Exercise Set 2 - Reinforcement Learning
Chapter 3,4 - Tabular Methods

Instructions

This is the second exercise booklet for Reinforcement Learning. It covers both ungraded exercises to practice at home or during the tutorial sessions as well as graded homework exercises and graded coding assignments. The graded assignments are clearly marked.

- Make sure you deliver answers in a clear and structured format. LaTeXhas our preference. Messy handwritten answers will not be graded.
- Pre-pend the name of your TA to the file name you hand in and remember to put your name and student ID on the submission;
- The deadline for this first assignment is **September 24th 2021 at 13:00** and will cover the material of chapter 3-4. All questions marked 'Homework' in this booklet need to be handed in on Canvas. The coding assignments need to be handed in separately through the Codegra.de platform integrated on canvas.

Contents

Chapter 3: Monte Carlo and TD methods

3.1 Monte Carlo

1. Consider an MDP with a single state s_0 that has a certain probability of transitioning back onto itself with a reward of 0, and will otherwise terminate with a reward of 5. Your agent has interacted with the environment and has gotten the following two trajectories: $[0, 0, 5], [0, 0, 0, 5]$. Use $\gamma = 0.9$.

   (a) Estimate the value of s_0 using first-visit MC.

   (b) Estimate the value of s_0 using every-visit MC.

3.2 Importance Sampling in Monte Carlo methods

1. What is a disadvantage of using *ordinary importance sampling* in off-policy Monte Carlo?

2. What is a disadvantage of using *weighted importance sampling* in off-policy Monte Carlo?

3.3 Temporal Difference Learning (application)

Consider an undiscounted Markov Decision Process (MDP) with two states A and B, each with two possible actions 1 and 2, and a terminal state T with $V(T) = 0$. The transition and reward functions are unknown, but you have observed the following episode using a random policy:

- $A \xrightarrow[r_3=-3]{a_3=1} B \xrightarrow[r_4=4]{a_4=1} A \xrightarrow[r_5=-4]{a_5=2} A \xrightarrow[r_6=-3]{a_6=1} T$

where the the arrow ($\rightarrow$) indicates a transition and a_t and r_t take the values of the observed actions and rewards respectively.

1. What are the state(-action) value estimates $V(s)$ (or $Q(s, a)$) after observing the sample episode when applying

   (a) TD(0) (1-step TD)

   (b) SARSA,

   where we initialize state(-action) values to 0 and use a learning rate $\alpha = 0.1$.

3.4 Temporal Difference Learning (Theory)

1. We can use Monte Carlo to get value estimates of a state with $V_M(S) = \frac{1}{M}\sum_{n=1}^{M} G_n(S)$ where $V_M(S)$ is the value estimate of state S after M visits of the state and $G_n(S)$ the return of an episode starting from S. Show that $V_M(S)$ can be written as the update rule $V_M(S) = V_{M-1}(S) + \alpha_M[G_M(S) - V_{M-1}(S)]$ and identify the learning rate α_M.

2. Consider the TD-error

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t). \tag{1}$$

   (a) What is $\mathbb{E}[\delta_t | S_t = s]$ if δ_t uses the true state-value function V^π

