RBE 595 — Reinforcement Learning
Midterm Exam

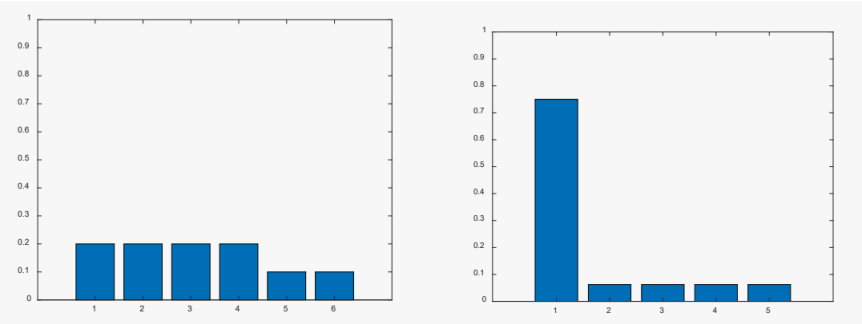
Arjan Gupta

Problem 1

Consider two random variables with distributions below:

$$p = \{0.2, 0.2, 0.2, 0.2, 0.1, 0.1\}$$

$$q = \{0.75, 0.0625, 0.0625, 0.0625, 0.0625\}$$



A. [4 points] Calculate the entropy for each variable.
B. [4 points] Intuitively how can you tell which variable has a higher entropy without calculating the entropy numerically? What does higher entropy mean?

Answer

A. The entropy for each variable is given by:

$$\begin{aligned} H(p) &= -\sum_{i=1}^{6} p_i \log_2 p_i \\ &= -\left(0.2\log_2 0.2 + 0.2\log_2 0.2 + 0.2\log_2 0.2 + 0.2\log_2 0.2 + 0.1\log_2 0.1 + 0.1\log_2 0.1\right) \\ &= -(4 * 0.2\log_2 0.2 + 2 * 0.1\log_2 0.1) \\ &= 2.5219 \end{aligned}$$

$$\begin{aligned} H(q) &= -\sum_{i=1}^{5} q_i \log_2 q_i \\ &= -\left(0.75\log_2 0.75 + 4 * 0.0625\log_2 0.0625\right) \\ &= 1.3112 \end{aligned}$$

B. Entropy is defined as the lack of expected information, or the 'surprise'/uncertainty of a random variable. We can tell that q has a higher entropy than p without calculating the entropy, because the histogram for q shows that there is a 'surprising' value of 0.75, which goes against the trend of the other values (which are all 0.0625). On the other hand, the histogram for p shows that all the values are fairly close to each other. In general, higher entropy means that the random variable has more uncertainty, so the likelihood of encountering a value closer to the expected value is lower.

Problem 2

2- [5 points] Which equation is correct? draw the corresponding backup-diagram and explain.

A. $v_*(s) = \sum_{r,s} \pi(a|s)p(s',r|s,a)[r + \gamma q_*(s')]$

B. $v_*(s) = \max_a q_*(s,a)$

C. $v_*(s) = \max_a \sum_{r,s} p(s',r|s,a)[r + \gamma q_*(s',a)]$

D. $v_*(s) = \max_a \sum_{r,s'} p(s',r|s,a)[r + \gamma v_*(s')]$

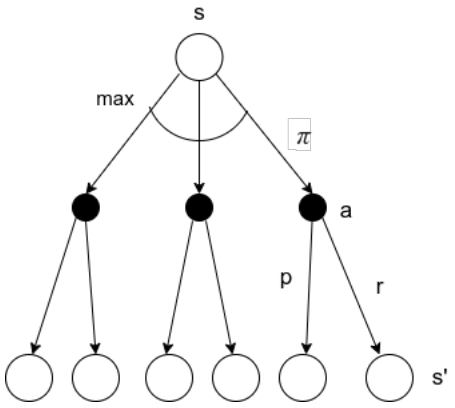
E. None of the above

F. B and D

Answer

Option F (B and D) is correct.

