Random Walk

Lidor Erez

July 2024

1 Introduction

In this paper I will guide through my way of estimating the true value function $V^{\pi}(s)$ in the random walk problem given in [1] using temporal difference learning.

2 Background

2.1 Temporal Difference Learning

Temporal Difference (TD) learning is a reinforcement learning algorithm that combines ideas from Monte Carlo methods with dynamic programming (DP). The advantage of TD methods lies in their ability to learn directly from raw experience, eliminating the need for a complete model of the environment's dynamics, which is typically required by DP methods. Additionally, TD methods overcome the drawback of Monte Carlo methods by not having to wait for an episode to terminate before updating the value function estimates.

More formally, TD methods base their updates on existing estimates. The value function update in TD learning can be expressed as:

$$V^{\pi}(s) = E_{\pi}[r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s]$$

rather than computing:

$$V^{\pi}(s) = E_{\pi}[R_t \mid s_t = s]$$

as the later expected value is unknown. This makes TD learning more efficient in practice. To solve the random walk problem, I will use the TD(0) algorithm illustrated in Figure 1

```
Algorithm: TD(0) for estimating $v(\pi)$

Input:
  $\alpha$: step size
  $\gamma$: discount factor
  $\pi_{rand}$: random policy, chooses actions with equal probability
  $num\_episodes$: total number of episodes
Output:
  $V$: converged state-value function for the random policy $\pi_{rand}$

Initialize $V(s)$ for all $s \in \mathcal{S}$: $V(\text{nonterminal}) \leftarrow 0.5$, $V(\text{terminal}) \leftarrow 0$

for $i \leftarrow 1$ to $num\_episodes$ do
  Initialize $S$
  while episode not terminated do
    $A \leftarrow \pi_{rand}(S)$
    Take action $A$ and observe $(S', R)$
    $V(S) \leftarrow V(S) + \alpha[R + \gamma V(S') - V(S)]$        ▷ update $V$ values
    $S \leftarrow S'$                                              ▷ replace $S$ with next state $S'$
return $V$
```

Figure 1: TD(0) algorithm for estimating $V^{\pi}(s)$. Taken from terrence-ou

3 Random Walk Problem

In the random walk problem, each episode starts in a designated state, which can be chosen by the user (C in this case). The agent in this problem can move either left or right from the current state with equal probability, depending on its current position. The objective is to estimate the value function for each state, given the reward structure. Figure 2 illustrates the random walk graph, where TL and TR are terminal states with specified rewards for each state transition.

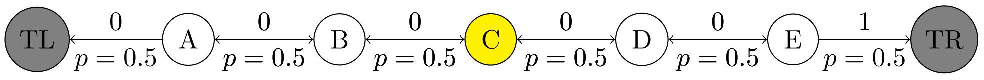

Figure 2: Random Walk Graph with C as the Start Node, TL and TR as the terminal nodes, showing the reward for each state transition and the probability of transitioning.

3.1 Solution

To solve this problem, I created two classes in Python: one for the nodes in the graph and another for the graph itself. This structure facilitated easier iteration through the graph. Additionally, I defined the following parameters, policy, and action space:

- $\gamma = 1$
- $\alpha = 0.1$
- $A = \{\text{Left}, \text{Right}\}$
- $\pi(s) = \begin{cases} 0.5 & \text{if action is Left} \\ 0.5 & \text{if action is Right} \end{cases}$

The problem was solved for different numbers of episodes (0, 1, 10, 100, 500) across different seeds (13, 14, 15, 16, 17). As the policy is stochastic, I estimated the value function for each episode across the different seeds and averaged the results for each state. Finally, the results were compared to the analytical solution of V^π which is:

- $V^\pi(A) = \frac{1}{6}$
- $V^\pi(B) = \frac{2}{6}$
- $V^\pi(C) = \frac{3}{6}$
- $V^\pi(D) = \frac{4}{6}$
- $V^\pi(E) = \frac{5}{6}$

4 Results

Figure 3 shows the estimated values of V^π across different numbers of episodes. The estimates were computed for 0, 1, 10, 100, and 500 episodes, averaged over different random seeds (13, 14, 15, 16, 17). The TD(0) algorithm managed to get closer to the real V^π when the number of episodes was 500.

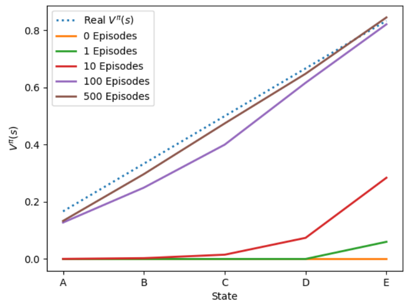

Figure 3: Estimated V^π across different episodes.

5 Summary and Conclusions

In this project, I applied Temporal Difference (TD) learning to solve the random walk problem. By creating Python classes for nodes and the graph, I streamlined the iteration process. Using a stochastic policy, I estimated the value function for various episodes and seeds. The results show that as the number of episodes increases, the TD(0) estimates get closer to the true value function, which demonstrates the power of TD learning to efficiently learn from raw experience without needing a complete model of the environment.

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, 2018.