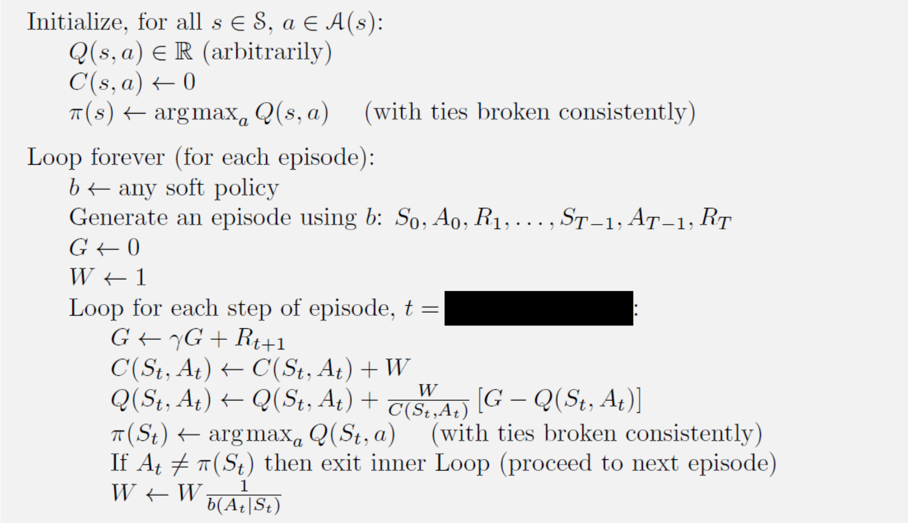
   (b) What is $\mathbb{E}[\delta_t | S_t = s, A_t = a]$ if δ_t uses the true state-value function V^π

3.5 * Exam question: Monte Carlo for control

This exercise has been taken from a previous exam (perhaps lightly edited) and can require a bit more insight. It should give you a good idea of the level of exam questions.

The following questions refer to the pseudo-code presented in Figure 1.

Figure 1: Algorithm pseudo-code.



```
Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
    $Q(s,a) \in \mathbb{R}$ (arbitrarily)
    $C(s,a) \leftarrow 0$
    $\pi(s) \leftarrow \operatorname{argmax}_a Q(s,a)$    (with ties broken consistently)

Loop forever (for each episode):
    $b \leftarrow$ any soft policy
    Generate an episode using $b$: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    $W \leftarrow 1$
    Loop for each step of episode, $t =$ [redacted]:
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t,A_t)} [G - Q(S_t, A_t)]$
        $\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$    (with ties broken consistently)
        If $A_t \neq \pi(S_t)$ then exit inner Loop (proceed to next episode)
        $W \leftarrow W \frac{1}{b(A_t|S_t)}$
```

1. Part of the algorithm is covered by a black square. What is the missing information?

2. Is this a Monte-Carlo algorithm or a TD-based algorithm? Explain your answer based on the given pseudo-code.

3. What is stored in $C(S_t, A_t)$?

4. Why is the inner loop stopped when $A_t \neq \pi(S_t)$?

3.6 Homework: Coding Assignment - Monte Carlo

1. In the chapter 5.6 of the book, we are given incremental update rule for weighted importance sampling (equations 5.7-5.8). However, in the coding assignment we will use ordinary importance sampling which uses a different update rule. Start with $V_n = \frac{\sum_{k=1}^{n} W_k G_k}{n}$ and derive the incremental update rule for ordinary importance sampling. Your final answer should be of the form: $V_n = V_{n-1} + a * (b - V_{n-1})$.

2. Download the notebook *RLLab2_MC.zip* from canvas assignments and follow the instructions.

3. What is the difference between Dynamic Programming and Monte Carlo? When would you use the one or the other algorithm? Include your answer in the submitted homework.

Chapter 4: Advanced TD Methods

4.1 Temporal Difference Learning (Application)

Consider the same setting as in Exercise 3.3, i.e. we have an undiscounted Markov Decision Process (MDP) with two states A and B, each with two possible actions 1 and 2, and a terminal state T with $V(T) = 0$. The transition and reward functions are unknown, but you have observed the following episode using a random policy:

- $A \xrightarrow[r_3=-3]{a_3=1} B \xrightarrow[r_4=4]{a_4=1} A \xrightarrow[r_5=-4]{a_5=2} A \xrightarrow[r_6=-3]{a_6=1} T$

where the the arrow (→) indicates a transition and a_t and r_t take the values of the observed actions and rewards respectively.

1. What are the state(-action) value estimates $V(s)$ (or $Q(s,a)$) after observing the sample episode when applying:

   (a) 3-step TD

   (b) Q-learning

   where we initialize state(-action) values to 0 and use a learning rate $\alpha = 0.1$

2. Choose a deterministic policy that you think is better than the random policy given the data. Refer to any of the state(-action) value estimates to explain your reasoning.

