EE675A

Lecture 14: MC vs TD and TD-λ

13/03/2023

Lecturer: Subrahmanya Swamy Peruru *Scribe: Aadeesh Jain and Riya Bhalla*

1 Recap and Overview

In the previous lecture, we discussed two kinds of model-free algorithms that are required to solve a MDP when we do not assume complete knowledge of the environment. One of the algorithms is Monte Carlo method which requires sample sequences of states, actions, and rewards from the environment, and the state value is estimated as the average of returns calculated after each episode. For every time step the return used to update $V_\pi(s)$ for any state s is calculated at the end of an episode, that is, when the agent reaches the terminal state following a . The first-visit MC method estimates $V_\pi(s)$ as the average of the returns following first visits to s, whereas the every-visit MC method averages the returns following all visits to s.
Another algorithm is the one-step temporal difference(TD) learning method which is a combination of Monte Carlo and Dynamic programming. One-step TD updates $V_\pi(s)$ using the immediate reward R_{t+1} and the previous value at some state S_{t+1}, thus they bootstrap.

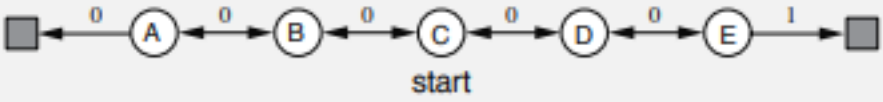
2 MC vs TD

MC	TD
Its useful only in **Episodic** setting	It can be used in **Continuous MDP too**
Offline based learning Update happens after an episode ends	**Online based learning** Update happens after every time stamp
Unbiased Estimates As it converges nearly to the true value	**Biased Estimates** As here they might not converge to the true value
Higher Variance As variance for all rewards in each state adds up	**Lower variance** In each update step actions are limited
No Boot Straping	**Boot Straping**

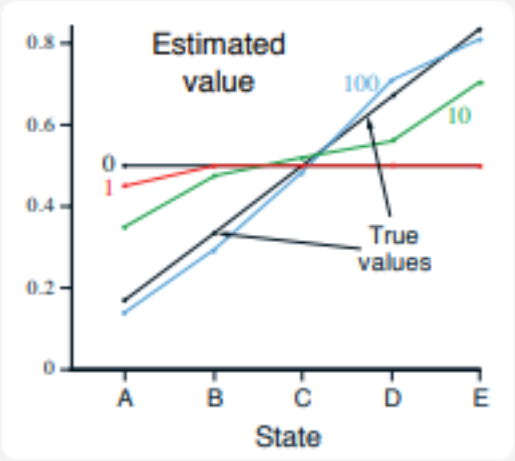
3 Comparision of MC vs TD

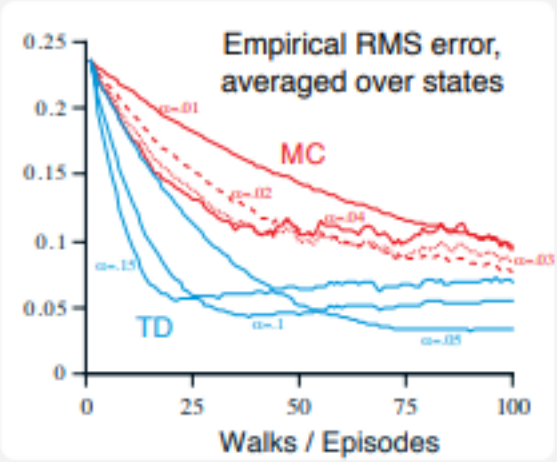
Example 6.2 Random Walk

In this example we empirically compare the prediction abilities of TD(0) and constant-α MC when applied to the following Markov reward process:



A *Markov reward process*, or MRP, is a Markov decision process without actions. We will often use MRPs when focusing on the prediction problem, in which there is no need to distinguish the dynamics due to the environment from those due to the agent. In this MRP, all episodes start in the center state, C, then proceed either left or right by one state on each step, with equal probability. Episodes terminate either on the extreme left or the extreme right. When an episode terminates on the right, a reward of +1 occurs; all other rewards are zero. For example, a typical episode might consist of the following state-and-reward sequence: C, 0, B, 0, C, 0, D, 0, E, 1. Because this task is undiscounted, the true value of each state is the probability of terminating on the right if starting from that state. Thus, the true value of the center state is $v_\pi(\mathsf{C}) = 0.5$. The true values of all the states, A through E, are $\frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6}$, and $\frac{5}{6}$.





