Monte Carlo Methods

Chapter 5

Our first **learning methods** for estimating
value functions and optimal policies

Assignment marking

- You all did very well :)
- Make sure you follow the instructions given, if it says submit the plot then do so
  - Matters for tests!
- Show your work! The last line, final equation is not the important part
- Bonus questions are bonus, not free marks
  - They require special effort and our expectations are higher

Bellman Equation for state values

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] && \text{(by (3.9))} \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r \,|\, s, a) \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s'] \Big] \\
&= \sum_a \pi(a|s) \sum_{s', r} p(s', r \,|\, s, a) \Big[r + \gamma v_\pi(s') \Big], \quad \text{for all } s \in \mathcal{S}, && (3.14)
\end{aligned}
$$

- Full details available in the course folder: Bellman_Derivation.pdf

Chapter 5 Summary

- Monte Carlo (MC) methods estimate value functions and optimal policies from **experience** or **sample episodes**
  Experience → values, policy
- **MC** methods simply average many returns that start in a state to estimate value functions
- These methods have 3 important advantages over DP methods
  1. learn from interaction with environment; no probability model required
  2. can also be used with **simulation models**
  3. can easily focus MC methods on a **subset** of the states
- Less sensitive to violations of the Markov Property
  - update estimates from sample trajectories not the values of other states
  - they do not **bootstrap**

…Summary

- MC methods fit nicely into GPI
  - MC methods provide an alternative policy evaluation process
  - MC methods do the policy evaluation and policy improvement on an **episode by episode basis**
- Only defined for episodic tasks (in this book)
- MC methods face the challenge of **sufficient exploration**
  - handled with *exploring starts*, learning about ϵ-*soft policies*, or *off-policy* learning
- Off-policy learning is the process of learning **about** one policy while following another
  - this requires importance sampling to re-weight the sample returns; either ordinary importance sampling or weighted

The Setting

- Unlike Chapter 4, we will not assume access to the probability model p(s',r | s,a)
- We will estimate value functions (V(s), Q(s,a)) from **experience**:
  - trajectories of states, actions, & rewards
  - starting in some state and ending in termination
- Focus on undercounted episodic tasks
  - all episodes eventually termination no matter what actions are selected
- Only make change our estimates for value functions and the policy at the **end of an episode**
  - **incremental** on an episode-by-episode basis
  - not step-by-step

Experience

- Sample trajectories or episodes can be **actually** generated by an agent interacting with the world
- Or we can **simulate** trajectories from a model
- We do not require a probability model to simulate
  - $p : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S} \times \mathcal{R};\ S_{t+1}, R_{t+1} = p(S_t, A_t)$
  - $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \times \mathcal{R} \rightarrow [0, 1];\ p(S_{t+1}, R_{t+1}, S_t, A_t) = pr$

MC Policy Evaluation

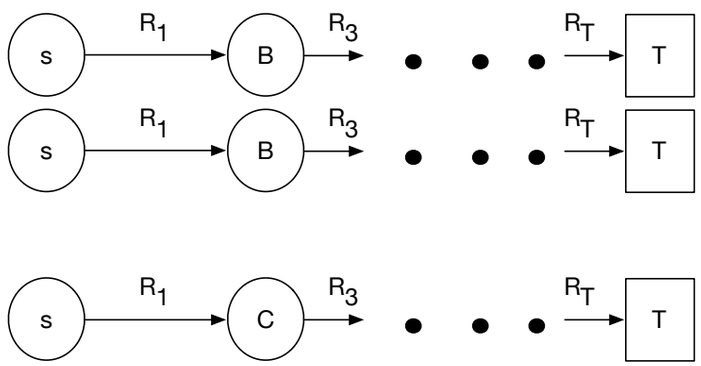
MC methods sample and average returns

- Lets consider **policy evaluation** or the *prediction* case, we will:
  - learn the state-value function for a given policy
  - estimate V(s) for policy π
  - approximate $v_\pi(s)$

MC Policy Evaluation

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

$$= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + ... + \gamma^{T-1} R_T | S_t = s]$$



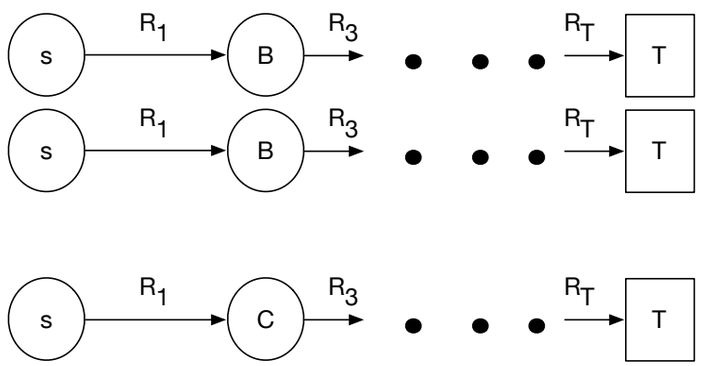
Note: each episode might be a different length (T is a random variable)

MC Policy Evaluation

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

$$= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + ... + \gamma^{T-1} R_T | S_t = s]$$

- How can we update our estimates of $v_\pi(s)$ from experience?



Note: each episode might be a different length (T is a random variable)

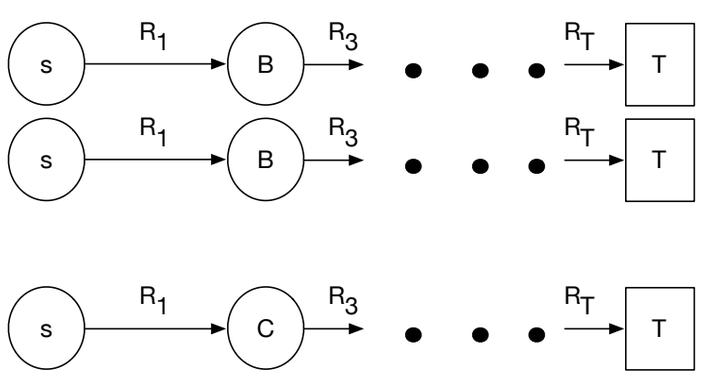
MC Policy Evaluation

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

$$= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + ... + \gamma^{T-1} R_T | S_t = s]$$

- How can we update our estimates of $v_\pi(s)$ from experience?

- collect a bunch of episodes starting in state *s*



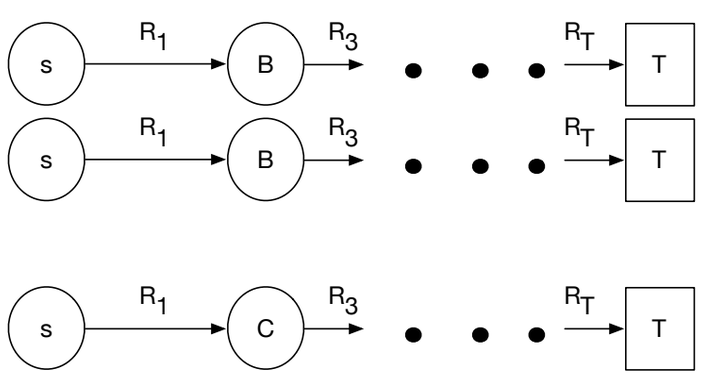
Note: each episode might be a different length (T is a random variable)

MC Policy Evaluation

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

$$= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + ... + \gamma^{T-1} R_T | S_t = s]$$

- How can we update our estimates of $v_\pi(s)$ from experience?
  - collect a bunch of episodes starting in state *s*
  - for each episode compute the sample return



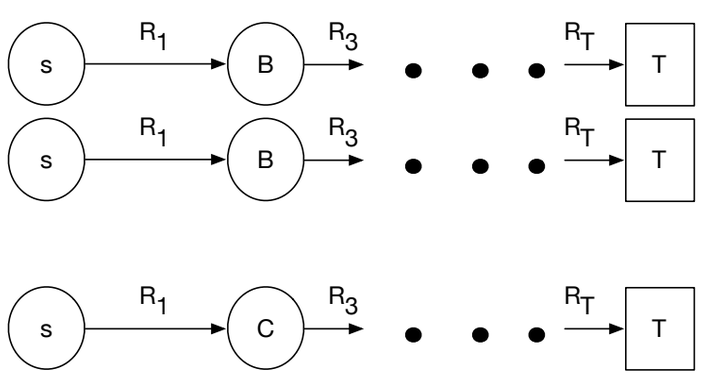
Note: each episode might be a different length (T is a random variable)

MC Policy Evaluation

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

$$= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + ... + \gamma^{T-1} R_T | S_t = s]$$