I have drawn the backup diagram for v_* as given below:



Explanation

v_* is the optimal state-value function. The backup diagram for v_* shows us that, if we start at a state s, we choose the action that maximizes the value of the state-action pair, and then we take the action. The action choice is taken using our current policy, π. Once we take the action, we end up in a new state, s', and we get a reward, r. The resultant state s' as well as the reward r are chosen according to the dynamics of the environment, $p(s', r|s, a)$, which is not something we can control. In summary, the optimal state value function chooses the action that maximizes the value of the state-action value.

Problem 3

Consider a vehicle with 4 actions (left, right, up, down). There's no uncertainty in the outcome of the action (i.e. when left is commanded, the left state is achieved). The actions that cause the vehicle outside the grid, leave the state unchanged. The reward for all transition is -1 except when the goal is reached where the reward is zero. Discount factor $\gamma = 1$.
The figure on left shows the name of the states and figure on the right shows the state-value $V(s)$, for each state under a uniform random policy.

Termination	a	b	c
d	e	f	g
h	i	j	k
l	m	n	Termination

Termination	-14	-20	-22
-14	-18	?	-20
-20	-20	-18	-14
-22	-20	-14	Termination

A. [4 points] What is $q(k, down)$?
B. [4 points] What is $q(g, down)$?
C. [4 points] What is $V(f)$?

Answer

A. We know the equation for $q(s, a)$ is given by,

$$q(s,a) = \sum_{s',r} p(s', r|s, a) \left[r + \gamma V(s')\right]$$

We can calculate $q(k, down)$ as follows:

$$\begin{aligned} q(k, down) &= \sum_{s',r} p(s', r|k, down) \left[r + \gamma V(s')\right] \\ &= 1 \left[0 + 1 \cdot 0\right] \\ &= 0 \end{aligned}$$

B. We can calculate $q(g, down)$ as follows:

$$\begin{aligned} q(g, down) &= \sum_{s',r} p(s', r|g, down) \left[r + \gamma V(s')\right] \\ &= 1 \left[-1 + 1 \cdot -14\right] \\ &= -15 \end{aligned}$$

C. We know the equation for $V(s)$ is given by,

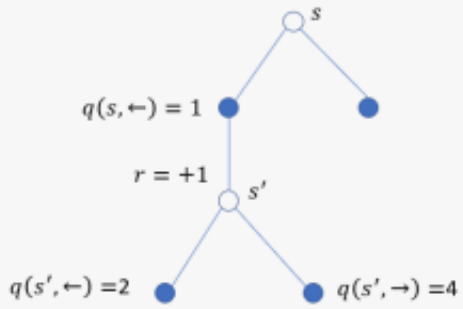
$$V(s) = \sum_{a} \pi(a|s) q(s, a)$$

We can calculate $V(f)$ as follows:

$$\begin{aligned}
V(f) &= \sum_{s',r} p(s', r|f, \pi(f)) \left[r + \gamma V(s')\right] \\
&= 0.25\left[-1 + 1 \cdot -18\right] + 0.25\left[-1 + 1 \cdot -20\right] + 0.25\left[-1 + 1 \cdot -20\right] + 0.25\left[-1 + 1 \cdot -18\right] \\
&= 0.25\left[-19\right] + 0.25\left[-21\right] + 0.25\left[-21\right] + 0.25\left[-19\right] \\
&= 0.25\left[-80\right] \\
&= -20
\end{aligned}$$

Problem 4

4- Consider the state-action interaction below. The $q(s,a)$ written next to each action (left and right) is the initial estimate.



consider discount factor $\gamma = 0.5$ and learning rate $\alpha = 0.1$:

