NAME1: Mahsa Mojtahedi
STUDENTID1:12166510
MAIL1:
mahsa.mojtahedi@student.uva.nl

NAME2: Melika Ayoughi
STUDENTID2:12224855
MAIL2: melika.ayoughi@uva.student.nl

Homework Assignment 2
Reinforcement Learning, 19/20

2019-10-06

1 5.4 Gradient Descent Methods

1.1 Why is the Monte Carlo target, G_t, an unbiased estimate of $v_\pi(S_t)$?

In MC, the states in the examples are the states generated by interaction (or simulated interaction) with the environment using policy π. Because the true value of a state is the expected value of the return following it, the Monte Carlo target $U_t = G_t$ is by definition an unbiased estimate of $v_\pi(S_t)$. Thus, the general SGD method converges to a locally optimal approximation to $v_\pi(S_t)$. Thus, the gradient-descent version of Monte Carlo state-value prediction is guaranteed to find a locally optimal solution.

1.2 Why does using the Dynamic Programming target result in a semi-gradient method?

Unlike MC, if a bootstrapping estimate of $v_\pi(S_t)$ is used as the target (Ut) like the DP target, it will depend on the current value of the weight vector w_t (while an assumption is made that the target is independent of w_t), which implies that they will be biased and that they will not produce a true gradient-descent method. Bootstrapping methods are not in fact instances of true gradient descent, because they take into account the effect of changing w_t on the estimate, but ignore its effect on the target. Also, they include only a part of the gradient and, thus called semi-gradient methods.

1.3 Despite not being unbiased, semi-gradient methods that use bootstrapping have certain advantages w.r.t. Monte Carlo approaches. Why, for example, would you prefer bootstrapping in the Mountain Car problem?

Although semi-gradient (bootstrapping) methods do not converge as robustly as gradient methods, they typically enable significantly faster learning. Another is that they enable learning to be continual and online, without waiting for the end of an episode. For example in the mountain car problem, when using MC, the reward and the termination is received only when the car reaches the top of the hill which happens rarely because a complex sequence of actions need to be taken. Thus, MC updates might take a long time while the semi gradient method updates per step with estimated values and leads to a faster convergence.

2 Geometry of linear value-function approximation

2.1 What is the Bellman error vector after initialization with $w = 1$ and using $\gamma = 1$?

$$\delta = \sum_a \pi(a|s) \sum_{s',r'} P(s', r|s, a)[r +_w (s')] - v_w(s)$$

We need to calculate the δ value for each state. For both cases the value of $\pi(a|s)$ and $P(s', r|s, a)$ is 1 since there is only one action and the environment is deterministic.

$$\delta_{s_1} = [r + \gamma v_w(s_2)] - v_w(s_1) = [r + \gamma w * \phi_2] - w * \phi_1 = 0 + 1.1.2 - 1.1 = 1$$
$$\delta_{s_2} = [r + \gamma v_w(s_1)] - v_w(s_2) = [r + \gamma w * \phi_1] - w * \phi_2 = 0 + 1.1.1 - 1.2 = -1$$
$$\bar{\delta}_w = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

2.2 What is the Mean Squared Bellman Error?

Note that we consider same weights for each state ($\mu = \frac{1}{2}$) because each state is equally important in this example.

$$\overline{BE}_w = \|\bar{\delta}_w\|^2_\mu = \frac{1}{2}(1)^2 + \frac{1}{2}(-1)^2 = 1$$

2.3 What w results in the value functions that is closest (in least-squares sense) to the target values $B_\pi v_w$?

We need to take the gradient with respect to w to find w^*.

$$
\begin{aligned}
w^* &= argmin_w \overline{BE}(w) \\
&= argmin_w \frac{1}{2}[(r + \gamma v(2) - \phi_1 w)]^2 + \frac{1}{2}[(r + \gamma v(1) - \phi_2 w)]^2 \\
&= argmin_w \frac{1}{2}[(2 - w)^2 + (1 - 2w)^2] \\
&= argmin_w \frac{1}{2}[5w^2 - 8w + 5] \\
\implies w^* &= \frac{\partial(5w^2 - 8w + 5)}{\partial w} = 0 \\
\implies w^* &= \frac{4}{5} \\
\implies \Pi B^\pi v_w &= v_{w^*} = \begin{bmatrix} \frac{4}{5} \\ \frac{8}{5} \end{bmatrix}
\end{aligned}
$$

2.4 Plot v_w, $B^\pi w$ and $\Pi B^\pi v_w$. Explain what is happening.

$$
\begin{aligned}
v_w &= \begin{bmatrix} 1 \\ 2 \end{bmatrix} \\
v_{w^*} = \Pi B^\pi v_w &= \begin{bmatrix} \frac{4}{5} \\ \frac{8}{5} \end{bmatrix} \\
B^\pi v_w &= \begin{bmatrix} 2 \\ 1 \end{bmatrix}
\end{aligned}
$$

In Figure 1, contrary to Figure 11.3 in the textbook, we have only two states, and as a result, the space of all value function is a two dimensional space. Also, since we have one w, the space of all possible weights would be a line in this space, this line is the linear subspace of all the value functions we can represent by our approximation. In the previous sections, we have first calculated v_w for $w = 1$. Applying the Bellman Operator on v_w ($B^\pi v_w$) will bring this approximate value out of the subspace, and find w^*, which minimizes the Bellman error. We can project this value back to the linear subspace with the projection operator Π, resulting in v_{w^*}. The blue line corresponds to this projection, and the red dotted line shows the linear subspace.

