SMU Lyle School of Engineering

Reinforcement Learning

Monte Carlo Methods

Sutton/Barto* Chapter 5

Michael Hahsler, SMU

With figures from Sutton/Barto*

*Sutton and Barto, Reinforcement Learning: An Introduction, 2nd edition, MIT Press, Cambridge, MA, 2018

CC BY NC SA

Topics of this Course

- Introduction to reinforcement learning
- Markov decision processes
- **Part I: Tabular Methods**
  - Dynamic programming
  - **Monte Carlo methods**
  - Temporal-difference learning
  - Multi-step bootstrapping
  - Planning and learning with tabular methods
- Part II: Approximate Solution Methods
  - Prediction and Control using Approximation
  - Eligibility Traces
  - Policy Gradient Methods
- Part III: Modern RL Methods
  - Deep Reinforcement Learning
  - Current Applications

Summary of Notation

General

X	capital letters: random variables
x, p	lower-case letters: realizations of random variables or scalar functions
$\boldsymbol{w}$	Bold lower-case letters: real-valued vectors (even if random variables)
$\boldsymbol{W}$	bold capitals: matrices
α	Greek letters: parameters (vectors if in bolt)
$\Pr\{X = x\}$	probability that a random variable X takes on the value x
$X \sim p$	random variable X selected from distribution $p(x) = \Pr\{X = x\}$
$\mathbb{E}[X]$	expectation of a random variable X, i.e., $\mathbb{E}[X] = \sum_x p(x)x$
$\text{argmax}_a f(a)$	a value of action a at which $f(a)$ takes its maximal value

Value Function

G_t	return (cumulative reward) following time t
$G_{t:h}$	return from t to h (discounted and corrected)
$v_\pi(s)$	value of state s under policy π (expected return)
$v_*(s)$	value of state s under the optimal policy
$q_\pi(s, a)$	value of taking action a in state s under policy π
$q_*(s, a)$	value of taking action a in state s under the optimal policy
V, V_t	array estimates of state-value function v_π or v_*
Q, Q_t	array estimates of action-value function q_π or v_*

MDP

s, s'	states
a	an action
r	a reward
$\mathcal{S}$	set of all (nonterminal) states, $\mathcal{S}^+$ are all states
$\mathcal{A}(s)$	set of all actions available in state s
γ	discount-rate parameter
t	discrete time step
T	final time step of an episode (a.k.a. horizon)
A_t	random variable for the action at time t
S_t	random variable for the state at time t
R_t	random variable for the reward at time t
$p(s', r \mid s, a)$	probability of transition to state s' and receiving reward r, from state s taking action a.
$p(s' \mid s, a)$	probability of transition to state s' fom state s taking action a.
$r(s, a)$	expected immediate reward from state s after action a.
$r(s, a, s')$	expected immediate reward from state s to s' with action a.
$\pi(a \mid s)$	probability of taking action a in state s under stochastic policy π
$\pi(s)$	action taken in state s under deterministic policy π

Introduction

- **Model-free method**:
  - We do not assume knowledge of the environment model (the MDP).
  - We use “experience” = **sample sequences** of states, actions, and rewards from actual or simulated interactions with the environment.
- **Remember from AI**: model-free means the agent faces an unknown, fully observable, possibly stochastic environment.
- **Advantages**:
  - We often have **sample access** to systems.
  - It is easier to simulate transitions than to specify the complete MDP.

- **Monte Carlo (MC) methods** are a family of algorithms that rely on repeated random sampling to obtain numerical results (see example to the right).