The left graph above shows the values learned after various numbers of episodes on a single run of TD(0). The estimates after 100 episodes are about as close as they ever come to the true values—with a constant step-size parameter ($\alpha = 0.1$ in this example), the values fluctuate indefinitely in response to the outcomes of the most recent episodes. The right graph shows learning curves for the two methods for various values of α. The performance measure shown is the root mean square (RMS) error between the value function learned and the true value function, averaged over the five states, then averaged over 100 runs. In all cases the approximate value function was initialized to the intermediate value $V(s) = 0.5$, for all s. The TD method was consistently better than the MC method on this task.

4 Batch Update

In batch update we update our estimates after interacting with our environment for k episodes.

$S_t^{(1)}, A_t^{(1)}, R_{t+1}^{(1)}, S_{t+1}^{(1)}, A_{t+1}^{(1)}, ...S_T^{(1)}$
$S_t^{(2)}, A_t^{(2)}, R_{t+1}^{(2)}, S_{t+1}^{(2)}, A_{t+1}^{(2)}, ...S_T^{(2)}$
$S_t^{(3)}, A_t^{(3)}, R_{t+1}^{(3)}, S_{t+1}^{(3)}, A_{t+1}^{(3)}, ...S_T^{(3)}$
. . . . .
. . . . .
$S_t^{(K)}, A_t^{(K)}, R_{t+1}^{(K)}, S_{t+1}^{(K)}, A_{t+1}^{(K)}, ...S_T^{(K)}$

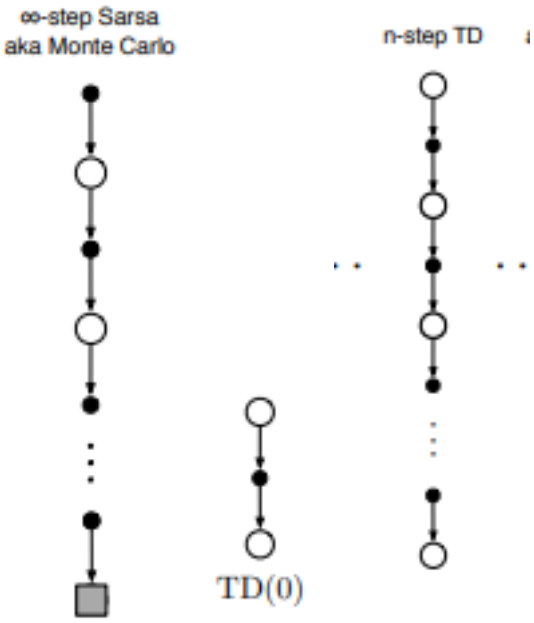
5 Example 2

Suppose you observe following 8 episodes. What will be the optimal predictions for V(A) and V(B) ?
A, 0, B, 0
B, 1
B, 1
B, 1
B, 1
B, 1
B, 1
B, 0

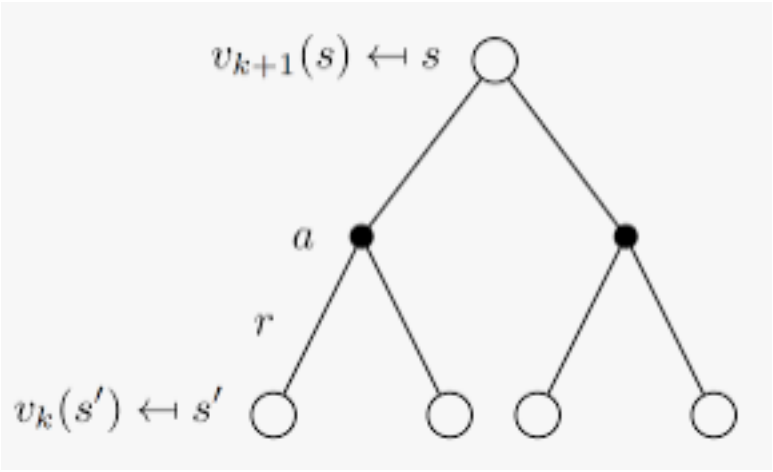
One can easily say that by observing all 8 episode that the value of V(B) is 6/8, as six out of the eight times in state B the process terminated immediately with a return of 1, and the other two times in B the process terminated immediately with a return of 0.
But what is the optimal value for the estimate V (A) given this data? Here there are two reasonable answers. One is to observe that 100 percent of the times the process was in state A it traversed immediately to B (with a reward of 0), and because we have already decided that B has value 3/4 , therefore A must have value 3/4 as well. One way of viewing this answer is that it is based on first modeling the Markov process, in this case as shown to the right, and then computing the correct estimates given the model, which indeed in this case gives V (A) = 3/4 . This is also the answer that batch TD(0) gives.
The other reasonable answer is simply to observe that we have seen A once and the return that followed it was 0; we therefore estimate V (A) as 0. This is the answer that batch Monte Carlo methods give.
This illustrates a general difference between the estimates found by batch TD(0) and batch Monte Carlo methods. Batch Monte Carlo methods always find the estimates that minimize mean square error on the training set, whereas batch TD(0) always finds the estimates that would be exactly

