Machine Learning for Large-Scale Data Analysis and Decision Making (MATH80629A)
Fall 2021

Week #14- **Summary**

Announcement

- **Office hour:** Today (hybrid) at 11:30-1 pm & Thursday (online) 12-1:30 pm
- **Project Presentation: in-person** on **December 6**
- Upload your slides/poster to Gradescope due **December 6**
- **Project Report: Group & Individual** due **December 20**

Today

- **Last Quiz** on Gradescope!
- Summary of Sequential decision making 2
- Q&A
- Hands on session

Quiz 6

Login to your Gradescope account

Brief recap

- **Markov Decision Processes (MDP)**
  - Offer a framework for sequential decision making

    $$\langle \mathsf{A}, \mathsf{S}, \mathsf{P}, \mathsf{R}, \gamma \rangle$$

  - **Goal:** find the **optimal policy**
    - Dynamic programming and several algorithms (e.g., VI,PI)

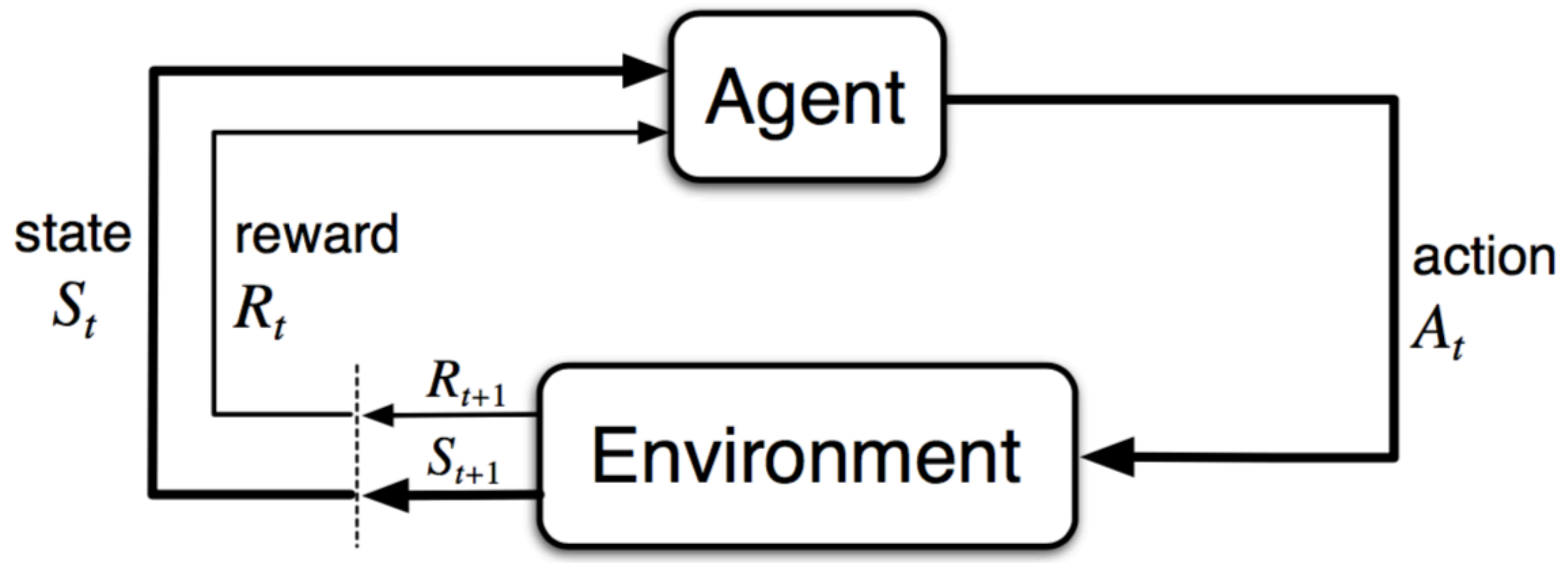


Figure 3.1, RL: An introduction

From MDPs to RL

- In MDPs we assume that we know

From MDPs to RL

- In MDPs we assume that we know

  1. **Transition probabilities:** $P(s' | s, a)$

From MDPs to RL

- In MDPs we assume that we know

1. **Transition probabilities:** $P(s' | s, a)$

2. **Reward function:** $R(s)$

From MDPs to RL

- In MDPs we assume that we know
  1. **Transition probabilities:** $P(s' \mid s, a)$
  2. **Reward function:** $R(s)$
- RL is more general

From MDPs to RL

- In MDPs we assume that we know

  1. **Transition probabilities:** $P(s' | s, a)$

  2. **Reward function:** $R(s)$

- RL is more general

  - **In RL both are typically unknown**

From MDPs to RL

- In MDPs we assume that we know

  1. **Transition probabilities:** $P(s' \mid s, a)$

  2. **Reward function:** $R(s)$

- RL is more general

  - **In RL both are typically unknown**

  - **RL agents navigate the world to gather this information**

Experience

A. **Supervised Learning:**

- Given fixed dataset
- Goal: maximize objective on test set (population)

B. **Reinforcement Learning**

- Collect data as agent interacts with the world
- Goal: maximize sum of rewards

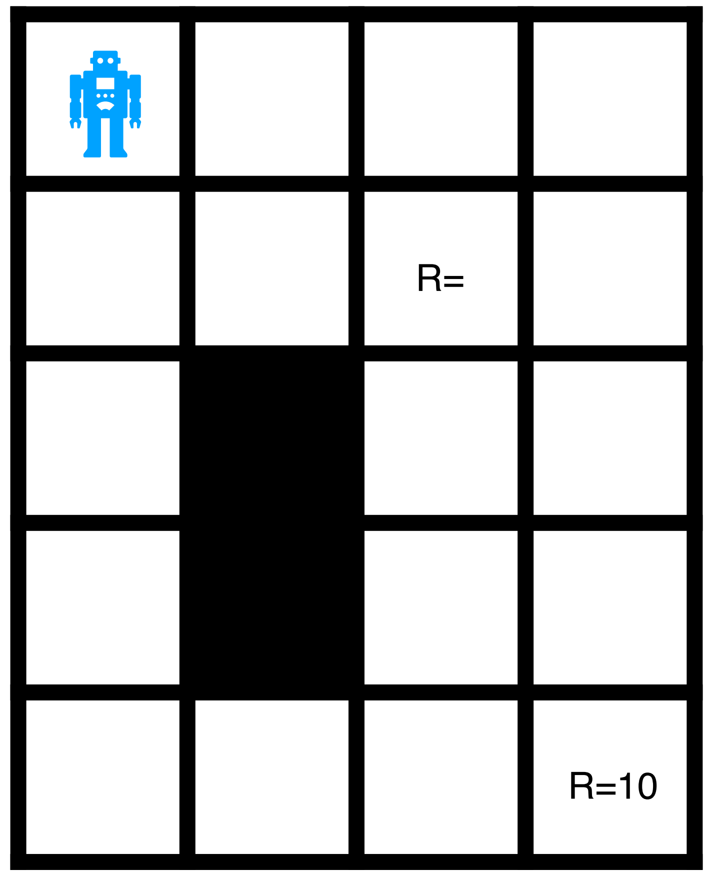
Challenges of reinforcement learning

Challenges of reinforcement learning

- **Credit assignment problem**: which action(s) should be credited for obtaining a reward
    - **A series of actions** (getting coffee from cafeteria)
    - **A small number of actions several time steps ago may be important** (test taking: study before, getting grade long after)