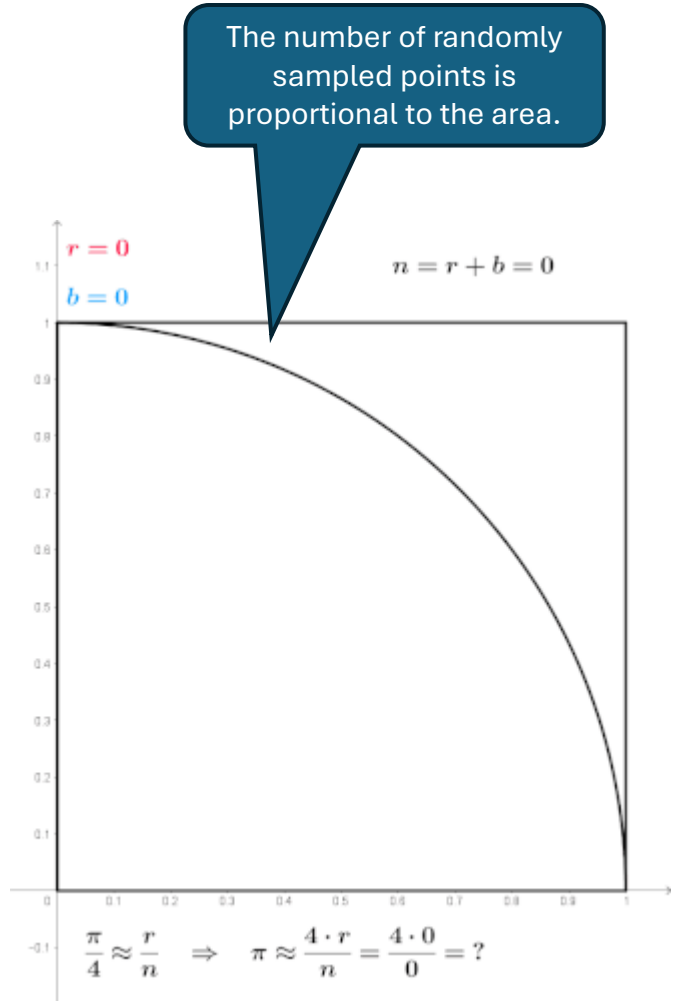
- **MC value function approximation:** Use the sample-average method to approximate the expectation in

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

The average return of many sampled sequences (called episodes or trajectories) starting at s and following π converges to the true expectation as the number of samples increases.

$$v_\pi(s) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} G_{t|S_t=s,\pi}$$

- **Note**: This methods needs to sample whole episodes before any calculation. It is not a true step-by-step online method!



Monte Carlo method applied to approximating the value of π
Source: Wikipedia

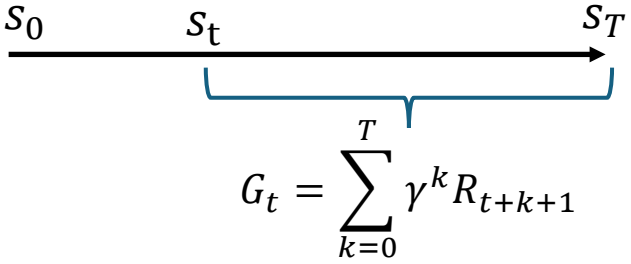
Monte Carlo Prediction

Estimate state values by sampling with a given policy.

MC Prediction: Estimate v_π for a given π

- **Goal**: Estimate $v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$ for a given deterministic or stochastic policy π by averaging the observed returns G for the rest of the episode after a visit of s.
- **Idea:** We can update all state values along the episode.
- **Convergence**: The average of many sampled returns converges to the expectation and thus $v_\pi(s)$.
- **First-visit**: Using the return only for the first visit of a state in the episode ensures that the sampled returns are i.i.d. distributed (not dependent on each other) and we get an **unbiased** estimate.

Episode



Calculate the return following s_t for all states 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, $t = T-1, T-2, \ldots, 0$:

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

Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

Append G to $Returns(S_t)$

$V(S_t) \leftarrow$ average$(Returns(S_t))$

We observe a sequence of rewards

Calculate remaining returns in reverse order!

Uses only the first visit.

This sample average approximates the expected return.
Note: This can be implemented with incremental updating of the averages!

Monte Carlo Control

Find a good policy while using it to sample the data.

MC Control as Generalized Policy Iteration (GPI)

- Approx. π_* using ideas from GPI with a Q-function estimated using MC prediction.

