5 Monte Carlo Methods

Monte Carlo methods learn state and action values by sampling and averaging returns (i.e. not from dynamics like DP). These methods learn from experience (real or simulated) and require no prior knowledge of the environments dynamics.

Monte Carlo methods thus require well defined returns, so we will consider them only for episodic tasks. Only on completion of an episode do values and policies change.

We still use the generalised policy iteration framework, but we adapt it so that we learn the value function from experience rather than compute it *a priori*.

5.1 Monte Carlo Prediction

The idea is to average the returns following each state to get an estimate of the state value

$$v_\pi(s) = \mathbb{E}_\pi[G_{t+1}|S_t = s].$$

Given enough observations, the sample average converges to the true state value under the policy π.

Given a policy π and a set of episodes, here are two ways in which we might estimate state values

- *First Visit MC* average returns from first visit to state s in order to estimate $v_\pi(s)$
- *Every Visit MC* average returns following every visit to state s.

First visit MC generates iid estimates of $v_\pi(s)$ with finite variance, so the sequence of estimates converges to the expected value by the law of large numbers as visits to s tend to ∞. Every visit MC does not generate independent estimates, but still converges.

An algorithm for first visit MS (what we will focus on) is below. Every visit is the same, just without the check for S_k occurring earlier in the episode.

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated
Initialize:
    $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$
    $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
        $G \leftarrow G + R_{t+1}$
        Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:
            Append G to $Returns(S_t)$
            $V(S_t) \leftarrow$ average$(Returns(S_t))$

Monte Carlo methods are often used even when the dynamics of the environment are knowable, e.g. in Blackjack. It is often much easier to create sample games than it is to calculate environment dynamics directly.

MC estimates for different states are independent (unlike bootstrapping in DP). This means that we can use MC to calculate the value function for a subset of the states, rather than the whole state space as with DP. Along with the ability to learn from experience and simulation, this is the another advantage that MC has over DP.

5.2 Monte Carlo Estimation of Action Values

If we don't have a model for the environment, then it is more useful to estimate action-values. With a model we can use state values to find a policy by searching possible actions, as with DP (value iteration, etc.). We can't do this without knowledge of the dynamics, so one of the primary goals of MC is to estimate q_*. We start with policy evaluation for action-values.

Policy Evaluation for Action-Values

The policy evaluation problem for action-values is to estimate $q_\pi(s, a)$ for some π. This is essentially the same as for state values, only we now talk about state-action pairs being visited, i.e. taking action a in state s, rather than just states being visited.

If π is deterministic, then we will only estimate the values of actions that π dictates. We therefore need to incorporate some exploration in order to have useful action-values (since, after all, we want to use them to make informed decisions).

One consideration is to make π stochastic, e.g. ε-soft. Another is the assumption of *exploring starts*, which specifies that ever state-action pair has non-zero probability of being selected as the starting state. Of course, this is not always posible in practice.

For now we assume exploring start. Later we will come back to the issue of *maintaining exploration*

5.3 Monte Carlo Control

We make use of the GPI framework for action-values. Policy evaluation is done as described. Policy improvement is done by making the policy greedy with respect to the action-value function, so no model is needed for this step

$$\pi(s) \doteq \operatorname*{argmax}_a q(s, a).$$

We generate a sequence of policies π_k each greedy with respect to $q_{\pi_{k-1}}(s, a)$. The policy improvement theorem applies: for all $s \in \mathcal{S}$

$$\begin{aligned} q_{\pi_k}(s, a = \pi_{k+1}(s)) &= q_{\pi_k}(s, \operatorname*{argmax}_a q_\pi(s, a)) \\ &= \max_a q_{\pi_k}(s, a) \\ &\geq q_{\pi_k}(s, \pi_k(s)) \\ &= v_{\pi_k}(s) \end{aligned}$$

So π_{k+1} uniformly better than π_k or it is optimal.

The above procedure's convergence depends on assumptions of exploring starts and infinitely many episodes. We will relax the first later, but we will address the second now.

Two approaches to avoid infinitely many episodes:

1. Stop the algorithm once the q_{π_k} stop moving within a certain error. (In practice this is only useful on the smallest problems.)

2. Stop policy evaluation after a certain number of episodes, moving the action value towards q_{π_k}, then go to policy improvement.

For MC policy evaluation, it is natural to alternate policy evaluation and improvement on a episode by episode basis. We give such an algorithm below (with the assumption of exploring starts).

Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:
$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$
$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Loop forever (for each episode):
Choose $S_0 \in \mathcal{S}$ and $A_0 \in \mathcal{A}(S_0)$ such that all pairs have probability > 0
Generate an episode from S_0, A_0, following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G \leftarrow 0$
Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow G + R_{t+1}$
Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:
Append G to $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

It is easy to see that optimal policies are a fixed point of this algorithm. Whether this algorithm converges in general is still, however, an open question.

5.4 Monte Carlo Control without Exploring Starts

On Policy vs. Off Policy

On-policy methods evaluate or improve the policy that is used to make decisions, whereas off-policy methods evaluate or improve one that is different than the one used to generate the data.

On-Policy Techniques without Exploring Starts

We consider ε-greedy policies that put probability $1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}(s)|}$ on the maximal action and $\frac{\varepsilon}{|\mathcal{A}(s)|}$ on each of the others. These are examples of ε-soft policies in which $\pi(a|s) \geq \frac{\varepsilon}{|\mathcal{A}(s)|}$.

We use this idea in the GPI framework:

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

We now show that an ε-greedy policy with respect to q_π, π', is an improvement over any ε-soft policy π. For any $s \in \mathcal{S}$

$$\begin{aligned} q_\pi(s, \pi'(s)) &= \sum_a \pi'(a|s) q_\pi(s,a) && (33) \\ &= \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s,a) + (1-\varepsilon) \max_a q_\pi(s,a) && (34) \\ &\geq \frac{\varepsilon}{|\mathcal{A}(s)|} \sum_a q_\pi(s,a) + (1-\varepsilon) \sum_a \frac{\pi(a|s) - \frac{\varepsilon}{|\mathcal{A}(s)|}}{1-\varepsilon} q_\pi(s,a) && (35) \\ &= \sum_a \pi(a|s) q_\pi(s,a) && (36) \\ &= v_\pi(s) && (37) \end{aligned}$$

(where line 3 follows because a weighted average with weights $w_i \geq 0$ and $\sum_i w_i = 1$ is $\leq$ the max term).

This satisfies the condition of the policy improvement theorem so we now know that $\pi' \geq \pi$.

Previously, with deterministic greedy policies, we would get automatically that fixed points of policy iteration are optimal policies since

$$v_*(s) \doteq \max_\pi v_\pi(s) \quad \forall s \in \mathcal{S}.$$

Now our policies are not deterministically greedy, our value updates do not take this form. We note, however, that we can consider an equivalent problem where we change the environment to select state and reward transitions at random with probability ε and do what our agent asks with probability $1 - \varepsilon$. We have moved the stochasticity of the policy into the environment, creating an equivalent

problem. The optimal value function in the new problem satisfies its Bellman equation

$$\tilde{v}_\pi(s) = (1-\varepsilon)\max_a \tilde{q}_\pi(s,a) + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a \tilde{q}_\pi(s,a) \tag{38}$$

$$= (1-\varepsilon)\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma\tilde{v}_\pi(s')] + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a\sum_{s',r} p(s',r|s,a)[r+\gamma\tilde{v}_\pi(s')]. \tag{39}$$

We also know that at fixed points of our algorithm

$$v_\pi(s) = (1-\varepsilon)\max_a q_\pi(s,a) + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a q_\pi(s,a) \tag{40}$$

$$= (1-\varepsilon)\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v_\pi(s')] + \frac{\varepsilon}{|\mathcal{A}(s)|}\sum_a\sum_{s',r} p(s',r|s,a)[r+\gamma v_\pi(s')]. \tag{41}$$

This is the same equation as above, so by uniqueness of solutions to the Bellman equation we have that $v_\pi = \tilde{v}_\pi$ and so π is optimal.

5.5 Off-Policy Prediction via Importance Sampling

Off-policy learning uses information gained by sampling the *behaviour policy* b to learn the *target policy* π. The behaviour policy explores the environment for us during training and we update the target policy accordingly.

In this section we consider the prediction problem: estimating v_π or q_π for a fixed and known π using returns from b. In order to do this we need the assumption of coverage:

$$\pi(a|s) \geq 0 \implies b(a|s) \geq 0. \tag{42}$$

This implies that b must be stochastic wherever it is not identical to π. The target policy π may itself be deterministic, e.g. greedy with respect to action-value estimates.