Challenges of reinforcement learning

- **Credit assignment problem**: which action(s) should be credited for obtaining a reward
  - **A series of actions** (getting coffee from cafeteria)
  - **A small number of actions several time steps ago may be important** (test taking: study before, getting grade long after)
- **Exploration/Exploitation tradeoff:** As agent interacts should it exploit its current knowledge (*exploitation*) or seek out additional information (*exploration*)



Application of RL

- **Robotics**
- **Video games**
- **Financial trading** is the buying and selling of financial assets.
- **Medical treatment/intervention** means the management and care of a patient to combat disease or disorder.
- A **self-driving car** is a vehicle that is capable of sensing its environment and moving safely with little or no human input.
- **Personalized tutoring** is an educational approach that aims to customize learning for each student's strengths, needs, skills, and interests.
- **Feed generation** is an automated platform that helps retailers build product feeds

Algorithm

Algorithm

- **Input:** an environment
  - actions, states, discount factor
  - starting state, method for obtaining next state

Algorithm

- **Input:** an environment
  - actions, states, discount factor
  - starting state, method for obtaining next state
- **Output:** an optimal policy

Algorithm

- **Input:** an environment
  - actions, states, discount factor
  - starting state, method for obtaining next state
- **Output:** an optimal policy
- **In practice:** need a simulator or a real environment for your agent to interact

Algorithms for RL

- **Two main classes of approach**

Algorithms for RL

- **Two main classes of approach**

  1. **Model-based**

     - Learns a model of the transition and uses it to optimize a policy given the model $P(s' \mid s, a)$

Algorithms for RL

- **Two main classes of approach**

  1. **Model-based**

     - Learns a model of the transition and uses it to optimize a policy given the model $P(s' \mid s, a)$

  2. **Model-free**

     - Learns an optimal policy without explicitly learning transitions π

Prediction vs. control

1. **Prediction:** evaluate a given policy
2. **Control:** Learn a policy

- Sometimes also called
  - **passive** (prediction)
  - **active** (control)

Monte Carlo Methods

MC for Prediction: First-visit Monte Carlo

- Given a fixed policy (**prediction**)

- Calculate the value function V(s) for each state

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

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:
    $G \leftarrow$ the return that follows the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow \text{average}(Returns(s))$

[Sutton & Barto, RL Book, Ch 5]

- Converges to $V_\pi(s)$ as the number of visits to each state goes to infinity

MC for Prediction: First-visit Monte Carlo

- Given a fixed policy (**prediction**)

- Calculate the value function V(s) for each state

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

Initialize:
  $\pi \leftarrow$ policy to be evaluated
  $V \leftarrow$ an arbitrary state-value function
  $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

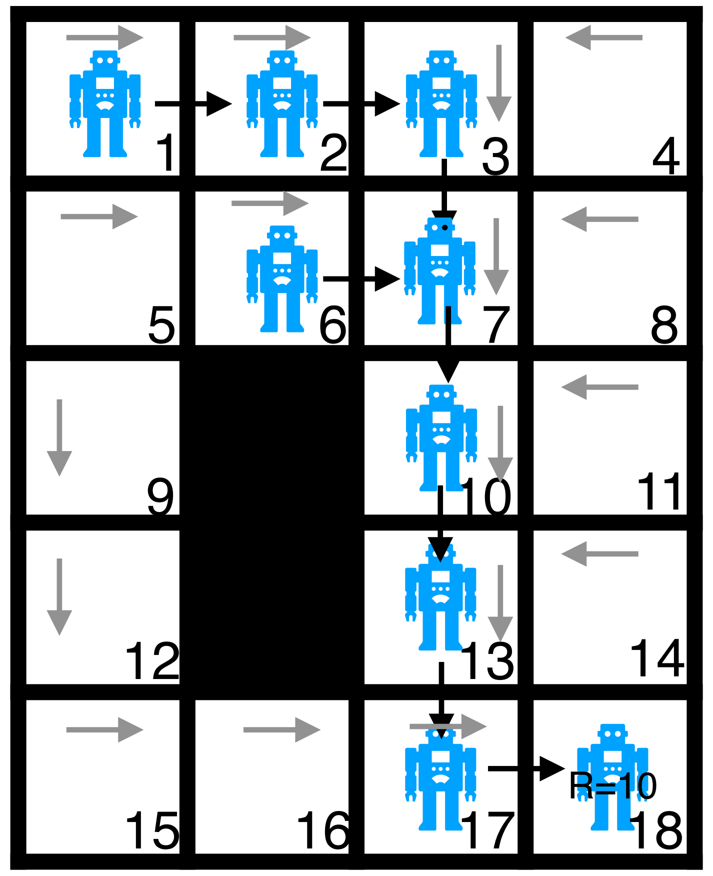
Repeat forever:
  Generate an episode using π
  For each state s appearing in the episode:
    $\boxed{G \leftarrow \text{the return that follows the first occurrence of } s}$
    Append G to $Returns(s)$
    $V(s) \leftarrow \text{average}(Returns(s))$

[Sutton & Barto, RL Book, Ch 5]

- Converges to $V_\pi(s)$ as the number of visits to each state goes to infinity

$$\boxed{V(s_t) = \max_{a_t} \left\{ R(s_t) + \gamma \sum_{s_{t+1}} P(s_{t+1} \mid s_t, a_t) V(s_{t+1}) \right\}}$$

Example: grid world



- **Start state is top-left (start of episode)**
- **Bottom right is absorbing (end of episode)**
- **Policy π is given (gray arrows)**

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

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:
$G \leftarrow$ the return that follows the first occurrence of s
Append G to $Returns(s)$
$V(s) \leftarrow \text{average}(Returns(s))$