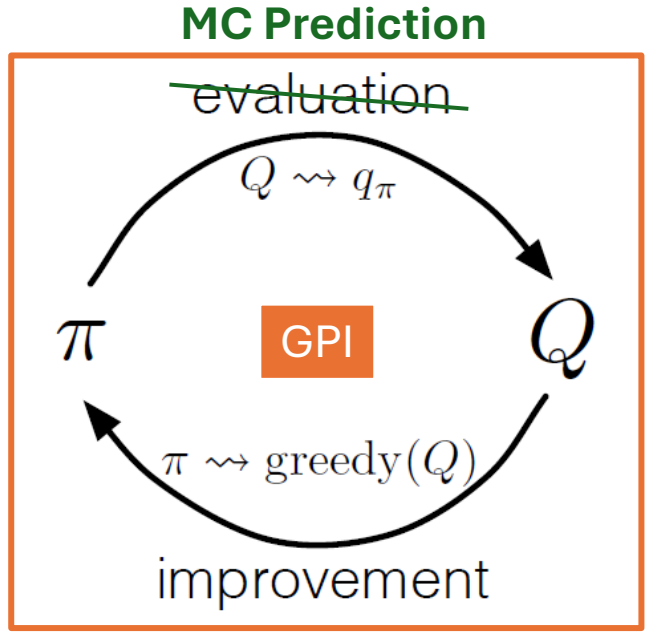
- MC policy iteration:

$$\pi_0 \overset{E}{\rightarrow} q_{\pi_o} \overset{I}{\rightarrow} \pi_1 \overset{E}{\rightarrow} q_{\pi_1} \overset{I}{\rightarrow} \pi_2 \overset{E}{\rightarrow} q_{\pi_2} \overset{I}{\rightarrow} \pi_3 \overset{E}{\rightarrow} q_{\pi_3} \overset{I}{\rightarrow} \dots \pi_* \overset{E}{\rightarrow} q_*$$

- Process:
  1. Start with an arbitrary policy.
  2. **Evaluation**: Update MC estimate by sampling some episodes using the current π.
  3. **Improvement**: $\text{greedy}(Q)$ means

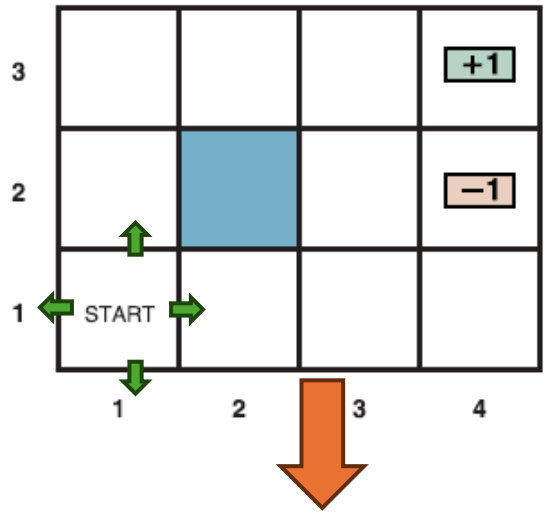
$$\pi(s) = \underset{a}{\text{argmax}}\, q(s,a) \quad \forall s \in \mathcal{S}$$

  4. Repeat evaluation and improvement till the policy does not change (= optimal policy is reached).



MC Prediction to Estimate Q-Values

- MC prediction for $q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$.
- Same as for $v_\pi(s)$, just record and average returns for action-state pairs (a,s) when following π.
- **Convergence**: Unbiased samples lead to convergence to q_π after ∞ episodes.
- **Exploration issue**: Many (a,s)-pairs that are important for π^* may never be visited when following π. To get Q-Value estimates, we need to keep exploring!
- Options to guarantee that all (a,s)-pairs are sampled :
  - **a)** **Exploring starts**: choose the starting (a,s) for each episode randomly.
  - b) Only consider **soft policies** like ϵ-greedy.
    Remember: Soft policies have non-zero probabilities for all actions so they will explore.



Q-Table

s	a	$q(s,a)$
(1,1)	up	Sample average
(1,1)	right	Sample average
(1,1)	down	Sample average
(1,1)	left	Sample average
…	…	…

Remember: Policy Improvement Theorem

If we update the MC $q(s,\cdot)$ estimate by adding more unbiased sample returns following π_k, then we can construct a new greedy policy π_{k+1} that is better (or equal) compared to π_k. Reason:

$$v_{\pi_{k+1}}(s) = q_{\pi_k}\big(s, \pi_{k+1}(s)\big) = q_{\pi_k}(s, \operatorname*{argmax}_a q_{\pi_k}(s,a))$$
$$= \max_a q_{\pi_k}(s,a)$$
$$\geq q_{\pi_k}\big(s, \pi_k(s)\big)$$
$$\geq v_{\pi_k}(s)$$

π_{k+1} chooses greedy action for s

Choosing the best action will never lead to a lower value!

