# Monte-Carlo Methods
## to Learn MDPs and Policies

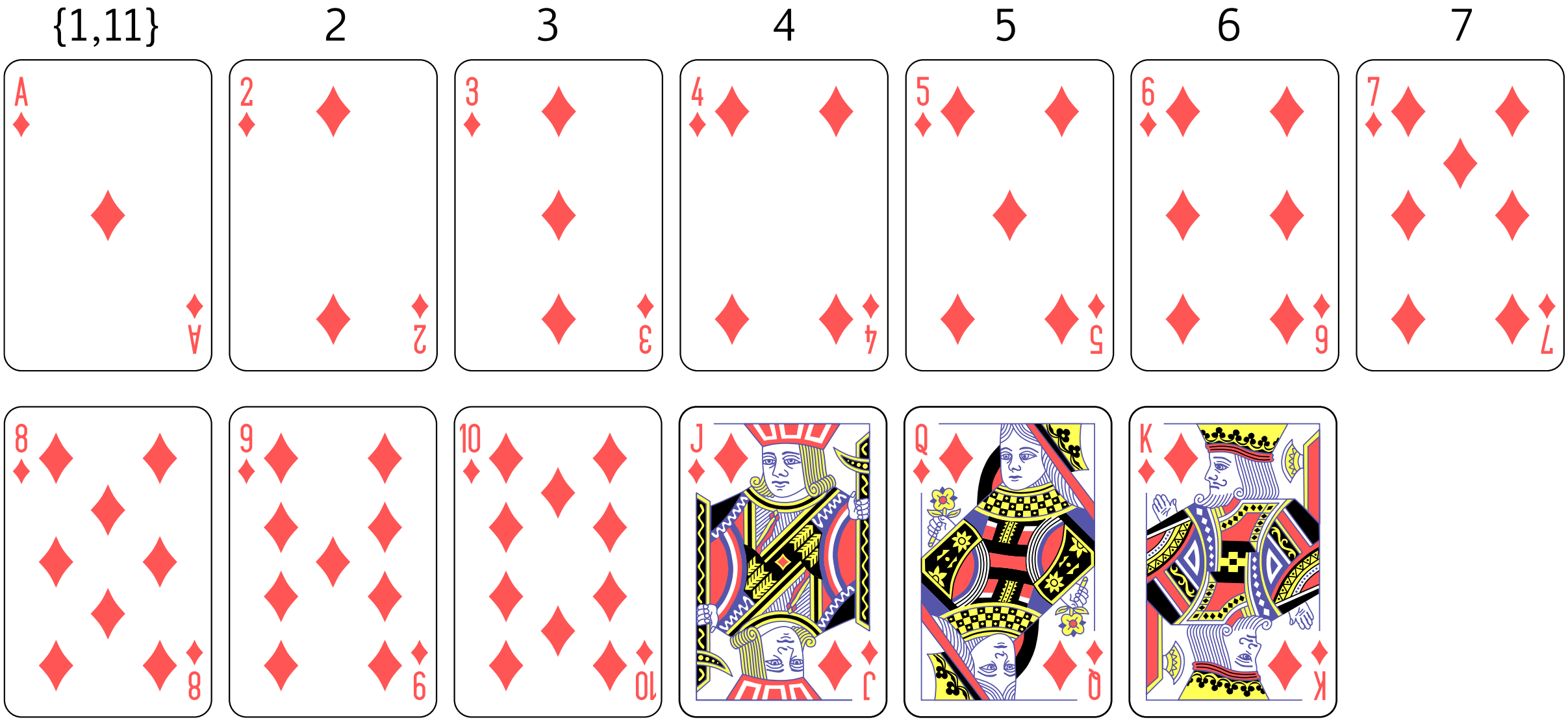
Prof. Matthias Grossglauser

Information and Network Dynamics (INDY) lab
School of Computer and Communication Sciences (I&C)
EPFL

# Recap

- So far, our discussion of RL was focused on the **(optimal) control** problem: given a perfectly known environment $p(s', r|s, a)$, find a policy $\pi(a|s)$ that optimizes long-term behavior
- Dynamic Programming: iterative method to evaluate a given policy $\pi$ (in terms of return = expected discounted cumulative reward), and to find optimal policies $\pi_*$
- Central notion: value functions for states and state-action pairs
  - $v_\pi(s)$ and $q_\pi(s, a)$: expected return starting in state $s$, and starting in $s$ and doing $a$
  - $v_*(s)$ and $q_*(s, a)$: best return starting in state $s$, and starting in $s$ and doing $a$
- Today we start exploring how to control an unknown system → we need to rely on interactions with environment to learn enough to device a good policy

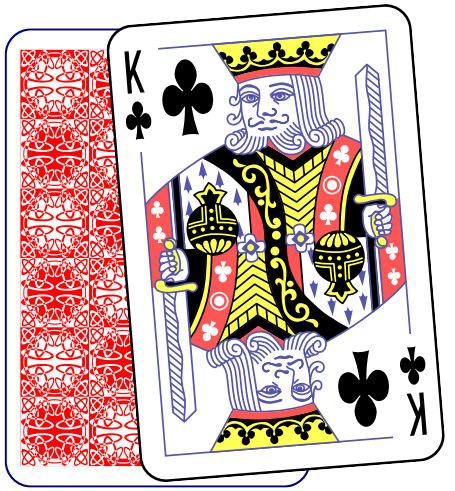
# Example 1: Blackjack



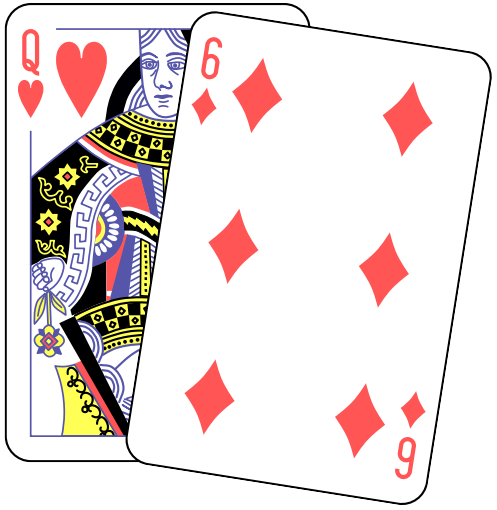
# Blackjack

- Game between dealer and player
- Both get two cards, one of dealer's face down
- Goal: win if own sum > sum of dealer's, but ≤ 21
  - Above 21: bust (lose, reward = -1)
  - Equal: draw (reward = 0)
- Ace: can count as 1 or 11
  - If there is an ace: if sum ≤ 21 with ace counting as 11 → "usable ace"
  - Usable ace: insurance policy, if player exceeds 21 → convert ace, subtract 10
- Player goes first: every round, hit (one more card) or stick (terminate)
- Then dealer follows fixed strategy: hit if ≤ 17