Episode: $(1, \rightarrow) \rightarrow (2, \rightarrow) \rightarrow (3, \downarrow) \rightarrow \boxed{(7, \downarrow)} \rightarrow (6, \rightarrow) \rightarrow (7, \downarrow) \rightarrow (10, \downarrow) \rightarrow (13, \downarrow) \rightarrow (17, \rightarrow)$

For state 7:

$$\text{return}(7) = \gamma R(6) + \gamma^2 R(7) + \gamma^3 R(10) + \gamma^4 R(13) + \gamma^5 R(17) + \gamma^6 R(18)$$
$$= \gamma^6 10$$

$$V(7) = \gamma^6 * 10$$

Q-value function for control

- We know about state-value functions V(s)

Q-value function for control

- We know about state-value functions V(s)
  - If state transitions are known then they can be used to derive an optimal policy [recall value iteration]:

$$\boldsymbol{\pi}^*(\mathbf{s}) = \arg\max_{\mathbf{a}} \left\{ \mathbf{R}(\mathbf{s}) + \gamma \sum_{\mathbf{s}'} \mathbf{P}(\mathbf{s}' \mid \mathbf{s}, \mathbf{a}) \mathbf{V}^*(\mathbf{s}') \right\} \; \forall \mathbf{s}$$

Q-value function for control

- We know about state-value functions V(s)
  - If state transitions are known then they can be used to derive an optimal policy [recall value iteration]:

$$\boldsymbol{\pi}^*(\mathbf{s}) = \arg\max_{\mathbf{a}} \left\{ \mathbf{R}(\mathbf{s}) + \gamma \sum_{\mathbf{s}'} \mathbf{P}(\mathbf{s}' \mid \mathbf{s}, \mathbf{a}) \mathbf{V}^*(\mathbf{s}') \right\} \ \forall \mathbf{s}$$

  - When state transitions are unknown what can we do?

Q-value function for control

- We know about state-value functions V(s)
  - If state transitions are known then they can be used to derive an optimal policy [recall value iteration]:

$$\pi^*(s) = \arg\max_{a} \left\{ R(s) + \gamma \sum_{s'} P(s' \mid s, a) V^*(s') \right\} \ \forall s$$

  - When state transitions are unknown what can we do?
    - **Q(s,a)** the value function of a **(state,action) pair**

$$\pi^*(s) = \arg\max_{a} \{ Q^*(s, a) \} \ \forall s$$

MC for Control: Monte Carlo ES

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

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:
        $G \leftarrow$ the return that follows 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:
        $\pi(s) \leftarrow \text{argmax}_a\, Q(s,a)$

[Sutton & Barto,
RL Book, Ch.5]

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

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:
        $G \leftarrow$ the return that follows the first occurrence of s
        Append G to $Returns(s)$
        $V(s) \leftarrow \text{average}(Returns(s))$

MC for Control: Monte Carlo ES

[Sutton & Barto, RL Book, Ch.5]

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

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:
        $G \leftarrow$ the return that follows 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:
        $\pi(s) \leftarrow \arg\max_a Q(s, a)$

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

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:
        $G \leftarrow$ the return that follows the first occurrence of s
        Append G to $Returns(s)$
        $V(s) \leftarrow \text{average}(Returns(s))$

- **Strong reasons to believe that it converges to the optimal policy**

- **“Exploring starts” requirement may be unrealistic**

Monte Carlo without exploring starts (on policy)

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

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:
        $G \leftarrow$ the return that follows 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:
        $A^* \leftarrow \arg\max_a Q(s, a)$ (with ties broken arbitrarily)
        For all $a \in \mathcal{A}(s)$:

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

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

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:
        $G \leftarrow$ the return that follows 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:
        $\pi(s) \leftarrow \arg\max_a Q(s, a)$

[Sutton & Barto, RL Book, Ch.5]

Monte Carlo without exploring starts (on policy)

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

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
 $Q(s,a) \leftarrow$ arbitrary
 $Returns(s,a) \leftarrow$ empty list
 $\boxed{\pi(a|s) \leftarrow \text{an arbitrary } \varepsilon\text{-soft policy}}$

Repeat forever:
 (a) Generate an episode using π
 (b) For each pair s, a appearing in the episode:
  $G \leftarrow$ the return that follows 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:
  $A^* \leftarrow \arg\max_a Q(s,a)$ (with ties broken arbitrarily)
  For all $a \in \mathcal{A}(s)$:

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

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

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:
  $G \leftarrow$ the return that follows 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:
  $\pi(s) \leftarrow \arg\max_a Q(s,a)$

[Sutton & Barto, RL Book, Ch.5]

Monte Carlo without exploring starts (on policy)

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

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
  $Q(s,a) \leftarrow$ arbitrary
  $Returns(s,a) \leftarrow$ empty list
  $\boxed{\pi(a|s) \leftarrow \text{an arbitrary } \varepsilon\text{-soft policy}}$

Repeat forever:
  (a) Generate an episode using π
  (b) For each pair s, a appearing in the episode:
    $G \leftarrow$ the return that follows 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:
    $A^* \leftarrow \arg\max_a Q(s,a)$ (with ties broken arbitrarily)
    For all $a \in \mathcal{A}(s)$:

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

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

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:
    $G \leftarrow$ the return that follows 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:
    $\pi(s) \leftarrow \arg\max_a Q(s,a)$

[Sutton & Barto, RL Book, Ch.5]

Monte Carlo without exploring starts (on policy)

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

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
  $Q(s,a) \leftarrow$ arbitrary
  $Returns(s,a) \leftarrow$ empty list
  $\boxed{\pi(a|s) \leftarrow \text{an arbitrary } \varepsilon\text{-soft policy}}$

Repeat forever:
  (a) Generate an episode using π
  (b) For each pair s, a appearing in the episode:
    $G \leftarrow$ the return that follows 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:
    $A^* \leftarrow \arg\max_a Q(s,a)$ (with ties broken arbitrarily)
    For all $a \in \mathcal{A}(s)$:

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

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

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:
    $G \leftarrow$ the return that follows 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:
    $\pi(s) \leftarrow \arg\max_a Q(s,a)$

[Sutton & Barto, RL Book, Ch.5]

- **Policy value cannot decrease**

$$v_{\pi'}(s) \geq v_\pi(s), \forall s \in S$$

π : policy at current step
π' : policy at next step

Monte-Carlo methods summary

- Allow a policy to be learned through interactions
  - **(Does not learn transitions)**
- States are effectively treated as being independent
  - Focus on a subset of states (e.g., states for which playing optimally is of particular importance)
- Episodic (with or without exploring starts)

