7. n-step Bootstrapping

Notes

- n-*steps TD methods* span a spectrum with MC methods at one end and one-step TD methods on the other, and the best ones are often those that unify these two. MC methods uses the entire sequence of rewards, whereas one-step TD methods are based on the next reward. Such methods allows us to bootstrap from over multiple time steps, not just a single time step.
- n-step TD Prediction:
  - The *n-step TD methods* essentially bootstrap from the next n steps in order to update the estimated value of the state. These are still TD methods because they change an earlier estimate based on how it differs from a later estimate, but this time the later estimates are n steps later. Implementation can be found in page 144.
  - The *n-step return* is given by:

    $$G_{t:t+n} \doteq R_{t+1} + R_{t+2} + \cdots + \gamma^n V_{t+n-1}(S_{t+n})$$

    for $0 \leq t < T - n$, and for $t \geq T - n$, the missing terms can be taken as zero and $G_{t:t+n} \doteq G_t$. The state value algorithm is:

    $$V_{t+n}(S_t) \doteq V_{t+n-1}(S_t) + \alpha[G_{t:t+n} - V_{t+n-1}(S_t)]$$

  - The n-step return uses V_{t+n-1} to correct for the missing rewards beyond R_{t+n}, and the expectation for then-step returns is guaranteed to be a better estimate of v_π than V_{t+n-1}:

    $$\max_s \left|\mathbb{E}_\pi[G_{t:t+n}|S_t = s] - v_\pi(s)\right| \leq \gamma^n \max_s \left|V_{t+n-1}(s) - v_\pi(s)\right|$$

    This is called the *error reduction property* of n-step returns. This means that all such methods converge to the correct predictions under appropriate conditions. One-step TD methods and MC methods form the extreme of n-step TD methods.
- n-step Sarsa:
  - A one-step Sarsa can also be denote by Sarsa(0). In n-step Sarsa, we switch the state value estimates for action value estimates, as well as using an ϵ-greedy policy.
  - The n-step returns in terms of the action values are:

    $$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n})$$

    where $n \geq 1$ and $0 \leq t < T - n$. The action value estimate is then

    $$Q_{t+n}(S_t, A_t) \doteq Q_{t+n-1}(S_t, A_t) + \alpha[G_{t:t+n} - Q_{t+n-1}(S_t, A_t)]$$

    where $0 \leq t < T$, and all other state-action pairs remain unchanged. Implementation can be found in page 147.
  - For expected Sarsa, the returns can be written a

    $$G_{t:t+n} \doteq R_{t+1} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \bar{V}_{t+n-1}(S_{t+n})$$

    where $\bar{V}_t(s) = \sum_a \pi(a|s) Q_t(s, a)$, which is the expected approximate value.

- The n-steps method can help speed up learning when compared to the one-step method because the n-steps method updates the value estimates for the previous n steps, whereas the one-step method only updates the previous step.

- n-step Off-policy Learning
  - For the state values, we can include the importance sampling ratio to update our value estimates:

    $$V_{t+n}(S_t) \doteq V_{t+n-1}(S_t) + \alpha\rho_{t:t+n-1}[G_{t:t+n} - V_{t+n-1}(S_t)]$$

    where $\rho_{t:t+h} = \prod_{k=t}^{\min(h,T-1)} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$. This means that if the action is never taken by the target policy π, then the n-step return is ignored, whilst if the probability of the action taken is much more favoured by π than b, then a much higher weight is given to the return.
  - The n-step Sarsa update can be written as

    $$Q_{t+n}(S_t, A_t) \doteq Q_{t+n-1}(S_t, A_t) + \alpha\rho_{t+1:t+n}[G_{t:t+n} - Q_{t+n-1}(S_t, A_t)].$$

    Note that the $\rho_{t+1:t+n}$ is shifted one time step later. This is to account for the fact that this is an update for the action value estimate where the action has already been selected, rather than a state value estimate.
  - The off-policy Expected Sarsa would use the same update as above, except that the the importance sampling ratio will be $\rho_{t+1:t+n-1}$ instead of $\rho_{t+1:t+n}$, and that the return will use the average $\bar{V}_t$. It is a different sampling ratio because the Expected Sarsa takes into account all possible actions in the last state.