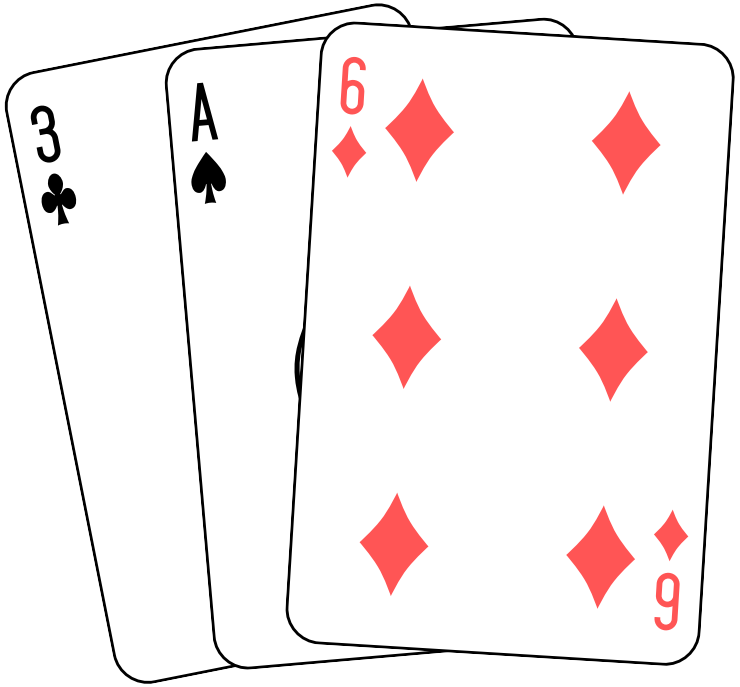
[10,21]



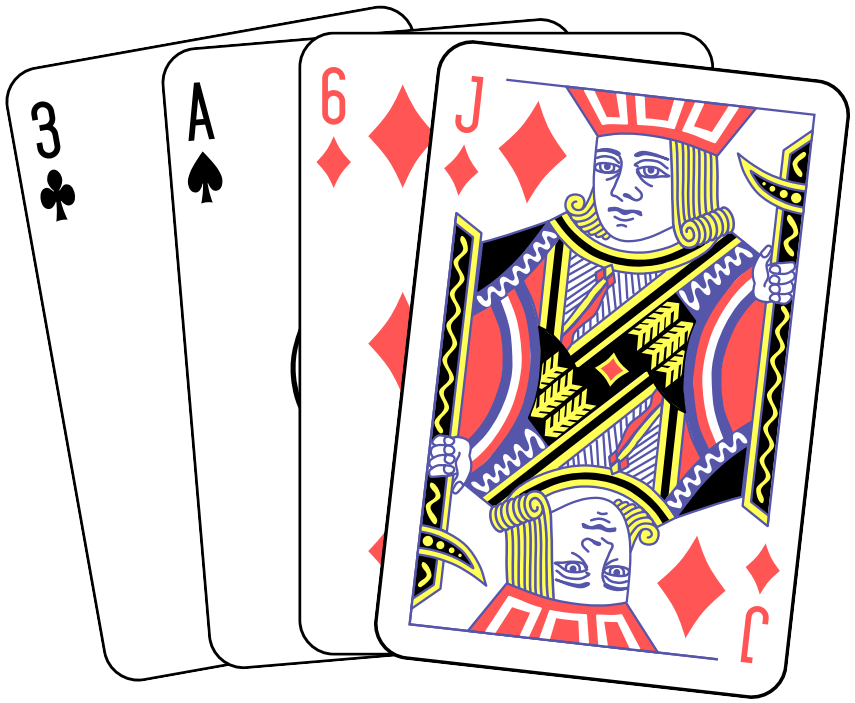
16



# Blackjack: usable vs unusable ace




20 or 10




20

# Blackjack: state and action space

- Player state: if sum < 11 → always hit, no risk of going bust
- Presence of usable ace for player → additional flexibility, should be more aggressive
- Dealer shows one card → 10 possibilities
- Total state space: 2 x 10 x 10 = 200

Dealer card showing

Player sum

| | A | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 11 | | | | | | | | | | |
| 12 | | | | | | | | | | |
| 13 | | | | | | | | | | |
| 14 | | | | | | | | | | |
| 15 | | | | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | | | | | | | | | | |
| 18 | | | | | | | | | | |
| 19 | | | | | | | | | | |
| 20 | | | | | | | | | | |

# Example 2: autopilot for a sailboat

- Environment can be difficult to model: fluid dynamics of air and water, friction, waves,… → learn from interaction!
- Sometimes simulation of a real system is within reach, but explicitly modeling $p(s', r|s, a)$ is not

# Model-free RL vs model-based RL

| Feature | Model-free (direct) | Model-based (indirect) |
|---|---|---|
| What is learned? | Values $v(s)$ or $q(s,a)$ | Dynamics $p(s',r\|s,a)$ |
| How is $\pi_*$ found? | Directly from values | DP (planning) using the learned model |
| Sample efficiency | Generally lower | Generally higher |
| Computational cost | Lower | Higher |
| Complexity | Simple schemes | Harder (model bias etc.) |

# Episodic MC for value estimation

- Estimates and policy change only between episodes
- Averaging complete returns from multiple episodes
- A bit like bandits: sampling returns instead of rewards
- But: multiple states → multiple bandits
  - But they are interrelated
- We will study the first-visit version of MC algorithms: basically the return is sampled once per state per episode
- Alternative: every-visit MC
  - At first sight, this would seem to make more sense: why ignore samples?
  - But actually even first-visit takes into account all the rewards until end of episode; it's just a question of different weighting

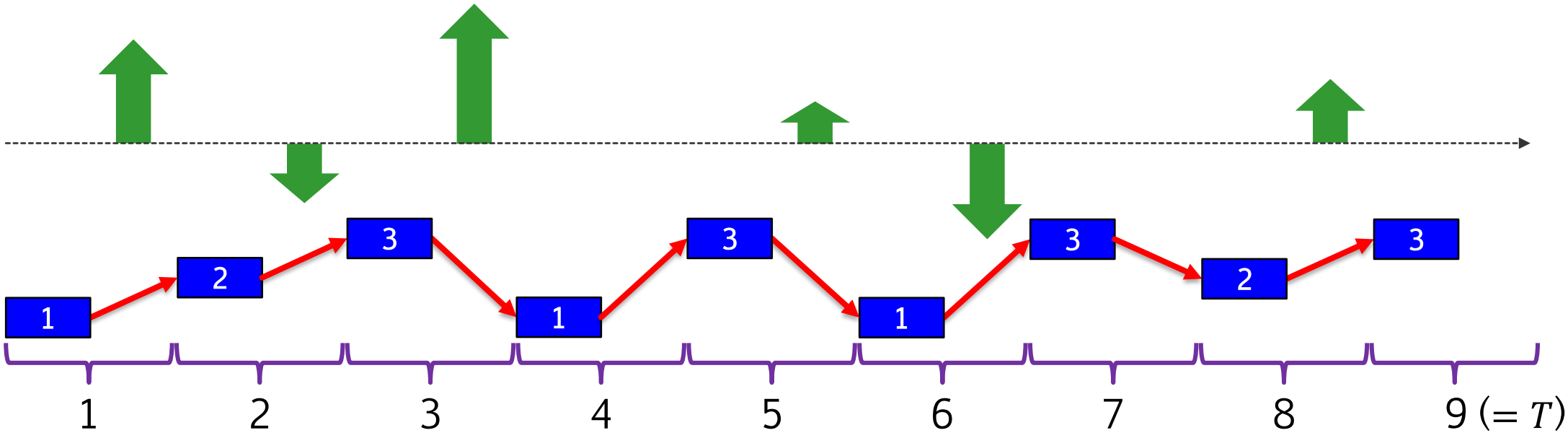
# MC value estimation

INPUT: policy $\pi$
Initialize:
    $v(s)$ arbitrary for $s \in S$, $v(\otimes) = 0$
    returns($s$) := [] /* empty list */

