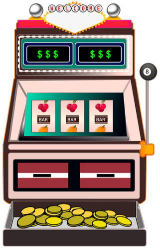
Reinforcement learning

Tomáš Svoboda, Petr Pošík

Vision for Robots and Autonomous Systems, Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague

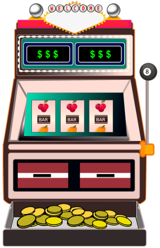
April 15, 2025

(Multi-armed) Bandits



Think about not one but 10 arms you may choose to pull.

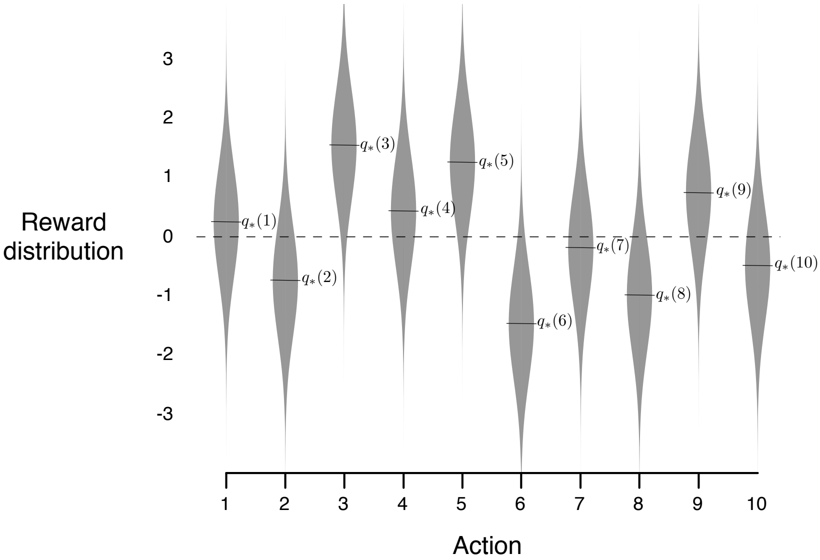
(Multi-armed) Bandits



Think about not one but 10 arms you may choose to pull.

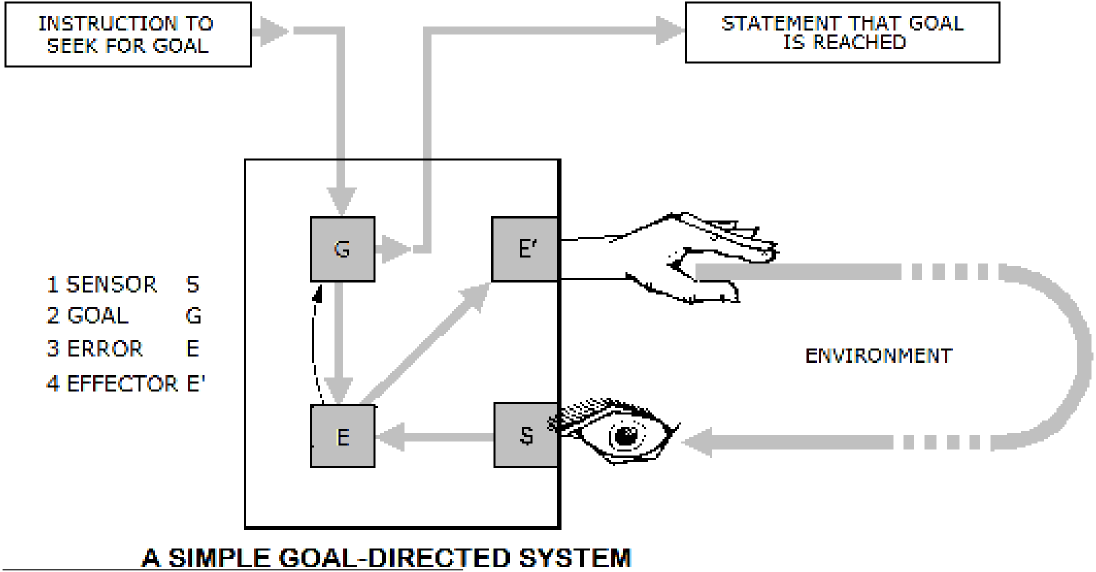
$p(s'|s,a)$ and $r(s,a,s')$ not known!

10 armed bandit, what arm to pull?



