Model Free Prediction : Multi-Step TD Methods

Easwar Subramanian

TCS Innovation Labs, Hyderabad

Email : easwar.subramanian@tcs.com / cs5500.2020@iith.ac.in

September 6, 2022

Overview

Review

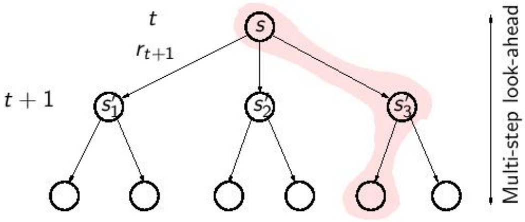
Monte Carlo Methods : Key Idea

$$\begin{aligned} V^\pi(s) &\overset{\text{def}}{=} \mathbb{E}_\pi(G_t|s_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}|s_t = s\right) \\ &= \mathbb{E}_\pi\left[r_{t+1} + \gamma V^\pi(s_{t+1})|s_t = s\right] \end{aligned}$$

How can we estimate the expectations?
Use samples!

- Monte Carlo methods estimate $V^\pi(s)$ by accumulating rewards along different trajectories of π starting from state s
- By the law of large numbers $V(s) \to V^\pi(s)$ as number of episodes increases

Monte Carlo Algorithms: A Schematic View



- Uses experience, rather than model
- Uses only experience; does not bootstrap
- Needs complete sequences; suitable only for episodic tasks
- Suited for off-line learning
- Time required for one estimate does not depend on total number of states
- Can be used in non-Markovian setting as well

Temporal Difference : Key Idea

$$V^\pi(s) \overset{\text{def}}{=} \mathbb{E}_\pi(G_t | s_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right)$$
$$= \mathbb{E}_\pi\left[r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\right]$$

- Estimate expectation from experience using the recursive decomposition formulation of the value function

One-Step TD

- We wish to approximate

$$V^\pi(s) = \mathbb{E}_\pi\left[r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\right]$$

- Approximate the expectation by a sample mean
  - ★ If the *transition* (s_t, r_{t+1}, s_{t+1}) is observed at time t under π, then

$$V(s_t) \leftarrow V(s_t) \ + \ \alpha_t[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

  - ★ Samples come from different visits to the state s, either from same or different trajectories
  - ★ Compute the sample mean incrementally

One-Step TD : TD(0) Algorithm

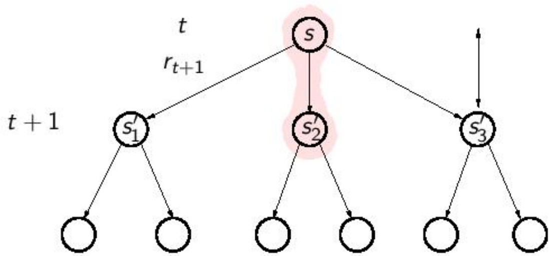
Algorithm TD(0) : Algorithm

```
1: Initialize $V(s)$ arbitrarily (say, $V(s) = 0 \quad \forall s \in \mathcal{S}$);
2: for $k = 1, 2, \cdots, K$ do
3:     Let $s$ be a start state for episode $k$
4:     for For each step in the $k$-th episode do
5:         Take action $a$ recommended by policy $\pi$ from state $s$
6:         Collect reward $r$ and reach next state $s'$
7:         Perform the following TD update
               $V(s) = V(s) + \alpha_{N(s)}[r + \gamma V(s') - V(s)]$
8:         Assign $s \leftarrow s'$
9:     end for
10: end for
```

Convergence of TD Algorithms

- For any fixed policy π, the TD(0) algorithm described above converges (asymptotically) to V^{π} under some conditions on the choice of α (Robbins Monroe Condition)
  - ★ $\sum \alpha_t = \infty$
  - ★ $\sum \alpha_t^2 < \infty$
- *Generally*, TD methods have usually been found to converge faster than MC methods on certain class of tasks

TD Algorithms: A Schematic View



- Uses experience without model like MC
- Bootstraps like DP
- Can work with partial sequences
- Suited for online learning

Connections between MC Error and TD Error

- The term $\delta_t = [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$ is called the (one step) **TD error**
- The term $G_t - V(s_t)$ is called the **MC error**
- If a trajectory has T time steps then

$$G_t - V(s_t) = \sum_{k=0}^{T-t-1} \gamma^k \delta_{t+k}$$

(Exercise : Prove it !)

