Model Free Prediction

Easwar Subramanian

TCS Innovation Labs, Hyderabad

Email : cs5500.2020@iith.ac.in

September 02, 2023

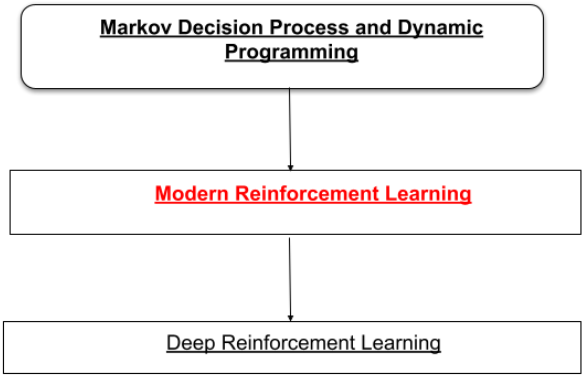
Overview

Review

MDP and RL setting

- **MDP Setting** : The agent has knowledge of the state transition matrices $\mathcal{P}^a_{ss'}$ and the reward function $\mathcal{R}$
- **RL Setting** : The agent does not have knowledge of the state transition matrices $\mathcal{P}^a_{ss'}$ and the reward function $\mathcal{R}$

Course Setup



DP Algorithms : A Closer Look

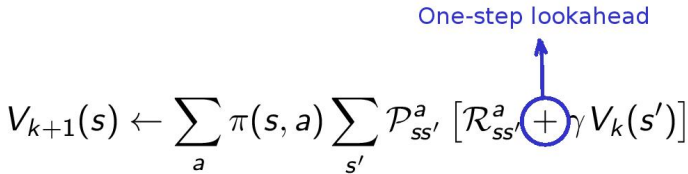
DP Algorithms : Terminology

Let us consider the policy evaluation formula

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(s,a) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_k(s') \right]$$

