Università degli Studi di Milano, Data Science for Economics Master programme

Control and VFA

Prof. Alfio Ferrara

Reinforcement Learning

Instability and divergence of off-policy methods with VFA

> Example and discussion from Section 11.2 of *Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.*

Example: Let's take an MDP where the parameters vector has a single component $\mathbf{w} = [w]$ and let's take two states $s_1 = w$ and $s_2 = 2w$, where w and $2w$ are the estimated values. We also have $\mathbf{x}(s_1) = [1]$ and $\mathbf{s_2} = 2$. We perform linear approximation. In s_1 the only possible action is a deterministic transition to s_2, with reward of 0.

Now suppose that initially $w = 10$. Since $(s_1, a, 0, s_2)$ leads from a value of 10 to a value of 20, w will be increased to raise the estimated value of s_1.

Assuming $\gamma \approx 1$ and $\alpha = 0.1$, the TD error will be ≈ 10 and the new value of s_1 is updated to $s_1 \approx 11$. But this way, also s_2 will be increased to $s_2 \approx 22$.

When the transition occurs again, we will have a TD error ≈ 11, which will end up in a further increment of w.

According to this schema, the estimated value w will diverge to infinity.

More formally, the TD error is:

$$\begin{aligned}\Delta_t &= r_{t+1} + \gamma \hat{V}(s_{t+1}, \mathbf{w}_t) - \hat{V}(s_t, \mathbf{w}_t) \\ &= r_{t+1} + \gamma \mathbf{x}(s_2)^T \mathbf{w} - \mathbf{x}(s_1)^T \mathbf{w} \\ &= 0 + \gamma 2 w_t - w_t \\ &= (2\gamma - 1) w_t\end{aligned} \tag{1}$$

Thus, performing TD(0) update, we obtain

$$\begin{aligned}w_{t+1} &= w_t + \alpha \Delta_t \nabla_{\mathbf{w}} \mathbf{x}(s_1)^T \mathbf{w} \\ &= w_t + \alpha(2\gamma - 1) w_t \cdot 1 \\ &= (w_t + \alpha(2\gamma - 1)) w_t\end{aligned} \tag{2}$$

Where, in our example $w_t + \alpha(2\gamma - 1) = 10 + 0.1(2 - 1) = 11$.

Intuition: The point is that, working off-policy, the behaviour policy (the one chosen according to the training strategy, such as ϵ-greedy for example) **might select actions which the target policy never would**. Under on-policy instead, for the transition to s_2 there would also be a transition out of s_2 to a new state s_3. Unless $\hat{V}(s_3)$ is higher than $2w$, this new transition will reduce $\hat{V}(s_2)$, so that also w will be reduced.

A complete example with instability is given in the *Baird's counterexample*, discussed in Figure 11.1 of *Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.*