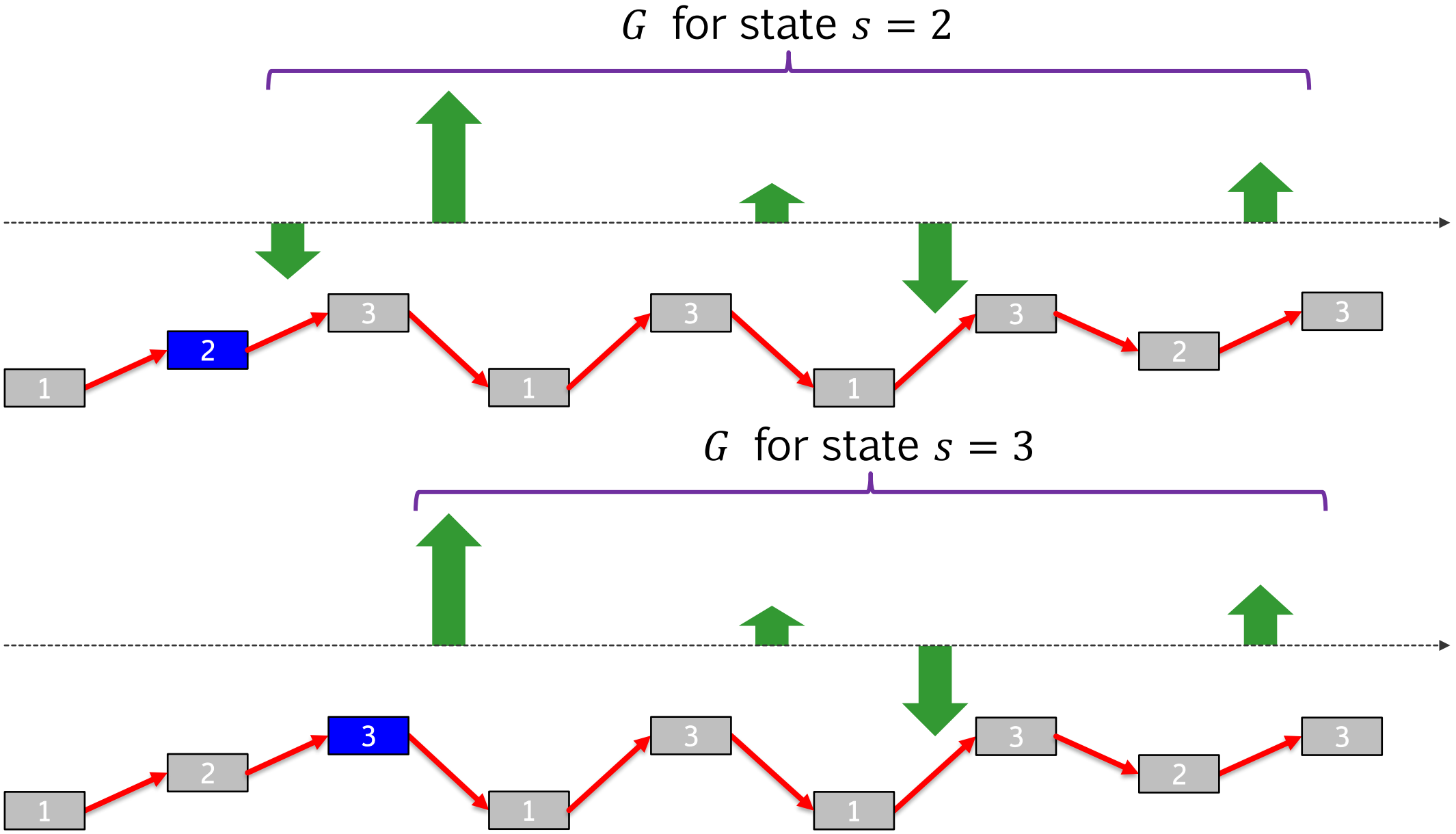
REPEAT:
    Read/generate a new episode: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G := 0$
    For $t := T-1, \ldots, 0$:
        $G := \gamma G + R_{t+1}$
        If $S_t \notin \{S_0, S_1, \ldots, S_{t-1}\}$:
            returns[$S_t$] = returns[$S_t$] + $G$
            $V(S_t) :=$ average (returns[$S_t$]) /* current avg */
UNTIL no-more-episodes