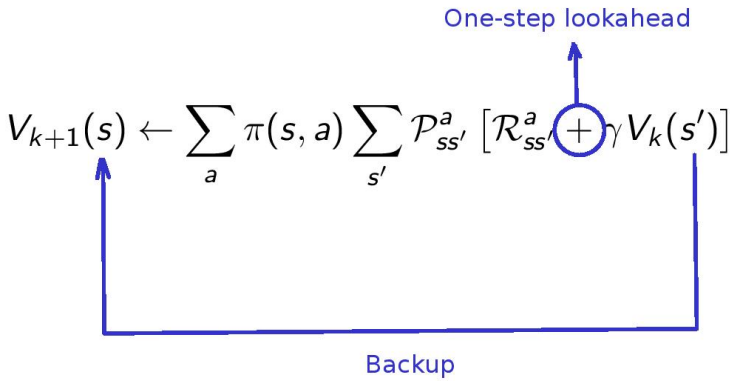
DP Algorithms : Terminology

Let us consider the policy evaluation formula



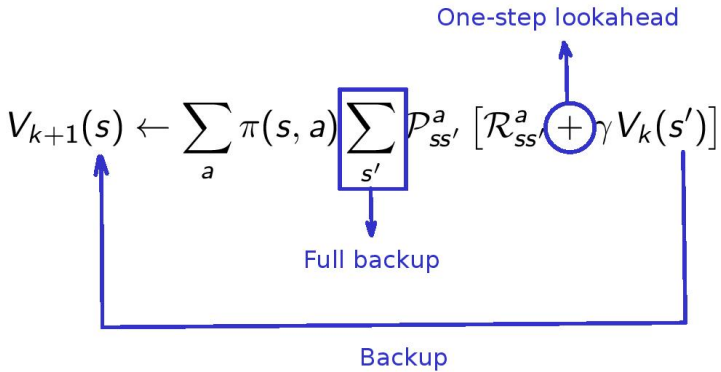
DP Algorithms : Terminology

Let us consider the policy evaluation formula



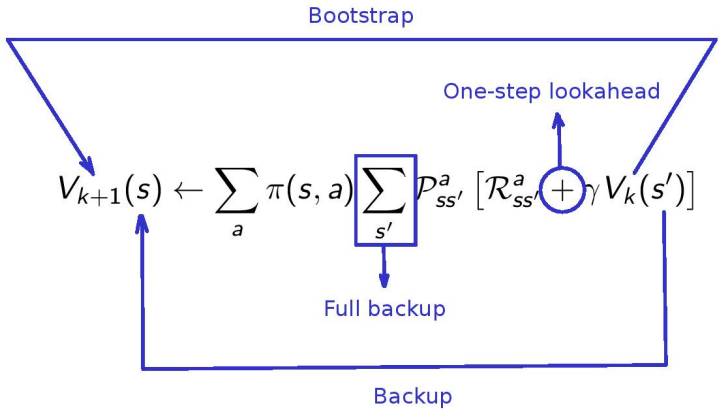
DP Algorithms : Terminology

Let us consider the policy evaluation formula

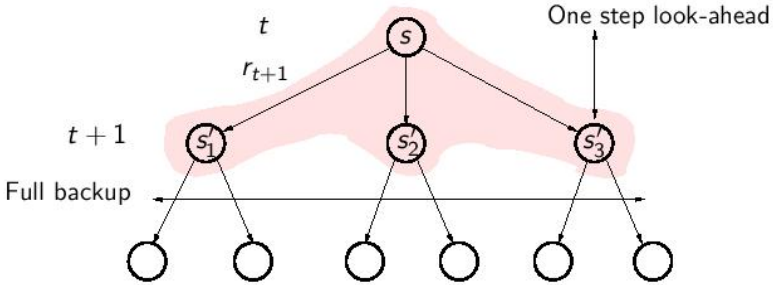


DP Algorithms : Terminology

Let us consider the policy evaluation formula



DP Algorithms : Schematic View



$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

$$V_{k+1}(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s'} \mathcal{P}^{a}_{ss'} \left[\mathcal{R}^{a}_{ss'} + \gamma V_{k}(s') \right]$$

Drawbacks of DP Algorithms

- Requires full prior knowledge of the dynamics of the environment
- Can be implemented only on small or medium sized discrete state spaces
  - ★ For large problems, DP suffers from Bellman's curse of dimensionality
- DP uses full width back-ups
  - ★ Every successor state and action is considered

Model Free Prediction: Monte Carlo Methods

Model Free Prediction : Key Idea

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

How can we estimate the expectations?
Use samples!

Monte Carlo Policy Evaluation

- Goal : Evaluate $V^{\pi}(s)$ using experiences (or trajectories) under policy π

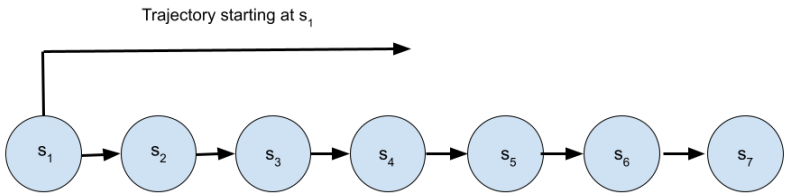
$$s_0, a_0, r_1, s_1, a_1, r_2, s_3, \cdots, s_T$$

- Recall that

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

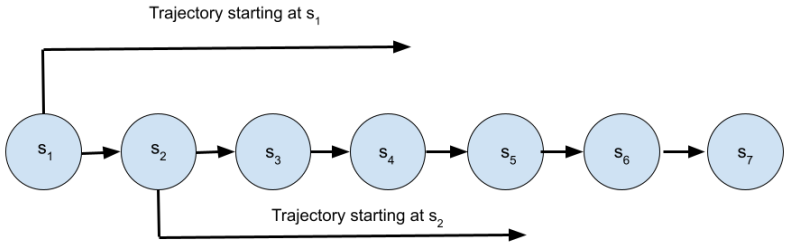
- The idea is to calculate **sample** mean return (G_t) starting from state s instead of expected mean return