- Per-decision Methods with Control Variates
  - The off-policy method above is simple but not the most efficient, and this method here is a similar idea to what we saw in Chapter 5. Recall that the n steps return ending at horizon h can be written as

    $$G_{t:h} = R_{t+1} + \gamma G_{t+1:h}, \quad t < h < T, \quad G_{h:h} = V_{h-1}(S_h).$$

    One might think that the reward R_{t+1} can just be weighted by the the importance sampling ratio of time t, $\rho_t = \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$, and so on when we expand $G_{t+1:h}$. However, if the action at time t would never be selected by our target policy π, then $\rho_t = 0$, resulting in the n-step return becoming 0. This can result in high variance of the return.
  - Instead, we use a slightly more sophisticated approach, where

    $$G_{t:h} \doteq \rho_t(R_{t+1} + \gamma G_{t+1:h}) + (1 - \rho_t)V_{h-1}(S_t), \quad G_{h:h} \doteq V_{h-1}(S_h).$$

    Here, if $\rho_t = 0$, this wouldn't cause the estimate to shrink, but it will keep the estimate to be unchanged. The second term above is called the *control variate*. Given that ρ_t has expected value of 1, the control variate will have an expected value of 0, which means that it will not change out expected update. Note that this is also a generalisation of the on-policy definition of the n-step return, where both the target and behaviour policy are the same.
  - For action-values, the importance sampling does not involve the first action, since the action has already been selected, so it does not matter that it is an unlikely, or impossible, action under the target policy. We still need to determine the return from the state-action pair. The n-step return involving action values can be written as

    $$\begin{aligned} G_{t:h} &\doteq R_{t+1} + \gamma\Big(\rho_{t+1}G_{t+1:h} + \bar{V}_{h-1}(S_{t+1}) - \rho_{t+1}Q_{h-1}(S_{t+1}, A_{t+1})\Big) \\ &= R_{t+1} + \gamma\rho_{t+1}\Big(G_{t+1:h} - Q_{h-1}(S_{t+1}, A_{t+1})\Big) + \gamma\bar{V}_{h-1}(S_{t+1}), \quad t < h \leq T \end{aligned}$$

where if $h < T,\ G_{h:h} \doteq Q_{h-1}(S_h, A_h)$ and if $h \geq T,\ G_{T-1:h} \doteq R_T$. More details, including derivation, can be found here: http://auai.org/uai2018/proceedings/papers/282.pdf.

- The purpose of the control variates introduced here is to reduce the variance caused by the importance sampling ratios.

- Off-policy Learning Without Importance Sampling: The n-step Tree Backup Algorithm
  - The *tree-backup algorithm* is a multi-step off-policy learning algorithm. Let's look at a qualitative description of a 3-step tree-backup algorithm:

    This algorithm uses the entire tree for estimated values, which includes actions that were not selected at all levels of the n-steps, to update our estimate. The update is from the *lead nodes* of the tree. The selected action node contributes to the next level and does not participate in the current level, but the leaf nodes contribute to the target with a weight proportional to $\pi(a|S)$. For example, the first-level action $a \neq A_{t+1}$ contributes with a weight of $\pi(a|S_{t+1})$. The second-level action $a' \neq A_{t+2}$ contributes $\pi(A_{t+1}|S_{t+1})\pi(a'|S_{t+2})$, and so on for other levels.

  - The general recursive definition of the tree-backup n-step return is

    $$G_{t:t+n} \doteq R_{t+1} + \gamma \sum_{a \neq A_{t+1}} \pi(a|S_{t+1}) Q_{t+n-1}(S_{t+1}, a) + \gamma \pi(A_{t+1}|S_{t+1}) G_{t+1:t+n} \quad \text{for } t < T-1, n \geq 2.$$

    This definition is used with the typical action-value update rule for n-step Sarsa. Implementation of this can be found in page 154.

- A unifying Algorithm: n-step $Q(\sigma)$:
  - The *n-step* $Q(\sigma)$ *algorithm* is a cross between the n-step Sarsa and tree-backup algorithms. The n-step Sarsa uses only sample transitions and importance sampling ratios, whilst the tree-backup algorithm has all state-to-action transitions fully branched without sampling. At each step, we decided whether we want to take the action as a sample like Sarsa, or take the expectation over all actions like tree-backup.
  - We can let the degree of sampling at time step t be $\sigma_t \in [0, 1]$, where $\sigma = 1$ denotes full sampling. This random variable can also be set as a function of state, action or state-action pair at each time step.
  - The tree-backup update algorithm is given by

    $$G_{t:t+n} \doteq R_{t+1} + \gamma \pi(A_{t+1}|S_{t+1}) \Big(G_{t+1:t+n} - Q_{h-1}(S_{t+1}, A_{t+1}) \Big) + \gamma \bar{V}_{h-1}(S_{t+1}).$$

    The n-step Sarsa with importance sampling is very similar, with just the $\pi(A_{t+1}|S_{t+1})$ replaced ρ_{t+1}.