Bias and Variance in Prediction Methods

Bias in MC Algorithms

- Bias of an estimator $\hat{\theta}$ (with true value θ) is given by $\mathbb{E}[\hat{\theta}] - \theta$
- Since

$$V^\pi(s) \overset{\text{def}}{=} \mathbb{E}_\pi(G_t | s_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right),$$

  the return G_t is an unbiased estimator of $V^\pi(s)$
- Since each total discounted return is sampled from independent trajectory, they are i.i.d., and hence the MC estimate of $V^\pi(s)$ has no bias

Variance in MC Algorithms

- Consider a sequence of estimates $V_k(s)$ for a particular state s computed as

$$V_k(s) = \frac{1}{k}\sum_{i=1}^{k} G_i$$

- We then have, $\text{Var}(V_k(s)) \propto \frac{1}{k}$ since,

$$\begin{aligned}\text{Var}(V_k(s)) &= \text{Var}\left(\frac{1}{k}\sum_{i=1}^{k} G_i\right) = \frac{1}{k^2}\text{Var}\left(\sum_{i=1}^{k} G_i\right) = \frac{1}{k^2}k\text{Var}\left(G_i\right) \\ &= \frac{1}{k}\text{Var}\left(G_i\right) \qquad (1)\end{aligned}$$

- Hence MC methods tend to have high variance as returns are a function of multi-step sequence of random actions, states and rewards

Monte Carlo Methods : Bias and Variance

- Both first visit MC and every visit MC converge to V^π as number of trajectories go to infinity
- In first visit MC this is easy to see as each return sample is independent of the another
- By the law of large numbers the sequence of averages of these estimates converges to their expected value
- Each average is itself an unbiased estimate, and the standard deviation of its error falls as $\sqrt{1/k}$ where k is the number of returns averaged
- The convergence of every visit MC is less straight forward to see but it also converges at a quadratic rate to V^π

In both MC methods, it is possible that we may leave out computing $V^\pi(s)$ for some $s \in \mathcal{S}$ because the state s was never visited by any of the trajectories

Bias in TD Algorithms

- In TD methods, value estimates are computed using,

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\right]$$

  with bootstrapping being used

- Bias exists in TD target $r_{t+1} + \gamma V^{\pi}(s_{t+1})$ as $V^{\pi}(s_{t+1})$ is an **estimate** of value function and not the true value function
- There is a lot of bias in the beginning of training where the estimate of V^{π} is far from true V^{π} and the bias reduces with training

Bias/Variance Trade-Off in TD Algorithms

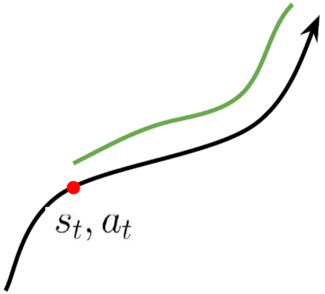
Monte Carlo Algorithms

- No Bias
  - ★ Sample average is an unbiased estimate of the expectation
- High Variance
  - ★ Returns are a function of multi-step sequence of random actions, states and rewards

Temporal Difference Algorithms

- Some Bias
  - ★ TD target $r_{t+1} + \gamma V(s_{t+1})$ is a biased estimate of $V(s)$
- Low Variance
  - ★ TD target only has one random action, reward and next state

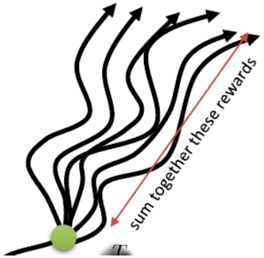
Bias and Variance of MC and TD Estimators : Schematics

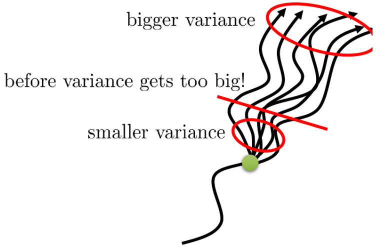


The rewards are counted along the green curve and hence the green curve represents the summation in the defintion of V^π

Figure Source: Jie-Han-Chen:SlideShare

Bias and Variance of MC and TD Estimators : Schematics





- In the MC method we need to wait till the end of the trajectory to make an update; It means lots of future rewards need to be summed. This makes the estimate have high variance
- In the TD, we cut off the path only until next state (in one step TD) and hence the estimate has low variance. Because of bootstrapping, the TD estimate has high bias

Figure Source: UCB : Sergey Levine

Properties of Different Policy Evaluation Algorithms

	DP Algorithms	MC Algorithms	TD Algorithms
Model Free	No	Yes	Yes
Non Episodic Domains	Yes	No	Yes
Non Markovian Domains	No	Yes	No
Bias	Not Applicable	Unbiased	Some Bias
Variance	Not Applicable	High Variance	Low Variance

Multi-Step Temporal Difference

Multi-step TD

- One-step TD

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi\left[r_{t+1} + \gamma V^\pi(s_{t+1}) | s_t = s\right] \\ V(s_t) &\leftarrow V(s_t) + \alpha_t[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \end{aligned}$$