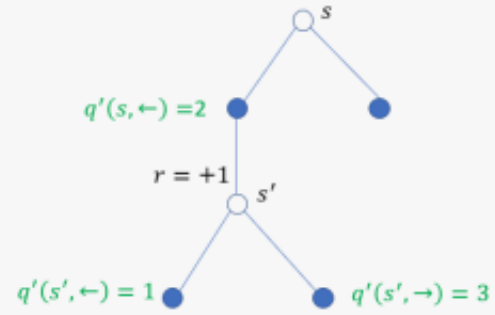
A. [4 points] What is the target value as well as updated value of $q(s,\leftarrow)$ using SARSA algorithm if the action at s' was the left action. What is the target value as well as updated value of $q(s,\leftarrow)$ using SARSA algorithm if the action at s' was the right action.

B. [4 points] Assume that the action at s' has a distribution in such a way that it is 30 percent left action and 70 percent right action. What is the expected SARSA target value and as well expected value of $q(s,\leftarrow)$ under SARSA algorithm?

C. [4 points] What is the target value as well as updated value of $q(s,\leftarrow)$ using Q-learning algorithm.

D. [4 points] Does the distribution of the action at s' have an effect on the Q-learning target value?

Consider now we have another batch of initial estimates for the action-values. We call then $q'(s,a)$. They are shown in the diagram below in green color:



E. [4 points] What is the updated value of $q(s,\leftarrow)$ using double-Q learning algorithm? (You need to use both green and black initial estimates)

F. [4 points] What is the updated value of $q'(s,\leftarrow)$ using double-Q learning algorithm? (You need to use both green and black initial estimates)

Problem 4 continued on next page...

Answer

A. The target value for SARSA is given by $R + \gamma Q(S_{t+1}, A_{t+1})$.
The update rule for SARSA is given by $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[target - Q(S_t, A_t)]$.

For left action at s':
The target SARSA value at $q(s, left)$ is given by $1 + 0.5 \cdot 2 = 2$
The update value for SARSA is given by $1 + 0.1 \cdot (2 - 1) = 1.1$

For right action at s':
The target SARSA value at $q(s, right)$ is given by $1 + 0.5 \cdot 4 = 3$
The update value for SARSA is given by $1 + 0.1 \cdot (3 - 1) = 1.2$

B. The target value for Expected SARSA is given by $R + \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1}, a)$.
The update rule for Expected SARSA is given by $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[target - Q(S_t, A_t)]$.

The policy distribution for s' is given by $\pi(left|s') = 0.3$ and $\pi(right|s') = 0.7$.

The target Expected SARSA value at $q(s, left)$ is given by $1 + 0.5(0.3 \cdot 2 + 0.7 \cdot 4) = 2.7$
The update value for Expected SARSA is given by $1 + 0.1 \cdot (2.7 - 1) = 1.17$

C. The target value for Q-learning is given by $R + \gamma \max_a Q(S_{t+1}, a)$.
The update rule for Q-learning is given by $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[target - Q(S_t, A_t)]$.

The target Q-learning value at $q(s, left)$ is given by $1 + 0.5 \cdot 4 = 3$
The update value for Q-learning is given by $1 + 0.1 \cdot (3 - 1) = 1.2$

D. The distribution of the action at s' does not affect the target value for Q-learning, because the target value for Q-learning is given by $R + \gamma \max_a Q(S_{t+1}, a)$, which does not depend on the action distribution at s'.

E. With the new batch of initial estimates, and applying double Q-learning,
our Q_1 target is given by $R + \gamma Q_2(S_{t+1}, \arg\max_a Q_1(S_{t+1}, a))$
and our Q_2 target is given by $R + \gamma Q_1(S_{t+1}, \arg\max_a Q_2(S_{t+1}, a))$

The update rule for Q_1 is given by $Q_1(S_t, A_t) \leftarrow Q_1(S_t, A_t) + \alpha\,[targetQ_1 - Q_1(S_t, A_t)]$.
The update rule for Q_2 is given by $Q_2(S_t, A_t) \leftarrow Q_2(S_t, A_t) + \alpha\,[targetQ_2 - Q_2(S_t, A_t)]$.