correct for the maximum-likelihood model of the Markov process. In general, the maximum-likelihood estimate of a parameter is the parameter value whose probability of generating the data is greatest. In this case, the maximum-likelihood estimate is the model of the Markov process formed in the obvious way from the observed episodes: the estimated transition probability from i to j is the fraction of observed transitions from i that went to j, and the associated expected reward is the average of the rewards observed on those transitions. Given this model, we can compute the estimate of the value function that would be exactly correct if the model were exactly correct. This is called the certainty-equivalence estimate because it is equivalent to assuming that the estimate of the underlying process was known with certainty rather than being approximated. In general, batch TD(0) converges to the certainty-equivalence estimate. This helps explain why TD methods converge more quickly than Monte Carlo methods. In batch form, TD(0) is faster than Monte Carlo methods because it computes the true certainty-equivalence estimate.

6 Backup Diagrams for v_π



DP



7 n-Step TD

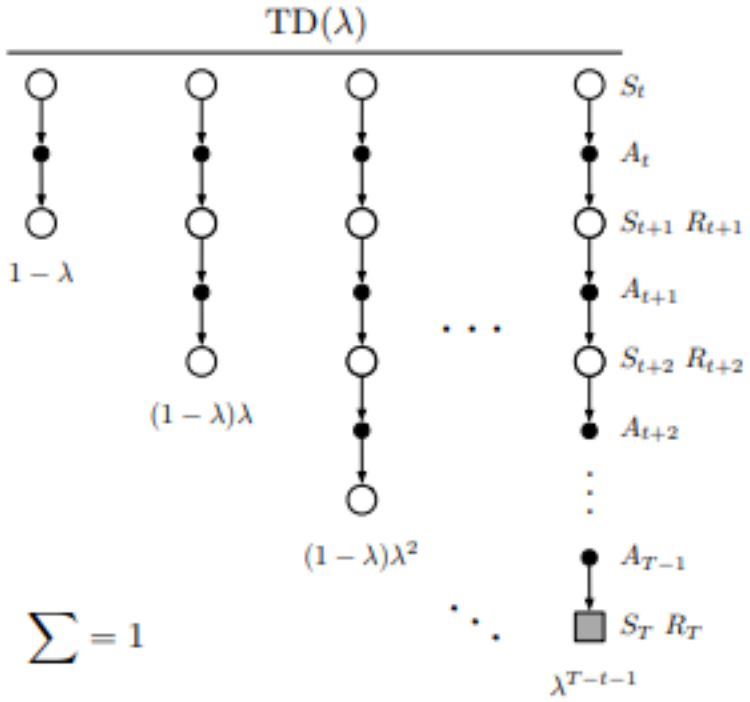
Monte Carlo methods perform an update for each state based on the entire sequence of observed rewards from that state until the end of the episode. The update of one-step TD methods, on the other hand, is based on just the one next reward, bootstrapping from the value of the state one step later as a proxy for the remaining rewards. n-TD perform an update based on an intermediate number of rewards: more than one, but less than all of them until termination.

$V_{new}(S_t) = V_{old}(S_t + \alpha[G_t^{(n)} - V_{old}(S_t)])$
$G_t^n = R_{t+1} + \gamma R_{t+2} .... + \gamma^{n-1} R_{t+n} + \gamma^n V_{old}(S_{t+n})$

Hybrid n-step : $G_t^{(1,2)} = 1/2 G_t^{(1)} + 1/2 G_t^{(2)}$

8 TD(λ)

TD(λ) improves over the offline -return algorithm in three ways. First it updates the weight vector on every step of an episode rather than only at the end, and thus its estimates may be better sooner. Second, its computations are equally distributed in time rather than all at the end of the episode. And third, it can be applied to continuing problems rather than just to episodic problems.



$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} G_t^{(n)} \lambda^{n-1}$, where
$G_t^{(n)} = G_t \forall n >= T - t$

Note
1. To use this method in its current form you have to wait till the end of episode and This version is called Forward View of TD(λ).
The material is primarily based on the content from [1]

References

[1] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2020.