# Monte Carlo Methods: Reinforcement Learning

Sanyam Singhal

March 23, 2022

## 1 Summary

1. Unlike dynamic programming, where we required the complete knowledge of the MDP dynamics to solve for optimal policy, we can do so as we experience the MDP i.e. through samples states and rewards. A model of the problem is still required but we can progress without knowing the dynamics. The term Monte Carlo is used for estimation methods which involves some form of random component, say random summands, random number of summands and so on. Here, we learn the state and state-action pair values from sample returns with the MDP.

2. Monte Carlo Prediction: solving for $v_\pi$ and $q_\pi$ for a fixed policy $\pi$. Each occurence of a state $s$ in an episode is called the visit to state $s$. First-visit Monte Carlo method estimates $v_\pi$(s) as the average of the returns following the first visit to state $s$. Similarly every visit MC method averages the returns following all visits to state $s$.

   **First-visit MC prediction for estimating $v_\pi$**

   (a) **Input:** A policy $\pi$ to be evaluated

   (b) **Initialize:** $v(s) \in \mathbb{R}$ arbitrarily for all $s \in S$.
   *Returns(s)* $\leftarrow$ an empty list for all $s \in S$.

   (c) **Loop over all episodes:**
   Generate an episode following $\pi$
   $G \leftarrow 0$
   For each step $t = T-1, T-2, \ldots, 0$:
   Loop from $t$ to $T$:
   $G \leftarrow \gamma G + R_{t+1}$
   Unless $S_t$ appears in $S_0, \ldots, S_{t-1}$:
   Append $G$ to *Returns*$(S_t)$
   $v(S_t) \leftarrow$average(Returns$(S_t)$)

---

   For every-visit MC, we drop the 'Unless $S_t$..' check.

   Both the methods converge to $v_\pi(s)$ as the number of visits to the state $s$ goes to infinity.

3. Estimates for each state is independent in Monte Carlo methods, unlike the dynamic programming algorithms. We say that the Monte Carlo methods do not bootstrap (using previously known values to estimate the new values). Moreover, the computational expense to implement Monte Carlo methods is independent of the number of states as we only generate sample averages. This implies that we indirectly focus only on the important states i.e. those that occur in the sampled episode of the underlying task.

4. To estimate the action values $q_\pi(s, a)$, we say that an action pair $(s, a)$ is visited if ever the state $s$ is visited and action $a$ is taken in it. Then, we apply the same Monte Carlo methods (first or every visit) to generate the estimates. Moreover, both the methods converge to $q_\pi(s, a)$ in the limit of infinitely many visits to the pair $(s, a)$.

5. There is one shortcoming in the action value estimation is that the boon of not visiting unimportant states becomes a bane here as not all actions may be taken from a visited state $s$. This means we would not have the entire $q_\pi(s, a)$ for a state $s$, resulting in lack of information to take optimal action. One way to tackle this could be to explicitly take action $a$ from a state $s$ and then follow $\pi$ till the end of episode. This exploration can generate episodes for each state-action pair $(s, a)$, giving us some tangible approximation for values of all actions from an important state (visited state) $s$. This method is called the method of exploring starts.

6. Monte Carlo Control: Just like the dynamic programming algorithms, we maintain an approximate value function and approximate policy. We alter the both and in the limit of infinite sampling, they converge to optimal value and policy respectively. The policy evaluation part is done as specified before, for finding $q_{\pi_i}(s, a)$ and then policy improvement is done by being greedy with respect to current action-value function:

$$\pi_{i+1} = arg\max_a q_{\pi_i}(s, a) \tag{1}$$

Note that they we assumed that we have exploring starts to evaluate $q_{\pi_i}$.

We consider a modified value iteration style method where we alternate between improvement and evaluation steps for single states instead of all states.

**Monte Carlo Control with Exploring Starts for estimating $\pi_*$**

(a) **Initialize:** $\pi(s)$ for all $s \in S$
$q(s, a) \in \mathbb{R}$ arbitrarily for all $s \in S$ and $a \in A(s)$.
*Returns(s)* $\leftarrow$ an empty list for all $s \in S$.

(b) **Loop over all episodes (foreover):**
Choose $s_0 \in S$ and $a_0 \in A(s_0)$ randomly such that all pairs have a non-zero probability.
Generate an episode,starting from $s_0, a_0$ following $\pi$
$G \leftarrow 0$
For each step $t = T-1, T-2, \ldots, 0$:
Loop from $t$ to $T$:
$G \leftarrow \gamma G + R_{t+1}$
Unless $S_t, A_t$ appears in $S_0, A_0 \ldots, S_{t-1}, A_{t-1}$:
Append $G$ to *Returns*$(S_t, A_t)$
$q(S_t, A_t) \leftarrow$average(Returns$(S_t, A_t)$)
$\pi(s_t) = arg\max_a q(s_t, a)$

---

7. To remove the constraint of exploring starts, we need an approach that ensures that all actions are selected infinitely often. There are two approaches: on-policy and off-policy. The on-policy methods evaluate or improve the policy used to make the decisions, unlike the off-policy methods which work on a different policy. Monte Carlo Control with ES was an example of an on-policy algorithm. Here, we implement an $\epsilon$-greedy policy and implement the Monte Carlo control without ES.