- Two-step TD

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 V^\pi(s_{t+2}) | s_t = s\right] \\ V(s_t) &\leftarrow V(s_t) + \alpha_t[r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2}) - V(s_t)] \end{aligned}$$

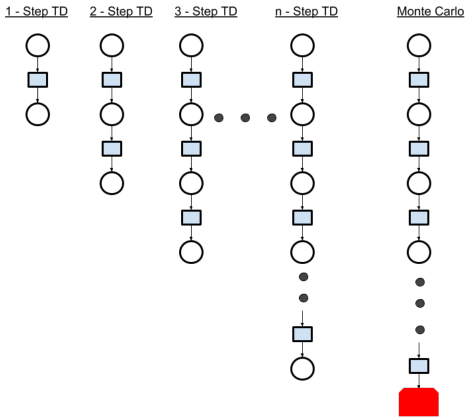
- More generally, define the n-step return

$$G_t^{(n)} \overset{\text{def}}{=} r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n})$$

- n-step TD

$$V(s_t) \leftarrow V(s_t) + \alpha_t[G_t^{(n)} - V(s_t)]$$

Why Multi-Step TD ?



- Multi-step TD methods tend to have <u>less bias</u> compared to 1-step TD method

λ-Return

- What if we average some or all the n-step returns?

Example : Target could be

$$\frac{1}{2}G_t^{(1)} + \frac{1}{2}G_t^{(4)}$$

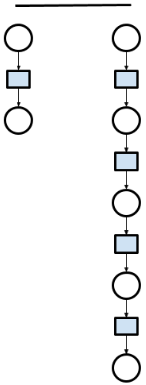


Figure Source: Sutton and Barto

λ-Return

- Any set of returns can be averaged, even an infinite set, as long as the weights on the component returns are positive and sum to 1
- Choose $\lambda \in [0, 1]$, and define the λ-return at time t as

$$G_t^\lambda \overset{\text{def}}{=} (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

- λ-return algorithm: use λ-return as the target

$$V(s_t) \leftarrow V(s_t) + \alpha_t [G_t^\lambda - V(s_t)]$$

- If the episode ends at $T > t$, define $G_t^{(n)}$ to be G_t for all $n > T - t$. Then

$$G_t^\lambda = (1 - \lambda) \sum_{n=1}^{T-t} \lambda^{n-1} G_t^{(n)} + \lambda^{T-t} G_t$$

- $\lambda = 0$ gives 1-step TD, $\lambda = 1$ gives MC update

λ-Return

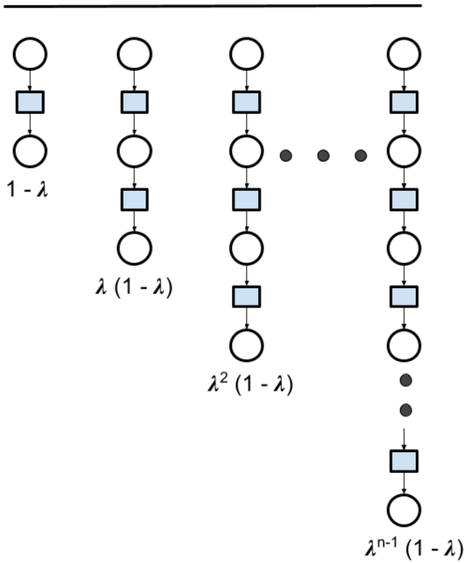
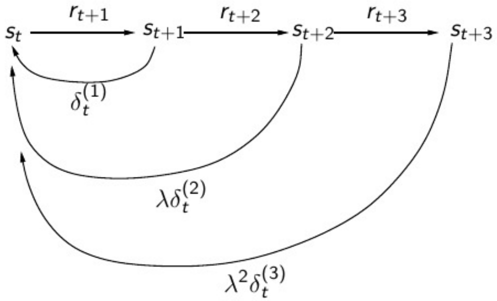


Figure Source: Sutton and Barto

Forward View

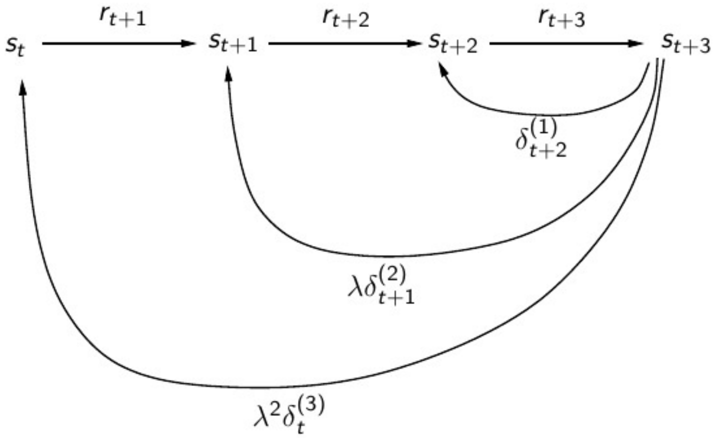
$$G_t^{\lambda} - V(s_t) = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}[G_t^{(n)} - V(s_t)] = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}\delta_t^{(n)}$$