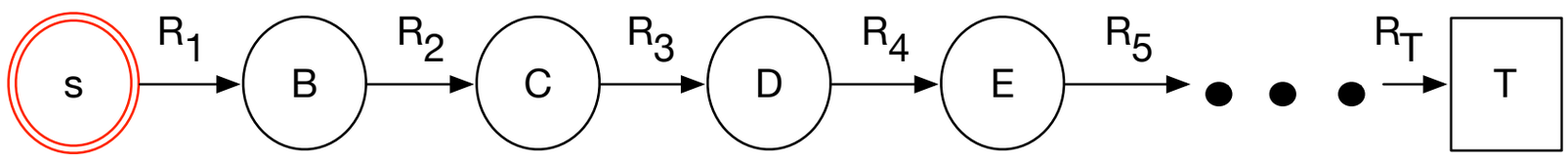
- How can we update our estimates of $v_\pi(s)$ from experience?
  - collect a bunch of episodes starting in state *s*
  - for each episode compute the sample return
  - average the sample returns, yielding V(s)



Note: each episode might be a different length (T is a random variable)

Consider processing a single sample episode

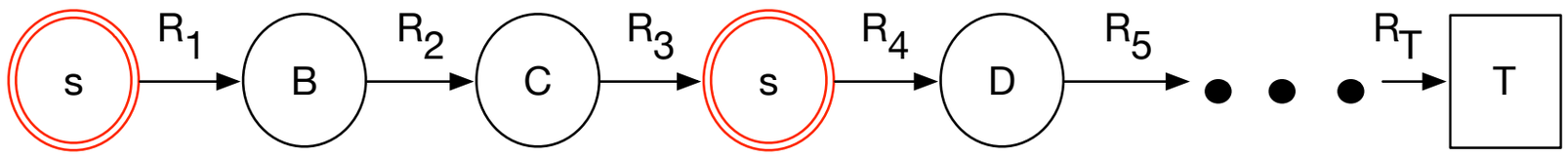
- For each state in the episode:
  - compute the return starting from each state



- This may generate many sample returns, for each state in the observed episode

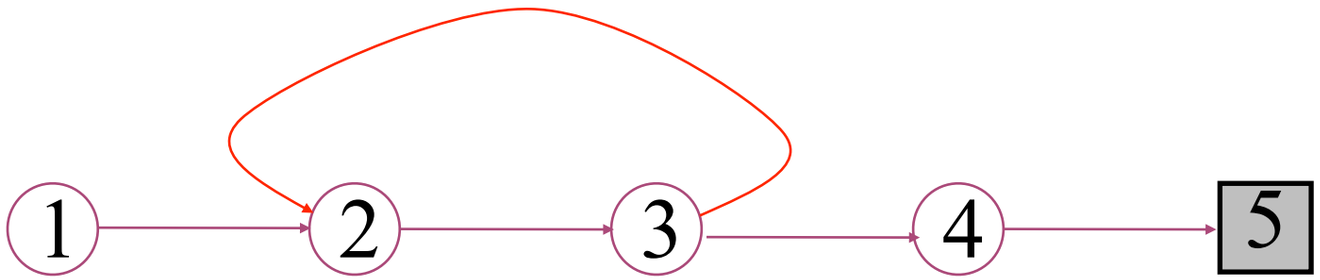
First visit and *every* visit MC

- **Every-visit MC:** averages returns **every** time *s* is visited in an episode
- **First-visit MC**: averages returns only for the **first** time is visited in an episode
- Both converge
- Imagine one sample episode, starting in state s:



Monte Carlo Policy Evaluation

- *Goal:* learn $v_\pi(s)$
- *Given:* some number of episodes under π which contain *s*
- *Idea:* Average returns observed after visits to s



- *Every-Visit MC:* average returns for *every* time *s* is visited in an episode
- *First-visit MC:* average returns only for *first* time *s* is visited in an episode
- Both converge asymptotically

First-visit MC policy evaluation algorithm

Initialize:

$\pi \leftarrow$ policy to be evaluated

$V \leftarrow$ an arbitrary state-value function

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

Repeat forever:

Generate an episode using π

For each state s appearing in the episode:

$G \leftarrow$ return following the first occurrence of s

Append G to $Returns(s)$

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

Blackjack

- Card game
- You against the dealer (no other players!)
- Game starts with you being dealt **2 cards** and the dealer gets 2 cards, but you can only see **one of the dealers cards** (infinite deck!)
- The face cards are worth 10, Ace can be 1 or 11
- You go first: you can decide to **stick** (get no more cards) or **hit** (get another card)
  - your goal is to get the sum of your cards to be higher than the dealer without exceeding 21: **bust**, game over
  - automatic win if 2 start cards = 21 & dealer does not have 21 to start
- After you go (& you don't bust), its the dealer's turn:
  - they **sticks** on any sum >= 17, otherwise they **hit** (he can go bust also)
- Game ends in win, lose, or draw

Blackjack s,r,a

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick
- **Rewards**?

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick
- **Rewards**?
  - +1 win, 0 for draw, -1 loss at the end, all other rewards are zero

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick
- **Rewards**?
  - +1 win, 0 for draw, -1 loss at the end, all other rewards are zero
- Continuing? Episodic?

Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick
- **Rewards**?
  - +1 win, 0 for draw, -1 loss at the end, all other rewards are zero
- Continuing? Episodic?
  - $\gamma = 1$; undiscounted episodic task

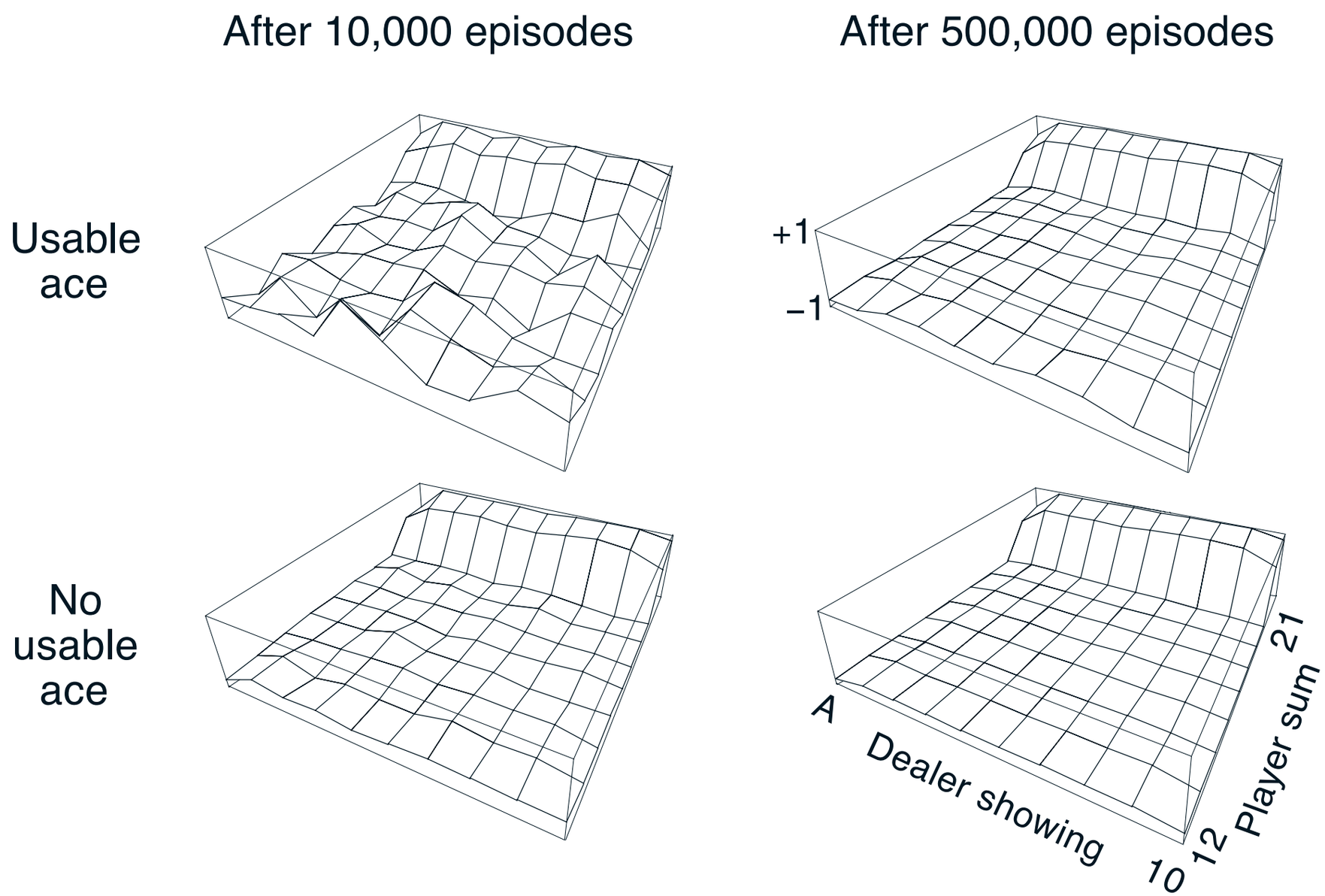
Blackjack s,r,a

- What are the **non-terminal states** (infinite deck with replacement)?
  - useable Ace means you **could** count it as 11 and sum **not** > 21
  - your **current sum** (12-to-21), **dealers card** (ace-to-10),
  - if you have a **useable ace:** could count as 11 without going bust
- **Actions**?
  - two actions: hit or stick
- **Rewards**?
  - +1 win, 0 for draw, -1 loss at the end, all other rewards are zero
- Continuing? Episodic?
  - $\gamma = 1$; undiscounted episodic task
- Estimate v_π for policy that **sticks on sum of 20 or 21 and hits otherwise**