$\rightarrow$ converges to π_*

Assumptions

- All state/action pairs are tried (e.g., start episodes with exploring states)
- Perfect policy evaluation with ∞ many episodes before each policy update. **Perfect policy evaluation is impractical!**

The effect of imperfect policy evaluation

- Individual q estimates will be wrong but unbiased return estimates will produce in expectation (on average) the correct estimate.
- Policy improvement sometimes may pick the wrong action, but **on average favors actions with true higher values.**

Alg. MC Control with Exploring Starts

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

Loop forever (for each episode):

Generate an episode following π: S

$G \leftarrow 0$

Loop for each step of episode

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

Unless S_t appears in $S_0, S_1, \ldots$

Append G to $Returns(S_t)$

$V(S_t) \leftarrow$ average($Returns($

So all state-action pairs are tired.

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

Initialize:

$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$

Learn policy

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

Q-Values instead of V.

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

Loop forever (for each episode):

Exploring starts

Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly 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, $t = T-1, T-2, \ldots, 0$:

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

First-visit

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 $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow$ average($Returns(S_t, A_t)$)

Greedy policy update

$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

Code...

MC Control without Exploring Starts

- **Issue**: The environment needs to let us start episodes at different (s, a) at will. This is fine for simulations, but hard in the real world.

- **Need**: For convergence, we need a guarantee that all (reachable) state-action combinations are tried.

- Options:
  a) **On-policy control**: learn a **soft policy** instead of the optimal policy.
  b) **Off-policy control**: use a soft **behavior policy** for sampling that tries all state-action pairs, but learn the optimal, deterministic **target policy**.

On-Policy Monte Carlo Control

Find a good policy while using it to sample.

On-Policy vs. Off-Policy Control

Dilemma:

- We need behavior that explores.
- We know that MDPs have optimal deterministic policies that do not explore.
- Learning optimal action values depends on observing optimal behavior.

On-policy: Can learn a near-optimal policy that still explores (i.e., ϵ-greedy with a small ϵ) and then converts it into a greedy deterministic policy.

Off-policy: Use two policies

- **Target policy**: learn π, which will become an optimal deterministic policy.
- **Behavior policy** b: guarantees exploration with a soft policy.

- Called "off-policy" because the generated data follows a different policy than the target policy.

- Off-policy methods are typically:
  - More complicated
  - Converge slower
  - Often more powerful
  - Can be applied to data externally generated using a fixed, potentially unknown, behavior policy.

Note: On-policy learning is a special case of off-policy learning with $b = \pi$

Types of Soft Policies for Exploration

Definition : A policy is soft if

$$\pi(a|s) > 0 \qquad \forall s \in \mathcal{S}, a \in \mathcal{A}$$

Soft policies are stochastic and used as behavior policies that guarantee exploration.

Important Soft Policy Types:

- **ϵ-soft policy**: agent takes with probability ϵ a random action.

$$\pi(a|s) \geq \frac{\epsilon}{|\mathcal{A}(s)|}$$

- **ϵ-greedy policy**: follows a greedy policy but chooses a random action with probability ϵ

$$\pi(a|s) = \begin{cases} 1 - \epsilon + \dfrac{\epsilon}{\mathcal{A}(s)} & \text{if } a = \text{greedy action} \\ \dfrac{\epsilon}{|\mathcal{A}(s)|} & \text{if } a \neq \text{greedy action} \end{cases}$$

Alg. On-Policy MC Control with ϵ-soft Policies

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

Loop forever (for each episode):
Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such
Generate an episode from S_0, A_0, fol
$G \leftarrow 0$
Loop for each step of episode, $t = T-1, T-$
$G \leftarrow \gamma G + R_{t+1}$
Unless the pair S_t, A_t appears in $S_0, A_0,$
Append G to $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow$ average$(Returns(S_t, A_t)$
$\pi(S_t) \leftarrow \text{argmax}_a\, Q(S_t, a)$

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, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
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 $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow$ average$(Returns(S_t, A_t))$
$A^* \leftarrow \text{argmax}_a\, Q(S_t, a)$ (with ties broken arbitrarily)
For all $a \in \mathcal{A}(S_t)$:

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

Start with soft policy

No exploring starts!