# MC value prediction (first-visit)

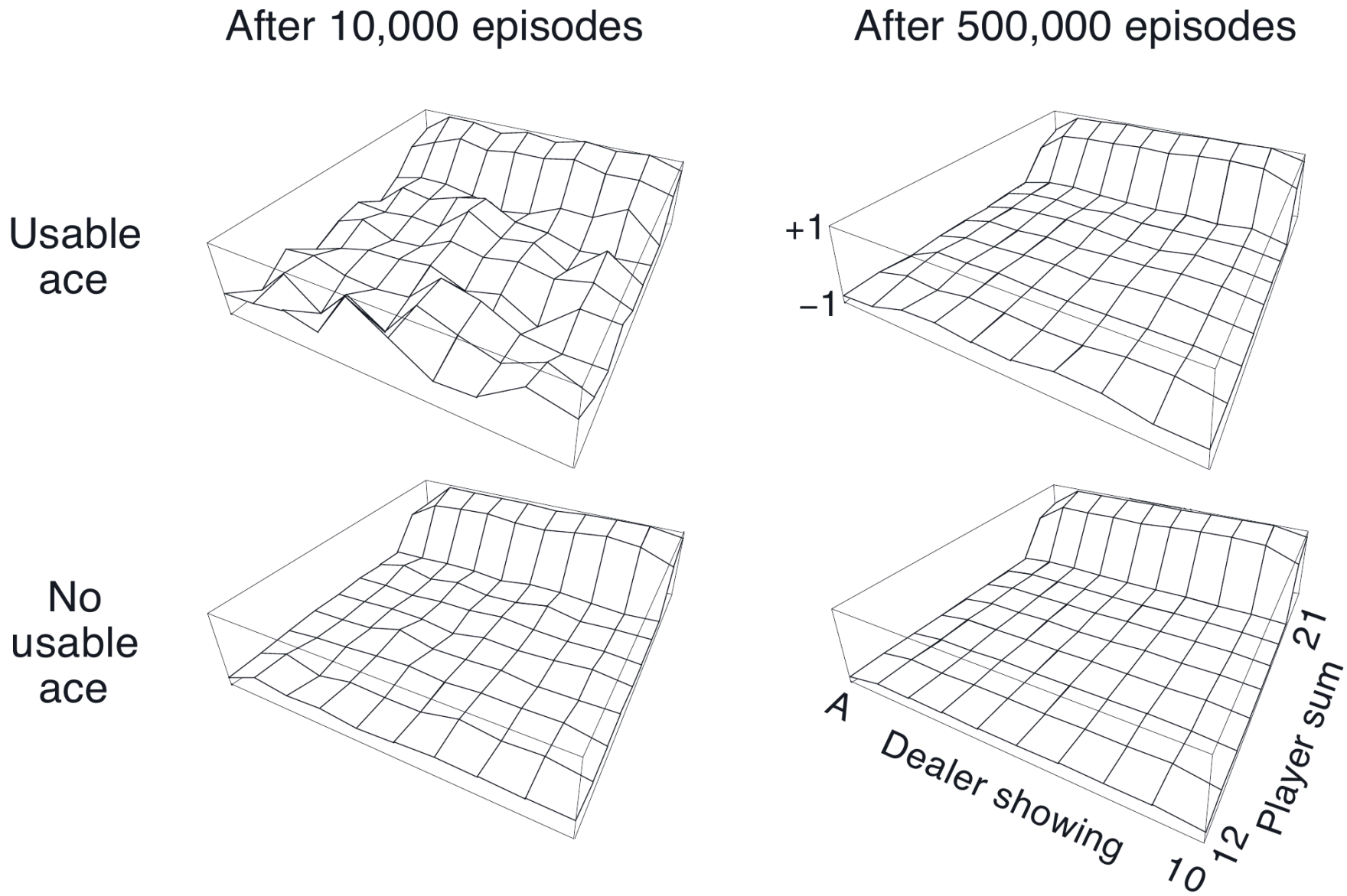
- Trajectory from one episode



# MC value prediction (first-visit)



# Blackjack: $\pi$ = "stick at 20 or 21"



# How to determine a good policy with MC prediction

- Note: with a model $p(s', r|s, a)$, $v(s)$ and $q(s, a)$ contain equivalent information:
  - $v_\pi(s) = \sum_a \pi(a|s)\, q_\pi(s, a)$
  - $q_\pi(s, a) = \sum_{s', r} p(s', r|s, a)[r + \gamma v_\pi(s')]$
- Once we know $v(s)$, compute $q(s, a)$ → greedy policy $\pi'$
- With MC this is not true: no model → no choice of the right action from state value alone
  - $q_\pi(s, a) = \sum_{s', r} p(s', r|s, a)[r + \gamma v_\pi(s')]$
- Ultimately, we will want to find good policies (not just value prediction of a given policy) → can we estimate $q_\pi(s, a)$ directly?

# First-visit MC estimation of action-value function

INPUT: policy $\pi$
Initialize:
$q(s,a)$ arbitrary for $s \in S$, $a \in \mathcal{A}$
returns$(s,a)$ := [] /* empty list */

REPEAT:
Read/generate a new episode: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, \ldots, S_{T-1}, A_{T-1}, R_T$
$G := 0$
For $t := T-1, \ldots, 0$:
$G := \gamma G + R_{t+1}$
If $(S_t, A_t) \notin \{(S_0, A_0), (S_1, A_1), \ldots, (S_{t-1}, A_{t-1})\}$:
returns$[(S_t, A_t)]$ = returns$[(S_t, A_t)]$ + $G$
$q(S_t, A_t) :=$ average (returns$[(S_t, A_t)]$) /* current avg */
UNTIL no-more-episodes

# Blackjack: why not apply DP directly?

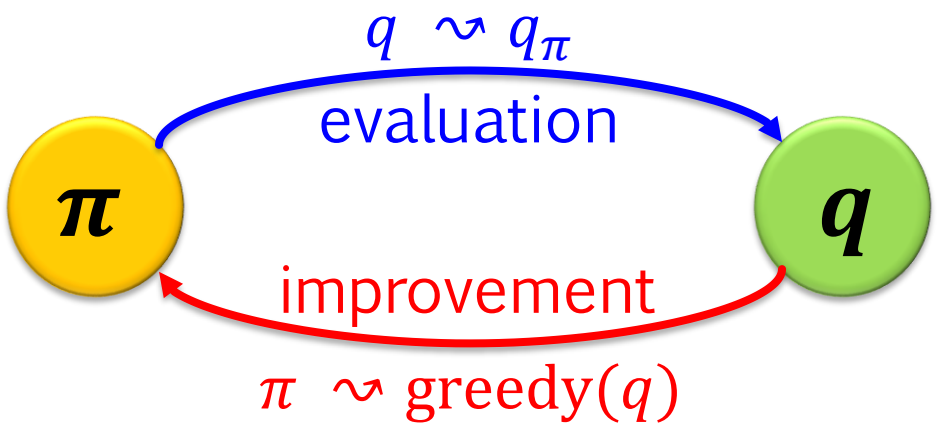
- The game is fully specified in probabilistic terms → we cannot we compute $p(s', r|s, a)$ and just find the optimal policy via DP?
- The full specification is deceptive: it does not mean we can easily find $p(s', r|s, a)$
- For example, suppose the player sum is 14, player chooses to stick → MDP goes into terminal state → reward depends on dealer card + sequence of cards the dealer draws; this is a complex probabilistic calculation
- On the other hand: very easy to simulate Blackjack from any desired state

# MC estimation of action-value function

- Problem: many pairs $(s, a)$ may be visited rarely (or never)
  - In particular: for a deterministic policy $\pi$, we can observe only one action per state
  - No information on a pair $(s, a)$ → we can never decide to take that action
- We have seen this problem before: **insufficient exploration!**
  - Like permanently forgetting the optimal arm in bandits
- For policy evaluation to work: ensure every possible $(s, a)$ gets sampled from time to time
- One approach: **exploring starts**
  - At the start of an episode, sample initial $(S_0, A_0)$ such that every possible pair has positive probability
  - This works for Blackjack (simulate games from some given intermediate constellation), but not for a sailboat (you cannot reset the sea and wind and the boat to some desired state to see what happens next)

# From MC estimation to MC control

- Inspiration: generalized policy iteration (GPI): two loosely interlocked processes
  - Evaluation (prediction): improve value estimates from current policy
  - Improvement: update policy given current value estimates




- MC version of policy iteration (PI):
  - $\pi_0 \xrightarrow{\text{eval}} q_{\pi_0} \xrightarrow{\text{impr}} \pi_1 \xrightarrow{\text{eval}} q_{\pi_1} \xrightarrow{\text{impr}} \pi_2 \xrightarrow{\text{eval}} v_{\pi_2} \xrightarrow{\text{impr}} \ldots \xrightarrow{\text{impr}} \pi_* \xrightarrow{\text{eval}} q_* \xrightarrow{\text{impr}} \pi_*$
- The eval step is now done with MC
- Under two assumptions, this is easy:
  - Exploring starts (SE): ensure that we sample every possible $(s, a)$
  - $\infty$ (or very large) # of episodes for **every** eval step!

