Answers to Exercises
Reinforcement Learning: Chapter 5

Exercise 5.1 Consider the diagrams on the right in Figure 5.1. Why does the estimated value function jump up for the last two rows in the rear? Why does it drop off for the whole last row on the left? Why are the frontmost values higher in the upper diagrams than in the lower?

Answer: The value function jumps up for the last two rows in the rear because these rows correspond to sums of 20 and 21, for which the policy is to stick, whereas for sums less than 20 the policy is to hit. Sticking is a much better action in this region of state space, so the values under the policy are much better for these states.

Value drops off for the leftmost row because these are the states for which the dealer is showing an ace. This is the worst case for the player because the dealer has a usable ace, giving him extra flexibility—in effect, two chances to get close to 21.

The values are generally higher in the upper diagrams because here the player has a usable ace and thus the extra flexibility. □

Exercise 5.2 Suppose every-visit MC was used instead of first-visit MC on the blackjack task. Would you expect the results to be very different? Why or why not?

Answer: The results would be exactly the same, because in this problem the same state is never visited more than once in a single episode. □

Exercise 5.3 What is the backup diagram for Monte Carlo estimation of q_π?

Answer:

□

Exercise 5.4 The pseudocode for Monte Carlo ES is inefficient because, for each state–action pair, it maintains a list of all returns and repeatedly calculates their mean. It would be more efficient to use techniques similar to those explained in Section 2.4 to maintain just the mean and a count (for each state–action pair) and update them incrementally. Describe how the pseudocode would be altered to achieve this.

Answer: The initialization section would be modified by replacing the line for the *Returns* with one that initialized a count $N(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$. Then, the second and third to last lines are replaced by:

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)} \left[G - Q(S_t, A_t) \right]. \qquad \square$$

Exercise 5.5 Consider an MDP with a single nonterminal state and a single action that transitions back to the nonterminal state with probability p and transitions to the terminal state with probability $1-p$. Let the reward be $+1$ on all transitions, and let $\gamma = 1$. Suppose you observe one episode that lasts 10 steps, with a return of 10. What are the first-visit and every-visit estimators of the value of the nonterminal state?

Answer: First-visit estimator: $V(s) = 10$
Every-visit estimator:

$$V(s) = \frac{10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1}{10} = 5.5 \qquad \square$$

Exercise 5.6 What is the equation analogous to (5.6) for *action* values $Q(s,a)$ instead of state values $V(s)$, again given returns generated using b?

Answer: There are two changes needed. First, we need a notation $\mathcal{T}(s,a)$ for the set of time steps in which the state–action *pair* occurred (or first occurred in the episode for a first-visit method). Second, we need to correct for one less importance-sampling ratio, because, in an action-value estimate, we don't have to correct for the first action. Thus:

$$Q(s,a) \doteq \frac{\sum_{t \in \mathcal{T}(s,a)} \rho_{t+1:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s,a)} \rho_{t+1:T(t)-1}}. \qquad \square$$

Exercise 5.7 In learning curves such as those shown in Figure 5.3 error generally decreases with training, as indeed happened for the ordinary importance-sampling method. But for the weighted importance-sampling method error first increased and then decreased. Why do you think this happened?

Answer: This shape of the learning curve for the weighted importance sampling estimator is quite robust. Recall that the weighted importance sampling estimator is initially biased. I believe this bias is toward zero, where in fact the true answer lies, and that this results in low initial mean square error. $\square$

Exercise 5.8 The results with Example 5.5 and shown in Figure 5.4 used a first-visit MC method. Suppose that instead an every-visit MC method was used on the same problem. Would the variance of the estimator still be infinite? Why or why not?

Answer: TBD. $\square$

Exercise 5.9 Modify the algorithm for first-visit MC policy evaluation (Section 5.1) to use the incremental implementation for sample averages described in Section 2.4.

Answer:

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated
Initialize:
  $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$
  $N(s) \leftarrow 0$, for all $s \in \mathcal{S}$

Loop forever (for each episode):
  Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
    $G \leftarrow \gamma G + R_{t+1}$
    Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:
      $N(S_t) \leftarrow N(S_t) + 1$
      $V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}\left[G - V(S_t)\right]$

□

Exercise 5.10 Derive the weighted-average update rule (5.8) from (5.7). Follow the pattern of the derivation of the unweighted rule (2.3).

Answer: Here is one way to do it:

$$
\begin{aligned}
V_{n+1} &= \frac{\sum_{k=1}^{n} W_k G_k}{\sum_{k=1}^{n} W_k} && \text{(from (5.7))} \\
&= \frac{W_n G_n + \sum_{k=1}^{n-1} W_k G_k}{W_n + \sum_{k=1}^{n-1} W_k} \\
&= \frac{W_n G_n + \left(\sum_{k=1}^{n-1} W_k\right) V_n + W_n V_n - W_n V_n}{W_n + \sum_{k=1}^{n-1} W_k} \\
&= V_n + \frac{W_n G_n - W_n V_n}{\sum_{k=1}^{n} W_k} \\
&= V_n + \frac{W_n}{\sum_{k=1}^{n} W_k}[G_n - V_n] \\
&= V_n + \frac{W_n}{C_n}[G_n - V_n].
\end{aligned}
$$

□

Exercise 5.11 In the boxed algorithm for off-policy MC control, you may have been expecting the W update to have involved the importance-sampling ratio $\frac{\pi(A_t|S_t)}{b(A_t|S_t)}$, but instead it involves $\frac{1}{b(A_t|S_t)}$. Why is this nevertheless correct?

Answer: In this algorithm, π is a deterministic policy, so, for the action actually taken, its probability of being taken is always 1. □

***Exercise 5.13** Show the steps to derive (5.14) from (5.12).

Answer: TBD. □

***Exercise 5.14** Modify the algorithm for off-policy Monte Carlo control (page 111) to use the idea of the truncated weighted-average estimator (5.10). Note that you will first need to convert this equation to action values.

Answer: TBD. □