We define $\epsilon$-soft policies to be those where $\pi(a|s) \geq \frac{\epsilon}{|A(s)|}$ for all states $s$ and actions $a \in A(s)$.

**On-policy First-visit MC Control with $\epsilon$-soft policies for estimating $\pi_*$**

(a) **Algorithm parameter:** small $\epsilon > 0$.

(b) **Initialize:**
$\epsilon$-soft policy $\pi(s)$ for all $s \in S$
$q(s, a) \in \mathbb{R}$ arbitrarily for all $s \in S$ and $a \in A(s)$.
*Returns(s)* $\leftarrow$ an empty list for all $s \in S$.

(c) **Loop over all episodes (foreover):**
Generate an episode following $\pi$

$G \leftarrow 0$
For each step $t = T-1, T-2, \ldots, 0$:
  Loop from $t$ to $T$:
    $G \leftarrow \gamma G + R_{t+1}$
  Unless $s_t, s_t$ appears in $s_0, a_0 \ldots, s_{t-1}, a_{t-1}$:
    Append $G$ to $Returns(s_t, a_t)$
    $q(s_t, a_t) \leftarrow$average(Returns$(S_t, A_t)$)
    $a^* \leftarrow arg\max_a q(s_t, a)$
    For all $a \in A(s_t)$:
      if $a = a^*$:
        $\pi(a|s_t) \leftarrow 1 - \epsilon + \frac{\epsilon}{|A(s_t)|}$
      else:
        $\pi(a|s_t) \leftarrow \frac{\epsilon}{|A(s_t)|}$

---

8. So far, we have seen learning regimes where the policy that we wish to evaluate or improve generated the episode trace. What if we are learning against an exploratory policy, say an adversary in a game like chess or learning a situation under the guidance of a human expert? Such situations are called off-policy learning. Our policy, that we wish to evaluate or improve is called the target policy, still denoted as $\pi$ whereas the policy that generates the environment behavior is called the behavior policy, denoted as $b$.

9. We need the assumption of coverage to have any meaningful discussion of successful learning: for every action taken under $\pi$, it is also taken atleast occasionally in $b$ i.e. $\pi(a|s) > 0 \implies b(a|s) > 0$.

10. We use a metric called the importance-sampling ratio, which is the ratio of the probability of a trajectory/trace occurring under target and behavior policies: Given a state $s_t$, the probability of the subsequent state-action trajectory being $a_t, s_{t+1}, a_{t+1}, \ldots, s_T$ under the policy $\pi$ is:

$$Pr(a_t, \ldots, s_T) = \Pi_{k=t}^{T} \pi(a_k|s_k) p(s_{k+1}|s_k, a_k) \tag{2}$$

Similarly for policy $b$. The importance sampling ratio is defined as:

$$\rho_{t:T-1} = \Pi_{k=t}^{T-1} \frac{\pi(a_k|s_k)}{b(a_k|s_k)} \tag{3}$$

We want the expected returns under the target policy but the returns that we have access to, converge to the value function determined by the behavior policy: $\mathbb{E}(G_t|s_t = s) = v_b(s)$. The ratio $\rho$ transforms $G_t$ to our value of interest:

$$\mathbb{E}(\rho_{t:T-1} G_t | s_t = s) = v_\pi(s) \tag{4}$$

We index our time steps in an increasing manner such that say an episode ends at $t = 100$ then the next episode time step starts from $t = 101$. We denote the set of all time steps in which state $s$ is visited as $\mathcal{T}(s)$ and let $T(t)$ be the termination time step, where the episode starts at $t$. We define ordinary importance sampling estimate as:

$$v(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|} \tag{5}$$

An alternative is the weighted importance sampling estimate (it is defined to be 0 if its denominator is zero):

$$v(s) = \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1}} \tag{6}$$

11. Ordinary sampling method is generally unbiased but generally has an unbounded variance since importance ratios are in general unbounded whereas weighted sampling method is biased (with bias asymptotically converging to 0) but the maximum weight of an individual return is 1 so the variance is significantly lower than the ordinary counterpart.

12. Off-policy prediction problem: We need to maintain estimate for $\rho_{t:T(t)-1}$ and we do using a variable $W$ such that $W_i = \rho_{t_i:T(t_i)-1}$ in the asymptotic limit. With slight algebraic manipulation, we can turn the weighted importance sampling estimate into an incremental calculation:

$$v_{n+1} = v_n + \frac{W_n}{c_n}[G_n - v_n],\ n \geq 1 \tag{7}$$

Here, $c_{n+1} = c_n + W_{n+1}$ with $c_0 = 0$. So, the off-policy MC prediction for estimating $q_\pi$ is as follows:

**Off-policy MC prediction with weighted importance sampling for estimating $q_\pi$**

(a) **Input:** A target policy $\pi$ to be evaluated

(b) **Initialize:**
$q(s,a) \in \mathbb{R}$ arbitrarily for all $s \in S$ and actions $a \in A(s)$.
$c(s,a) \leftarrow 0$ for all states and actions.