Simulate many games of Blackjack

- each game produces a sample episode, where the return is the final reward
- compute sample returns from states observed in each episode
- average the returns for each state to find V

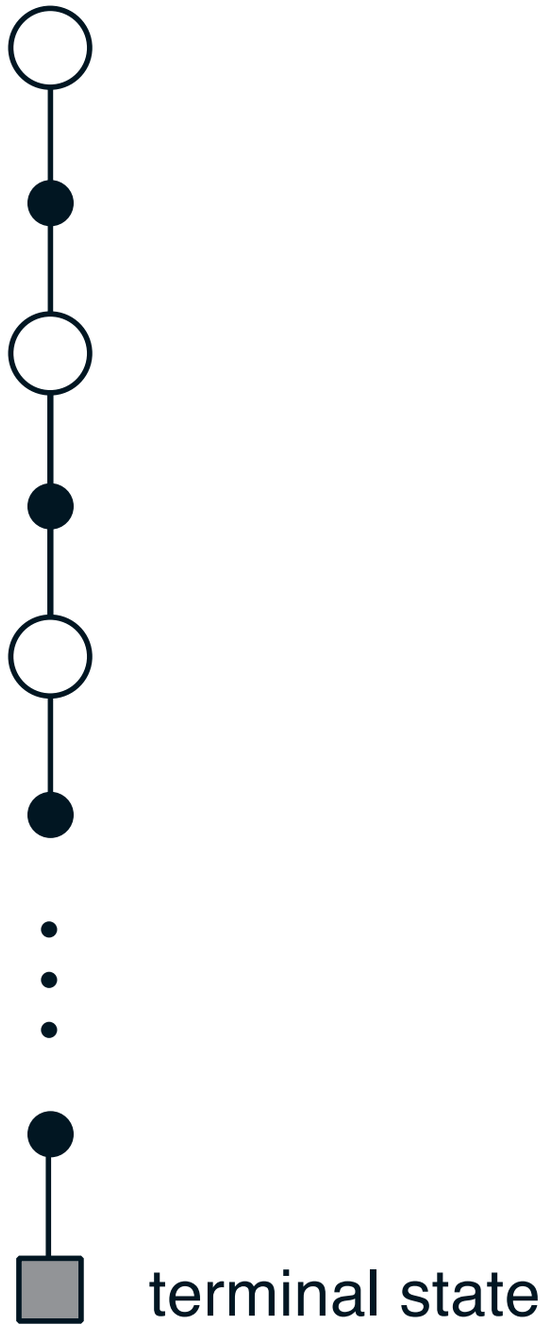
Learned Blackjack state-value function



Imagine solving this problem with DP

Backup diagram for Monte Carlo

- Entire rest of episode included
- Only one choice considered at each state (unlike DP)
  - in DP we consider all possible next states
  - thus, there will be an explore/exploit dilemma



Bootstrapping

- In MC we do not **bootstrap** from possible next state values like we do in DP:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \Big[r + \gamma v_\pi(s') \Big]$$

- The time to update a states value does not depend on the total number of states. We don't **sweep** like in DP

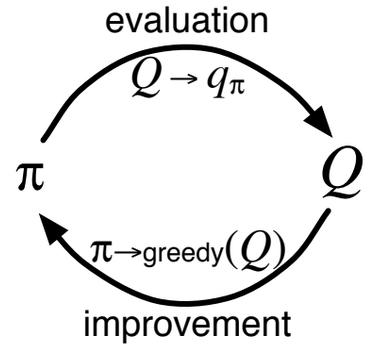
State-action value functions and MC

- In Chapter 4 we assumed access to $\boldsymbol{p}(s',r|s,a)$
- Using the state-value function and $\boldsymbol{p}$ we could look ahead one step choose the action that gave best sum of reward and next state value
  - Policy iteration used the current estimate of v* and **p**, instead of q*(s,a)
- In MC we don't assume a probability model so we will instead attempt to estimate $Q(s,a)$
- Just like with state-values we can use first-visit and every-visit policy evaluation MC methods to estimate $q_\pi(s,a)$
  - using these methods our Q will converge to q_π

Maintaining exploration

- Imagine π is deterministic—say greedy; in every state we select one action with probability 1
  - we will only observe returns from one action from each state
  - for all other actions b!=a, estimates of $Q_\pi(s,b)$ will **not improve** no matter how many returns we average
- The purpose of learning q_π is help choose amongst different actions in each state: arg_max Q(s,a)
- We have so somehow insure that we continue to explore
- **Exploring starts**: every state-action pair has nonzero prob of being the start state and first action of each episode
  - when might this be impractical?

Monte Carlo Control



- Goal: find optimal policy: π^*
- We again use Generalize policy iteration
  1. policy evaluation: MC method with exploring starts
     - many episodes will be experienced and Q->q^*
  2. policy improvement: $\pi(s) \stackrel{\text{def}}{=} \arg\max_a Q(s,a)$
- We can use MC methods within GPI framework to find π^* and q^* in the limit
  - assuming exploring starts and an infinite # of episodes in policy evaluation step
  - no knowledge of environment dynamics is required

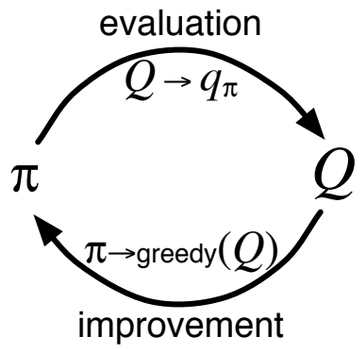
Policy improvement

- Monte Carlo Control algorithm satisfies the policy improvement theorem
- Ensuring the whole process converges to the optimal policy and optimal value function
- Using only sample episodes, and no other information about the dynamics
- But we needed to made two unrealistic assumptions:
  - The ability to use exploring starts
  - infinite number of episodes

Infinite sampling

- We made this assumption in DP as well
  - DP policy evaluation converges only asymptotically to the true value function
- Solution: don't worry about converging to q_π on each policy evaluation step
  - instead move q_{π_k} toward q_π
  - just like value iteration
  - results in GPI say this is ok
- So lets operate on an episode by episode basis

Monte Carlo ES



- Alternate between PE and PI on an episode by episode basis
- All returns are accumulated and averaged for each (s,a)-pair, no matter what policy was used to generate them
- MC ES cannot converge to a suboptimal policy. Say π stops changing and its not π^*:
  - if it did, then PE would cause Q_π to converge to q_π
  - then the PI step would improve it, since π is not optimal
  - stability is only achieved when both π^* and q^* are reached