Create ϵ-greedy policy

Important: On-policy learns the best ϵ-greedy policy, but not the optimal deterministic policy!

GLIE: Greedy in the Limit with Infinite Exploration

- **Issue**: On-policy MC control algorithms change the policy and average over returns from different policies. This could be a problem for convergence!

GLIE Condition

Many on-policy control methods converge if:

- **Infinite explorations**: Every state-action pair is visited infinitely often.

$$\lim_{t\to\infty} N_t(s,a) = \infty$$

- **Greedy in the limit**: The policy becomes greedy with respect to Q over time.

$$\lim_{t\to\infty} \pi\big(\text{argmax}_a(Q_t(s,a),s)\big) = 1$$

- **Reason**: Early wrong return samples from "bad policies" will eventually be dominated by many new samples from the optimal/near-deterministic policy.

- **Implementation**:
  - Infinite exploration: Use many episodes and an ϵ-greedy policy for exploration.
  - Greedy in the limit: Use an ϵ-greedy policy where ϵ is reduced over time. E.g., $\epsilon_t = \frac{1}{t}$

Off-Policy Prediction

Estimate a value function for a policy different from the sampling policy.

Off-Policy Prediction

- **Goal**: Estimate v_π or q_π given episodes generated with $b \neq \pi$.

- **Assumptions**:
  - b is fixed.
  - **Coverage**: Every action taken under π will occasionally also be taken under b

$$\pi(a|s) > 0 \Rightarrow b(a|s) > 0$$

  - That is, b needs to be stochastic (soft) for states where it differs from π (typically b is ϵ-greedy)

- **Issue**: Returns and sample averages are valid for b but not for π! We need to correct them.

Importance Sampling

- **Importance sampling** is a method to estimate the expected values under a desired distribution, given samples from another distribution
  - Expectation under desired distribution: $\mathbb{E}_\pi(G_t|S_t = s)$
  - Observed expectation from sampled distribution: $\mathbb{E}_b(G_t|S_t = s)$
- **Idea**: correct returns observed under b to estimate returns for π

- Probability of a trajectory starting at S_t under π

$$\Pr\{A_t, S_{t+1}, A_{t+1}, \dots, A_{T-1}, S_T | S_t, A_{t:T-1} \sim \pi\} = \prod_{k=t}^{T-1} \pi(A_k|S_k) p(S_{k+1}|S_k, A_k)$$

Needs unknown transition probabilities!

- **Importance sampling ratio:** How much more likely are the remaining actions in the sequence chosen under π than under b:

$$\rho_{t:T-1} \stackrel{\text{def}}{=} \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)\, p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)\, p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

Luckily, the unknown transition probabilities cancel out!

- Now we get:

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

- **Note**: We only get $\rho_{t:T-1} > 0$ when all A_k chosen by b have a $\pi(A_k|S_k) > 0$!

This especially a problem if π is a deterministic policy very different from b!

Ordinary vs. Weighted Importance Sampling

- **Ordinary importance sampling** averages the corrected sample returns

$$V(s) \stackrel{\text{def}}{=} \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|} \longrightarrow \mathbb{E}_b[\rho_{t:T-1} G_t | S_t = s] = \mathbb{E}_\pi(G_t | S_t = t)$$

  - $\mathcal{T}(s)$ are the time steps that s was (first) visited.
  - Gives an unbiased estimate of $v_\pi(s)$ but the variance is extreme for low $|\mathcal{T}(s)|$.

Note: the $\frac{|\mathcal{T}(s)|}{|\mathcal{T}(s)|}$ is missing since it canceled out.