Monte Carlo Evalution : Schematics



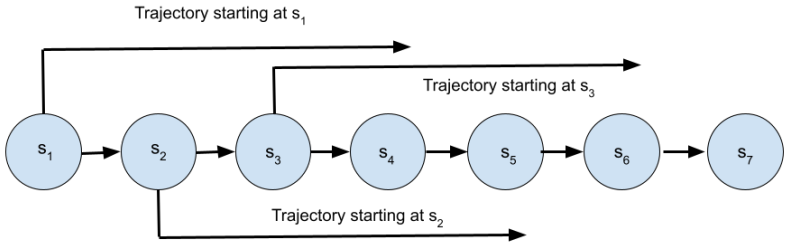
- Use G_1 to update $V^{\pi}(s_1)$

Monte Carlo Evalution : Schematics



- Use G_1 to update $V^\pi(s_1)$
- Use G_2 to update $V^\pi(s_2)$

Monte Carlo Evalution : Schematics



- Use G_1 to update $V^\pi(s_1)$
- Use G_2 to update $V^\pi(s_2)$
- Use G_3 to update $V^\pi(s_3)$

First-visit Monte Carlo Policy Evaluation

- To evaluate $V^\pi(s)$ for some given state s, repeat over several episodes
  - ★ The first time t that $s_t = s$ in the episode
    1. Increment counter for number of visits to s: $N(s) \leftarrow N(s) + 1$
    2. Increment running sum of total returns with return from current episode: $S(s) \leftarrow S(s) + G_t$
- Monte Carlo estimate of value function $V(s) \leftarrow S(s)/N(s)$

By the law of large numbers $V(s) \rightarrow V^\pi(s)$ as number of episodes increases

Every-visit Monte Carlo Policy Evaluation

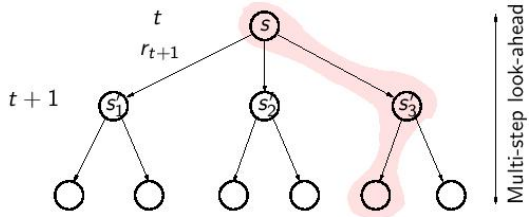
- To evaluate $V^\pi(s)$ for some given state s, repeat over several episodes
  - ★ Every time t that $s_t = s$ in the episode
    1. Increment counter for number of visits to s: $N(s) \leftarrow N(s) + 1$
    2. Increment running sum of total returns with return from current episode: $S(s) \leftarrow S(s) + G_t$
- Monte Carlo estimate of value function $V(s) \leftarrow S(s)/N(s)$

By the law of large numbers $V(s) \rightarrow V^\pi(s)$ as number of episodes increases

Monte Carlo Method : Example

- Consider an MDP with two states $\mathcal{S} = \{A, B\}$ with $\gamma = 1$
- $\mathcal{P}$ and $\mathcal{R}$ are unknown
- Consider a policy π that gives rise to following state-reward sequence
  - $A(+3), A(+2), B(-4), A(+4), B(-3)$
  - $B(-2), A(+3), B(-3)$
- What is $V^\pi(A)$ and $V^\pi(B)$ if we use first visit MC and every visit MC respectively ?
- First visit MC : $V(A) = \frac{1}{2}(2+0) = 1$; $V(B) = \frac{1}{2}(-3-2) = -5/2$
- Every visit MC : $V(A) = \frac{1}{4}(2-1+1+0) = 1/2$; $V(B) = \frac{1}{4}(-3-3-3-2) = -11/4$

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
- Estimates for each state are independent

Model Free Prediction : Temporal Difference

Temporal Difference : Key Idea

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

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

Incremental Calculation of Mean

$$
\begin{aligned}
\mu_{k+1} &\stackrel{\text{def}}{=} \frac{1}{k+1}\sum_{i=1}^{k+1} x_i \\
&= \frac{1}{k+1}\sum_{i=1}^{k} x_i + \frac{1}{k+1}x_{k+1} \\
&= \frac{k}{k+1}\left(\frac{1}{k}\sum_{i=1}^{k} x_i\right) + \frac{1}{k+1}x_{k+1} \\
&= \frac{k}{k+1}\mu_k + \frac{1}{k+1}x_{k+1} \\
&= \mu_k + \frac{1}{k+1}(x_{k+1} - \mu_k)
\end{aligned}
$$

Update = learning rate $\times$ (Target $-$ Previous Value)