3. Let π_{random} denote the random policy used so far and $\pi_{student}$ denote the new policy you proposed. Suppose you can draw new sample episodes indefinitely until convergence of the value estimates.

   (a) Discuss how do you expect the final value estimates to differ if you ran Q-Learning with π_{random} as compared to $\pi_{student}$.

   (b) What problems may arise with π_{random} or $\pi_{student}$ respectively?

   (c) Do you think using an ϵ-greedy policy as behavior policy would be beneficial? Explain why/why not?

4.2 Off-policy TD

Consider the MDP in Figure 2. Consider a uniform behavior policy b (probability of a_1 and a_2 is 0.5 in both states). Additionally, consider the target policy π which takes a_1 with probability 0.1 and a_2 with probability 0.9 in both states. We consider the undiscounted case ($\gamma = 1$).

1. What are the Q functions Q^b and Q^π under both policies?

2. Now consider a dataset gathered using b $(s_1, a_2, 0, s_2, a_1, -1), (s_1, a_2, 0, s_2, a_2, +1)$. Consider a Q-function that is initialized as per the following table

	a_1	a_2
s_1	-1	0.5
s_2	-1	+1

   Apply one pass of Sarsa on the dataset with a learning rate of 0.1. How does the change in Q function relate to the two functions you calculated in sub-question 1?
   hint: throughout this question, only $Q(s_1, a_2)$ will change. Why?

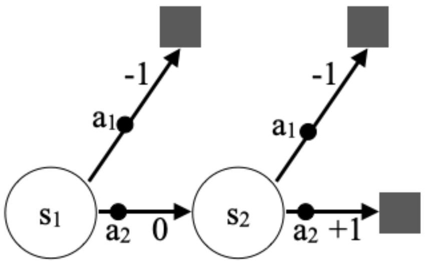


Figure 2: Example MDP

3. We can use expected Sarsa to estimate the Q-function Q^π. Apply a single pass, and note how the change in Q function relates to the two functions of sub-question 1.

4. Another possibility for off-policy learning is applying Sarsa with importance weight. Again do one pass and notice the change in Q-function.

5. We could also learn a V function, e.g. through TD(0), off policy. For example, by using importance weights. Why do you think the book doesn't cover this?

6. Could you do something like expected Sarsa for learning a V function? If yes, apply one pass. If not, explain why this is the case.

7. Why is Q-learning considered an off-policy control method? (ex. 6.11 in RL:AI)

4.3 *Exam question - N-Step Temporal Difference Learning

This exercise has been taken from a previous exam (perhaps lightly edited) and can require a bit more insight. It should give you a good idea of the level of exam questions.

Consider the undiscounted and deterministic random walk environment in Figure 3 with 19 states and two terminal states. The state in the middle (J) is always the start state and we want to apply n-step TD using a random behavior policy choosing between going left or right with equal probability at each step. The rewards are indicated above each transition arrow. The initial value of each state is set to 0. We run n-step TD for different values of n and learning rate α. To the right you can see the average RMS error over all states compared to the true state values after 10 episodes.

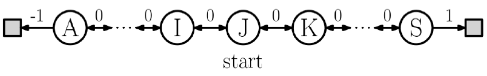


Figure 3: Random walk environment.

1. We observe that we need a small learning rate when n is big in order to reduce the error. Why is this the case compared to when using a smaller n?

2. Why does an intermediate value for n work best in this scenario (assuming a good choice for α)? You can argue about the disadvantages of the corner cases.

3. To use off-policy data with n-step methods, we can use importance weights. Do you think off-policy n-step learning with a greedy target policy (like in Q-learning) would be effective? Explain your answer.