- **Weighted importance sampling**

$$V(s) \stackrel{\text{def}}{=} \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1}} \longrightarrow \frac{\mathbb{E}_b[\rho_{t:T-1} G_t | S_t = s]}{\mathbb{E}_b\left[\rho_{t:T(t)-1} \middle| S_t = s\right]} = \frac{\mathbb{E}_\pi(G_t | S_t = t)}{1}$$

  - Set 0 if denominator is 0.
  - Initially biased to $v_b(s)$. The ρ terms cancel for a single observation and G_t is sampled using b.
  - Bias falls asymptotically with more visits
  - Preferred method with much lower errors (variance) for smaller number of episodes.

Alg. Off-Policy Prediction

Weighted importance sampling + incremental average update

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

$Returns(s,a) \leftarrow$ empty list, for a

Repeat forever (for each epis

Generate an episode follo

$G \leftarrow 0$

Loop for each step of episode, t =

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

Unless the pair S_t, A_t appears

Append G to $Returns(S_t,$

$Q(S_t, A_t) \leftarrow$ average($Retu$

$A^* \leftarrow \text{argmax}_a Q(S_t, a)$

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

$$\pi(a|S_t) \leftarrow \begin{cases} 1-\varepsilon + \\ \varepsilon/|\mathcal{A}(S \end{cases}$$

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)

$C(s,a) \leftarrow 0$

cumulative sum of weights for (s,a) for incremental update of the average return.

Loop forever (for each episode):

$b \leftarrow$ any policy with coverage of π

b can change every episode! Practical choice is $b = \epsilon$-greedy(π)

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

$G \leftarrow 0$

$W \leftarrow 1$

Importance sampling ratio ρ

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

Can only learn from the end of the episode where the action chosen from b can also be chosen by π!

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

$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$

$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$

$W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$

Incremental update + Correct the update

Off-Policy Control

Find a good policy while following a different behavior policy.

Off-Policy MC Control

- **Remember**: on-policy control tries to estimate a policy while using it for sampling. This has the following implications:
  - The **behavior changes** with the learned policy.
  - The policy that is learned needs to explore and therefore **may not be able to learn the optimal deterministic policy**.

- Off-policy control uses two policies:

  - **Behavior policy b**: Needs to have coverage. That is, it eventually takes every action taken under π. This typically means that we will use a **soft policy**.

  - **Target policy π**: Typically, a **deterministic greedy policy** which can become optimal.

Alg. Off-Policy MC Control

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 episod
$b \leftarrow$ any policy with cove
Generate an episode followin
$G \leftarrow 0$
$W \leftarrow 1$
Loop for each step of episode, t =
$G \leftarrow \gamma G + R_{t+1}$
$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S}$
$W \leftarrow W \frac{\pi(A_t|S_t)}{b(A_t|S_t)}$

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)
$C(s,a) \leftarrow 0$
$\pi(s) \leftarrow \arg\max_a Q(s,a)$ (with ties broken consistently)

Add the greedy target policy

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, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
$C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)} [G - Q(S_t, A_t)]$
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken consistently)
If $A_t \neq \pi(S_t)$ then exit inner Loop (proceed to next episode)
$W \leftarrow W \frac{1}{b(A_t|S_t)}$

We use π to select a. So the probability is 1.

Can only uses the end of the episode that uses greedy actions for π
Issue: Not very sample efficient!

What you Should Know

MC works with sampled episodes:

- **Policy Evaluation**: Averages many sampled returns.
- **Control**: Interleaves pol. eval. and pol. improvement on an episode-by-episode basis.
- **On vs. off-policy control.**

Advantages of MC methods vs. DP:

- **Model-free**: No need for an environment model. Sampling from a simulation is often easy.
- Can evaluate a small **state set** of interest. DP always has to sweep over all states.
- **Unbiased estimates:** Estimates are based on returns from complete sampled sequences. We will learn about other methods that introduce bias by bootstrapping (estimating a value from other estimates).
- Using complete sequences may be less harmed by violations of the Markov property/issues with observability.

Issues:

- **Needs to maintain sufficient exploration**. Options are:
  1. Exploring starts can learn the optimal policy.
  2. On-policy control with an ϵ-soft policy and GLIE.
  3. Off-policy control with a covering behavior policy. Note: only a part of the data can be used.
- MC methods are **not sample efficient**. I.e., sampled episodes are used only once and then discarded.
- **Not a truly online method**: Updates are not done after each action, but are delayed until the end of each episode.