See chapters 2.2 and 2.3 in [4] for more detailed discussion

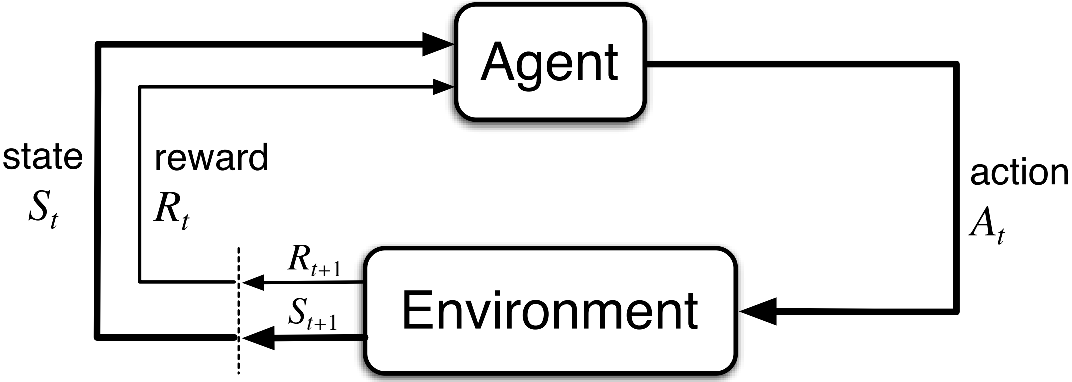
Goal-directed system



1

[1] Figure from http://www.cybsoc.org/gcyb.htm

Reinforcement Learning - performing actions, learning from rewards



2

- Feedback in form of Rewards
- Learn to act so as to maximize expected rewards.

[2] Scheme from [4]

Reinforcement Learning - performing actions, learning from rewards

Reinforcement Learning - performing actions, learning from rewards

Examples, robot learning, Atari games, . . .

Video: Learning safe policies[3]

[3] M. Pecka, V. Salansky, K. Zimmermann, T. Svoboda. Autonomous flipper control with safety constraints. In Intelligent Robots and Systems (IROS), 2016, https://youtu.be/_oUMbBtoRcs

From off-line (MDPs) to on-line (RL)

Markov decision process – MDPs. Off-line search, we know:

- A set of states $s \in \mathcal{S}$ (map)
- A set of actions per state. $a \in \mathcal{A}$
- A transition model $T(s, a, s')$ or $p(s'|s, a)$ (robot)
- A reward function $r(s, a, s')$ (map, robot)

Looking for the optimal policy $\pi(s)$. We can plan/search before the robot enters the environment.

From off-line (MDPs) to on-line (RL)

Markov decision process – MDPs. Off-line search, we know:

- A set of states $s \in \mathcal{S}$ (map)
- A set of actions per state. $a \in \mathcal{A}$
- A transition model $T(s, a, s')$ or $p(s'|s, a)$ (robot)
- A reward function $r(s, a, s')$ (map, robot)

Looking for the optimal policy $\pi(s)$. We can plan/search before the robot enters the environment.

On-line problem:

- Transition model p and reward function r not known.
- Agent/robot must act and learn from experience.

(Transition) Model-based learning

The main idea: Do something, and:

- Learn an approximate model from experiences.
- Solve as if the model was correct.

(Transition) Model-based learning

The main idea: Do something, and:

- Learn an approximate model from experiences.
- Solve as if the model was correct.

Learning MDP model:

- In s try a, observe s', count (s, a, s').
- Normalize to get and estimate of $p(s' \mid s, a)$.
- Discover (by observation) each $r(s, a, s')$ when experienced.

(Transition) Model-based learning

The main idea: Do something, and:

- Learn an approximate model from experiences.
- Solve as if the model was correct.

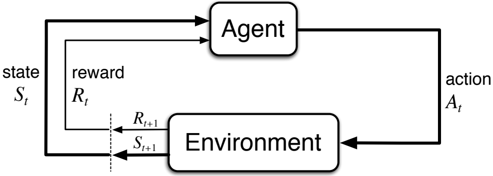
Learning MDP model:

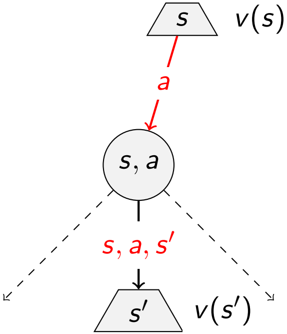
- In s try a, observe s', count (s, a, s').
- Normalize to get and estimate of $p(s' \mid s, a)$.
- Discover (by observation) each $r(s, a, s')$ when experienced.

