Reinforcement Learning

Alejandro Ribeiro

April 19, 2024

Consider a dynamical system f with state trajectory s_t which we control through the choice of actions a_t. We say that (s_t, a_t) are a state-action pair. Actions are chosen according to a policy π such that whenever the state of the system is $s_t = s$ we choose to execute action $a = \pi(s)$. It then follows that the system's state evolves as

$$s_{t+1} = f\big(s_t,\, a_t\big) = f\big(s_t,\, \pi(s_t)\big)\,. \tag{1}$$

We postulate that whenever we visit the state s_t and take action a_t we collect a reward $r(s_t, a_t)$. Our goal is to find the policy π that results in trajectories that accumulate as much reward as possible.

Reinforcement Learning (RL) refers to the problem of finding such optimal policies through interactions with the system. I.e., we want to find optimal policies when we do not know the system f but have the ability to observe state trajectories s_t, probe it with actions a_t, and observe the collected rewards $r(s_{t,t})$.

1 Value Functions

We are given a discount factor $\gamma < 1$, we fix a policy π, and we initialize the system with state-action pair $(s, a) = (s_0, a_0)$. We define the Q-function $Q(s, a; \pi)$ as the reward accumulated by executing policy π when the system is initialized at state-action pair (s, a),

$$Q(s, a; \pi) = \sum_{t=0}^{\infty} \gamma^t\, r(s_t, a_t), \tag{2}$$

In (2), the system starts at the state-action pair $(s, a) = (s_0, a_0)$ and collects the reward $r(s, a) = r(s_0, a_0)$ – the action a may or may not be the policy action $\pi(s)$. As per (1), the system moves to state $s_1 = f(s_0, a_0) = f(s, a)$. At this point in time we choose action $a_1 = \pi(s_1)$ to collect the reward $\gamma r(s_1, a_1)$ and effect the system to transition into state $s_2 = f(s_1, a_1)$. We then execute action $a_2 = \pi(s_2)$ to collect reward $\gamma^2 r(s_2, a_2)$ and transition into state $s_3 = f(s_2, a_2)$. We keep executing actions as dictated by the policy π so that at arbitrary time t we execute action $a_t = \pi(s_t)$ to collect reward $\gamma^t r(s_t, a_t)$ and transition into state $s_{t+1} = f(s_t, a_t)$. We point out that the purpose of the discount factor γ is to give more weight to earlier collection of rewards.

The Q-function in (2) describes the reward that we collect when the system is initialized at state-action pair $(s, a) = (s_0, a_0)$. In general, we care about the system when it is initialized at a number of different states s. To capture this interest we consider a distribution of initial states $s = s_0$ and define the value function of policy π as

$$V(\pi) = \int Q(s, \pi(s); \pi)\, ds. \tag{3}$$

A policy π^* is optimal if it maximizes $V(\pi)$. Although we can attempt to solve (3) directly over the policy π we chose to introduce a learning parameterization $\pi(s, \alpha)$. This leads to the optimization problem

$$\alpha^* = \underset{\alpha}{\text{argmax}}\, V(\pi(\alpha)) = \underset{\alpha}{\text{argmax}} \int Q(s, \pi(s; \alpha); \pi(\alpha))\, ds. \tag{4}$$

The advantage of introducing a learning parameterization is that after solving (4) we can execute optimal actions $\pi(s_t; \alpha^*)$ with (simple) evaluations of the learning parameterization.

1.1 Circuit Navigation

The trajectory control problem we addressed in Labs 5A and 5B can be written as a particular case of (4). Indeed, suppose that we are given a reference trajectory p_t. The problem of controlling the car to follow the reference trajectory is described by the reward function,

$$r(s_t, a_t) = -\frac{1}{2}\|s_t - p_t\|^2. \tag{5}$$

If the car is initialized at state s with action a and we execute policy π, the Q-function reduces to

$$Q(s,a;\pi) = -\sum_{t=0}^{\infty} \frac{\gamma^t}{2} \|s_t - p_t\|^2. \tag{6}$$

If we further consider states s_i drawn from a set of N possible states we conclude that the value function is

$$V(\pi) = \frac{1}{N} \sum_{i=1}^{N} Q(s_i, \pi(s_i); \pi) \tag{7}$$

If we think of the initial state s_i as drawn from a set of optimal trajectories, this is almost the same problem we solved in Lab 5B using MPC. The only difference is that when we evaluate the merit of a policy in following the reference trajectory we consider the discounted cost $\sum_t (\gamma^t/2)\|s_t - p_t\|^2$ instead of the average cost $(1/T)\sum_t \|s_t - p_t\|^2$. This is a minor difference.

2 Actor Critic Policy Optimization

Our experience with optimization problems indicates that to solve (4) we need to compute gradients of the value function with respect to the parameter α. This is actually a little more involved than it seems but a reasonable approximation to the value of this gradient is

$$g(\alpha) := \frac{\partial V(\pi(\alpha))}{\partial \alpha} = \int \frac{\partial Q(s,a;\pi(\alpha))}{\partial a} \frac{\partial \pi(s)}{\partial \alpha}\, ds. \tag{8}$$

In (8) the gradient $\partial \pi(s)/\partial \alpha$ can be computed as usual. The gradient $\partial Q(s,a;\pi(\alpha))/\partial a$ is more difficult to evaluate because we do not know the Q-function.

We solve the latter problem with the introduction of a critic. A critic is a learned parameterization that we use to estimate the Q-function. I.e., we consider a function $\tilde{Q}(s,a;\beta)$ that maps the state-action pair (s,a) and the parameter β to an estimate of the Q-function's value $Q(s,a)$. We can then replace the gradient in (8) by the estimated gradient

$$\tilde{g}(\alpha) = \int \frac{\partial \tilde{Q}(s,a;\beta)}{\partial a} \frac{\partial \pi(s)}{\partial \alpha}\, ds. \tag{9}$$

The advantage of (9) is that we have a closed form expression for $\tilde{Q}(s, a; \beta)$. We can therefore evaluate the gradient with respect to the action a.

We are left with the problem of estimating the Q function. To do that we consider the mean squared loss

$$\ell(\beta) = \int \frac{1}{2} \| Q(s, \pi(s), \pi) - \tilde{Q}(s, \pi(s); \beta) \|^2. \tag{10}$$

We minimize (10) by taking gradients with respect to β and approximating the Q function as

$$Q(s, \pi(s), \pi) = r(s, a) + \gamma \tilde{Q}(\tilde{s}, \pi(\tilde{s}); \beta), \tag{11}$$

where $\tilde{s} = f(s, a)$.

3 Project

Solve the car navigation problem of Lab 5B using actor critic policy optimization.