General Form of Update Rule

The general form for the update rule that is present in the incremental calculation is,

$$\text{New Estimate} \leftarrow \text{Old Estimate} + \text{Learning Rate}(\text{Target - Old Estimate})$$

- The expression (Target - Old Estimate) is an error of the estimate
- The error is reduced by taking steps towards the "Target"
- The target is persumed to indicate a desirable direction to move
- In the incremental calculation of mean, the term x_{k+1} is the target

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

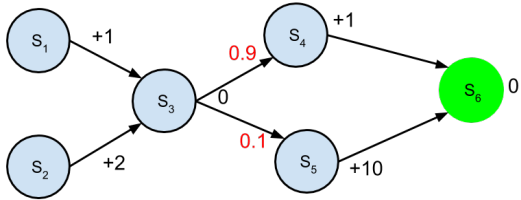
Algorithm TD(0) : Algorithm

Initialize $V(s)$ arbitrarily (say, $V(s) = 0 \quad \forall s \in \mathcal{S}$);
for $k = 1, 2, \cdots, K$ **do**
    Let s be a start state for episode k
    for For each step in the k-th episode **do**
        Take action a recommended by policy π from state s
        Collect reward r and reach next state s'
        Perform the following TD update

$$V(s) = V(s) + \alpha[r + \gamma V(s') - V(s)]$$

        Assign $s \leftarrow s'$
    end for
end for

TD vs MC : Example



(1) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

(2) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6$

(3) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

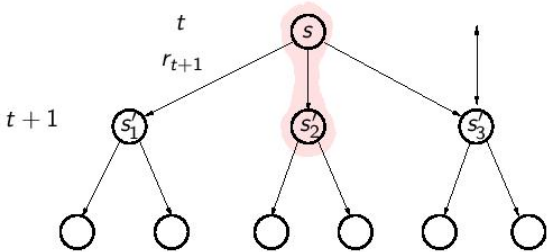
(4) $s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6$

(5) $s_2 \xrightarrow{2} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6$

TD vs MC : Example

- True value of each state is given by
  $V(s_6) = 0, V(s_5) = 10, V(s_4) = 1, V(s_3) = 1.9, V(s_2) = 3.9$ and $V(s_1) = 2.9$
- Evaluate $V(s_1)$ and $V(s_2)$ using MC $V(s_1) = 4.25$ and $V(s_2) = 12$
- Evaluate $V(s_1)$ and $V(s_2)$ using TD
  - First trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1$; $V(s_3) = 0$; $V(s_4) = 1$; $V(s_6) = 0$
  - Second trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6)$
    $V(s_1) = 1$; $V(s_3) = 0$; $V(s_5) = 10$; $V(s_6) = 0$
  - Third trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1$; $V(s_3) = 0.33$; $V(s_4) = 1$; $V(s_6) = 0$
  - Fourth trajectory $(s_1 \xrightarrow{1} s_3 \xrightarrow{0} s_4 \xrightarrow{1} s_6)$
    $V(s_1) = 1.08$; $V(s_3) = 0.5$; $V(s_4) = 1$; $V(s_6) = 0$
  - Fifth trajectory $(s_2 \xrightarrow{2} s_3 \xrightarrow{0} s_5 \xrightarrow{10} s_6)$
    $V(s_2) = 2.5$; $V(s_3) = 2.4$; $V(s_5) = 10$; $V(s_6) = 0$

TD Algorithms: A Schematic View



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

Schematic View of Various Algorithms

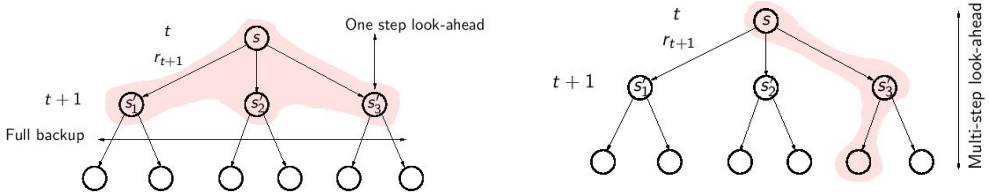
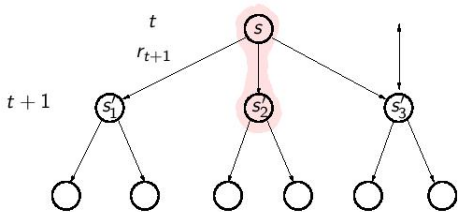