Solve the learned MDP.

Reward function $r(s, a, s')$

- $r(s, a, s')$ - reward for taking a in s and landing in s'.
- In Grid world, we assumed $r(s, a, s')$ to be the same everywhere.
- In the real world, it is different (going up, down, ...)

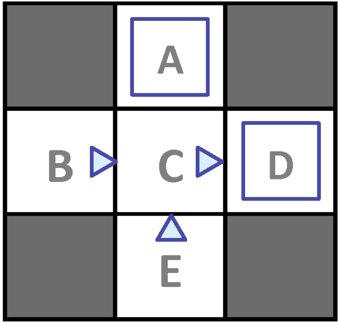




In ai-gym `env.step(action)` returns s', $r(s, \texttt{action}, s')$.

Model-based learning: Grid example

Input Policy π



Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

[4]

[4] Figure from [1]

Learning transition model

$\hat{p}(\mathrm{D} \mid \mathrm{C}, \mathrm{east}) = ?$

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Learning reward function

$\hat{r}(\mathrm{C}, \mathrm{east}, \mathrm{D}) = ?$

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Model based vs model-free: Expected age $\mathrm{E}[A]$

Random variable age A.

$$\mathrm{E}[A] = \sum_a P(A = a)a$$

We do not know $P(A = a)$. Instead, we collect N samples $[a_1, a_2, \ldots a_N]$.

Model based vs model-free: Expected age $\mathrm{E}[A]$

Random variable age A.

$$\mathrm{E}[A] = \sum_a P(A = a)a$$

We do not know $P(A = a)$. Instead, we collect N samples $[a_1, a_2, \ldots a_N]$.

Model based

$$\hat{P}(a) = \frac{\mathrm{num}(a)}{N}$$

$$\mathrm{E}[A] \approx \sum_a \hat{P}(a)a$$

Model based vs model-free: Expected age $\mathrm{E}[A]$

Random variable age A.

$$\mathrm{E}[A] = \sum_a P(A = a)a$$

We do not know $P(A = a)$. Instead, we collect N samples $[a_1, a_2, \ldots a_N]$.

Model based

$$\hat{P}(a) = \frac{\mathrm{num}(a)}{N}$$

$$\mathrm{E}[A] \approx \sum_a \hat{P}(a)a$$

Model free

$$\mathrm{E}[A] \approx \frac{1}{N} \sum_i a_i$$

Model-free learning

Passive learning (evaluating given policy)

- **Input:** a fixed policy $\pi(s)$
- We want to know how good it is.
- r, p not known.
- Execute policy ...
- and learn on the way.
- **Goal:** learn the state values $v^{\pi}(s)$

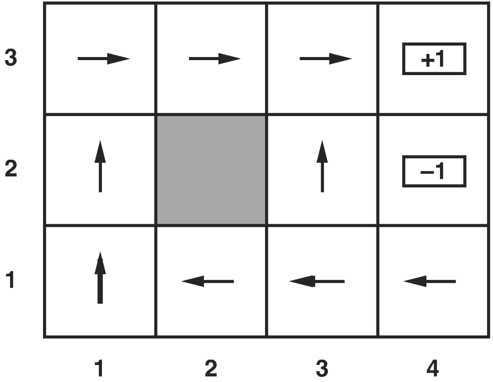


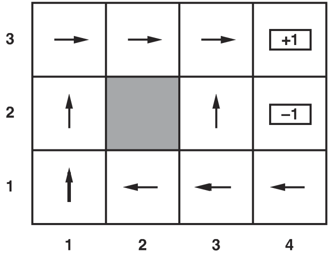
Image from [2]

Direct evaluation from episodes

Value of s for π – expected sum of discounted rewards – expected return

$$v^\pi(S_t) = \mathrm{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}\right]$$

$$v^\pi(S_t) = \mathrm{E}\left[G_t\right]$$

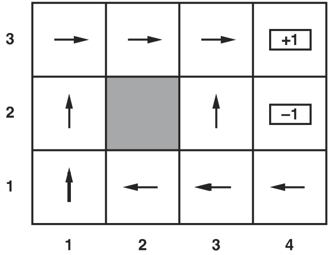


Direct evaluation from episodes

Value of s for π – expected sum of discounted rewards – expected return

$$v^{\pi}(S_t) = \mathrm{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}\right]$$

$$v^{\pi}(S_t) = \mathrm{E}\left[G_t\right]$$



$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$
$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$
$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (2,1)_{\mathbf{-.04}} \rightsquigarrow (3,1)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (4,2)_{\mathbf{-1}}\ .$$