# Policy improvement with MC

- We estimate the action-value function $q(s,a)$ → no model needed for the greedy policy:
  - Simply choose $\pi(s) \coloneqq \arg\max_a q(s,a)$
- So $\pi_{k+1}$ is the greedy policy with respect to $q_{\pi_k}$
- Conditions of the policy improvement theorem (last week) is satisfied:

$$\begin{aligned} q_{\pi_k}(s, \pi_{k+1}(s)) &= q_{\pi_k}\left(s, \arg\max_a q_{\pi_k}(s,a)\right) \\ &= \max_a q_{\pi_k}(s,a) \\ &\geq q_{\pi_k}(s, \pi_k(s)) \\ &\geq v_{\pi_k}(s) \end{aligned}$$

- Policy improvement theorem ensures that:
  - Either $\pi_{k+1}$ is uniformly better than $\pi_k$
  - Or they are both optimal

# From policy improvement to value improvement with MC

- PI requires a large number of episodes for each policy-improvement step (because we need small error on $q_{\pi_k}$)
- Note: after each evaluation, we "throw away" the trajectories (the returns will be different under an updated policy)
- In DP:
  - We accelerated convergence by meshing updates to policy and value more finely
  - We used in-place updates (faster convergence and less space)
- Do the same here:
  - After every episode, update both $q(s,a)$ and $\pi$
  - Have $q(s,a)$ in-place: values mix estimates during iteration $k$ and $k+1$
- Note: this adds a new source of error: estimates for $q(s,a)$ combine empirical returns from all episodes... where different policies were in place!
- In practice: it always converges -- in theory: not formally proven!

# MC ES (exploring starts) control to estimate $\pi_*$

Initialize:

$\pi(s) \in \mathcal{A}$ arbitrarily

$q(s,a)$ arbitrarily for $s \in S$, $a \in \mathcal{A}$

returns$(s,a)$ := [] /* empty list */

REPEAT:

Choose $S_0, A_0$ randomly such that their support = $S \times \mathcal{A}$

Read/generate a new episode starting from $S_0, A_0$: $R_1, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G := 0$

For $t := T-1, \ldots, 0$:

$G := \gamma G + R_{t+1}$

If $(S_t, A_t) \notin \{(S_0, A_0), (S_1, A_1), \ldots, (S_{t-1}, A_{t-1})\}$:

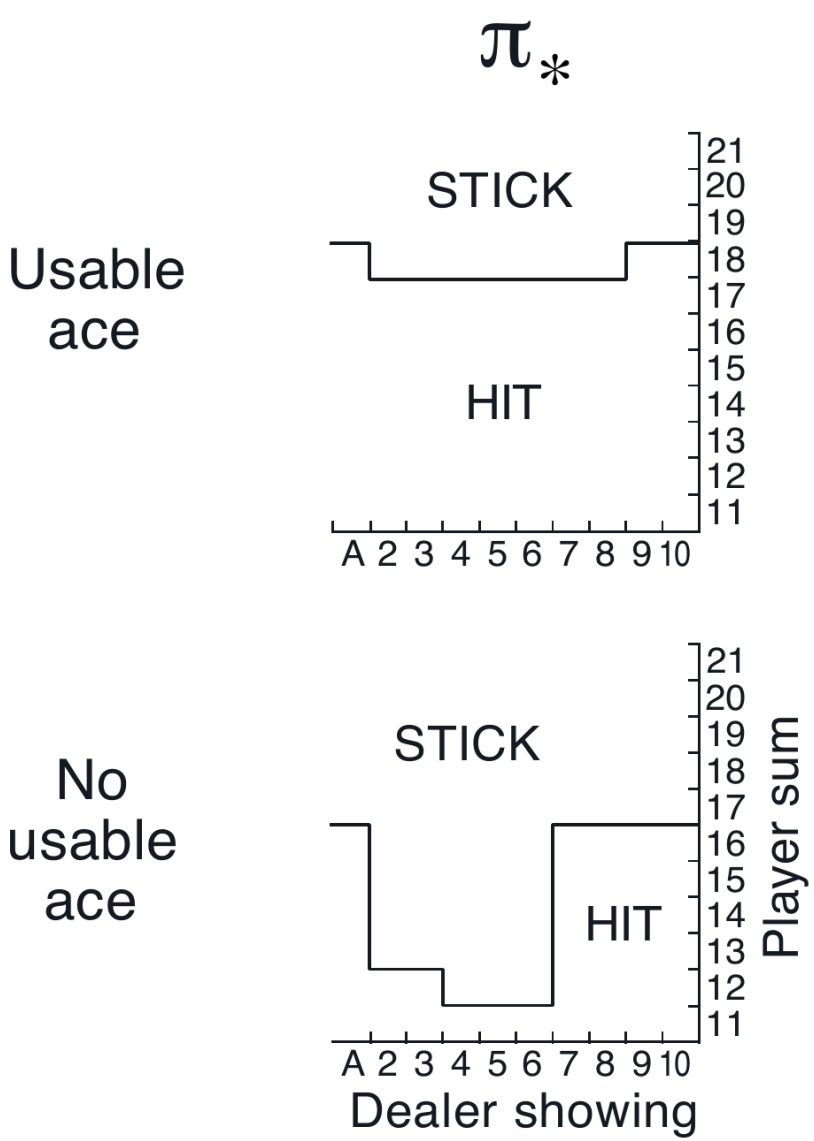
returns$[(S_t, A_t)]$ = returns$[(S_t, A_t)]$ + $G$

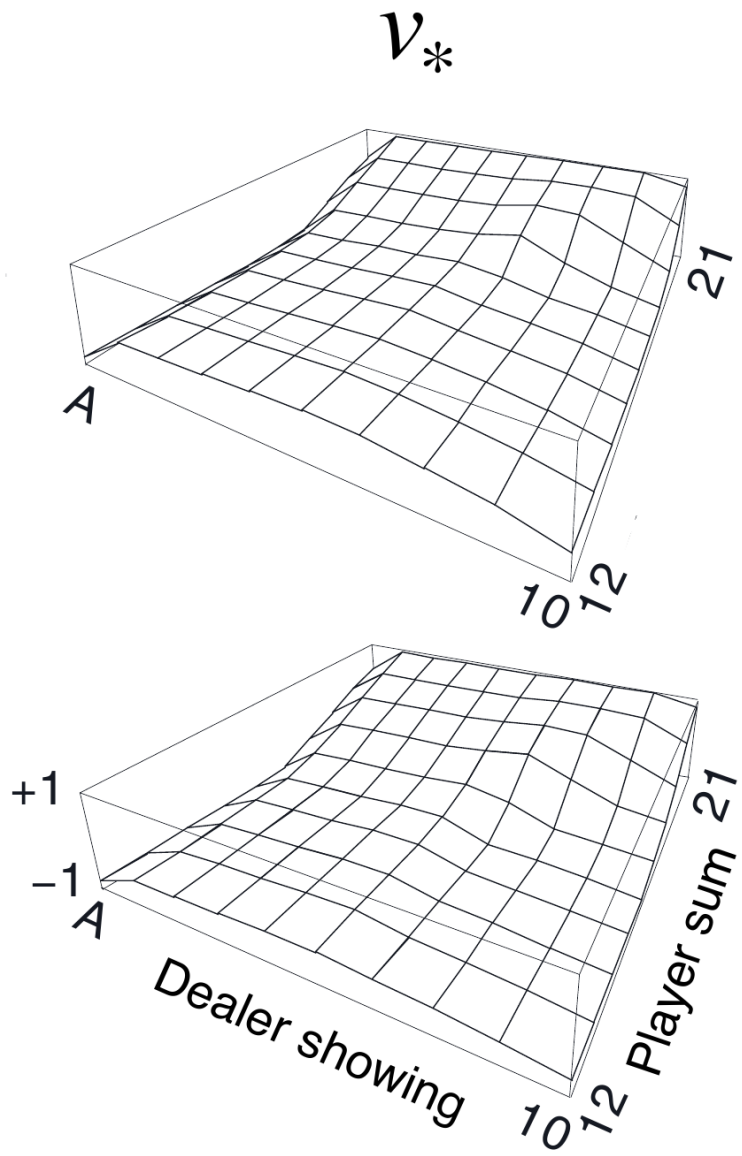
$q(S_t, A_t) :=$ average (returns$[(S_t, A_t)]$) /* current avg */