Monte Carlo ES Algorithm

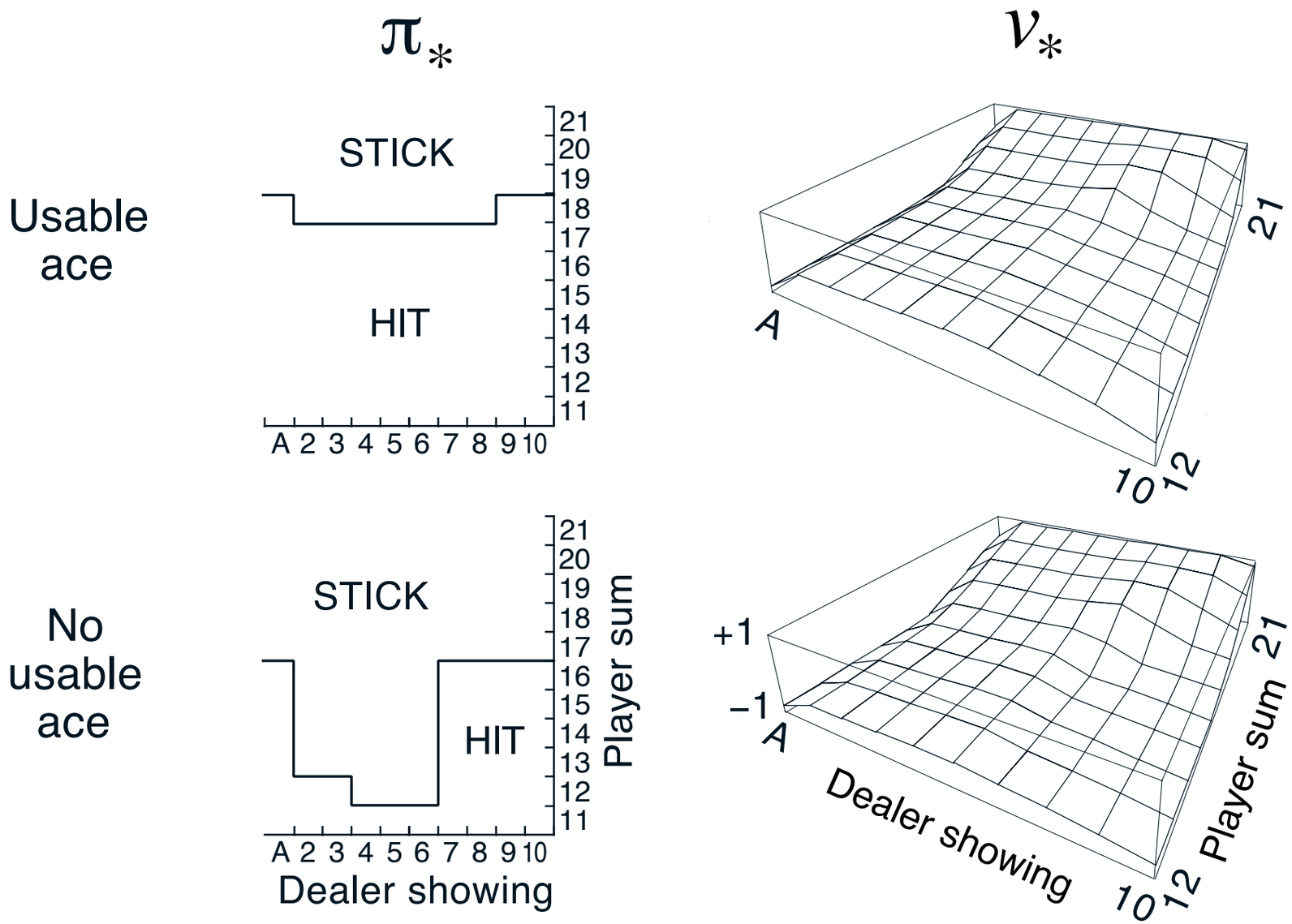
Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
    $Q(s,a) \leftarrow$ arbitrary
    $\pi(s) \leftarrow$ arbitrary
    $Returns(s,a) \leftarrow$ empty list

Repeat forever:
    Choose $S_0 \in \mathcal{S}$ and $A_0 \in \mathcal{A}(S_0)$ s.t. all pairs have probability > 0
    Generate an episode starting from S_0, A_0, following π
    For each pair s, a appearing in the episode:
        $G \leftarrow$ return following the first occurrence of s, a
        Append G to $Returns(s,a)$
        $Q(s,a) \leftarrow \text{average}(Returns(s,a))$
    For each s in the episode:
        $\pi(s) \leftarrow \arg\max_a Q(s,a)$

Back to Blackjack

- We will use MC ES on blackjack
- Initial policy: stick on 20,21, otherwise hit
  - Why is this important?
- How do we do exploring starts?
  - we had to write a program to simulate blackjack
  - randomly generate dealer's cards, the player's sum and, if we have a useable ace all with equal probability

Optimal policy for Blackjack



MC control with soft policies

- Exploring starts is unrealistic in many domains:
  - a robot, controlling an elevator
- To get rid of this assumption we must consider **soft** policies: $\pi(a|s) > 0 \text{ for all } s \in \mathcal{S}$
  - for example, **ϵ**-greedy
- An **ϵ**-soft policy: $\pi(a|s) \geq \frac{\epsilon}{|\mathcal{A}(s)|} \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s)$
- For **ϵ**-greedy for the greedy action π(a|s) =
  - and for each non-greedy action π(b|s) =

MC control with soft policies

- Exploring starts is unrealistic in many domains:
  - a robot, controlling an elevator
- To get rid of this assumption we must consider **soft** policies: $\pi(a|s) > 0 \text{ for all } s \in \mathcal{S}$
  - for example, **ϵ**-greedy
- An **ϵ**-soft policy: $\pi(a|s) \geq \frac{\epsilon}{|\mathcal{A}(s)|} \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s)$
- For **ϵ**-greedy for the greedy action π(a|s) = $1 - \epsilon + \frac{\epsilon}{|\mathcal{A}(s)|}$
  - and for each non-greedy action π(b|s) =

MC control with soft policies

- Exploring starts is unrealistic in many domains:
  - a robot, controlling an elevator
- To get rid of this assumption we must consider **soft** policies: $\pi(a|s) > 0 \text{ for all } s \in \mathcal{S}$
  - for example, **ϵ**-greedy
- An **ϵ**-soft policy: $\pi(a|s) \geq \frac{\epsilon}{|\mathcal{A}(s)|} \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}(s)$
- For **ϵ**-greedy for the greedy action π(a|s) = $1 - \epsilon + \frac{\epsilon}{|\mathcal{A}(s)|}$
  - and for each non-greedy action π(b|s) = $\frac{\epsilon}{|\mathcal{A}(s)|}$

GPI and ϵ-greedy policies

- GPI allows the policy improvement step to make the new policy (π'), nearly greedy with respect to Q_π
  - recall if we make it purely greedy and do not use exploring starts we would not explore
  - instead we will make π' **ϵ**-greedy wrt Q_π and π' is guaranteed to be an improvement over π
- Instead of learning π_*, we are learning the **best ϵ-soft policy** $\widetilde{\pi}_\star$, a **near-optimal policy**
- Can be shown that this process converges to the best **ϵ**-soft policy

MC control algorithm with ϵ-soft policies

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:

$Q(s,a) \leftarrow$ arbitrary

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

$\pi(a|s) \leftarrow$ an arbitrary ε-soft policy

Repeat forever:

(a) Generate an episode using π

(b) For each pair s, a appearing in the episode:

$G \leftarrow$ return following the first occurrence of s, a

Append G to $Returns(s,a)$

$Q(s,a) \leftarrow \text{average}(Returns(s,a))$

(c) For each s in the episode:

$A^* \leftarrow \arg\max_a Q(s,a)$

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

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

Your thought questions

Your thought questions

- In complex tasks, how do we include subgoals?

Your thought questions

- In complex tasks, how do we include subgoals?
- How do we deal with delays?

Your thought questions

- In complex tasks, how do we include subgoals?
- How do we deal with delays?
  - Delayed reward?

Your thought questions

- In complex tasks, how do we include subgoals?
- How do we deal with delays?
  - Delayed reward?
  - Delayed impact of our action selection on state dynamics?

Your thought questions

- In complex tasks, how do we include subgoals?
- How do we deal with delays?
  - Delayed reward?
  - Delayed impact of our action selection on state dynamics?
  - How do we handle partial information?

Your thought questions

Your thought questions

- Humans often learn by mimicking others. How can we integrate this into the reinforcement learning framework?

Your thought questions

- Humans often learn by mimicking others. How can we integrate this into the reinforcement learning framework?
  - What if you don't really know what actions the other is taking? What if they have a different action set?

Your thought questions

- Humans often learn by mimicking others. How can we integrate this into the reinforcement learning framework?
  - What if you don't really know what actions the other is taking? What if they have a different action set?
  - Is learning language first essential for learning? Communicating with other agents …

Your thought questions

- Humans often learn by mimicking others. How can we integrate this into the reinforcement learning framework?
  - What if you don't really know what actions the other is taking? What if they have a different action set?
  - Is learning language first essential for learning? Communicating with other agents …
  - What if the other agent has vastly different experience or a different way of understanding the world?

Your thought questions

Your thought questions

- Lots of questions about applications of the RL framework:

Your thought questions

- Lots of questions about applications of the RL framework:
  - Emotions

Your thought questions

- Lots of questions about applications of the RL framework:
  - Emotions
  - Malware design (state?)

Your thought questions

- Lots of questions about applications of the RL framework:
  - Emotions
  - Malware design (state?)
  - Drug addiction

Your thought questions

- Lots of questions about applications of the RL framework:
  - Emotions
  - Malware design (state?)
  - Drug addiction
  - humor and joke telling (reward, prior knowledge, good data?)

Sample thought questions about MC

Sample thought questions about MC

- How does non-stationarity of the environment impact MC methods?

Sample thought questions about MC

- How does non-stationarity of the environment impact MC methods?
- How well does MC handle massive state-spaces, say Chess with 10^46 states?

Sample thought questions about MC

- How does non-stationarity of the environment impact MC methods?
- How well does MC handle massive state-spaces, say Chess with 10^46 states?
- What aspects of the methods in Ch5 seem particularly bad for scaling?

Sample thought questions

Sample thought questions

- Can we change the behavior policy in off-policy learning?

Sample thought questions

- Can we change the behavior policy in off-policy learning?

  - what if its epsilon-greedy?