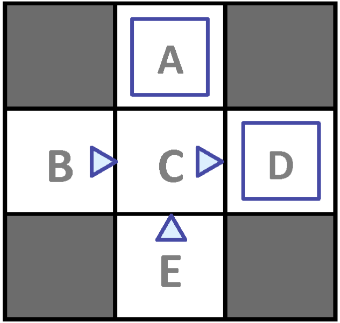
Direct evaluation from episodes, $v^\pi(S_t) = \mathrm{E}[G_t]$, $\gamma = 1$

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \; .$$

What is $v(3,2)$ after these episodes?

Direct evaluation: Grid example

Input Policy π



Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Direct evaluation: Grid example, $\gamma = 1$

What is $v(C)$ after the 4 episodes?

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Direct evaluation: Grid example, $\gamma = 1$

What is v(C) after the 4 episodes?

Let M be the number of recorded episodes.
Let N be the number of samples used to compute the averages.
What is the relation of M and N?

- **A** $N = M$
- **B** $N \leq M$
- **C** $N \geq M$
- **D** N has no relation to M

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Direct evaluation algorithm (every-visit version)

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \ .$$

Input: a policy π to be evaluated
Initialize:
    $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$
    $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$
Loop forever (for each episode):
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    Loop backwards for each step of episode, $t = T-1, T-2, \ldots, 0$:
        $G \leftarrow R_{t+1} + \gamma G$
        Append G to $Returns(S_t)$
        $V(S_t) \leftarrow$ average($Returns(S_t)$)

Direct evaluation algorithm (first-visit version)

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} .$$

Input: a policy π to be evaluated
Initialize:
    $V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$
    $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$
Loop forever (for each episode):
    Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
    $G \leftarrow 0$
    Loop backwards for each step of episode, $t = T-1, T-2, \ldots, 0$:
        $G \leftarrow R_{t+1} + \gamma G$
        If S_t does not appear in $S_0, S_1, \ldots, S_{t-1}$:   *// Use the return for the first visit only*
            Append G to $Returns(S_t)$
            $V(S_t) \leftarrow$ average$(Returns(S_t))$

Direct evaluation: analysis

The good:

- Simple, easy to understand and implement.
- Does not need p, r and eventually it computes the true v^{π}.

Direct evaluation: analysis

The good:

- Simple, easy to understand and implement.
- Does not need p, r and eventually it computes the true v^π.

The bad:

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} .$$

Direct evaluation: analysis

The good:

- Simple, easy to understand and implement.
- Does not need p, r and eventually it computes the true v^π.

The bad:

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \, .$$

- Each state value learned in isolation.

Direct evaluation: analysis

The good:

- Simple, easy to understand and implement.
- Does not need p, r and eventually it computes the true v^π.

The bad:

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}}\ .$$

- Each state value learned in isolation.
- State values are not independent
- $v^\pi(s) = \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s,\pi(s),s') + \gamma\, v^\pi(s')\big]$

(on-line) Policy evaluation?

In MDP, we did:

- Initialize the values: $V_0^\pi(s) = 0$
- In each iteration, replace V with a one-step-look-ahead:
  $V_{k+1}^\pi(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V_k^\pi(s')\big]$

(on-line) Policy evaluation?

In MDP, we did:

- Initialize the values: $V_0^\pi(s) = 0$
- In each iteration, replace V with a one-step-look-ahead:
  $V_{k+1}^\pi(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\left[r(s, \pi(s), s') + \gamma\, V_k^\pi(s')\right]$

Problem: both $p\big(s' \mid s, \pi(s)\big)$ and $r(s, \pi(s), s')$ unknown!

Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:

$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^{\pi}_{k}(s')\big]$

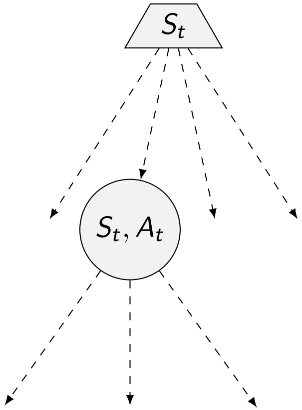
Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:

$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^{\pi}_{k}(s')\big]$

What about stop, try, try, . . . , and average?