  - For $Q(\sigma)$, we slide between the two cases by

    $$G_{t:t+n} \doteq R_{t+1} + \gamma \Big(\sigma_{t+1} \rho_{t+1} + (1 - \sigma_{t+1}) \pi(A_{t+1}|S_{t+1}) \Big) \Big(G_{t+1:t+n} - Q_{h-1}(S_{t+1}, A_{t+1}) \Big) + \gamma \bar{V}_{h-1}(S_{t+1}).$$

    where if $h < T,\ G_{h:h} \doteq Q_{h-1}(S_h, A_h)$ and if $h = T,\ G_{T-1:T} \doteq R_T$. Implementation can be found on page 156.

Exercises

7.1

$$
\begin{aligned}
G_{t:t+n} - V_{t+n-1}(S_t) &= R_{t+1} + \gamma G_{t+1:t+n} - V_{t+n-1}(S_t) + \gamma V_{t+n}(S_{t+1}) - \gamma V_{t+n}(S_{t+1}) \\
&= \delta_t + \gamma[G_{t+1:t+n} - V_{t+n}(S_{t+1})] \\
&= \delta_t + \gamma R_{t+2} + \gamma^2 G_{t+2:t+n} - \gamma V_{t+n}(S_{t+1}) + \gamma V_{t+n+1}(S_{t+1}) - \gamma V_{t+n+1}(S_{t+1}) \\
&= \delta_t + \gamma\delta_{t+1} + ... + \gamma^{t+n-1}\delta_{t+n} \\
&= \sum_{k=t}^{t+n} \gamma^{k-1}\delta_k
\end{aligned}
$$

7.2 Not too sure if my answer in 7.1 is even valid...

7.3 Here we have $1 \leq n \leq 512$. With a larger random walk task, the number of time steps in an episode will increase, allowing us to observe a better result from the n-step TD methods with large n. With a small task of only 5 states, it would be extremely difficult to discern the performance difference between large n values, since n will be larger than the length of the episode, essentially becoming a MC method. Thus, the small task will favour small values of n. The change in the left-side outcome from 0 to -1 increases the range of possible value estimates for each of the state. However, there wouldn't be much change to the difference in the best value of n, I think...

7.4

$$
\begin{aligned}
G_{t:t+n} &\doteq R_{t+1} + \cdots + \gamma^{n-1}R_{t+n} + \gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n}) \\
&= R_{t+1} + Q_{t-1}(S_t, A_t) - Q_{t-1}(S_t, A_t) + \gamma Q_t(S_{t+1}, A_{t+1}) - \cancel{\gamma Q_t(S_{t+1}, A_{t+1})} \\
&\quad + \gamma R_{t+2} + \cancel{\gamma Q_t(S_{t+1}, A_{t+1})} - \gamma Q_t(S_{t+1}, A_{t+1}) + \gamma^2 Q_{t+1}(S_{t+2}, A_{t+2}) \\
&\quad\quad - \cancel{\gamma^2 Q_{t+1}(S_{t+2}, A_{t+2})} \\
&\quad + \dots \\
&\quad + \gamma^{n-1}R_{t+n} + \gamma^{n-1}Q_{t+n-2}(S_{t+n-1}, A_{t+n-1}) - \gamma^{n-1}Q_{t+n-2}(S_{t+n-1}, A_{t+n-1}) \\
&\quad\quad + \gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n}) - \cancel{\gamma^n Q_{t+n-1}(S_{t+n}, A_{t+n})} \\
&\quad + \cancel{\gamma^n Q_{t+n-1}(S_{t_n}, A_{t+n})} \\
&= Q_{t-1}(S_t, A_t) + \sum_{k=t}^{\min(t+n,T)-1} \gamma^{k-t}[R_{k+1} + \gamma Q_k(S_{k+1}, A_{k+1}) - Q_{k-1}(S_k, A_k)]
\end{aligned}
$$

7.5 Use a similar pseudocode to that of the off-policy n-step Sarsa, but with $V(s)$ instead of $Q(s,a)$, and a tweaked section under $\text{If } \tau \geq 0$.