Sample thought questions

- Can we change the behavior policy in off-policy learning?
  - what if its epsilon-greedy?
  - what would be the benefit of changing it?

Sample thought questions

- Can we change the behavior policy in off-policy learning?
  - what if its epsilon-greedy?
  - what would be the benefit of changing it?
- How do MC methods relate to evolutionary / genetic methods?

Sample thought questions

- Can we change the behavior policy in off-policy learning?
  - what if its epsilon-greedy?
  - what would be the benefit of changing it?
- How do MC methods relate to evolutionary / genetic methods?
  - here we learn value functions

Sample thought questions

- Can we change the behavior policy in off-policy learning?
  - what if its epsilon-greedy?
  - what would be the benefit of changing it?
- How do MC methods relate to evolutionary / genetic methods?
  - here we learn value functions
  - probably would need multiple episodes before changing the policy

Sample thought questions

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?
- How would an off-policy control method's performance compare with an on-policy method?

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?
- How would an off-policy control method's performance compare with an on-policy method?
  - assume the off-policy method learns the optimal policy

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?
- How would an off-policy control method's performance compare with an on-policy method?
  - assume the off-policy method learns the optimal policy
  - the optimal policy might be more risky

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?
- How would an off-policy control method's performance compare with an on-policy method?
  - assume the off-policy method learns the optimal policy
  - the optimal policy might be more risky
  - the on-policy method takes it's own exploration into account

Sample thought questions

- What advantages do off-policy methods hold over on-policy methods?
- When do we switch to the target policy?
- How would an off-policy control method's performance compare with an on-policy method?
  - assume the off-policy method learns the optimal policy
  - the optimal policy might be more risky
  - the on-policy method takes it's own exploration into account
  - more about this in the next chapter …

Basic structure of ES MC policy iteration

- Initialize π and Q

1. loop until your are satisfied with π

   1.1. generate an episode **e** by following π, where:

      1.1.1. $s_1 \sim$ uniform(S); $a_1 \sim$uniform(A(s1)) // exploring start

      1.1.2. $a_{t>1} = \arg\max_a Q(s_t,a)$

   1.2. **for** each **s,a** in **e**

      1.2.1. update the Q function for each sample (average returns from s,a)

   1.3. ~~for each **s** in **e**~~

      1.3.1. ~~$\pi(s) = \arg\max_a Q(s,a)$~~

   1.4. goto 1

Basic structure of **ϵ**-soft policy iteration MC algorithm

- Initialize π and Q

1. loop until your are satisfied with π

    1.1. generate an episode **e**, by following π, where:

        1.1.1. $a_t \sim \pi(s_t| \ .)$

    1.2. **for** each **s,a** in **e**

        1.2.1. update the Q function for each sample (average returns from s,a)

    1.3. $$A^* \leftarrow \arg\max_a Q(s,a)$$
    $$\text{For all } a \in \mathcal{A}(s):$$
    $$\pi(a|s) \leftarrow \begin{cases} 1-\varepsilon+\varepsilon/|\mathcal{A}(s)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(s)| & \text{if } a \neq A^* \end{cases}$$

    1.6. goto 1

ϵ-greedy MC policy iteration

- Initialize π and Q

1. loop until your are satisfied with π

   1.1. generate an episode **e** by following π, where:

      1.1.1. a_t = ϵ-greedy(Q)

   1.2. **for** each **s,a** in **e**

      1.2.1. update the Q function for each sample (average returns from s,a)

   1.3. goto 1

// with pr ϵ return random action

// with pr 1-ϵ return $\arg\max_a Q(s,a)$

On-policy and Off-policy learning

- MC control with **ϵ**-soft policies is an example of an **on-policy** learning algorithm
  - the policy that is used to generate sample episodes is the same as the policy we are learning about— the **ϵ**-greedy policy
- Imagine we generate sample episodes using some **behavior policy**: b: S X A -> [0,1]
  - perhaps a policy that explores a lot, like uniform random action selection in each state
- The **target policy**, π: S X A -> [0,1], is **not necessarily** the policy used to generate sample episodes
  - π is the policy we wish to learn about:
    - for Policy Iteration MC exploring starts, that was π_*,
    - for MC control (with **ϵ**-soft policies) that was $\widetilde{\pi}_\star$

Off-policy learning

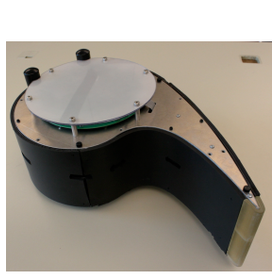
- What do we mean by learn **about** the target policy, π?
  - policy evaluation π: we learn the value function for π: v_π or q_π, OR
  - GPI: we the optimal policy π_*
  - all while generating episodes according to the behavior policy b
- This is called **off-policy** learning
- The special case where $\pi = b$ is called **on-policy** learning

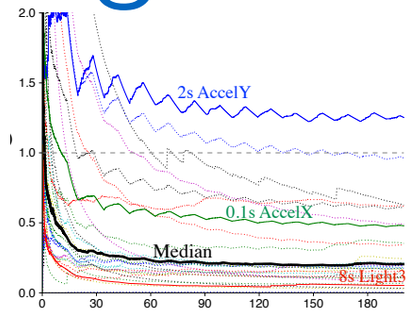
How does off-policy learning help for MC learning?

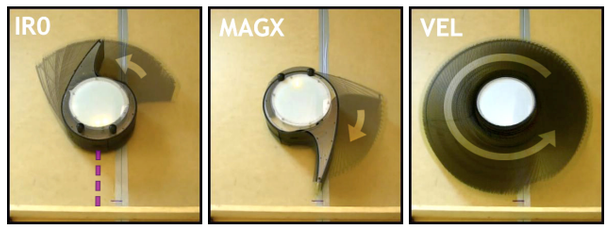
- It solves the problem of maintaining exploration in Monte Carlo GPI
  - we don't have to use unrealistic exploring starts
  - let b take care of exploration
    - just have to ensure:
      b(a|s) > 0 for every s,a that $\pi(a|s)>0$
  - we can learn π_* instead an ϵ-soft policy
- What famous algorithm is off-policy?

How else is off-policy learning useful?

- Massively parallel prediction learning: learn many V_π at once

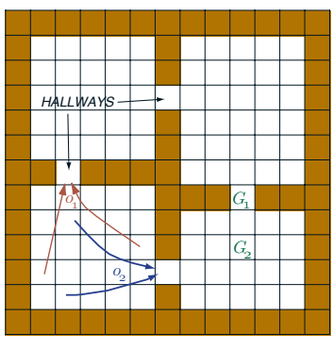






- Learn many policies in parallel

- Learn models of the world



- Learn from data generated by a human

- Learn in the batch RL setting

Using IS with MC methods

- How does this relate to the sample episodes, value functions, target and behavior policies?

- Todays lecture!

Off-policy of MC policy iteration

- Initialize π(target) & b(behavior) and **Q_π**

1. loop until your are satisfied with π or Q_π
   1.1. generate an episode **e** by following b:
      1.1.1. $a_t \sim b(s_t | .)$
   1.2. **for** each **s,a** in **e**
      1.2.1. update the **Q_π** function for each sample
         (but sample **s,a** are generated by following **b not π**)
         (simple average of returns is not statistically correct)
   1.3. optionally update/modify b
   1.4. goto 1

Importance sampling

- Most off-policy methods use importance sampling (IS)
- Imagine you want to compute the expected value of a random variable X with prob. mass func. **p**, **but** you can only generate samples according to some other pmf **f** != **p**

$$\mathbb{E}_p[X] = \sum_{i=1}^{k} p(x_i)x_i$$

$$\sum_{i=1}^{k} f(x_i)x_i \neq \mathbb{E}_p[X]$$

$$\sum_{i=1}^{k} \frac{p(x_i)}{f(x_i)} f(x_i)x_i = \sum_{i=1}^{k} p(x_i)x_i = \mathbb{E}_p[X]$$

- This process is called Importance sampling
- $\boldsymbol{\rho}_i = p(x_i)/f(x_i)$ is called the Importance sampling ratio.
- computing the same average of $\boldsymbol{\rho}_i * x_i$ is an unbiased estimator of $E_p[X]$

Importance sampling algorithm for sample mean

- Algorithm to estimate $E_p[X]$ with samples from f:

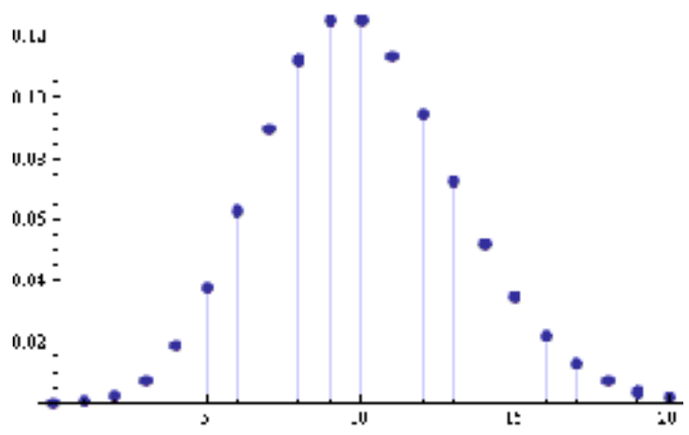
$$\frac{1}{N}\sum_{i=1}^{N}\rho_i * x_i = \frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{f(x_i)} * x_i$$