Temporal Difference

Temporal Difference (TD) Learning

Temporal Difference (TD) Learning

- **One of the "central ideas of RL"** [Sutton & Barto, RL book]

Temporal Difference (TD) Learning

- **One of the "central ideas of RL"** **[Sutton & Barto, RL book]**

- **Monte Carlo methods**

Observed returned : $G_t = \sum_{t}^{T} \gamma^t R(s_t)$

$$V'(s_t) = V(s_t) + \alpha[G_t - V(s_t)]$$

Step size

Temporal Difference (TD) Learning

- **One of the “central ideas of RL”** **[Sutton & Barto, RL book]**
- **Monte Carlo methods**

Observed returned :

$$G_t = \sum_{t}^{T} \gamma^t R(s_t)$$

$$V'(s_t) = V(s_t) + \boxed{\alpha}[G_t - V(s_t)]$$

Step size

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

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:
        $G \leftarrow$ the return that follows the first occurrence of s
        Append G to $Returns(s)$
        $V(s) \leftarrow$ average$(Returns(s))$

Temporal Difference (TD) Learning

- One of the “central ideas of RL” [Sutton & Barto, RL book]

- Monte Carlo methods

Observed returned : $G_t = \sum_{t}^{T} \gamma^t R(s_t)$

$$V'(s_t) = V(s_t) + \boxed{\alpha}[G_t - V(s_t)]$$

Step size

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

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:
    $G \leftarrow$ the return that follows the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow$ average$(Returns(s))$

- TD(0)
  - updates “instantly”

Temporal Difference (TD) Learning

- One of the “central ideas of RL” [Sutton & Barto, RL book]
- Monte Carlo methods

Observed returned : $G_t = \sum_{t}^{T} \gamma^t R(s_t)$

$$V'(s_t) = V(s_t) + \boxed{\alpha}[G_t - V(s_t)]$$

Step size

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

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:
    $G \leftarrow$ the return that follows the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow \text{average}(Returns(s))$

- TD(0)
  - updates “instantly”

$$V'(s_t) = V(s_t) + \alpha[\underbrace{R(s_t) + \gamma V(s_{t+1})}_{\approx G_t} - V(s_t)]$$

TD(0) for prediction VS TD for control

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
Initialize $V(s)$ arbitrarily (e.g., $V(s) = 0$, for all $s \in \mathcal{S}^+$)
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

[Sutton & Barto,
RL Book, Ch.6]

Q-learning for control

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Initialize $Q(s,a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
    Initialize S
    Repeat (for each step of episode):
        Choose A from S using policy derived from Q (e.g., ϵ-greedy)
        Take action A, observe R, S'
        $\boxed{Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]}$
        $S \leftarrow S'$
    until S is terminal

[Sutton & Barto, RL Book, Ch.6]

Q-learning for control

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$
    $S \leftarrow S'$
  until S is terminal

[Sutton & Barto, RL Book, Ch.6]

ϵ -greedy policy

$$a = \begin{cases} \arg\max_a Q(a, s) & \text{with probability } 1 - \epsilon, \\ \text{random } a & \text{with probability } \epsilon. \end{cases}$$

Q-learning for control

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Initialize $Q(s,a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
    $S \leftarrow S'$
  until S is terminal

[Sutton & Barto, RL Book, Ch.6]

ϵ -greedy policy

$$a = \begin{cases} \arg\max_a Q(a,s) & \text{with probability } 1-\epsilon, \\ \text{random } a & \text{with probability } \epsilon. \end{cases}$$

Q-learning for control

Q-learning $\boxed{\text{(off-policy TD control)}}$ for estimating $\pi \approx \pi_*$

Initialize $Q(s,a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
    Initialize S
    Repeat (for each step of episode):
        Choose A from S using policy derived from Q (e.g., ϵ-greedy)
        Take action A, observe R, S'
        $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
        $S \leftarrow S'$
    until S is terminal

[Sutton & Barto, RL Book, Ch.6]

ϵ -greedy policy

$$a = \begin{cases} \arg\max_a Q(a,s) & \text{with probability } 1-\epsilon, \\ \text{random } a & \text{with probability } \epsilon. \end{cases}$$

Q-learning for control

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Initialize $Q(s, a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

[Sutton & Barto, RL Book, Ch.6]

ϵ -greedy policy

$$\mathsf{a} = \begin{cases} \arg\max_{\mathsf{a}} \mathsf{Q}(\mathsf{a}, \mathsf{s}) & \text{with probability } 1 - \epsilon, \\ \text{random a} & \text{with probability } \epsilon. \end{cases}$$

Q-learning for control

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Initialize $Q(s,a)$, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
    $S \leftarrow S'$
  until S is terminal

[Sutton & Barto, RL Book, Ch.6]

- Converges to Q* as long as all (s,a) pairs continue to be updated and with minor constraints on learning rate

Comparing TD and MC

- MC requires going through full episodes before updating the value function. Episodic.
- Converges to the optimal solution

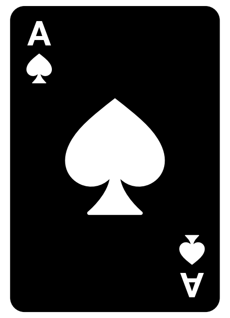
- TD updates each V(s) after each transition. Online.
- Converges to the optimal solution (some conditions on) α
- Empirically TD methods tend to converge faster

Practical difficulties

- **Compared** to **supervised learning** setting up an RL problem is often **harder**
  - Need an **environment** (or at least a simulator)
- **Rewards**
  - In some domains it's clear (e.g., in games)
  - In others it's much more subtle (e.g., you want to please a human)

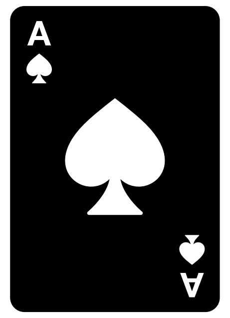
Hand on Session
+
Extra material
(Some will be used for this week’s exercises)

Black Jack



The most widely played casino banking game in the world, also known as Twenty-One.