Importance Sampling

We use *importance sampling* to evaluate expected returns from π given returns from b.

Define the importance sampling ratio as the relative probability of a certain trajectory from S_t

$$\rho_{t:T-1} = \frac{\mathbb{P}(A_t, S_{t+1}, A_{t+1}, \dots)|S_t, A_{t:T-1}\sim\pi}{\mathbb{P}(A_t, S_{t+1}, A_{t+1}, \dots)|S_t, A_{t:T-1}\sim b} \tag{43}$$

$$= \frac{\prod_{k=t}^{T-1}\pi(A_k|S_k)\mathbb{P}(S_{k+1}|S_k,A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)\mathbb{P}(S_{k+1}|S_k,A_k)} \tag{44}$$

$$= \prod_{k=t}^{T-1}\frac{\pi(A_k|S_k)}{b(A_k|S_k)} \tag{45}$$

where the state transition dynamics $\mathbb{P}$ cancel out.

If we have returns G_t from evaluating policy b, so $v_b(s) = \mathbb{E}[G_t|S_t = s]$, then we can calculate

$$v_\pi(s) = \mathbb{E}[\rho_{t:T-1}G_t|S_t = s]$$

Estimation

Introduce new notation:

- Label all time steps in a single scheme. So maybe episode 1 is $t = 1, \ldots, 100$ and episode 2 is $t = 101, \ldots, 200$, etc.
- Denote the set times of first/every visit to s by $\mathcal{T}(s)$ (spanning episodes).
- Let $T(t)$ be the first termination after t
- Let G_t be the returns from t to $T(t)$

We can now give two methods of values for π from returns from b:
Ordinary Importance Sampling

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T-1} G_t}{|\mathcal{T}(s)|} \tag{46}$$

Weighted Importance Sampling

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T-1}} \tag{47}$$

or 0 if the denominator is 0.

Weighted importance sampling is biased (e.g. it's expectation is $v_b(s)$ after 1 episode) but has bounded variance. The ordinary importance sampling ratio is unbiased, but has possibly infinite variance, because the variance of the importance sampling ratios themselves is unbounded.

Assuming bounded returns, the variance of the weighted importance sampling estimator converges to 0 even if the variance of the importance sampling ratios is infinite. In practice, this estimator usually has dramatically lower variance and is strongly preferred.

5.6 Incremental Implementation

We look for incremental calculations of the averages that make up the estimates, as in Chapter 2.

For on-policy methods the incremental averaging is the same as in Chapter 2. For off-policy methods, but with ordinary importance sampling, we only need to multiply the returns by the importance sampling ratio and then we can average as before.

We will now consider weighted importance sampling. We have a sequence of returns G_i, all starting in the same state s and each with a random weight W_i (e.g. $W_i = \rho_{i:T(i)-1}$). We want to iteratively calculate (for $n \geq 2$)

$$V_n = \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k}.$$

We can do this with the following update rules

$$V_{n+1} = V_n + \frac{W_n}{C_n}[G_n - V_n] \tag{48}$$

$$C_{n+1} = C_n + W_{n+1} \tag{49}$$

where $C_0 = 0$ and V_1 is arbitrary (notice that it cancels out as $V_2 = G_1$).

Below is an algorithm for off-policy weighted importance sampling (set $b = \pi$ for on policy). The estimator Q converges to q_π for all encountered state-action pairs.

Off-policy MC prediction (policy evaluation) for estimating $Q \approx q_\pi$

Input: an arbitrary target policy π
Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
    $Q(s,a) \in \mathbb{R}$ (arbitrarily)
    $C(s,a) \leftarrow 0$

Loop forever (for each episode):
    $b \leftarrow$ any policy with coverage of π
    Generate an episode following b: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    $W \leftarrow 1$
    Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
        $W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$
        If $W = 0$ then exit For loop

5.7 Off-Policy Monte Carlo Control

Below is an algorithm for estimating π_* and q_* in the GPI framework. The target policy π is the greedy policy with respect to Q, which is an estimate of q_π. This algorithm converges to q_π as long as an infinite number of returns are observed for each state-action pair. This can be achieved by making b ε-soft. The policy π converges to π_* at all encountered states even if b changes (to another ε-soft policy) between or within episodes.