Figure: DP Algorithm and MC Algorithm



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

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

- Consider a statistical mode represented by parameter θ giving rise to distribution of observed data $P(x|\theta)$
- Let $\hat{\theta}$ be an estimator of θ, that depends on observations from $P(x|\theta)$
- Bias of the estimator $\hat{\theta}$ (with true value θ) is given by $\mathbb{E}_{x|\theta}[\hat{\theta}] - \theta$
- Since

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

  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,

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

- 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
- FVMC 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

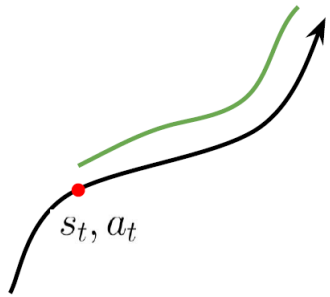
Monte Carlo Algorithms

- 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

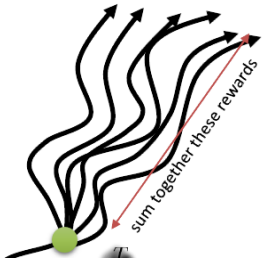
Bias and Variance of MC and TD Estimators : Schematics

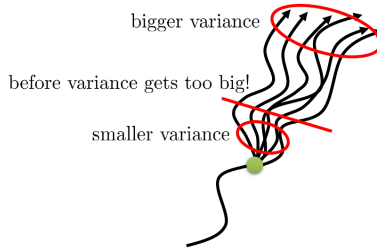


The rewards are counted along the green curve and hence the green curve represents the summation in the defintion of V^{π}

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

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

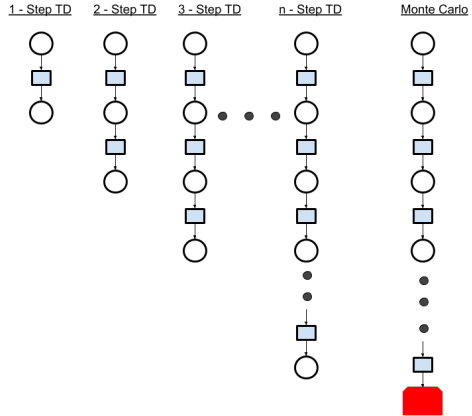
- More generally, define the n-step return

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

- 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 less bias compared to 1-step TD method

λ-Return

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

Example : Target could be

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

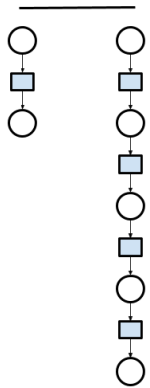


 Figure Source: Sutton and Barto

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

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

- 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

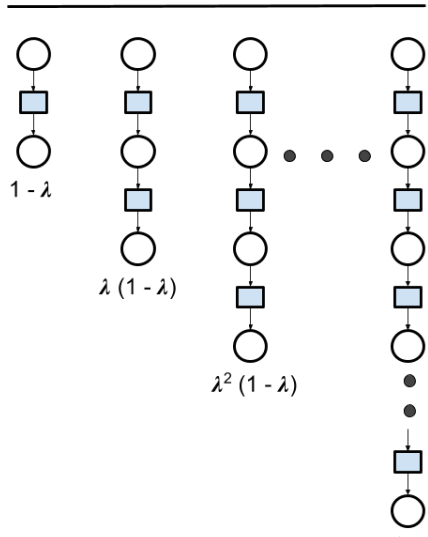
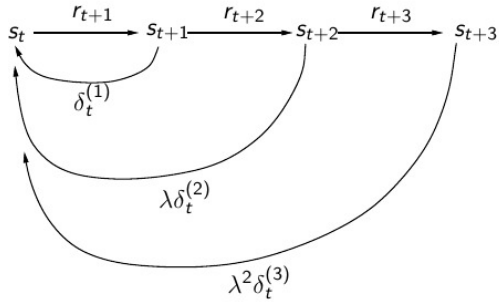