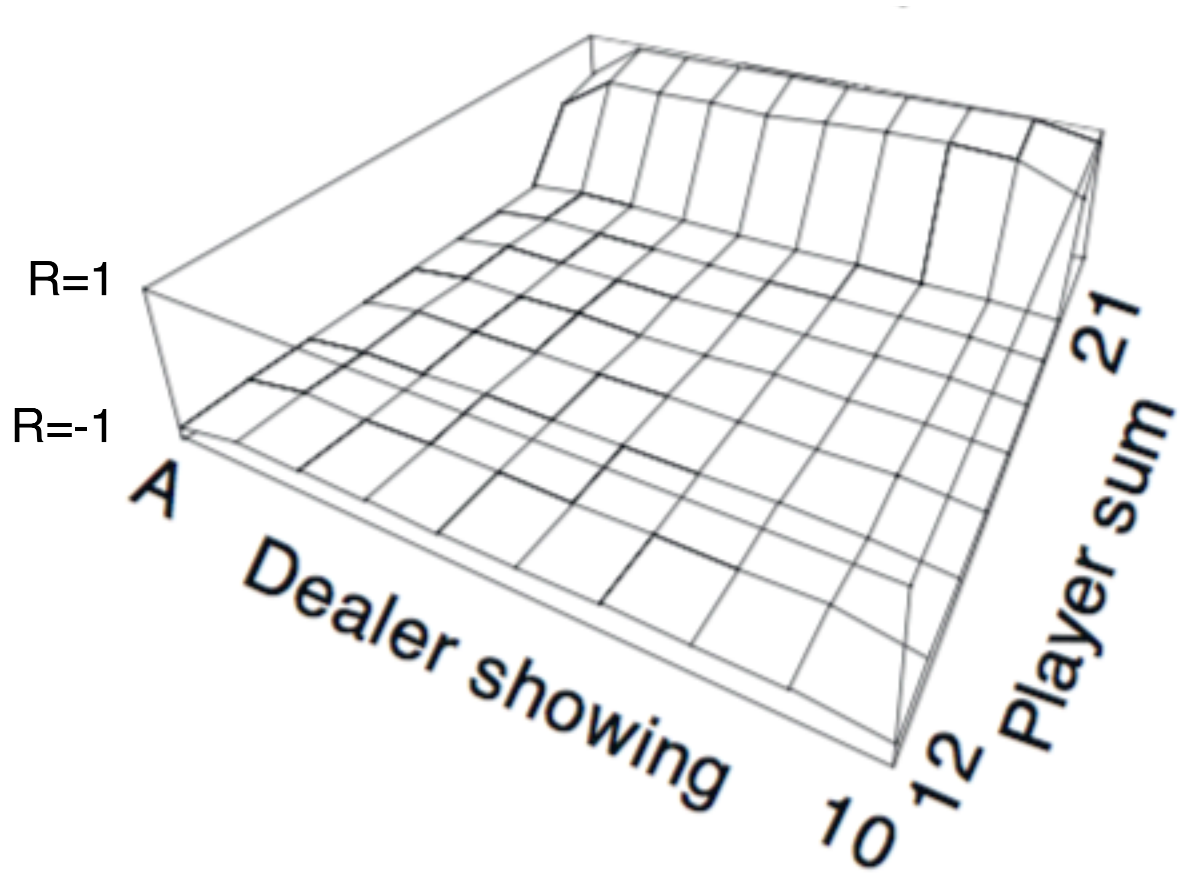
Black Jack



- Episode: one hand
- States: Sum of player’s cards, dealer’s card, usable ace
- Actions: {Stay, Hit}
- Rewards: {Win +1, Tie 0, Loose -1}
- A few other assumptions: infinite deck

- **Evaluates the policy that hits except when the sum of the cards is 20 or 21**

No usable ace

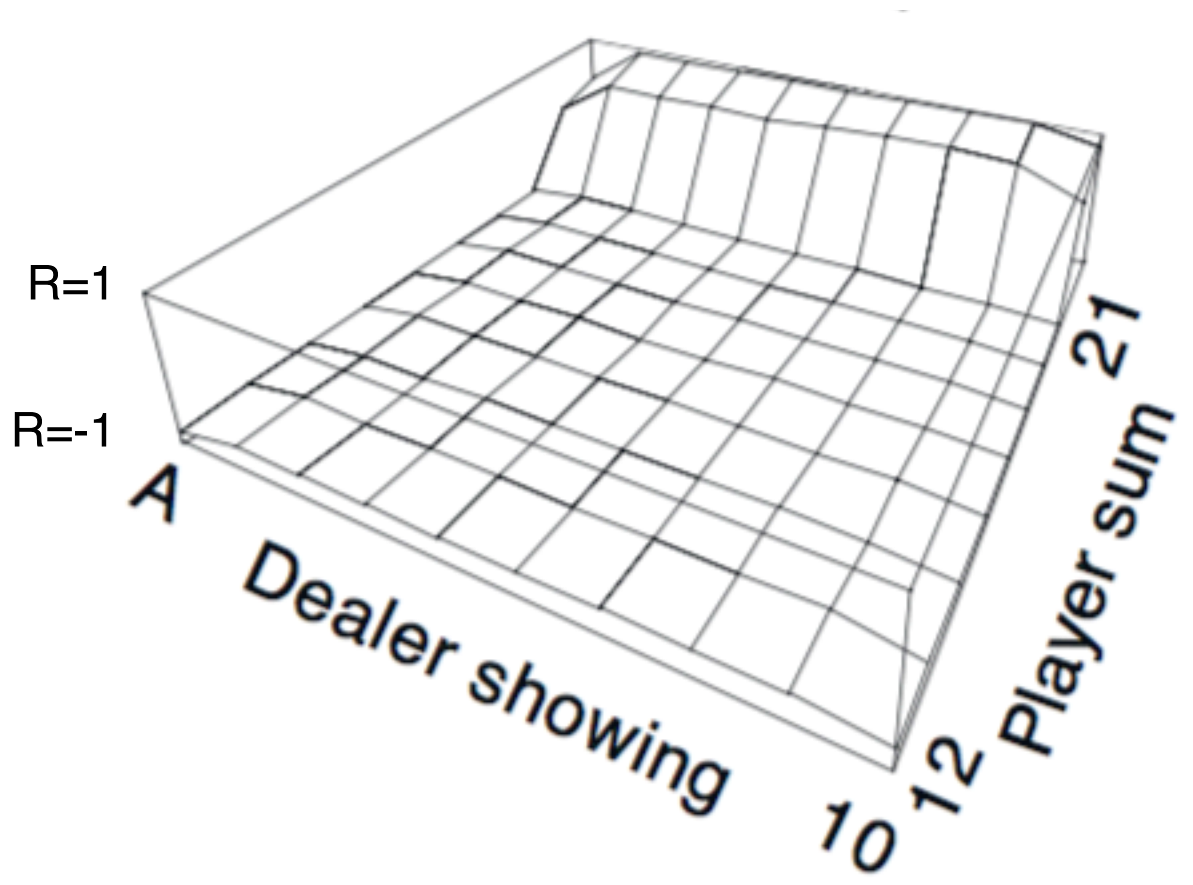


[Figure 5.1, Sutton & Barto]

