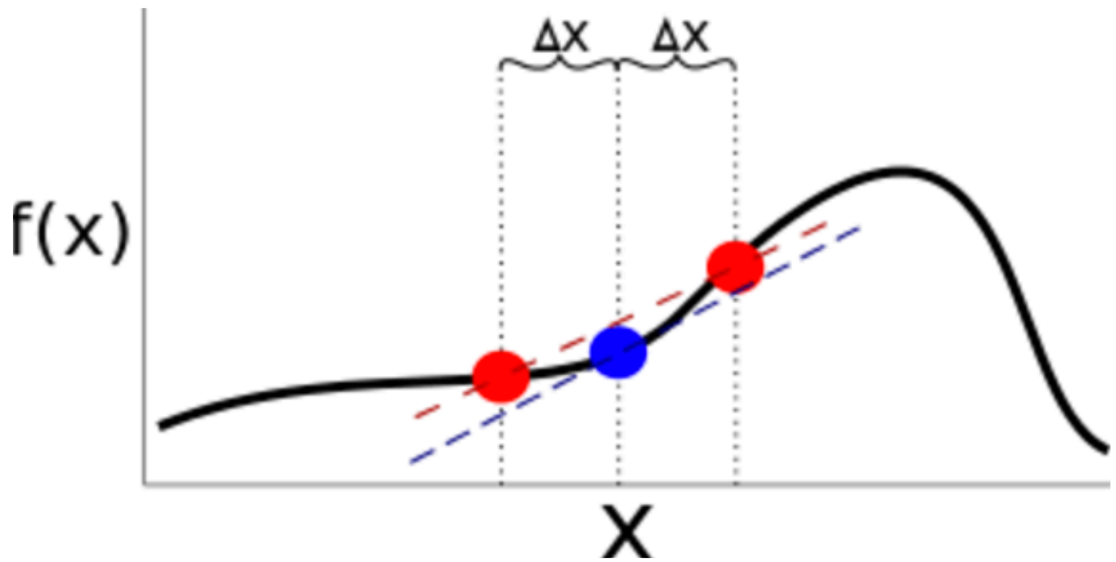
- The above controller strikes the right balance between minimizing the cost and maximize exploration

Guided Policy Search, Levine and Koltun 2013

Fitting Dynamics(1): Compute analytically derivatives of the “true” non-linear dynamics

- We may not have such analytic non linear dynamic equations available
- Very limiting: under modeling errors
- Complicated derivations

Fitting Dynamics(2): Finite Differences



$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\dot{f}(x) \approx \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x}$$

We need 2 samples per state dimension

Use linear regression to fit A,B,D to samples $\{\mathbf{x}_i, \mathbf{u}_i, \mathbf{x}'_i\}$

Time varying linear dynamics

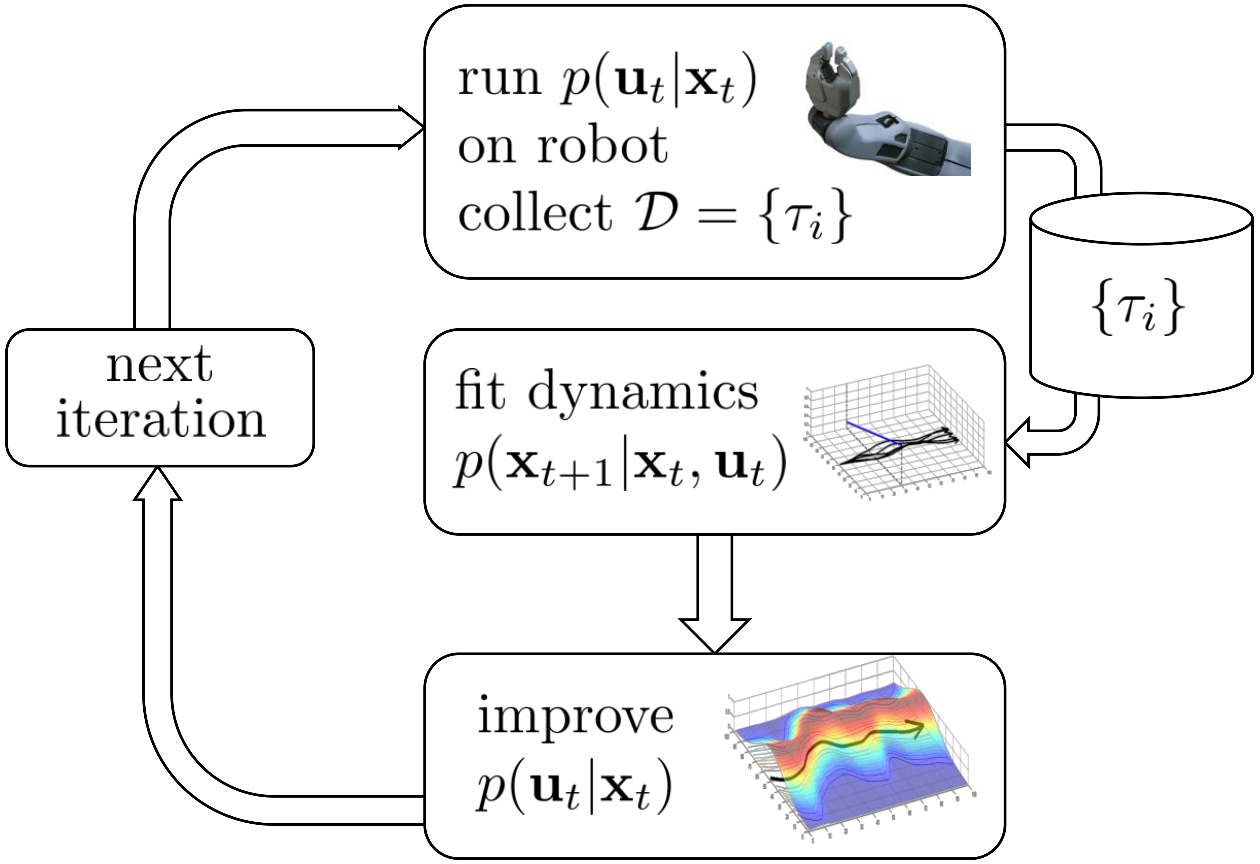
We iteratively fit dynamics and update the policy. Why such iteration is important?

So that the space (state, actions) our dynamics are estimated from is similar to the one our policy visits.

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(f(\mathbf{x}_t, \mathbf{u}_t), \Sigma)$$

$$f(\mathbf{x}_t, \mathbf{u}_t) \approx \mathbf{A}_t\mathbf{x}_t + \mathbf{B}_t\mathbf{u}_t$$

$$\mathbf{A}_t = \frac{df}{d\mathbf{x}_t} \qquad \mathbf{B}_t = \frac{df}{d\mathbf{u}_t}$$



run $p(\mathbf{u}_t|\mathbf{x}_t)$
on robot
collect $\mathcal{D} = \{\tau_i\}$

$\{\tau_i\}$

fit dynamics
$p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$

improve
$p(\mathbf{u}_t|\mathbf{x}_t)$

next
iteration