Figure Source: Sutton and Barto

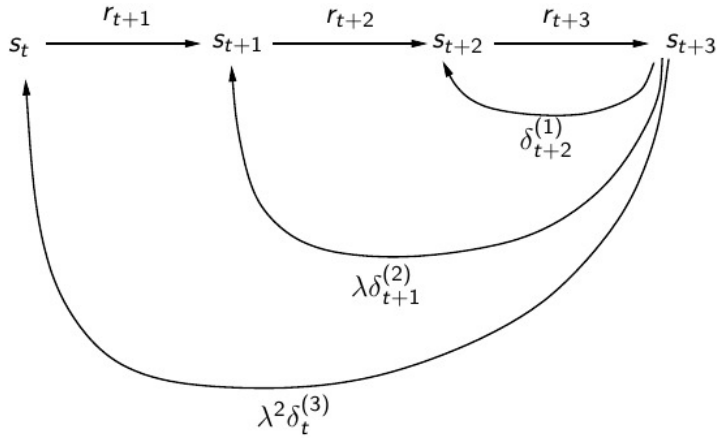
Forward View

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



- Not suitable for online implementation;

A Possible Online Implementation



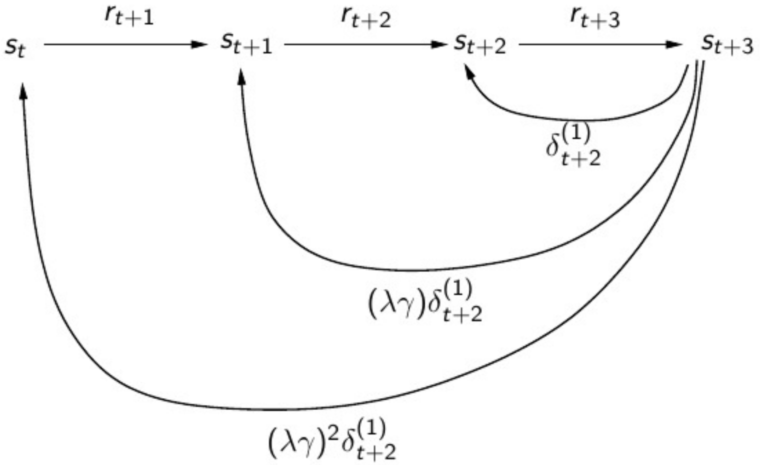
- Requires storing all rewards from the episode

A Rearrangement of n-step TD Errors

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

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

Online Implementation



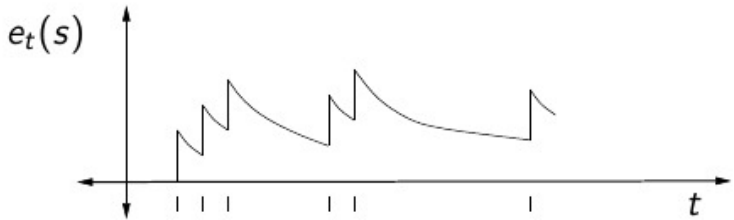
...ll previously encountered states

Figure Source: Sutton and Barto

Eligibility Traces

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

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



Algorithm : TD(λ)

Algorithm TD(λ) : Algorithm

Initialize $e(s) = 0$ for all s, $V(s)$ arbitrarily
for For each episode **do**
    Let s be a start state for episode k
    for For each step of the episode **do**
        Take action a recommended by policy π from state s
        Collect reward r and reach next state s'
        Form the one-step TD error $\delta \leftarrow r + \gamma V(s') - V(s)$
        Increment eligibility trace of state s, $e(s) \leftarrow e(s) + 1$
        for For all states $S \in \mathcal{S}$ **do**
            Update $V(S)$: $V(S) \leftarrow V(S) + \alpha e(S)\delta$
            Update eligibility trace: $e(S) \leftarrow \lambda\gamma e(S)$
        end for
        Move to next state: $s \leftarrow s'$
    end for
end for