- Called Ordinary importance sampling

Importance sampling and variance

$$\frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{f(x_i)} * x_i$$

- Imagine pmf f(x) is very small in parts of the sample space where p(x) is large
  - that means $\boldsymbol{\rho}_i = p(x_i)/f(x_i)$ could be large
    - our estimator could suffer from high variance:
      - variance does not goto zero with more samples
      - variance can be much larger than the variance of X
    - Import. Samp. is unbiased, but high variance in practice



Weighted Importance Sampling (WIS)

$$\frac{1}{\sum_{k=1}^{N} \rho_k} \sum_{i=1}^{N} \rho_i * x_i = \frac{1}{\sum_{k=1}^{N} \frac{p(x_k)}{f(x_k)}} \sum_{i=1}^{N} \frac{p(x_i)}{f(x_i)} * x_i$$

- This weighted estimator of $E_p[X]$ is **biased**, but bias vanishes with infinite sampling (a consistent estimator)
- Variance is typically much smaller than the variance of **ordinary importance sampling**

IS for off-policy MC

- In Monte Carlo we want to estimate value functions
  - value functions are **expected** returns
  - we estimate value functions by averaging returns
- In off-policy Monte Carlo, we generate episodes according to a behavior policy b, but want to estimate v_π
- **Key idea:** Use IS to compute: $v_\pi(s) = E_\pi$[returns from s] by computing E_b[importance weighted returns from s]
- We can use OIS or WIS to estimate v_π
  - what is our importance sampling correction?
  - $\boldsymbol{\rho}$?

Importance sampling applied to returns

- ρ = {**prob** of sample under target distribution} / {**prob** of sample under sampling distribution}
- **target dist.** is π, and **sampling dist.** is b
- ρ = {probability of episode, *starting in s* under target policy} / {probability of episode under behavior policy}
- probability of an episode depends on policy and the transition dynamics of the world …

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

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

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

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

- This is called the *importance sampling ratio*

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

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

- This is called the *importance sampling ratio*
- All importance sampling ratios have expected value 1

Importance Sampling Ratio

- Probability of the rest of the trajectory, after S_t, under π:

$$\begin{aligned}
&\Pr\{A_t, S_{t+1}, A_{t+1}, \ldots, S_T \mid S_t, A_{t:T-1} \sim \pi\} \\
&\quad = \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \cdots p(S_T|S_{T-1}, A_{T-1}) \\
&\quad = \prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k),
\end{aligned}$$

- In importance sampling, each return is weighted by the relative probability of the trajectory under the two policies

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

- This is called the *importance sampling ratio*
- All importance sampling ratios have expected value 1

$$\mathbb{E}\left[\frac{\pi(A_k|S_k)}{b(A_k|S_k)}\right] \doteq \sum_a b(a|S_k)\frac{\pi(a|S_k)}{b(a|S_k)} = \sum_a \pi(a|S_k) = 1$$

Correcting a sample episode

- The importance weight applied to a sample episode g_t , which corrects for sampling according to b.

- Therefore policy evaluation for policy π using state values functions

  - V(s) = average (returns starting in s weighted by $\boldsymbol{\rho}_t$)

  - E [$\boldsymbol{\rho}_{t:T-1} G_t \mid S_t = s$] = $v_\pi(s)$

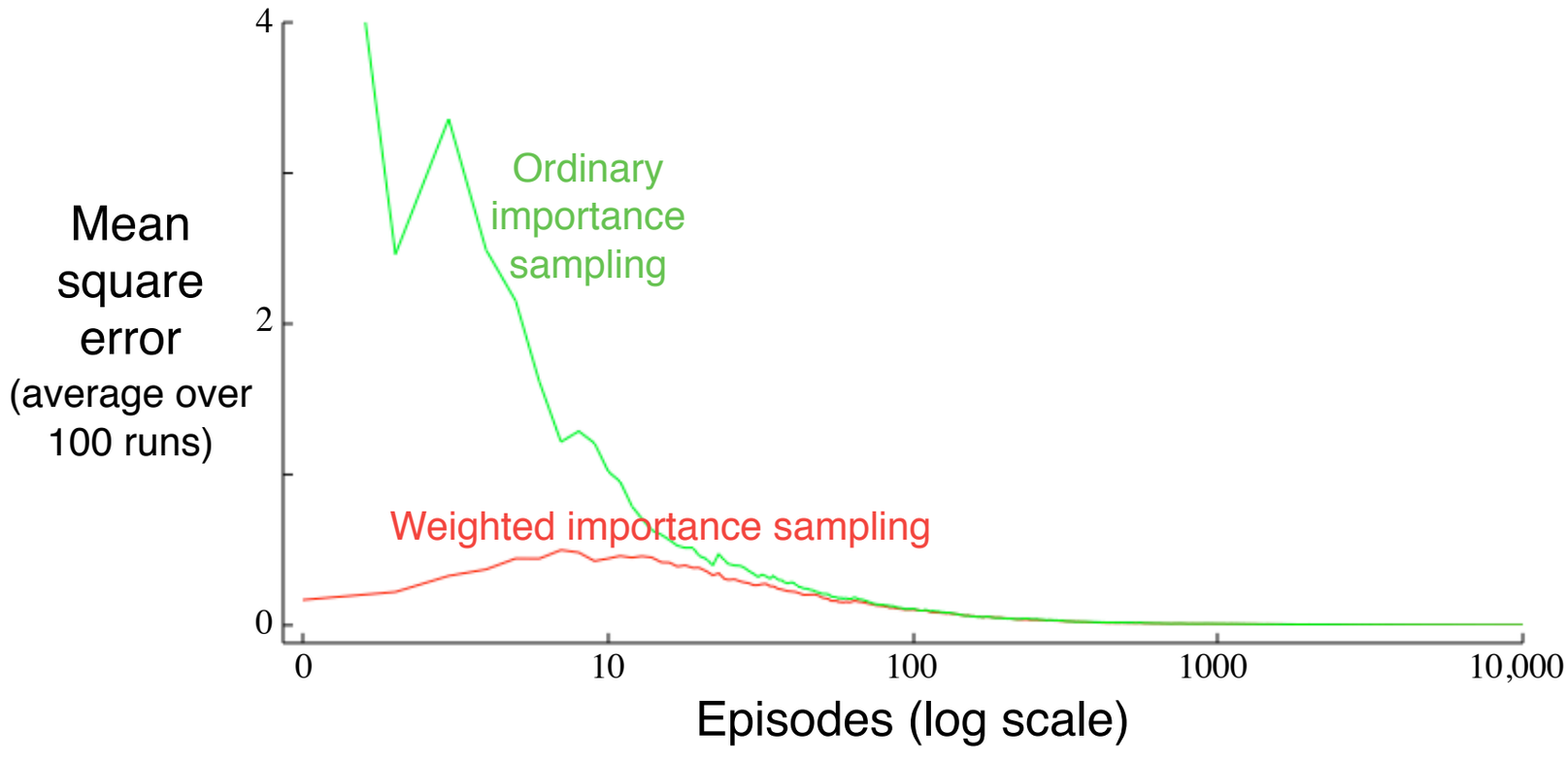
$$\rho_{t:T-1} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

off-policy policy evaluation
WIS vs OIS

- Blackjack estimate v_π for one state s
- s=[sum=13 (Ace,2), useable-ace, dealer has a two]
- π = stick on 20 or 21, otherwise hit
- b = hit or stick equally at random