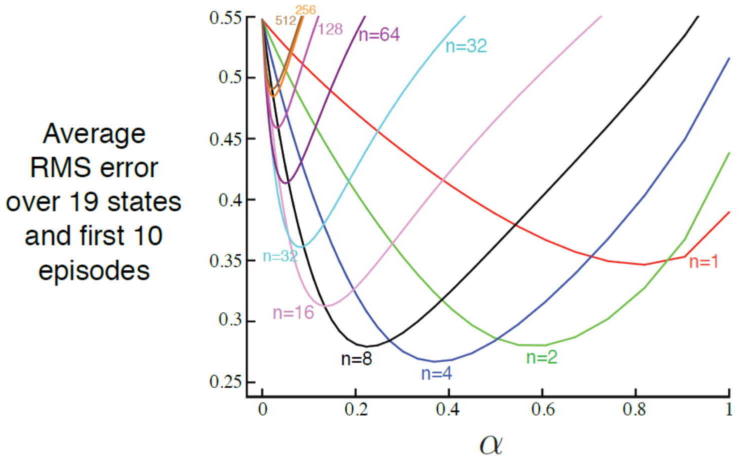


Figure 4: Performance of n-step TD methods as a function of α for various values of n on a 19-state random walk task.

4.4 Homework: Coding assignment - Temporal Difference Learning

1. Download the notebook *RLLab3_TD.zip* from canvas assignments and follow the instructions.
2. In the lab notebook you plotted the average returns during training for Q-learning and SARSA algorithms. Which algorithm achieves higher average return? Do you observe the same phenomenon as in the Example 6.6 in the book? Explain why or why not.

4.5 Homework: Maximization Bias

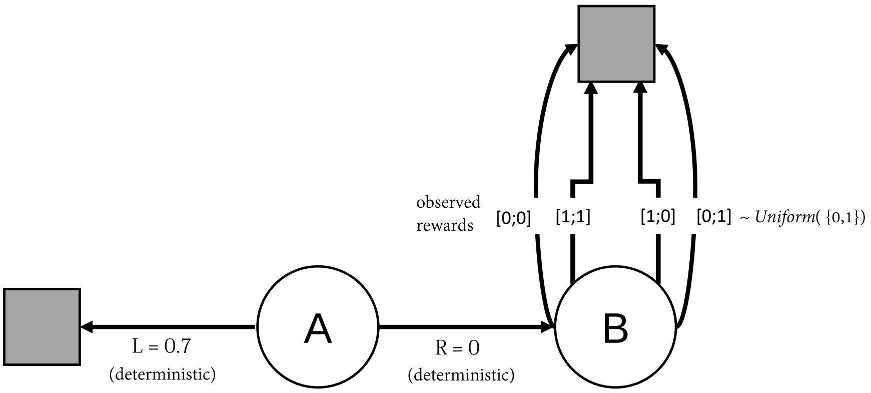


Figure 5: MDP: Maximization Bias

Consider the undiscounted MDP in Figure 5 where we have a starting state A with two actions (L,R), one ending in the terminal state T (with $V(T) = 0$) and always yielding a reward of 0.7 and another action that transitions to state B with zero reward. From state B we can take four different actions each transitioning to the terminal state and yielding a reward of either 0 or 1 with equal probability. Suppose that we sample two episodes where we try action L and eight episodes where try action R followed by an action from state B (each action in B is used twice). The observed rewards for each of the four actions from state B are indicated in the Figure, e.g. the rightmost action received a reward of 0 and a reward of 1. *Note: The scenario in this exercise is a bit artificial, as it is designed highlight the effect of maximization bias.*

1. Suppose we repeatedly apply Q-learning and SARSA on the observed data until convergence. Give all final state-action values for Q-learning and SARSA respectively.

   Note: Repeated application of an update means the value will converge to 'target value' for that state-action pair (and to the average of 'target values' if there are multiple datapoints available for that state-action pair).

2. What are the true state-action values that we would expect to get (after convergence) if we continued sampling episodes.

3. This problem suffers from maximization bias. Explain where this can be observed. Do both Q-learning and SARSA suffer from this bias? Why/why not?

4. To circumvent the issue of maximization bias, we can apply Double Q-learning. Use the given example to explain how Double Q-learning alleviates the problem of maximization bias.