Differential Dynamic Programming

Ramona Stefanescu

October 18, 2018

MDPs

- A (discounted infinite horizon) Markov Decision Process (MDP) is a tuple (S,A,T, γ, D,R)
- S is the set of possible states for the system
- A is the set of possible actions
- T represents the (typically stochastic) system dynamics
- D is the initial-state distribution, from which state s_0 is drawn
- $R : S \to R$ is the reward function

Acting in a Markov decision process results in a sequence of states and actions s_0,a_0,s_1,$s_2, \dots$

A policy π is a sequence of mappings $(\mu_0, \mu_1, \mu_2 \dots)$, where, at time t the mapping $\mu_t(\cdot)$ determines the action $a_t = \mu_t(s_t)$ to take when in state s_t

- The objective is to find policies that maximize the expected sum of rewards accumulated over time
- In particular, a policy π is good if its utility is high:

$$U(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi] \tag{1}$$

- To represent the system dynamics, we can use the state-transition distribution notation

$$s_{t+1} \propto P_{sa}(\cdot|s_t, a_t) \tag{2}$$

- Or using a deterministic function F and a random disturbance ω_t

$$s_{t+1} = F(s_t, a_t, \omega_t) \tag{3}$$

Example - car

- One way to model the state of a car is to use the following six state variables: northing (n), easting (e), north velocity ($\dot{n}$), east velocity ($\dot{e}$), heading (θ), angular rate ($\dot{\theta}$). Hence the state space $\mathbb{S} = \mathbb{R}^6$
- The actions (or control inputs) are (i) steering angle, (ii) throttle, (iii) brake
- The perturbances capture both environmental perturbations as well as unmodeled aspects of the car dynamics
- We could have the followinh dynamics model $s_{t+1} = F(s_t, a_t, \omega_t)$:

$$n_{t+1} = n_t + \dot{n}_t \Delta_t, \tag{4}$$

$$e_{t+1} = e_t + \dot{e}_t \Delta_t, \tag{5}$$

$$\theta_{n+1} = \theta + \dot{\theta} \Delta_t \tag{6}$$

$$\dot{n}_{t+1} = f_n(\dot{n}_t, \dot{e}_t, \dot{\theta}_t, a_t, \omega_t) \tag{7}$$

$$\dot{e}_{t+1} = f_e(\dot{n}_t, \dot{e}_t, \dot{\theta}_t, a_t, \omega_t) \tag{8}$$

$$\dot{\theta}_{t+1} = f_\theta(\dot{n}_t, \dot{e}_t, \dot{\theta}_t, a_t, \omega_t) \tag{9}$$

- The reward function could be $R(s_t) = 1$ (in goal region) and 100(in collision)

Finding optimal policies: value iteration

The goal is to find an optimal policy - the policy that maximizes the expected total of rewards earned over the period of our decision process.

- **Finite horizon:** we are concerned with decisions and rewards only up until a given time $t = H$, with state space S and action space A finite.
- The value of any policy π is:

$$V_\pi(s_0) = \mathbb{E}[\sum_{t=0}^{H} \gamma^t R(s_t)|\pi; s_0] \tag{10}$$

we are interested in finding

$$\max_\pi V_\pi(s_0) \tag{11}$$

$$\pi = \{\mu_0, \mu_1, \ldots, \mu_{H-1}\} \tag{12}$$

where $\mu_i : S \to A$

Finite horizon

- Since there are $|A|^{|S|}$ possible mapping for each μ_i, there are $(|A|^{|S|})^H$ possible policies π, which is far too many to compute the value of all possible options
- Instead, we apply a dynamic programming algorithm known as value iteration to find the optimal policy efficiently
- Intuitively, we are applying the notion that given a state, the past and future are independent (the "Markov property")
- Define the value function at the kth time-step as

$$V_k(s_k) = \max_{\mu_k, \mu_{k+1}, \dots, \mu_{H-1}} \mathbb{E}[\sum_{t=k}^{H} \gamma^{t-k} R(s_t)|s_k] \tag{13}$$

- It can be shown that the optimal policy can be found by working backwards from H:

$$V_k(s_k) = \max_{a \in A}[R(s_k) + \gamma \sum_{s'} P(s'|s,a) V_{k+1}(s')] \tag{14}$$

Infinite horizon optimal policies

- Now we are interested in finding an optimal policy when our horizon in infinite
- The value of a policy is now

$$V_\pi(s_0) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t) | s_0, \pi] \tag{15}$$

- And we are interested in finding $V^*(s) = \max_\pi V_\pi(s)$ and $\pi^* = argmax_\pi V_\pi(s)$
- Despite the fact the expectation is taken over an infinite sum, it is guaranteed to converge. This is because R is a function over a finite state space, and is therefore bounded, and since $\gamma \in (0, 1)$

Bellman Backup Operator

- Let $V : S \rightarrow R$ be our value function and let $V_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t)|s_0, \pi]$.
- Define the operator $T : V \leftarrow TV$ as

$$(TV)(s) = \max_{a \in A}[R(s) + \gamma \sum_{s'} P(s'|s,a)V(s')] \tag{16}$$

LQR - Finite Horizon Value Iteration Case Study

- We assume a linear dynamic system:

$$x_{t+1} = A_t x_t + B_t u_t, \tag{17}$$

where $x(t)$ denotes the state at time t and $u(t)$ denotes the input at time t

- We assume a quadratic cost function:

$$J = \sum_{t=0}^{H-1} (x_t^T Q_t x_t + u_t^T R_t u_t) + x_H^T P_H x_H \tag{18}$$

with for all t, $Q_t \geq 0, R_t \geq 0$

- Our goal is to find the input sequence $\{u_0, u_1, \ldots, u_{H-1}\}$ that minimizes the cost: $\min_{u_0 \ldots u_{H-1}} J$

LQR - cont

$$
\begin{aligned}
&\min_{u_0 \ldots u_{H-1}} J \\
&= \min_{u_0 \ldots u_{H-1}} \sum_{t=0}^{H-1} (x_t^T Q_t x_t + u_t^T R_t u_t) + x_H^T P_H x_H \\
&= \min_{u_0 \ldots u_{H-2}} \sum_{t=0}^{H-2} (x_t^T Q_t x_t + u_t^T R_t u_t) + x_{H-1}^T Q_{H-1} x_{H-1} + \\
&+ \min_{u_{H-1}} u_{H-1}^T R_{H-1} u_{H-1} + x_H^T Q_H x_H
\end{aligned}
$$

LQR - cont

Solving the equation the previous page leads to the following dynamic programming algorithm to find the optimal controller for a linear system with quadratic costs:

- for $t = H - 1$ to 0
- $\rightarrow K_t \leftarrow -(R_t + B_t^T P_{t+1} B_t)^{-1} B_t^T P_{t+1} A_t$
- $\rightarrow P_t \leftarrow Q_t + K_t^T R_t K_t + (A_t + B_t K_t)^T P_{t+1} (A_t + B_t K_t)$
- next t

The optimal inputs are computed as follows $u_t = K_t x_t$. The optimal cost-to-go (=cost incurred in all future steps) for being in state x_t at time t is given by $x_t^T P_t x_t$.

Controllability vs Observability

▶