The target value for $q(s, left)$ is given by $1 + 0.5 \cdot q'(s', right) = 1 + 0.5 \cdot 3 = 2.5$
The update value for $q(s, left)$ is given by $1 + 0.1 \cdot (2.5 - 1) = 1.15$

F.
The target value for $q'(s, left)$ is given by $1 + 0.5 \cdot q(s', right) = 1 + 0.5 \cdot 4 = 3$
The update value for $q'(s, left)$ is given by $2 + 0.1 \cdot (3 - 2) = 2.1$

Problem 5

As we discussed, n-step off-policy return via bootstrapping can be written as:

$$G_{t:h} = \rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right) \tag{1}$$

where $\rho_t = \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$ is the importance sampling ratio between target and behavior policy.
Using control variates, the return can be written as:

$$G_{t:h} = \rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right) + (1 - \rho_t) V_{h-1}(S_t) \tag{2}$$

• [8 points] Prove that introducing the control variates in equation (2) does not add any bias to the original return (i.e. equation (1)) in expectation.

Answer

We know that the expectation of the return is given by,

$$\begin{aligned}
\mathbb{E}[G_{t:h}] &= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right) + (1 - \rho_t) V_{h-1}(S_t)] \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + \mathbb{E}[(1 - \rho_t) V_{h-1}(S_t)] \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + (1 - \mathbb{E}[\rho_t]) \, \mathbb{E}[V_{h-1}(S_t)] \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + (1 - \mathbb{E}[\rho_t]) V_{h-1}(S_t)
\end{aligned}$$

We know that the expectation of the importance sampling ratio is given by,

$$\begin{aligned}
\mathbb{E}[\rho_t] &= \mathbb{E}\left[\frac{\pi(A_t|S_t)}{b(A_t|S_t)}\right] \\
&= \sum_a \pi(a|S_t) \frac{\pi(a|S_t)}{b(a|S_t)} \\
&= \sum_a \pi(a|S_t) \frac{\pi(a|S_t)}{\sum_{a'} \beta(a'|S_t)} \\
&= \sum_a \pi(a|S_t) \frac{\pi(a|S_t)}{\sum_{a'} \pi(a'|S_t)} \\
&= \sum_a \pi(a|S_t) \\
&= 1
\end{aligned}$$

Therefore, we can write the expectation of the return as,

$$\begin{aligned}
\mathbb{E}[G_{t:h}] &= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + (1 - \mathbb{E}[\rho_t]) V_{h-1}(S_t) \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + (1 - 1) V_{h-1}(S_t) \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)] + 0 \\
&= \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)]
\end{aligned}$$

We know that the expectation of the return without control variates is given by,

$$\mathbb{E}[G_{t:h}] = \mathbb{E}[\rho_t \left(R_{t+1} + \gamma G_{t+1:h}\right)]$$

We can see that the expectation of the return without control variates is the same as the expectation of the return with control variates. Therefore, introducing the control variates in equation (2) does not add any bias to the original return (i.e. equation (1)) in expectation.

Problem 6

(Exercise 8.1, page 166) The nonplanning method looks particularly poor in Figure 8.3 because it is a one-step method; a method using multi-step bootstrapping would do better. Do you think one of the multi-step bootstrapping methods from Chapter 7 could do as well as the Dyna method? Explain why or why not.

Answer

Let us analyze how the n-step TD method would be applied to this problem.

Firstly, in n-step TD, the $G_{t:t+n}$ is given as,

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}, A_{t+n})$$

One thing to notice is that, as given in the problem prompt, the reward for all actions is 0 except for the goal state, where the reward is 1. Therefore, we have two cases, depending on whether the goal state is reached within n steps or not.