TCS Research & Innovation

Few More Concepts on Model Free Prediction

Certainity Equivalence Estimate

- Model based policy evaluation with model estimted from samples
- Given an experience quadruple (s, a, r, s'), we can estimate
  - ★ Compute MLE for model using (s, a, s')

$$\hat{P}(s'|s,a) = \frac{1}{N(s,a)} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')$$

  - ★ Compute MLE estimate of the reward function

$$\hat{R}(s,a,s') = \frac{1}{N(s,a,s')} \sum_{k=1}^{K} \sum_{t=1}^{L_k - 1} \mathbb{1}(s_{k,t} = s, a_{k,t} = a, s_{k,t+1} = s')\ r_{t,k}$$

- Once MLE estimates of $\hat{P}(s'|s,a)$ and $\hat{R}(s,a,s')$ use a known policy evaluation method to compute a MLE based estimate for V^{π}.

Off-Policy Learning

- Can we estimate the value V^π or Q^π of a *target policy* π ...
- While following a *behavior policy* μ?
  - Trajectories or transitions are sampled from μ
  - Expected values have to be estimated w.r.t. π
- Possible benefits:
  - Learn by observing other agents
  - Re-use previous experience generated from earlier policies
  - Learn about optimal policy while following exploratory policy
  - Learn about multiple policies while following one policy

Importance Sampling : Review

Let $P(x)$ be the target distribution and $Q(x)$ be the behaviour distribution for some random variable x

$$\begin{aligned}\mathbb{E}_{X\sim P}[f(X)] &= \sum_i P(x_i)f(x_i) \\ &= \sum_i Q(x_i)\left[\frac{P(x_i)}{Q(x_i)}f(x_i)\right] \\ &= \mathbb{E}_{X\sim Q}\left[\frac{P(X)}{Q(X)}f(X)\right]\end{aligned}$$

- We have samples of X drawn from Q, but we wish to estimate expectation under P

$$\mathbb{E}_{X\sim P}[f(X)] \simeq \frac{1}{n}\sum_{i=1}^{n}\left[\frac{P(X_i)}{Q(X_i)}f(X_i)\right],\ X_i \sim Q$$

- Caveat: Q should not assign zero probability to any outcome that is assigned non-zero probability by P

The ratio $\frac{P(x)}{Q(x)}$ is the importance sampling (IS) weight for x.

What about the variance of IS estimator $\widehat{\mu}_Q \approx \frac{1}{N}\sum_{x_i \in D}\left[\frac{P(x_i)}{Q(x_i)}f(x_i)\right]$?

$$\begin{aligned}
\text{var}_Q\left[\widehat{\mu}_Q\right] &= \text{var}_Q\left[\frac{P(x)}{Q(x)}f(x)\right] \\
&= \left[\mathbb{E}_Q\left[\left(\frac{P(x)}{Q(x)}f(x)\right)^2\right] - \mathbb{E}_Q\left(\left[\frac{P(x)}{Q(x)}f(x)\right]\right)^2\right] \\
&= \mathbb{E}_P\left[\left(\frac{P(x)}{Q(x)}f(x)^2\right)\right] - \mathbb{E}_P\left(f(x)\right)^2
\end{aligned}$$

If the likelihood ratio $\frac{P(x)}{Q(x)}$ is large, the variance of the estimator explodes

Off-Policy TD using Importance Sampling

- Evaluate target policy π using TD targets $r_{t+1} + \gamma V(s_{t+1})$ generated from μ
- Weigh each TD target by the importance sampling factor $\pi(s_t, a_t)/\mu(s_t, a_t)$

$$V(s_t) \leftarrow V(s_t) + \alpha_t \left[\frac{\pi(s_t|a_t)}{\mu(s_t|a_t)} (r_{t+1} + \gamma V(s_{t+1})) - V(s_t) \right]$$

- π may be deterministic and greedy, μ should be stochastic and exploratory
- The case $\mu = \pi$ is called *on-policy* learning
- On Policy Learning : Learn about policy π from experience sampled from π
- Off Policy Learning : Learn about policy π from experience sampled from μ