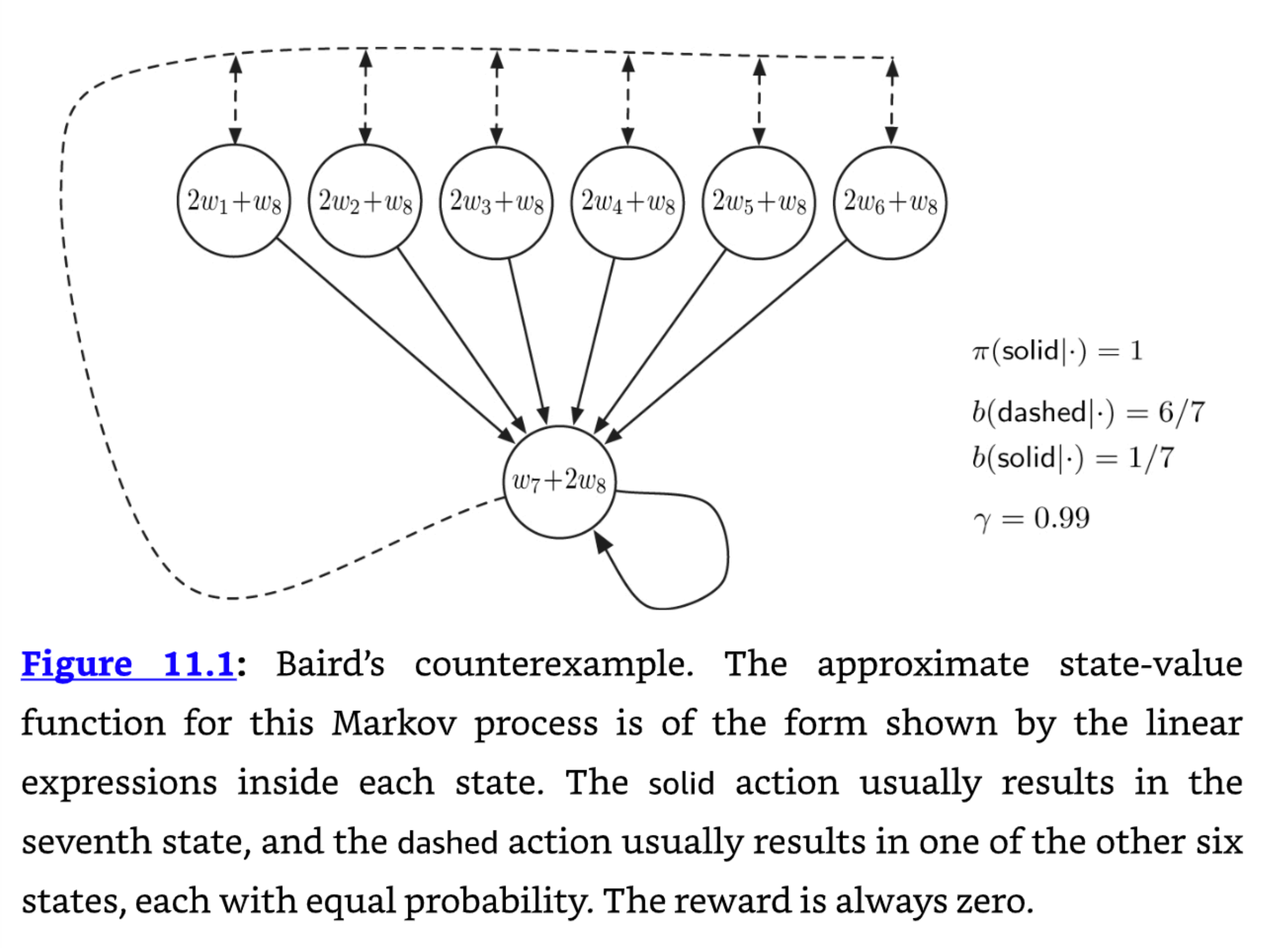
$2w_1+w_8$
$2w_2+w_8$
$2w_3+w_8$
$2w_4+w_8$
$2w_5+w_8$
$2w_6+w_8$
w_7+2w_8
$\pi(\text{solid}|\cdot) = 1$
$b(\text{dashed}|\cdot) = 6/7$
$b(\text{solid}|\cdot) = 1/7$
$\gamma = 0.99$

Figure 11.1: Baird's counterexample. The approximate state-value function for this Markov process is of the form shown by the linear expressions inside each state. The solid action usually results in the seventh state, and the dashed action usually results in one of the other six states, each with equal probability. The reward is always zero.

Discussion

There are three components that, when combined in training, may generate instability:

1. **Function approximation**: Any possible generalization over the states, either linear, neural, etc.
2. **Bootstrapping**: Any method that implies updating targets using existing estimates (such as dynamic programing and TD learning). Note that other methods, such as Monte Carlo, rely exclusively on actual rewards and complete returns, thus do not diverge.
3. **Off-policy training**: Training on a distribution of transitions other than those produced by the target policy.

Convergence of Control Methods

We ask if it is possible to approximate any V or Q value function by a linear Function Approximator. In general, given a sufficient number of features, yes. But it is not alway easy to find such an approximation.

Let's take an example of GridWorld where we have $r = -1$ on all the tiles except for $r(x = 0, y = 0) = 10$. (see `simplegrid.py` in the `rl` repository).

Let's also suppose to have $\mathbf{x}(s) = [1 \quad x \quad y]$.

Which parameters $\mathbf{w}$ are suitable to approximate $V(s; \mathbf{w})$ in this case?

Since $\hat{V}(s; \mathbf{w}) = \mathbf{x}(s)^T \mathbf{w} = w_0 + x w_1 + y w_2$, it's easy to see that parameters such $\mathbf{w} = [10 \quad -1 \quad -1]$ will do the work.

Now, what happens if $r(x = 2, y = 2) = 10$? It's also easy to see that there is not way to easily find a good approximation without providing a new feature.