Input: an arbitrary behaviour policy b such that $b(a|s) > 0$, for all $s \in \mathcal{S}, a \in \mathcal{A}$
Initialise $V(s)$ arbitrarily, for all $s \in \mathcal{S}$
Initialise π to be greedy with respect to V, or as a fixed given policy
Algorithm parameters: step size $\alpha \in (0, 1]$, a positive integer n
All store and access operations (for S_t, A_t, and R_t) can take their index mod $n + 1$

Loop for each episode:
  Initialise and store $S_0 \neq$ terminal
  Select and store an action $A_0 \sim b(\cdot|S_0)$
  $T \leftarrow \infty$
  Loop for $t = 0, 1, 2, \ldots$:
  | If $t < T$, then:
  | | Take action A_t
  | | Observe and store the next reward as R_{t+1} and the next state as S_{t+1}
  | | If S_{t+1} is terminal, then:
  | | | $T \leftarrow t + 1$
  | | else:
  | | | Select and store an action $A_{[}t + 1] \sim b\cdot|S_{t+1}$
  | $\tau \leftarrow t - n + 1$ (τ is the time whose estimate is being updated)
  | If $\tau \geq 0$:
  | | $h \leftarrow \min(\tau + n, T - 1)$
  | | $G \leftarrow V(S_h)$
  | | Loop for $k = h - 1, h - 2, \ldots, \tau + 1, \tau$:
  | | | $\rho \leftarrow \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$
  | | | $G \leftarrow \rho(R_{k+1} + \gamma G) + (1 - \rho)V_h(S_\tau)$
  | | $V(S_\tau) \leftarrow V(S_\tau) + \alpha[G - V(S_\tau)]$
  | | If π is being learned, then ensure that $\pi(\cdot|S_\tau)$ is greedy wrt V
  Until $\tau = T - 1$

7.6 We can see that the final line is the recursion equation for the importance-sampled n-step return for off-policy Expected Sarsa, without the control variate. Hence, the control variate does not change the expected value of the return. (Note that π and b are policies independent of one another, thus the nested expectation can just be swapped round, as well as ignoring the outer expectation)

$$
\begin{aligned}
\mathbb{E}_b[G_{t:h}] &= \mathbb{E}_b\Big[R_{t+1} + \gamma\Big(\rho_{t+1}G_{t+1:h} + \bar{V}_{h-1}(S_{t+1}) - \rho_{t+1}Q_{h-1}(S_{t+1}, A_{t+1})\Big)\Big] \\
&= \mathbb{E}_b\Big[R_{t+1} + \gamma\rho_{t+1}G_{t+1:h}\Big] + \gamma\mathbb{E}_b\Big[\bar{V}_{h-1}(S_{t+1}) - \rho_{t+1}Q_{h-1}(S_{t+1}, A_{t+1})\Big] \\
\left\{\mathbb{E}_b\big[\bar{V}_t(s)\big] = \mathbb{E}_b\big[\mathbb{E}_\pi[Q_t(s,a)]\big] = \mathbb{E}_\pi\big[\mathbb{E}_b[Q_t(s,a)]\big] = \mathbb{E}_b[Q_t(s,a)], \quad \mathbb{E}_b[\rho_t] = 1\right\} \\
&= \mathbb{E}_b\Big[R_{t+1} + \gamma\rho_{t+1}G_{t+1:h}\Big] \\
&\quad + \gamma\Big[\mathbb{E}_b[Q_{h-1}(S_{t+1}, A_{t+1})] - \mathbb{E}_b[Q_{h-1}(S_{t+1}, A_{t+1})]\Big] \\
&= \mathbb{E}_b\Big[R_{t+1} + \gamma\rho_{t+1}G_{t+1:h}\Big]
\end{aligned}
$$

7.7 The pseudocode here will be the same as in **7.5**, except that we would need to take care of how the recursion ends, where if $h < T,\ G_{h:h} \doteq Q_{h-1}(S_h, A_h)$ and if $h \geq T,\ G_{T-1:h} \doteq R_T$.

7.8 The TD error can be written as $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$.

$$
\begin{aligned}
G_{t:h} &\doteq \rho_t(R_{t+1} + \gamma G_{t+1:h}) + (1 - \rho_t)V_{h-1}(S_t) \\
&= \rho_t\big[R_{t+1} + \gamma[\rho_{t+1}(R_{t+2} + \gamma G_{t+2:h}) + (1 - \rho_{t+1})V_{h-1}(S_{t+1})]\big] \\
&\quad + (1 - \rho_t)V_{h-1}(S_t) \\
&= \rho_t\big[(R_{t+1} + V(S_{t+1}) - V(S_t)) + \gamma\rho_{t+1}[R_{t+2} + \gamma G_{t+2:h} - V(S_{t+1})]\big] + V(S_t) \\
&= \rho_t\Big[\delta_t + \gamma\rho_{t+1}\big[R_{t+2} + \gamma[\rho_{t+2}(R_{t+3} + \gamma G_{t+3:h}) + (1 - \rho_{t+2})V(S_{t+2})] \\
&\quad - V(S_{t+1})\big]\Big] + V(S_t) \\
&= \rho_t\Big[\delta_t + \gamma\rho_{t+1}\big[\delta_{t+1} + \gamma\rho_{t+2}[R_{t+2} + \gamma G_{t+3:h} - V(S_{t+2})]\big]\Big] + V(S_t) \\
&= \sum_{k=t}^{\min(h,T-1)} \big[\gamma^{k-t}\rho_{t:k}\delta_k\big] + V(S_t), \quad \text{where } \rho_{t:k} = \prod_{l=t}^{k}\rho_l = \prod_{l=t}^{k}\frac{\pi(A_l|S_l)}{b(A_l|S_l)}
\end{aligned}
$$