- **Evaluates the policy that hits except when the sum of the cards is 20 or 21**

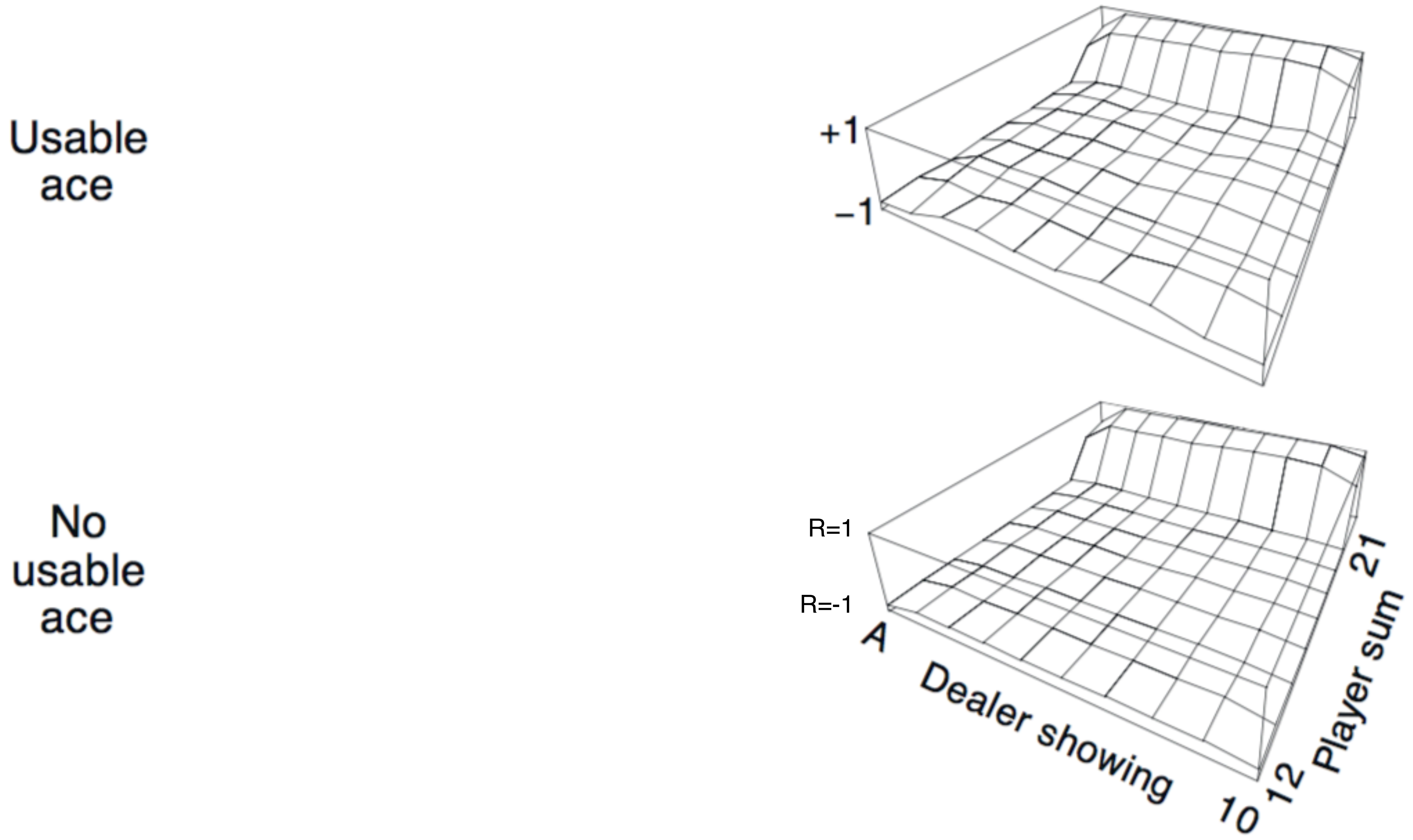
Usable ace

No usable ace



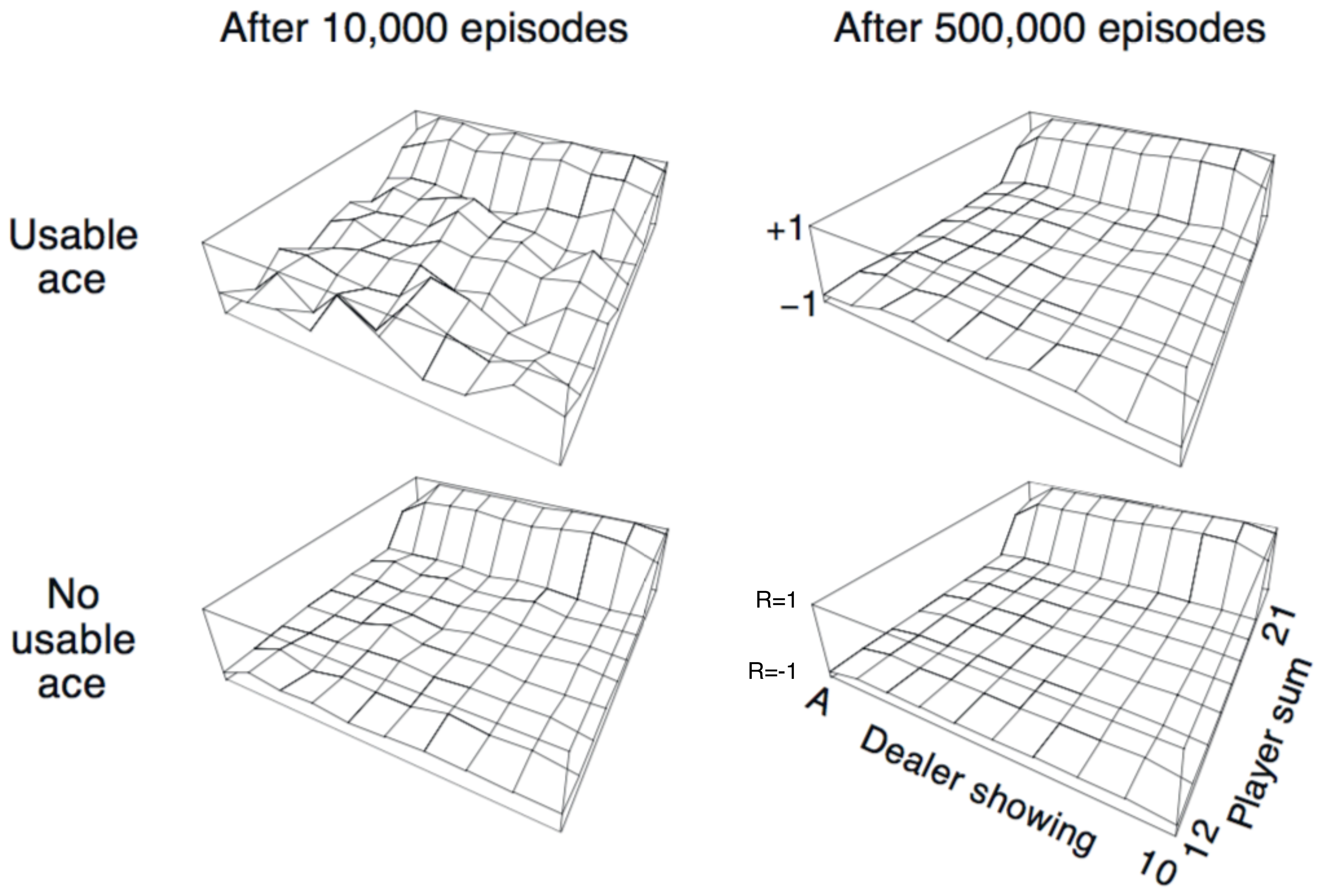
[Figure 5.1, Sutton & Barto]