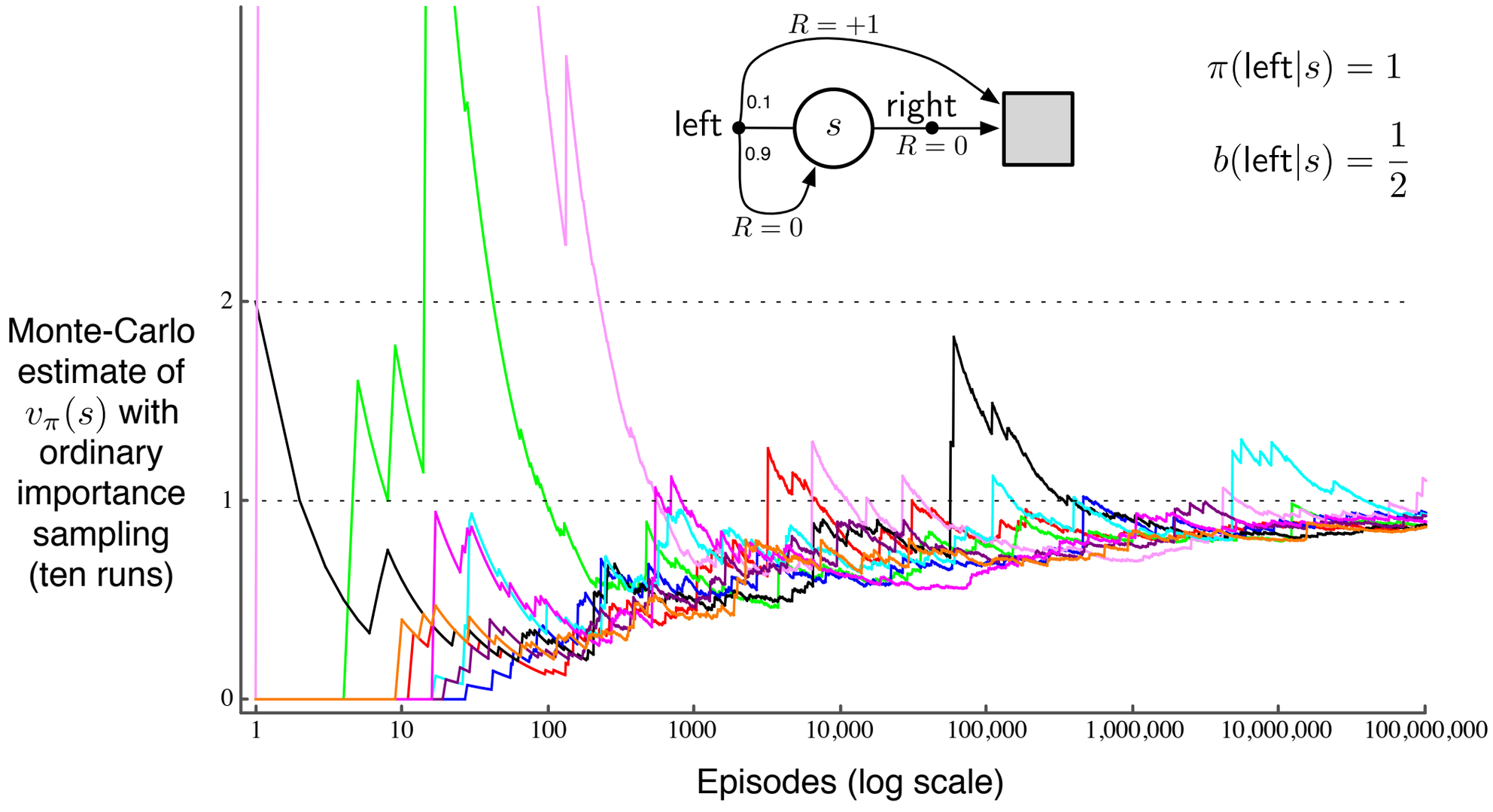
off-policy policy evaluation
WIS vs OIS

- Blackjack estimate v_π for one state s
- s=[sum=13 (Ace,2), useable-ace, dealer has a two]
- π = stick on 20 or 21, otherwise hit
- b = hit or stick equally at random

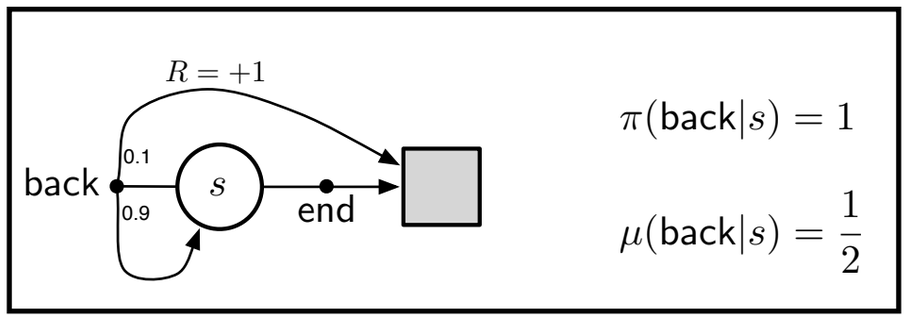


Instability of OIS

- First-visit MC policy evaluation, with OIS



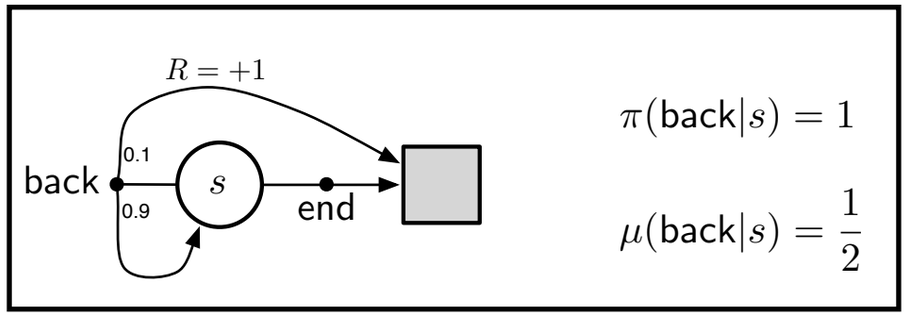
Infinite Variance of OIS



- Using μ to denote the behavior policy in this example

- The true value of state s is one; $v_\pi(s) = 1$; $E_\mu[\boldsymbol{\rho} G_0] = 1$
- But the variance of the importance weighted returns is infinite; $\text{Var}[\boldsymbol{\rho} G_0] = \infty$. How do we show this?

Infinite Variance of OIS



- Using μ to denote the behavior policy in this example
- The true value of state s is one; $v_\pi(s) = 1$; $E_\mu[\boldsymbol{\rho} G_0] = 1$
- But the variance of the importance weighted returns is infinite; $\text{Var}[\boldsymbol{\rho} G_0] = \infty$. How do we show this?

$$\text{Var}[X] \doteq \mathbb{E}\Big[\big(X - \bar{X}\big)^2\Big] = \mathbb{E}\big[X^2 - 2X\bar{X} + \bar{X}^2\big] = \mathbb{E}\big[X^2\big] - \bar{X}^2.$$

Infinite Variance of OIS

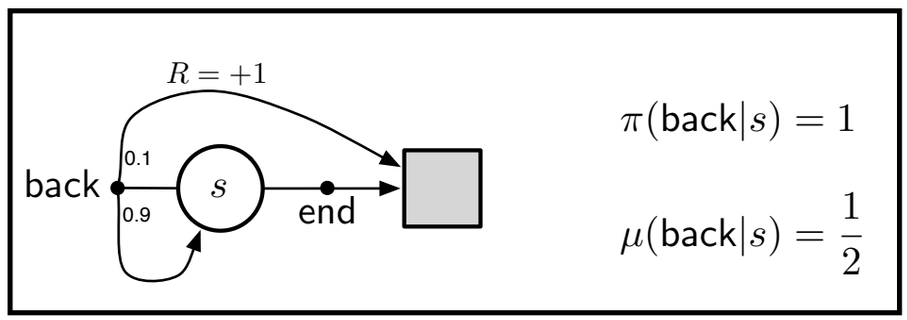
$$\mathbb{E}_\mu \left[\left(\prod_{t=0}^{T-1} \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} G_0 \right)^2 \right] =$$

$$\mu(\text{back}|s)p(\text{terminal}|s,\text{back}) \left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)} * 1 \right)^2 \qquad \text{(the length 1 episode)}$$

$$+ \mu(\text{back}|s)p(s|s,\text{back})\mu(\text{back}|s)p(\text{terminal}|s,\text{back}) \qquad \text{(the length 2 episode)}$$

$$\left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)} \frac{\pi(\text{back}|s)}{\mu(\text{back}|s)} * 1 \right)^2 + ...$$

- Using μ to denote the behavior policy in this example



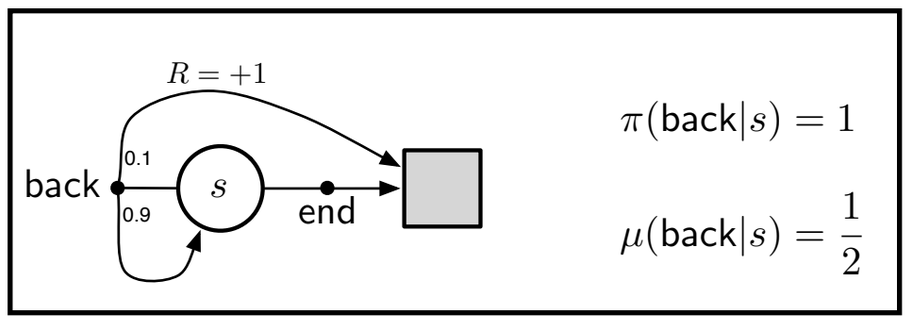
Infinite Variance of OIS

$$\mathbb{E}_\mu\left[\left(\prod_{t=0}^{T-1}\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}G_0\right)^2\right] =$$

$$\begin{aligned}
&\mu(\text{back}|s)p(\text{terminal}|s,\text{back})\left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}*1\right)^2 && \text{(the length 1 episode)}\\
&+\mu(\text{back}|s)p(s|s,\text{back})\mu(\text{back}|s)p(\text{terminal}|s,\text{back}) && \text{(the length 2 episode)}\\
&\left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}*1\right)^2+\ldots
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\right)^2 && \text{(the length 1 episode)}\\
&+\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\frac{1}{0.5}\right)^2 && \text{(the length 2 episode)}\\
&+\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\frac{1}{0.5}\frac{1}{0.5}\right)^2 && \text{(the length 3 episode)}\\
&+\cdots
\end{aligned}$$