(c) **Loop over all episodes:**
$b \leftarrow$any policy with coverage of $\pi$.
Generate an episode following $b$
$G \leftarrow 0$
$W \leftarrow 1$
For each step $t = T-1, T-2, \ldots, 0$, while $W \neq 0$:
Loop from $t$ to $T$:
$G \leftarrow \gamma G + R_{t+1}$
$c(s_t, a_t) \leftarrow c(s_t, a_t) + W$
$q(s_t, a_t) \leftarrow q(s_t, a_t) + \frac{W}{c(s_t, a_t)}[G - q(s_t, a_t)]$
$W \leftarrow W\frac{\pi(a_t|s_t)}{b(a_t|s_t)}$

---

13. We now see the off-policy MC control with the coverage assumption. We improve the target policy greedily with respect to the estimate $q(s,a)$. As long as we sample the returns of each state-action pair infinitely often, the policy estimate converges to optimal target policy $\pi_*$.

**Off-policy MC Control with weighted importance sampling for estimating $\pi_*$**

(a) **Initialize:**
$q(s,a) \in \mathbb{R}$ arbitrarily for all $s \in S$ and actions $a \in A(s)$.
$c(s,a) \leftarrow 0$ for all states and actions.
$\pi(s) \leftarrow arg\max_a q(s,a)$ with ties broken arbitrarily.

(b) **Loop over all episodes:**
$b \leftarrow$any policy with coverage of $\pi$.
Generate an episode following $b$
$G \leftarrow 0$
$W \leftarrow 1$
For each step $t = T-1, T-2, \ldots, 0$:
Loop from $t$ to $T$:
$G \leftarrow \gamma G + R_{t+1}$
$c(s_t, a_t) \leftarrow c(s_t, a_t) + W$
$q(s_t, a_t) \leftarrow q(s_t, a_t) + \frac{W}{c(s_t, a_t)}[G - q(s_t, a_t)]$

$\pi(s_t) \leftarrow arg\max_a q(s_t, a)$
If $a_t \neq \pi(s_t)$ then exit the loop and proceed to next episode
$W \leftarrow W\dfrac{1}{b(a_t|s_t)}$

---

The numerator in the $W$ update is 1 because we are choosing the greedy action with probability 1 and that is why we added the if exit condition because $\pi(a_t|s_t) = 0$ for sub-optimal action so, that would have made $W = 0$ (recall the $W \neq 0$ condition in the prediction algorithm)

14. Since, we exit if we encounter the sub-optimal action in the episode trajectory, this algorithm often results in slow learning.

15. Returns we saw in our MC methods were non-discounted. We can introduce discounting. The idea is to think of discounting as determining a probability of termination or equivalently a degree of partial termination. We define $\bar{G}_{t:h} = R_{t+1} + \cdots + R_h$ , $h \leq T$ called the flat partial return. With some algebraic manipulation, we can write the discounted return $G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-t-1} R_T$ as:

$$G_t = \left[(1-\gamma)\sum_{h=t+1}^{T-1} \gamma^{h-t-1}\bar{G}_{t:h}\right] + \gamma^{T-t-1}\bar{G}_{t:T} \tag{8}$$

(Valid for $\gamma \in [0, 1)$) If we replace this $G_t$ in the ordinary and weighting importance sampling estimation for $v(s)$ seen above, then we call them discounting-aware importance sampling estimators. Note that for $\gamma = 1$, we can not write $G_t$ as in equation (8).

16. With some algebraic manipulation, it can be seen that:

$$\mathbb{E}(\rho_{t:T-1}R_{t+1}) = \mathbb{E}(\rho_{t:t}R_{t+1}) \tag{9}$$

and

$$\mathbb{E}(\rho_{t:T-1}R_{t+k}) = \mathbb{E}(\rho_{t:t+k-1}R_{t+k}) \tag{10}$$

So, if we define $G'_t = \rho_{t:t}R_{t+1} + \gamma\rho_{t:t+1}R_{t+2} + \gamma^2\rho_{t:t+2}R_{t+3} + \cdots + \gamma^{T-t-1}\rho_{t:T-1}R_T$, then, we get:

$$\mathbb{E}(\rho_{t:T-1}G_t) = \mathbb{E}(G'_t) \tag{11}$$

Using this, we can modify the ordinary importance sampling estimator to the following:

$$v(s) = \frac{\sum_{t\in\mathcal{T}(s)} G'_t}{|\mathcal{T}(s)|} \tag{12}$$

This has the same unbiased expectation as the usual ordinary importance sampling method. This idea is called per-decision importance sampling method.