- **Evaluates the policy that hits except when the sum of the cards is 20 or 21**



[Figure 5.1, Sutton & Barto]

- **Evaluates the policy that hits except when the sum of the cards is 20 or 21**



[Figure 5.1, Sutton & Barto]

Approximation techniques

- Methods we studied are “tabular”
- State value functions (and Q) can be approximated
  - **Linear approximation:** $\mathbf{V}(\mathbf{s}) = \mathbf{w}^\top \mathbf{x}(\mathbf{s})$
    - Coupling between states through x(s)
  - **Adapt the algorithms for this case.**
    - Updates to the value function now imply updating the weights w using a gradient

Approximation techniques (prediction)

- **Linear approximation:** $V(s) = w^\top x(s)$

Approximation techniques (prediction)

- **Linear approximation:** $V(s) = w^\top x(s)$
- **Objective:** $\sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right]^2$

Approximation techniques (prediction)

- **Linear approximation:** $V(s) = w^\top x(s)$
- **Objective:** $\sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right]^2$
- **Gradient update:** $w_{t+1} = w_t - 2\alpha \sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right] x(s)$

Approximation techniques (prediction)

- **Linear approximation:** $V(s) = w^\top x(s)$
- **Objective:** $\sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right]^2$
- **Gradient update:** $w_{t+1} = w_t - 2\alpha \sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right] x(s)$

Gradient Monte Carlo Algorithm for Estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S} \times \mathbb{R}^d \to \mathbb{R}$

Initialize value-function weights $\mathbf{w}$ as appropriate (e.g., $\mathbf{w} = \mathbf{0}$)
Repeat forever:
  Generate an episode $S_0, A_0, R_1, S_1, A_1, \ldots, R_T, S_T$ using π
  For $t = 0, 1, \ldots, T-1$:
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha \big[G_t - \hat{v}(S_t,\mathbf{w}) \big] \nabla \hat{v}(S_t,\mathbf{w})$

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

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:
    $G \leftarrow$ the return that follows the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow$ average$(Returns(s))$

[Sutton & Barto, RL Book, Ch.9]

Approximation techniques (prediction)

- **Linear approximation:** $V(s) = w^\top x(s)$
- **Objective:** $\sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right]^2$
- **Gradient update:** $w_{t+1} = w_t - 2\alpha \sum_{s \in S} \left[v_\pi(s) - w^\top x(s) \right] x(s)$

Gradient Monte Carlo Algorithm for Estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S} \times \mathbb{R}^d \to \mathbb{R}$

Initialize value-function weights $\mathbf{w}$ as appropriate (e.g., $\mathbf{w} = \mathbf{0}$)
Repeat forever:
  Generate an episode $S_0, A_0, R_1, S_1, A_1, \ldots, R_T, S_T$ using π
  For $t = 0, 1, \ldots, T-1$:
    $\boxed{\mathbf{w} \leftarrow \mathbf{w} + \alpha \left[G_t - \hat{v}(S_t, \mathbf{w}) \right] \nabla \hat{v}(S_t, \mathbf{w})}$

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

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:
    $G \leftarrow$ the return that follows the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow$ average$(Returns(s))$

[Sutton & Barto, RL Book, Ch.9]

