Lecture 19, Feb 24, 2026

Dynamic Programming (Infinite Time Horizon)

- In infinite horizon, we have the additional problem of convergence
- For simplicity, consider a nonlinear time-invariant system $x(k+1) = f(x(k), u(k))$
- As with before, assume inputs can be selected from $\mathcal{U}_k(x) = \mathcal{U}(x)$, and states are $x \in \mathcal{X}$; we want to select a *policy* $\pi = \{ \mu_0, \mu_1, \dots \}$, but now we consider an infinite horizon
  - Note that each $\mu_k(x)$ is a function of x since it's a feedback; we use μ to denote a feedback and u to denote a particular input
  - Usually we will work with *stationary policies*, i.e. policies which do not vary in time, so the policy is the same $\mu(x)$
- Let $\mathcal{M} = \{ \mu \mid \mu(x) \in \mathcal{U}(x), x \in \mathcal{X} \}$ be the set of admissible feedbacks
  - The policy is a choice of admissible feedbacks at each time step
- Let $\mathcal{P} = \{ \{ \mu_0, \mu_1, \dots \} \mid \mu_k \in \mathcal{M}, k = 0, 1, \dots \}$ be the set of admissible policies
- Let the cost for $\pi \in \mathcal{P}$ with initial condition $x_0 \in \mathcal{X}$ be $V^\pi(x_0) = \sum_{k=0}^{\infty} \gamma^k r(x(k), \mu_k(x(k)))$
  - $r(x, u) \geq 0, x \in \mathcal{X}, u \in \mathcal{U}$, and $x(k)$ is a solution to the system $x(k+1) = f(x(k), \mu_k(x(k))), x(0) = x_0$
  - $\gamma \in (0, 1)$ is the *discount* or *forgetting factor*, which is added to make the cost converge
- Let $V^*(x_0) = \inf_{\pi \in \mathcal{P}} \{ V^\pi(x_0) \}$ be the optimal cost of starting from $x_0 \in \mathcal{X}$
  - Note technically we have to use the infimum instead of minimum, since this is an open set, but they work similarly
- $$\begin{aligned} V^\pi(x_0) &= r(x_0, \mu_0(x_0)) + \sum_{k=1}^{\infty} \gamma^k r(x(k), \mu_k(x(k))) \\ &= r(x_0, \mu_0(x_0)) + \gamma \sum_{k=0}^{\infty} \gamma^k r(x(k+1), \mu_{k+1}(x(k+1))) \\ &= r(x_0, \mu_0(x_0)) + \gamma V^{\pi^1}(f(x_0, \mu_0(x_0))) \qquad \text{where } \pi^1 = \{ \mu_1, \mu_2, \dots \} \end{aligned}$$
  - Therefore $V^*(x_0) = \inf_{\pi = \{ \mu_0, \pi^1 \} \in \mathcal{P}} \{ r(x_0, \mu_0(x_0)) + \gamma V^{\pi^1}(f(x_0, \mu_0(x_0))) \}$
    - The choice of our first input affects both our first cost and where we end up for the rest of the optimization
  - Applying the principle of optimality, $$\begin{aligned} V^*(x_0) &= \inf_{\mu_0 \in \mathcal{M}} \{ r(x_0, \mu_0(x_0)) + \gamma \inf_{\pi^1 \in \mathcal{P}} \{ f(x_0, \mu_0(x_0)) \} \} \\ &= \inf_{\mu_0 \in \mathcal{M}} \{ r(x_0, \mu_0(x_0)) + \gamma V^*(f(x_0, \mu_0(x_0))) \} \\ &= \inf_{u \in \mathcal{U}(x_0)} \{ r(x_0, u) + \gamma V^*(f(x_0, u)) \} \end{aligned}$$
    - As with the infinite time case, we can split the optimal cost into two parts: the first step's cost, and the optimal cost for the rest of the steps
    - This holds for all $x \in \mathcal{X}$
    - This is known as the *Hamilton-Jacobi-Bellman equation* (HJB), also known as the *Bellman optimality condition*
  - Since V^* appears on both sides of the HJB equation, we can use an iterative method for reinforcement learning (value iteration and policy iteration)