$\pi(S_t) := \arg\max_a q(S_t, a)$

UNTIL no-more-episodes

# Optimal policy for Blackjack



# Removing the exploring-starts assumption

- Goal: find another way of ensuring that all $(s, a)$ pairs are sampled infinitely often
- Two general approaches:
  - On-policy: evaluate and improve the actual policy that makes the decisions (like MC ES just seen)
  - Off-policy: one policy generates the data, another policy gets optimized
- On-policy: restrict to a soft policy with $\pi(s, a) > 0$ for all $s$ and $a$
  - Similar in spirit to bandits: play some arms even if our current estimates say they are not optimal
- One version: $\epsilon$-greedy
  - With probability $\epsilon$, choose an action uniformly at random
  - With probability $(1 - \epsilon)$, choose the action greedily (max current $q(s, a)$)
- In general: $\epsilon$-soft policy → $\pi(a|s) \geq \epsilon/|\mathcal{A}|$ for all $(s, a)$

# On-policy MC control (no ES) to estimate $\epsilon$-soft $\pi_*$

INPUT: $\epsilon > 0$:
Initialize:
$\pi(s) \in \mathcal{A}$ arbitrarily, $q(s,a)$ arbitrarily for $s \in S$, $a \in \mathcal{A}$
returns$(s,a)$ := [] /* empty list */
REPEAT:
Read/generate a new episode following $\pi$: $S_0, A_0, R_1, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G := 0$
For $t := T-1, \ldots, 0$:
$G := \gamma G + R_{t+1}$
If $(S_t, A_t) \notin \{(S_0, A_0), (S_1, A_1), \ldots, (S_{t-1}, A_{t-1})\}$:
returns$[(S_t, A_t)]$ = returns$[(S_t, A_t)]$ + $G$
$q(S_t, A_t) :=$ average (returns$[(S_t, A_t)]$) /* current avg */
$a_* := \arg\max_a q(S_t, a)$
FOR $a \in \mathcal{A}$:

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

UNTIL no-more-episodes

# $\epsilon$-greedy is best among all $\epsilon$-soft

- For $\epsilon = 0$: policy improvement theorem (PIT) says: deterministic greedy policy $\pi'$ w.r.t. $q_\pi$ improves $\pi$
- Theorem: for $\epsilon > 0$, $\epsilon$-greedy policy $\pi'$ w.r.t. $q_\pi$ improves $\epsilon$-soft policy $\pi$ (i.e., $\pi' \geq \pi$)
- Proof: conditions of PIT are satisfied here:

$$q_\pi\big(s, \pi'(s)\big) = \sum_a \pi'(a|s) q_\pi(s,a)$$

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

$$\geq \frac{\epsilon}{|\mathcal{A}|} \sum_a q_\pi(s,a) + (1-\epsilon) \sum_a \frac{\pi(a|s) - \frac{\epsilon}{|\mathcal{A}|}}{1-\epsilon} q_\pi(s,a)$$

$$= \frac{\epsilon}{|\mathcal{A}|} \sum_a q_\pi(s,a) - \frac{\epsilon}{|\mathcal{A}|} \sum_a q_\pi(s,a) + \sum_a \pi(a|s) q_\pi(s,a)$$

$$= v_\pi(s)$$

Weighted sum (with sum of weights = 1) $\leq$ largest element

# Equality means optimal within $\epsilon$-soft

- Theorem (part 2): If $\pi$ and $\pi'$ are equal, then they are optimal within the class of $\epsilon$-soft policies
- Proof:
  - Define a new environment, behaves like original except that policies are "$\epsilon$-softened"
  - New environment: in state $s$, with action $a$:
    - With probability $1-\epsilon$: same behavior as original
    - With probability $\epsilon$: sample an action $a'$ uniformly at random and execute that
  - Best general policy in new environment = best $\epsilon$-soft policy in original environment → we can find the optimal general policy for the modified environment instead

  - Define $\tilde{v}_*$ and $\tilde{q}_*$ the opt value functions for the new environment
  - Policy $\pi$ is optimal among $\epsilon$-soft class if $v_\pi = \tilde{v}_*$
  - But $\tilde{v}_*$ is the unique solution to the Bellman optimality equations (just with modified transition probabilities)

# Equality means optimal within $\epsilon$-soft

- $\tilde{v}_*(s) = \max_a \sum_{s',r} \left[(1-\epsilon)p(s',r|s,a) + \sum_{a'} \frac{\epsilon}{|\mathcal{A}|} p(s',r|s,a')\right][r + \gamma \tilde{v}_*(s')]$

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

Does not depend on $a$

- When we have equality, $\epsilon$-soft policy does not greedily improve any longer:

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

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

- Same equation system as above → unique solution ⟹ same solution

- Policy iteration works analogously to general case: within $\epsilon$-soft: $\epsilon$-greedy always improves until optimal

# Off-policy prediction

- Dilemma of on-policy:
  - Need to learn action values assuming everything after a decision is optimal
  - But to explore all actions, need to behave sub-optimally
- Question: can we learn an optimal policy while exploring according to another non-optimal policy?
  - I.e., exploring while learning a policy that exploits well
- On-policy: compromise between policy optimality and exploration
- Off-policy: two active policies:
  - **Target policy**: approximates the optimal policy
  - **Behavior policy**: generates the data

# Off-policy prediction

- We wish to estimate $v_\pi$ or $q_\pi$, but assuming that the trajectories are generated following some other policy $b(a|s)$ instead of $\pi(a|s)$
- We need an assumption: coverage

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

  i.e. every action taken under $\pi$ is sometimes taken under $b$
  - Policy $\pi$ may be stochastic or deterministic
  - But $b$ must be stochastic in any state $s$ where $b$ differs from $\pi$
- General technique to estimate values under one distribution when samples come from another distribution
- Idea: weight samples (returns) according to the relative probability of the trajectory occurring under $b$ and $\pi$