If $t < T$, then $G_{t:t+n}$ is given as,

$$G_{t:t+n} = 0.95^n Q(S_{t+n}, A_{t+n})$$

If $t \geq T$, then $G_{t:t+n}$ is given as,

$$G_{t:t+n} = 0.95^{k-1} + 0.95^k Q(S_{t+n}, A_{t+n})$$

Where k is the number of steps taken before reaching the goal state (i.e. $k = T - t$).

And $Q(S_t, A_t)$ is given as,

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \left[G_{t:t+n} - Q(S_t, A_t) \right]$$

We can see that if we use 1-step TD, the algorithm's environment-interaction steps will be very similar to Dyna-Q with $n = 0$ planning steps.

However, I believe that in this particular problem, **the general n-step TD method will not perform as well as n-planning step Dyna-Q**, because it will take longer than Dyna-Q to learn the state-action values of the states leading up to the goal state. This is because, initially the state-action values of all the states will be 0. Then, the n-step TD method will only update the value of the state-action pair that encounters the goal state. All the states leading up to the goal state will simply have their value updated to 0. After a few episodes, the state-action values will be updated to non-zero values, but it will take longer than Dyna-Q.

On the other hand, Dyna-Q will use the simulated experience to update the value of all the state-action pairs that were encountered in the simulated experience. It will make good use of computational resources in its planning phase. With only 1 episode and a sufficiently large number of planning steps, Dyna-Q will be able to learn non-zero values for the state-action values leading up to the goal state.

Problem 6 continued on next page...

By the second or third episode, Dyna-Q will have learned the state-action values of the states leading up to the goal state much better than the n-step TD method. Perhaps after a high number of episodes, the n-step TD method will catch up to Dyna-Q, but it will be behind initially. Therefore, I think if we compare the general performance of the two algorithms for this particular problem, Dyna-Q will perform better than the n-step TD method.

Problem 7

(Exercise 8.2, page 168) Why did the Dyna agent with exploration bonus, Dyna-Q+, perform better in the first phase as well as in the second phase of the blocking and shortcut experiments?

Answer

In the first phase, as described in Example 8.2, the maze has an opening on the near side, and a wall on the far side. After 1000 time steps, the short path is blocked and the far side opening is created. The graph for the performance of Dyna-Q and Dyna-Q+ is shown below:
We can see that Dyna-Q+ performs better than Dyna-Q before and after the 1,000 time step mark. Before the 1,000 time step mark, Dyna-Q+ performs better because it is able to explore the grid-world more than Dyna-Q. This is because Dyna-Q+ uses the exploration bonus in its planning phase, given by $R = r + \kappa\sqrt{\tau(s,a)}$. This encourages the agent to find the far edge faster than Dyna-Q. After the 1,000 time step mark, Dyna-Q+ performs better again for the same reason. Since the short path is blocked, Dyna-Q+ is able to explore the grid-world faster than Dyna-Q.

In the second phase, as described in Example 8.3, the maze has an opening on far side, and after the 3,000 time step mark, the short path is additionally opened. The graph for the performance of Dyna-Q and Dyna-Q+ is shown below:
While the performance of Dyna-Q+ and Dyna-Q is similar before the 3,000 time step mark, we can see that around roughly the 4,000 time step mark, Dyna-Q+ begins to perform noticeably better than Dyna-Q. Here, the reward bonus is helping Dyna-Q+ create trajectories that lead to the short path. This is because the reward bonus is given by $R = r + \kappa\sqrt{\tau(s,a)}$, where $\tau(s,a)$ is the number of time steps since the last visit to state s after taking action a. Therefore, the reward bonus is higher for states that have not been visited in a long time. This encourages the agent to explore the grid-world more, and find the short path. The Dyna-Q agent does not have this advantage, and in fact never finds the short path, as noted by the book, "even with an ϵ-greedy policy, it is very unlikely that an agent will take so many exploratory actions as to discover the shortcut." So, exploration in the planning phase is what gives Dyna-Q+ the advantage over Dyna-Q.