Off-policy MC control, for estimating $\pi \approx \pi_*$

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
$Q(s,a) \in \mathbb{R}$ (arbitrarily)
$C(s,a) \leftarrow 0$
$\pi(s) \leftarrow \operatorname{argmax}_a Q(s,a)$ (with ties broken consistently)

Loop forever (for each episode):
$b \leftarrow$ any soft policy
Generate an episode using b: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G \leftarrow 0$
$W \leftarrow 1$
Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
$\pi(S_t) \leftarrow \operatorname{argmax}_a Q(S_t, a)$ (with ties broken consistently)
If $A_t \neq \pi(S_t)$ then exit For loop
$W \leftarrow W \frac{1}{b(A_t|S_t)}$

Notice that this policy only learns from episodes in which b selects only greedy actions after some timestep. This can greatly slow learning.

5.8 *Discounting Aware Importance Sampling

We present a method of importance sampling that recognises the return as a discounted sum of rewards. This can help in estimation, since if an episode is of length 100 and $\gamma = 0$ then the final 99 terms of the importance sampling ration contribute nothing to the expected value of our estimator (they have expected value of 1) but can greatly increase its variance. We therefore construct a method of importance sampling that takes into account discounting.

Introduce the *flat partial returns*

$$\bar{G}_{t:h} \doteq \sum_{i=t+1}^{h} R_i \quad 0 \leq t \leq h \leq T$$

then it can be shown (by rearranging) that

$$G_t \doteq \gamma^{i-t} R_{i+1} \tag{50}$$

$$= (1-\gamma) \sum_{h=t+1}^{T-1} \gamma^{h-t-1} \bar{G}_{t:h} + \gamma^{T-t-1} \bar{G}_{t:T}. \tag{51}$$

Now we can scale each flat partial return by a truncated importance sampling ratio (hence reducing variance).

Ordinary Importance Sampling Ratio

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \left[(1-\gamma) \sum_{h=t+1}^{T(t-1)} \gamma^{h-t-1} \rho_{t:h-1} \bar{G}_{t:h} + \gamma^{T(t)-t-1} \rho_{t:T(t)-1} \bar{G}_{t:T(t)} \right]}{|\mathcal{T}(s)|} \tag{52}$$

Weighted Importance Sampling Ratio

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)}\left[(1-\gamma)\sum_{h=t+1}^{T(t)-1}\gamma^{h-t-1}\rho_{t:h-1}\bar{G}_{t:h}+\gamma^{T(t)-t-1}\rho_{t:T(t)-1}\bar{G}_{t:T(t)}\right]}{\sum_{t\in\mathcal{T}(s)}\left[(1-\gamma)\sum_{h=t+1}^{T(t)-1}\gamma^{h-t-1}\rho_{t:h-1}+\gamma^{T(t)-t-1}\rho_{t:T(t)-1}\right]} \tag{53}$$

5.9 *Per-Decision Importance Sampling

There is another way in which we may be able to reduce variance in off-policy importance sapling, even in the absence of discounting ($\gamma = 1$). Notice that the off-policy estimators are made up of terms like

$$\rho_{t:T-1}G_t = \rho_{t:T-1}(R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-t-1}R_T)$$

and that each of these terms is of the form

$$\rho_{t:T-1}R_{t+1} = \frac{\pi(A_t|S_t)}{b(A_t|S_t)}\cdots\frac{\pi(A_{T-1}|S_{T-1})}{b(A_{T-1}|S_{T-1})}R_{t+1}.$$

Now notice that only the first and last terms here are correlated, while all the others have expected value 1 (taken with respect to b). Clearly this is also the case at each t. This means that

$$\mathbb{E}[\rho_{t:T-1}R_{t+k}] = \mathbb{E}[\rho_{t:t+k-1}R_{t+k}]$$

therefore

$$\mathbb{E}[\rho t : T-1 G_t] = \mathbb{E}[\tilde{G}_t]$$

where

$$\tilde{G}_t \doteq \sum_{i=t}^{T-1}\gamma^{i-t}\rho_{t:i}R_{i+1}.$$

Now we can write the ordinary importance sampling estimator as

$$V(s) \doteq \frac{\sum_{t\in\mathcal{T}(s)}\tilde{G}_t}{|\mathcal{T}(s)|}$$

possibly reducing variance in the estimator.

The weighted importance sampling estimators of this form that have so far been found have been shown to not be consistent (in the statistical sense). We don't know if a consistent weighted average form of this exists.