- Using μ to denote the behavior policy in this example

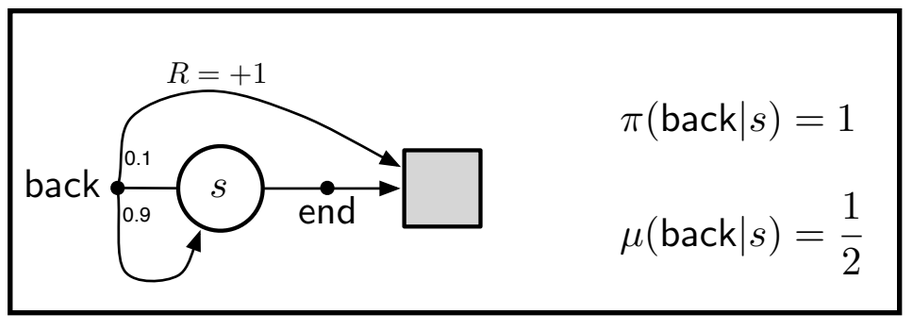


Infinite Variance of OIS

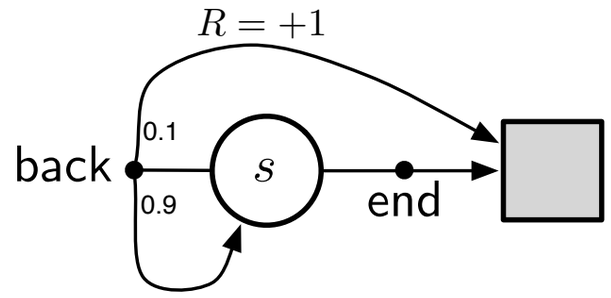
$$\mathbb{E}_\mu\left[\left(\prod_{t=0}^{T-1}\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}G_0\right)^2\right] =$$

$$\begin{aligned}
&\mu(\text{back}|s)p(\text{terminal}|s,\text{back})\left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}*1\right)^2 && \text{(the length 1 episode)}\\
&+\mu(\text{back}|s)p(s|s,\text{back})\mu(\text{back}|s)p(\text{terminal}|s,\text{back}) && \text{(the length 2 episode)}\\
&\left(\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}\frac{\pi(\text{back}|s)}{\mu(\text{back}|s)}*1\right)^2+\ldots\\
&=\frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\right)^2 && \text{(the length 1 episode)}\\
&\quad+\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\frac{1}{0.5}\right)^2 && \text{(the length 2 episode)}\\
&\quad+\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.9\cdot\frac{1}{2}\cdot 0.1\left(\frac{1}{0.5}\frac{1}{0.5}\frac{1}{0.5}\right)^2 && \text{(the length 3 episode)}\\
&\quad+\cdots\\
&=0.1\sum_{k=0}^{\infty}0.9^k\cdot 2^k\cdot 2\\
&=0.2\sum_{k=0}^{\infty}1.8^k\\
&=\infty.
\end{aligned}$$

- Using μ to denote the behavior policy in this example



Infinite Variance



$$\pi(\text{back}|s) = 1$$

$$\mu(\text{back}|s) = \frac{1}{2}$$

$$\mathrm{Var}[X] \doteq \mathbb{E}\Big[\big(X - \bar{X}\big)^2\Big] = \mathbb{E}\big[X^2 - 2X\bar{X} + \bar{X}^2\big] = \mathbb{E}\big[X^2\big] - \bar{X}^2.$$

$$\mathbb{E}\left[\left(\prod_{t=0}^{T-1} \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} G_0\right)^2\right]$$

$$\begin{aligned}
&= \frac{1}{2} \cdot 0.1 \left(\frac{1}{0.5}\right)^2 \\
&\quad + \frac{1}{2} \cdot 0.9 \cdot \frac{1}{2} \cdot 0.1 \left(\frac{1}{0.5}\frac{1}{0.5}\right)^2 \\
&\quad + \frac{1}{2} \cdot 0.9 \cdot \frac{1}{2} \cdot 0.9 \cdot \frac{1}{2} \cdot 0.1 \left(\frac{1}{0.5}\frac{1}{0.5}\frac{1}{0.5}\right)^2 \\
&\quad + \cdots \\
&= 0.1 \sum_{k=0}^{\infty} 0.9^k \cdot 2^k \cdot 2 \\
&= 0.2 \sum_{k=0}^{\infty} 1.8^k \\
&= \infty.
\end{aligned}$$

- Using μ to denote the behavior policy in this example

Incremental every-visit MC policy-evaluation with WIS

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?
- How can we avoid multiple passes over each sample episode to compute each sample return?

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?
- How can we avoid multiple passes over each sample episode to compute each sample return?

- $Gt = R_{t+1} + \gamma G_{t+1}$

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?
- How can we avoid multiple passes over each sample episode to compute each sample return?
  - $G_t = R_{t+1} + \gamma G_{t+1}$
- Answer:

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?
- How can we avoid multiple passes over each sample episode to compute each sample return?
  - $Gt = R_{t+1} + \gamma G_{t+1}$
- Answer:
  - use incremental update rule for Q(s,a)

Incremental every-visit MC policy-evaluation with WIS

- How can we avoid adding to a list of returns for each state-action pair?
- How can we avoid multiple passes over each sample episode to compute each sample return?
  - $G_t = R_{t+1} + \gamma G_{t+1}$
- Answer:
  - use incremental update rule for Q(s,a)
  - process episode backwards from T-1 to first step (each return is an independent sample order does not matter)

Incremental every-visit MC policy-evaluation with WIS

Off-policy MC prediction, for estimating $Q \approx q_\pi$

Input: an arbitrary target policy π

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
$Q(s,a) \leftarrow$ arbitrary
$C(s,a) \leftarrow 0$

Repeat forever:
$b \leftarrow$ any policy with coverage of π
Generate an episode using b:
$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
$G \leftarrow 0$
$W \leftarrow 1$
For $t = T-1, T-2, \ldots$ downto 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 ExitForLoop

Incremental every-visit MC policy-evaluation with WIS

Off-policy MC prediction, for estimating $Q \approx q_\pi$

Input: an arbitrary target policy π

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
$Q(s,a) \leftarrow$ arbitrary
$C(s,a) \leftarrow 0$

Repeat forever:
$b \leftarrow$ any policy with coverage of π
Generate an episode using b:
$S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T, S_T$
$G \leftarrow 0$
$W \leftarrow 1$
For $t = T-1, T-2, \ldots$ downto 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 ExitForLoop

If π=b what is:

1. W=?
2. C=?
3. W/C=?

Off-policy MC Control

- Recall one benefit of off-policy is we can learn about a deterministic policy(greedy) while following an exploratory policy that samples all the actions
- Target policy (π) is greedy with respect to Q
- Behavior policy (b) is ϵ-soft (e.g., ϵ-greedy)
- Working backwards, the updates from each sample episode are cut-off when a non-greedy action is chosen. Because ρ becomes zero!!
  - learning from tails of episodes
  - Limited utility is b often takes non-greedy action
- We will revisit *Off-policy every-visit MC with WIS* later when we discuss off-policy temporal difference learning

Chapter 5 Summary

- Monte Carlo (MC) methods estimate value functions and optimal policies from **experience** or **sample episodes**
- **MC** methods simply average many returns that start in a state to estimate value functions
- These methods have 3 important advantages over DP methods
  1. learn from interaction with environment
  2. no probability model required
  3. Less harmed by violating the Markov Property (more later…)

     - they do not **bootstrap**

…Summary

- MC methods fit nicely into GPI
  - MC methods provide an alternative policy evaluation process
  - MC methods do the policy evaluation and policy improvement on an episode by episode basis
- MC methods face the challenge of **sufficient exploration**
  - handled with *exploring starts*, learning about ϵ-*soft policies*, or *off-policy* learning
- Off-policy learning is the process of learning **about** one policy while following another
  - this requires importance sampling to re-weight the sample returns; either ordinary importance sampling or weighted