# Importance sampling

- Suppose we start in a state $S_t$
- Probability of a trajectory under policy $\pi$ is:

$$\mathbb{P}\left[A_t, S_{t+1}, A_{t+1}, \ldots, S_T | S_t, A_{t:T} \sim \pi\right]$$

$$= \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1}) \ldots p(S_T|S_{T-1}, A_{T-1})$$

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

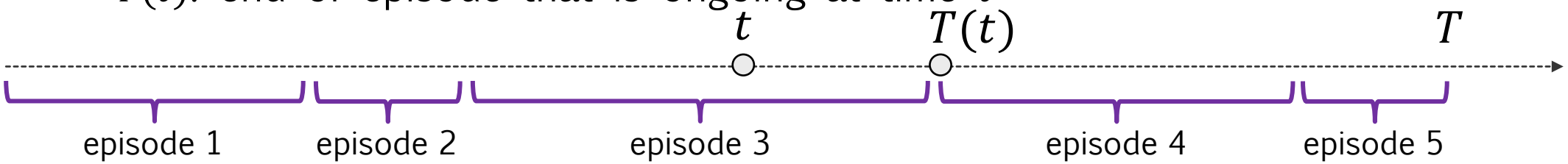
- Importance-sampling ratio:

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

- Depends only on $\pi$ and $b$, not on environment

# Importance sampling

- Note: $\mathbb{E}[G_t|S_t = s] = v_b(s)$
  but: $\mathbb{E}[\rho_{t:T-1}G_t|S_t = s] = v_\pi(s)$ bingo!
- Notation:
  - Time $t$ indexes all episodes
  - $T(t)$: end of episode that is ongoing at time $t$

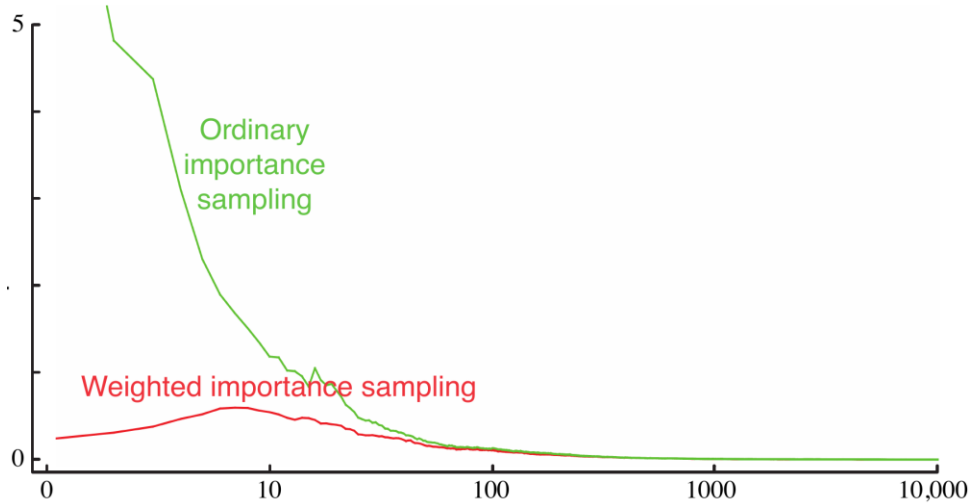



- Call $\mathcal{T}(s)$ the set of time steps where state $s$ is visited
  - First-time version: $\mathcal{T}(s)$ is first-visits of every period (if any)
  - All-time version: $\mathcal{T}(s)$ is all visits to state $s$
- Ordinary IS: $\hat{v}(s) \coloneqq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}G_t}{|\mathcal{T}(s)|}$ Weighted IS: $\hat{v}(s) \coloneqq \frac{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}G_t}{\sum_{t\in\mathcal{T}(s)} \rho_{t:T(t)-1}}$

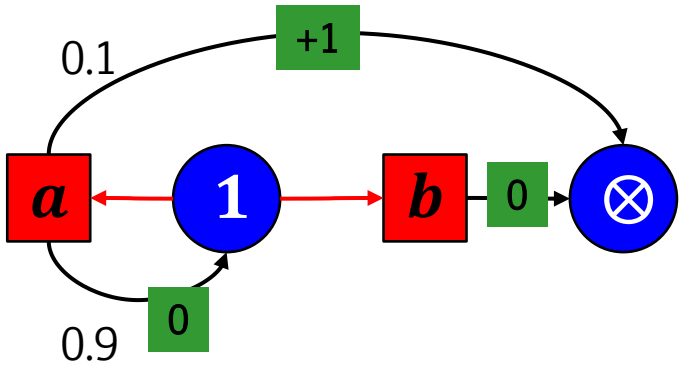
# Difference between ordinary and weighted IS

- First-visit version of **ordinary IS**:
  $\mathbb{E}[\hat{v}(s)] = v_\pi(s)$ good!
- First-visit version of **weighted IS**: suppose we had a single sample return
  $\mathbb{E}[\hat{v}(s)] \coloneqq \mathbb{E}_b[G_t|S = s] = v_b(s) \neq v_\pi(s)$ bias!
- WIS is a biased estimator, OIS is not → should we not just prefer OIS?
- No, because OIS has in general higher variance than WIS
  - In particular, for OIS, $\mathbb{V}[\hat{v}(s)] = \infty$ is possible even when returns are bounded!
- Therefore: in general bias of WIS is preferable over large variance of OIS
  - Often much faster convergence for WIS

Mean squared error for value estimate of one state
$\pi$: stick above 19
$b$: hit/stick randomly



# Example: high variance of OIS in a small MDP

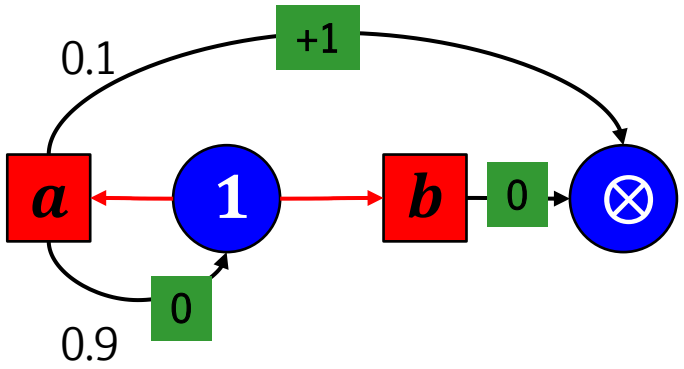



- Consider target policy $\pi(1) = a$, behavior policy $b(a|1) = \frac{1}{2}, b(b|1) = \frac{1}{2}$
- Estimate value $v_\pi(1) = \mathbb{E}[G_0|S_0 = 1]$ of state $s = 1$:
  - Note: $G_0 = 0$ if episode ends on action $b$, $G_0 = 1$ otherwise
- $\mathbb{V}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$
- Let $X$ be the ordinary IS estimator $\hat{v}(s)$:

$$\mathbb{E}[X^2] = \mathbb{E}_b\left[\left(\prod_{t=0}^{T-1}\frac{\pi(A_k|S_k)}{b(A_k|S_k)}G_0\right)^2\right]$$