- Not suitable for online implementation;

Figure Source: Sutton and Barto

A Possible Online Implementation



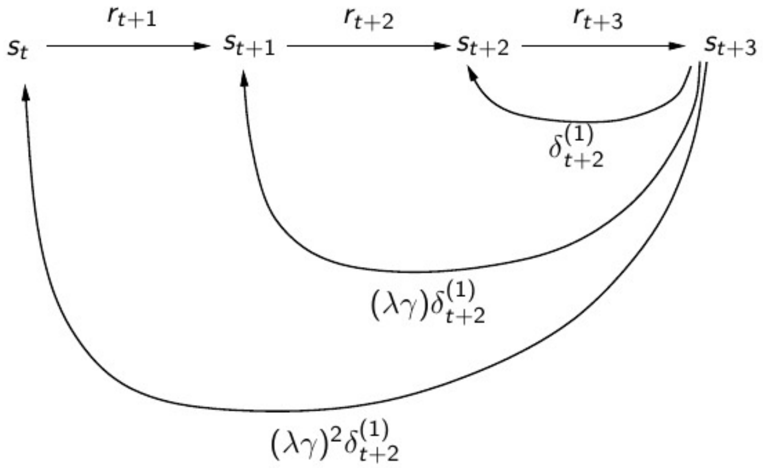
- Requires storing all rewards from the episode

A Rearrangement of n-step TD Errors

$$\begin{aligned}
\delta_t^{(n)} &\overset{\text{def}}{=} r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n}) - V(s_t) \\
&= \quad \gamma^0 [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \\
&\quad + \gamma^1 [r_{t+2} + \gamma V(s_{t+2}) - V(s_{t+1})] \\
&\vdots \\
&\quad + \gamma^{n-1} [r_{t+n} + \gamma V(s_{t+n}) - V(s_{t+n-1})] \\
&= \sum_{i=t}^{t+n-1} \gamma^{i-t} \delta_i^{(1)}
\end{aligned}$$

$$\begin{aligned}
G_t^\lambda - V(s_t) &= (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \delta_t^{(n)} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \sum_{i=t}^{t+n-1} \gamma^{i-t} \delta_i^{(1)} \\
&= \sum_{i=0}^{\infty} (\lambda\gamma)^i \delta_{t+i}^{(1)}
\end{aligned}$$

Online Implementation



ll previously encountered states

Figure Source: Sutton and Barto

Eligibility Traces

- The eligibility trace of a state $s \in \mathcal{S}$ at time t is defined recursively by

$$\begin{aligned} e_0(s) &= 0 \\ e_t(s) &= \begin{cases} (\lambda\gamma)e_{t-1}(s), & s_t \neq s \\ (\lambda\gamma)e_{t-1}(s) + 1, & s_t = s \end{cases} \end{aligned}$$

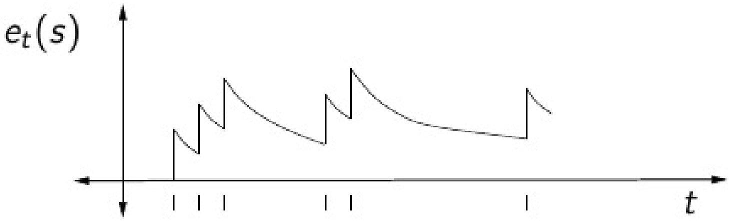


Figure Source: Sutton and Barto

Algorithm : TD(λ)

Algorithm TD(λ) : Algorithm

```
1: Initialize $e(s) = 0$ for all $s$, $V(s)$ arbitrarily
2: for For each episode do
3:     Let $s$ be a start state for episode $k$
4:     for For each step of the episode do
5:         Take action $a$ recommended by policy $\pi$ from state $s$
6:         Collect reward $r$ and reach next state $s'$
7:         Form the one-step TD error $\delta \leftarrow r + \gamma V(s') - V(s)$
8:         Increment eligibility trace of state $s$, $e(s) \leftarrow e(s) + 1$
9:         for For all states $S \in \mathcal{S}$ do
10:            Update $V(S)$: $V(S) \leftarrow V(S) + \alpha e(S)\delta$
11:            Update eligibility trace: $e(S) \leftarrow \lambda\gamma e(S)$
12:        end for
13:        Move to next state: $s \leftarrow s'$
14:    end for
15: end for
```