G_t is an unbiased estimator of $v_\pi(s_t)$

Approximation techniques

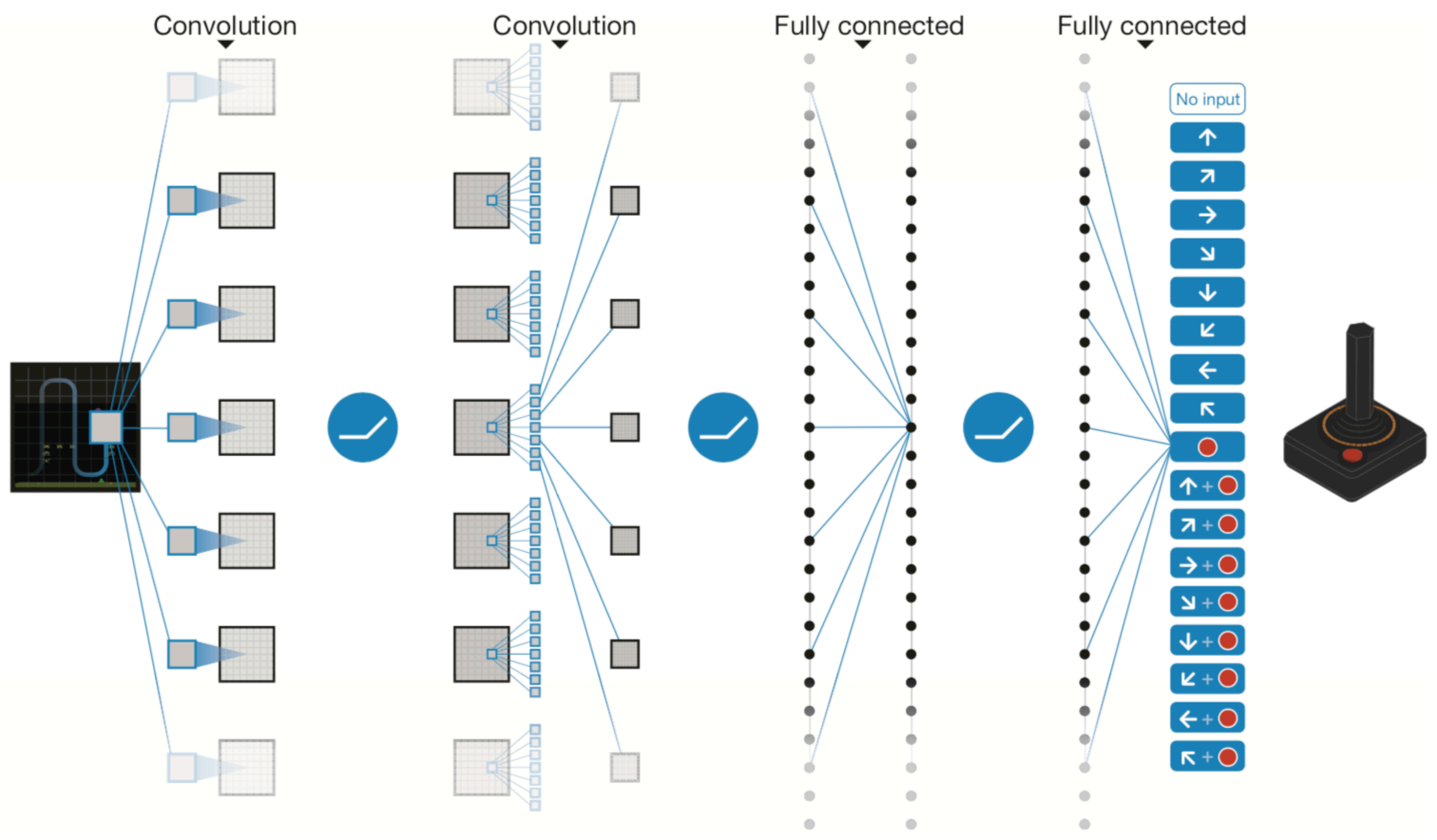
- Works both for prediction and control

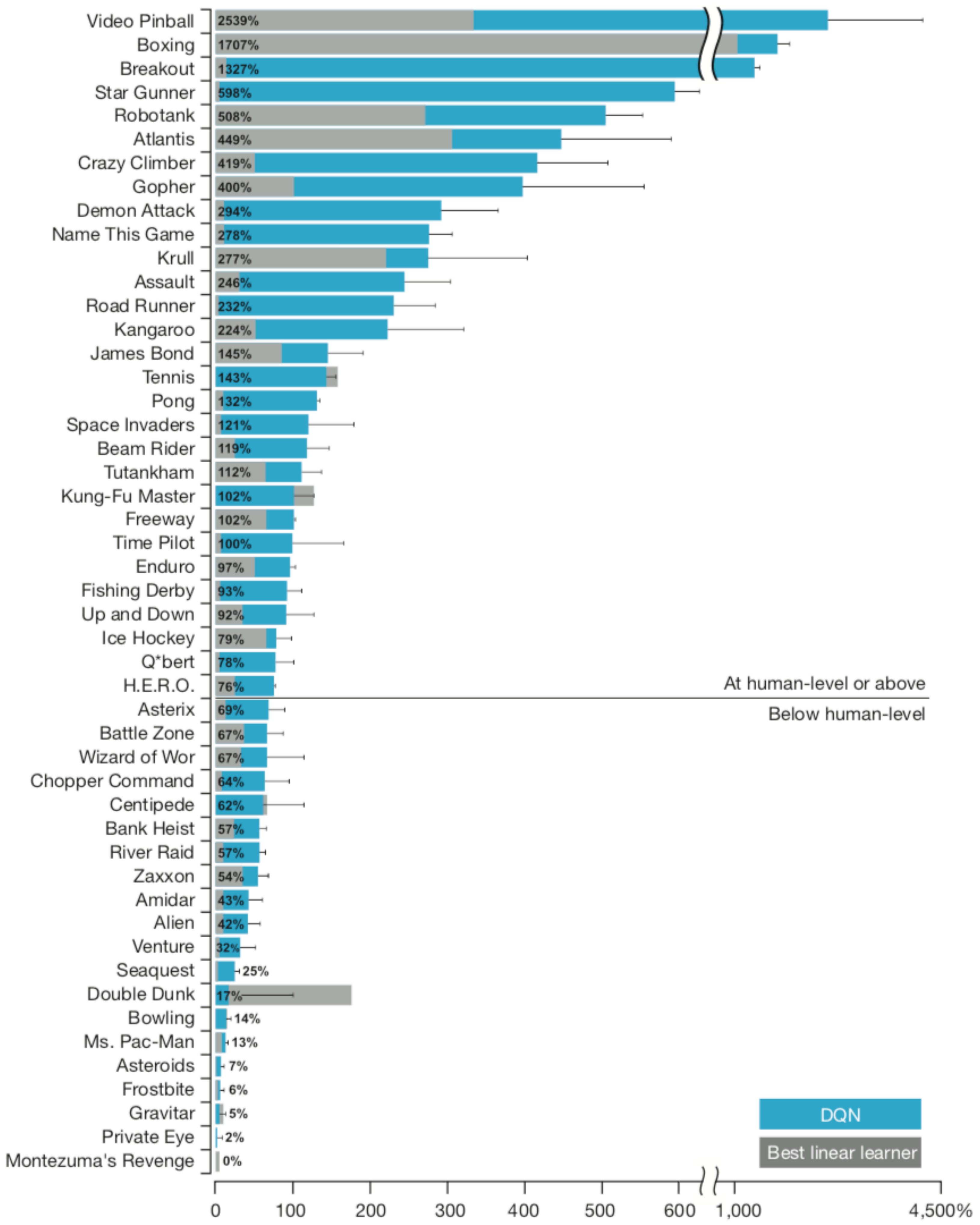
Approximation techniques

- Works both for prediction and control
- Any model can be used to approximate

Approximation techniques

- Works both for prediction and control
- Any model can be used to approximate
- Recent work using deep neural networks yield impressive performance on computer (Atari) games





Summary

- Today we have defined RL studied several algorithms for solving RL problems (mostly for for tabular case)
- **Main challenges**
  - Credit assignment
  - Exploration/Exploitation tradeoff
- **Algorithms**
  - Prediction
    - Monte Carlo and TD(0)
  - Control
    - Q-learning
- **Approximation algorithms can help scale reinforcement learning**