- Decompose by episode length:
  - Any episode ending in action $b$: $\rho = 0$ because $\pi$ would never take this action → we only need to consider episodes that take $\geq 1$ actions $a$ → $G_0 = 1$

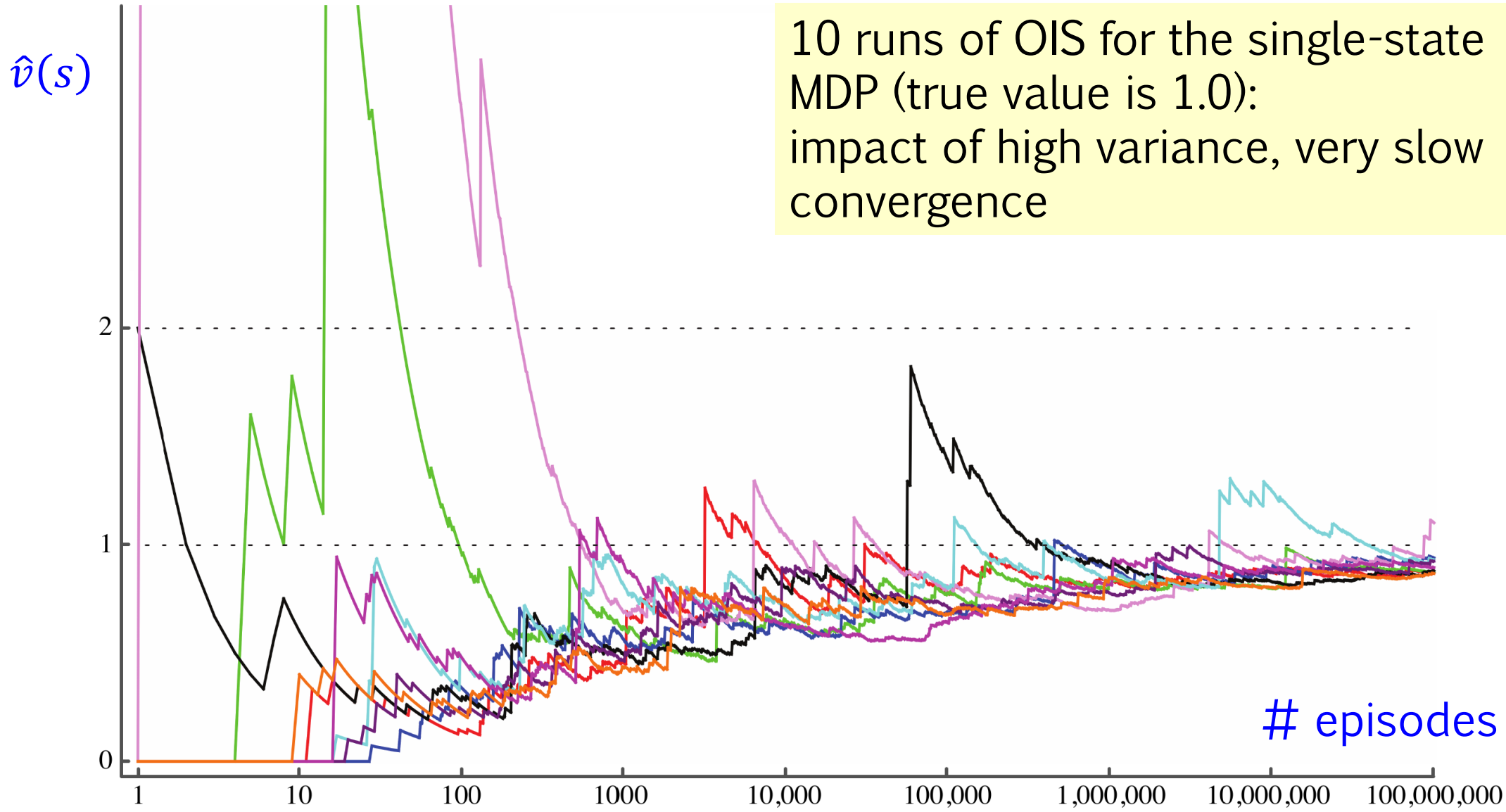
# Example: high variance of OIS in a small MDP




- $\mathbb{E}_b\left[\left(\prod_{t=0}^{T-1}\frac{\pi(A_k|S_k)}{b(A_k|S_k)}\right)^2\right] =$

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

# Example estimator convergence for ordinary IS




10 runs of OIS for the single-state MDP (true value is 1.0):
impact of high variance, very slow convergence

# Off-policy MC prediction (weighted IS)

INPUT: Arbitrary target policy $\pi$

Initialize: for all $(s, a)$:

$q(s, a)$ arbirary

$c(s, a) \coloneqq 0$

REPEAT:

$b \coloneqq$ any policy with coverage of $\pi$

Read/generate a new episode following $b$: $S_0, A_0, R_1, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \coloneqq 0$

$W \coloneqq 1$

For $t \coloneqq T-1, \ldots, 0$:

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

$c(S_t, A_t) \coloneqq c(S_t, A_t) + W$

$q(S_t, A_t) \coloneqq q(S_t, A_t) + \frac{W}{c(S_t, A_t)} (G - q(S_t, A_t)$

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

UNTIL no-more-episodes

# Off-policy MC control (weighted IS)

Initialize: for all $(s, a)$:

$q(s, a)$ arbirary

$c(s, a) \coloneqq 0$

$\pi(s) \coloneqq \arg\max_a q(s, a)$

REPEAT:

$b \coloneqq$ any $\epsilon$-soft policy

Read/generate a new episode following $b$: $S_0, A_0, R_1, S_1, A_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \coloneqq 0$

$W \coloneqq 1$

For $t \coloneqq T-1, \ldots, 0$:

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

$c(S_t, A_t) \coloneqq c(S_t, A_t) + W$

$q(S_t, A_t) \coloneqq q(S_t, A_t) + \frac{W}{c(S_t, A_t)}(G - q(S_t, A_t)$

$\pi(S_t) \coloneqq \arg\max_a q(S_t, a)$

IF $A_t \neq \pi(S_t)$ THEN break from FOR loop

$W \coloneqq W \frac{1}{b(S_t|A_t)}$

UNTIL no-more-episodes

# Summary

- Now we're finally doing reinforcement learning: find good policies for unknown environments via experience
- We study model-free (direct) methods: no model of environment inside the agent – we just learn from experience which policies get good returns
- Some problems are like Blackjack: easy to simulate and start in a desired state; others are like sailboats: no control over initial conditions → difficult to do exploring starts
  - Exploring starts: ensuring coverage of all state-action pairs by starting episodes
- Off-policy: evaluating and optimizing one policy while behavior (data generation) is driven by another
  - Unbiased but high variance: ordinary IS (normalization = # samples)
  - Lower variance but biased: weighted IS (normalization = total weight of samples)
- Suggested reading: [S&B] chapter 5