1 *Off-policy Methods with Approximation

Function approximation turns out to be more difficult in the case of off-policy learning than it is in the on-policy case. This is because both the update target and the state distribution are different between the target and behaviour policies.

1.1 Semi-gradient Methods

Semi-gradient methods alter the update target to correspond to the target policy, but do not address the issue of the update distribution. As such, they may diverge in some cases (but they are often successfully used in practice nonetheless).

The tabular off-policy algorithms can be applied in this case, where we simply exchange the estimated value arrays for their equivalents under function approximation and incorporate the importance sampling ratio. For instance, the one-step, state-value algorithm is semi-gradient off-policy TD(0). This has the update

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha\rho_t\delta_t\nabla\hat{v}(S_t, \boldsymbol{w}_t), \tag{1}$$

where $\rho_t \doteq \rho_{t:t}$ is per-step importance sampling ratio. The TD error δ_t is defined as appropriate, with respect to the episodic and discounted reward or the continuing and undiscounted average reward according to the setting.

The corresponding one-step algorithm for state-values is semi-gradient Expected Sarsa. In the tabular case we did not use importance sampling for one-step, action-value methods, but with function approximation (and corresponding generalisation) it is not clear that all actions should be weighted equally.

We now give some of the multi-step generalisations.

n-step Semi-Gradient Expected Sarsa

The update is

$$\boldsymbol{w}_{t+n} = \boldsymbol{w}_t + \alpha\rho_{t+1}\cdots\rho_{t+n-1}[G_{t+n} - \hat{q}(S_t, A_t, \boldsymbol{w}_{t+n-1})]\nabla\hat{q}(S_t, A_t, \boldsymbol{w}_{t+n-1}) \tag{2}$$

where $\rho_k = 1$ when $k \geq T$ and $G_{t:n} = G_t$ when $t + n \geq T$. The returns targets are defined for the episodic case as

$$G_{t:t+n} \doteq \sum_{i=0}^{n-1}\gamma^i R_{t+i+1} + \gamma^n\hat{q}(S_{t+n}, A_{t+n}, \boldsymbol{w}_{t+n-1})$$

and in the continuing case as

$$G_{t:t+n} \doteq \sum_{i=0}^{n-1}\left(R_{t+i+1} - \bar{R}_{t+i}\right) + \hat{q}(S_{t+n}, A_{t+n}, \boldsymbol{w}_{t+n-1}).$$

n-step Tree Backup Algorithm

The updates are

$$\boldsymbol{w}_{t+n} = \boldsymbol{w}_{t+n-1} + \alpha[G_{t:t+n} - \hat{q}(S_t, A_t, \boldsymbol{w}_{t+n-1})]\nabla\hat{q}(S_t, A_t, \boldsymbol{w}_{t+n-1}) \tag{3}$$

$$G_{t:t+n} \doteq \hat{q}(S_t, A_t, \boldsymbol{w}_{t-1}) + \sum_{k=t}^{t+n-1}\delta_k\prod_{i=t+1}^{k}\gamma\pi(A_i|S_i), \tag{4}$$

where δ_t is defined in the episodic case as

$$\delta_t \doteq R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})\hat{q}(S_{t+1}, a, \boldsymbol{w}_t) - \hat{q}(S_t, A_t, \boldsymbol{w}_t)$$

and in the continuing case as

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \sum_a \pi(a|S_{t+1})\hat{q}(S_{t+1}, a, \boldsymbol{w}_t) - \hat{q}(S_t, A_t, \boldsymbol{w}_t)$$

1.3 The Deadly Triad

Instability and divergence arise whenever we have all three of *function approximation*, *bootstrapping*, *off-policy training*. Each of these has their benefits, so it is unclear which (if any!) is to be given up. Note that the difficulties are not due to uncertainty in the environment (they arise with DP), nor are they due specifically to control or generalised policy iteration.

Function Approximation allows for generalisation, dimensionality reduction and reduction in complexity. Potentially also

Bootstrapping Data and computationally efficient. Memory efficient.

Off-policy Learning Essential to some use cases (not yet mentioned in this book). Seems important to be able to learn from hypothetical actions.

1.4 Linear Value-function Geometry