7.9 The Expected Sarsa TD error is $\delta'_t = R_{t+1} + \gamma\bar{V}(S_{t+1}) - Q(S_t, A_t)$. For the action value version:

$$
\begin{aligned}
G_{t:h} &\doteq R_{t+1} + \gamma\rho_{t+1}\big(G_{t+1:h} - Q_{h-1}(S_{t+1}, A_{t+1})\big) + \gamma\bar{V}_{h-1}(S_{t+1}) \\
&= R_{t+1} + \gamma\rho_{t+1}\Big(R_{t+2} + \gamma\rho_{t+2}\big(G_{t+2:h} - Q(S_{t+2}, A_{t+2})\big) + \gamma\bar{V}(S_{t+2}) \\
&\quad - Q(S_{t+1}, A_{t+1})\Big) + \gamma\bar{V}(S_{t+1}) \\
&= R_{t+1} + \gamma\rho_{t+1}\Big(\big[R_{t+2} + \gamma\bar{V}(S_{t+2}) - Q(S_{t+1}, A_{t+1})\big] + \gamma\rho_{t+2}\big(R_{t+3} \\
&\quad + \gamma\rho_{t+3}\big(G_{t+3:h} - Q(S_{t+3}, A_{t+3})\big) + \gamma\bar{V}(S_{t+3}) - Q(S_{t+2}, A_{t+2})\big)\Big) + \gamma\bar{V}(S_{t+1}) \\
&= R_{t+1} + \gamma\rho_{t+1}\Big(\delta'_{t+1} + \gamma\rho_{t+2}\big(\delta'_{t+2} + \gamma\rho_{t+3}\big(G_{t+3:h} - Q(S_{t+3}, A_{t+3})\big)\big)\Big) \\
&\quad + \gamma\bar{V}(S_{t+1}) \\
&= R_{t+1} + \gamma\bar{V}(S_{t+1}) + \sum_{k=t+1}^{\min(h,T-1)} \big[\gamma^{k-t}\rho_{t+1:k}\delta'_k\big], \quad \text{where } \rho_{t:k} = \prod_{l=t}^{k}\rho_l
\end{aligned}
$$

7.10 Can't be bothered...

7.11 The expectation-based TD error is $\delta_t = R_{t+1} + \gamma\bar{V}(S_{t+1}) - Q(S_t, A_t)$.

$$
\begin{aligned}
G_{t:t+n} &\doteq R_{t+1} + \gamma\sum_{a \neq A_{t+1}}\pi(a|S_{t+1})Q(S_{t+1}, a) + \gamma\pi(A_{t+1}|S_{t+1})G_{t+1:t+n} \\
&= R_{t+1} + \gamma\bar{V}(S_{t+1}) - \gamma\pi(A_{t+1}|S_{t+1})Q(S_{t+1}, A_{t+1}) \\
&\quad - Q(S_t, A_t) + Q(S_t, A_t) + \gamma\pi(A_{t+1}|S_{t+1})G_{t+1:t+n} \\
&= Q(S_t, A_t) + \delta_t + \gamma\pi(A_{t+1}|S_{t+1})\big[G_{t+1:t+n} - Q(S_{t+1}, A_{t+1})\big] \\
&= Q(S_t, A_t) + \delta_t + \gamma\pi(A_{t+1}|S_{t+1})\big[\cancel{Q(S_{t+1}, A_{t+1})} + \delta_{t+1} + \\
&\quad + \pi(A_{t+2}|S_{t+2})[\gamma G_{t+2:t+n} - Q(S_{t+2}, A_{t+2})] - \cancel{Q(S_{t+1}, A_{t+1})}\big] \\
&= Q(S_t, A_t) + \sum_{k=t}^{\min(t+n-1,T-1)}\gamma^{k-t}\delta_k\prod_{i=t+1}^{k}\pi(A_i|S_i)
\end{aligned}
$$