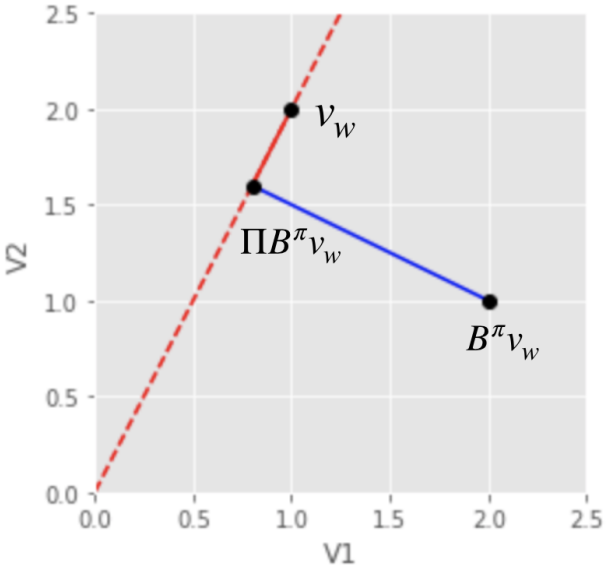

Figure 1: Red line: the space of all possible weights, v_{w^*} is a projection of $B^\pi v_w$ on the red line.

3 Compatible Function Approximation Theorem

3.1 Show that $\mathbb{E}_a[\hat{q}_w(s,a)] = 0, \forall s \in S$. Briefly interpret this result.

$$\begin{aligned}\mathbb{E}_a[\hat{q}_w(s,a)] &= \mathbb{E}_a[w^T \nabla_\theta log\pi_\theta(s,a)] \\ &= w^T \mathbb{E}_a[\nabla_\theta log\pi_\theta(s,a)]\end{aligned}$$

Assuming that the actions are discrete, we can open the expectation as follows. A similar calculation would apply to the continuous action case, only then the sum would be replaced with an integral over all the actions.

$$\begin{aligned}w^T \mathbb{E}_a[\nabla_\theta log\pi_\theta(s,a)] &= w^T \sum_a \pi_\theta(s,a) \nabla_\theta log\pi_\theta(s,a) \\ &= w^T \sum_a \pi_\theta(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} \\ &= w^T \sum_a \nabla_\theta \pi_\theta(s,a) \\ &= w^T \nabla_\theta \sum_a \pi_\theta(s,a)\end{aligned}$$

Since the sum (or integral in the continuous case)of $\pi_\theta(s,a)$ over all the actions is always 1, and since the derivative of a constant is 0, we have:

$$\mathbb{E}_a[\hat{q}_w(s,a)] = 0$$

This result tells us that our approximation of the state-action value is biased, since, contrary to the real state-action value, its expectation over all actions does not equal the value of that state.

3.2 Evaluate the expectation $\mathbb{E}_a[q_\pi(s,a) - v_\pi(s)] \forall s \in S$.

$$\begin{aligned}\mathbb{E}_a[q_\pi(s,a) - v_\pi(s)] &= \mathbb{E}_a[q_\pi(s,a)] - v_\pi(s) \\ &= v_\pi(s) - v_\pi(s) \\ &= 0\end{aligned}$$

3.3 What do you conclude from the results in 1. and 2.

We see that the expectation of both the advantage function and the $\hat{q}_w(s,a)$ equals zero. Therefore, $\hat{q}_w(s,a)$ is a better approximation of the advantage function, rather than the true state-action value ($q_\pi(s,a)$)

3.4 Considering the policy given in the homework, give an expression for the parametrization of $\hat{q}_w$ that satisfies the compatible function approximation theorem.

We know that a function in the form of $w^T \nabla_\theta log\pi_\theta(s,a)$ would satisfy the compatible function approximation theorem. Therefore, we simply need to insert our policy in this formula:

$$\pi_\theta(s,a) = \frac{e^{\theta^T \phi_{sa}}}{\sum_b e^{\theta^T \phi_{sb}}}$$

$$\implies log(\pi_\theta(s,a)) = \theta^T \phi_{sa} - log \sum_b e^{\theta^T \phi_{sb}}$$

$$\begin{aligned}\implies \nabla_\theta log(\pi_\theta(s,a)) &= \phi_{sa} - \nabla_\theta log \sum_b e^{\theta^T \phi_{sb}} \\ &= \phi_{sa} - \frac{\nabla_\theta \sum_b e^{\theta^T \phi_{sb}}}{\sum_b e^{\theta^T \phi_{sb}}} \\ &= \phi_{sa} - \frac{\sum_b e^{\theta^T \phi_{sb}} \phi_{sb}}{\sum_b e^{\theta^T \phi_{sb}}} \\ &= \phi_{sa} - \sum_b \pi_\theta(s,b) \phi_{sb}\end{aligned}$$

Therefore, the following function will satisfy the compatible function theorem:

$$\hat{q}_w(s,a) = w^T[\phi_{sa} - \sum_b \pi_\theta(s,b)\phi_{sb}]$$