Samples at time t. $\pi(S_t) \rightarrow A_t$, repeat A_t.

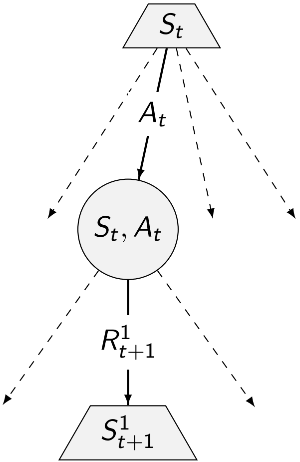


Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:
$V^\pi_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^\pi_k(s')\big]$

What about stop, try, try, ..., and average?
Samples at time t. $\pi(S_t) \to A_t$, repeat A_t.

$$\text{sample}^1 \quad = \quad R^1_{t+1} + \gamma\, V(S^1_{t+1})$$

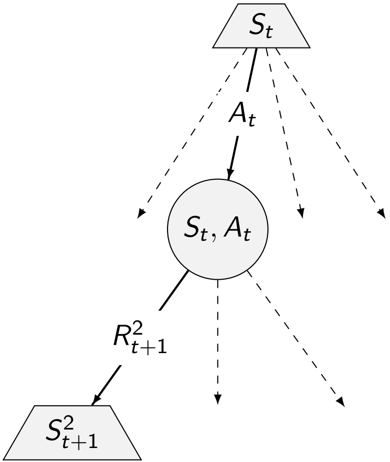


Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:
$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^{\pi}_{k}(s')\big]$

What about stop, try, try, . . . , and average?
Samples at time t. $\pi(S_t) \rightarrow A_t$, repeat A_t.

$$
\begin{aligned}
\text{sample}^1 &= R^1_{t+1} + \gamma\, V(S^1_{t+1}) \\
\text{sample}^2 &= R^2_{t+1} + \gamma\, V(S^2_{t+1})
\end{aligned}
$$

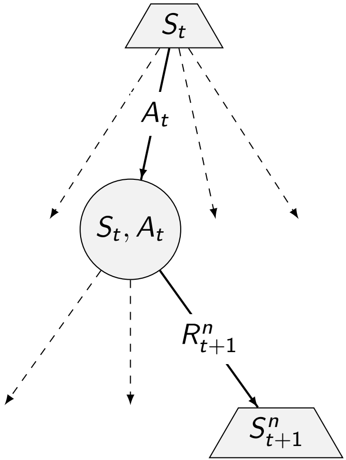


Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:
$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^{\pi}_{k}(s')\big]$

What about stop, try, try, . . . , and average?
Samples at time t. $\pi(S_t) \rightarrow A_t$, repeat A_t.

$$
\begin{aligned}
\text{sample}^1 &= R^1_{t+1} + \gamma\, V(S^1_{t+1}) \\
\text{sample}^2 &= R^2_{t+1} + \gamma\, V(S^2_{t+1}) \\
\vdots &= \vdots \\
\text{sample}^n &= R^n_{t+1} + \gamma\, V(S^n_{t+1})
\end{aligned}
$$



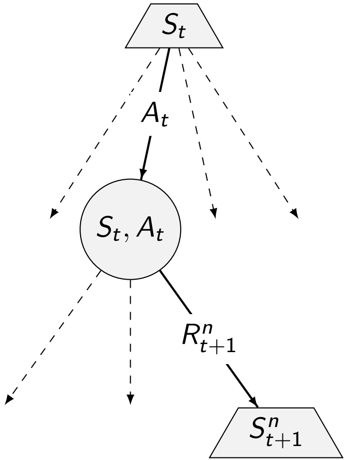
Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:
$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s, \pi(s), s') + \gamma\, V^{\pi}_k(s')\big]$

What about stop, try, try, ..., and average?
Samples at time t. $\pi(S_t) \to A_t$, repeat A_t.

$$
\begin{aligned}
\text{sample}^1 &= R^1_{t+1} + \gamma\, V(S^1_{t+1}) \\
\text{sample}^2 &= R^2_{t+1} + \gamma\, V(S^2_{t+1}) \\
\vdots &= \vdots \\
\text{sample}^n &= R^n_{t+1} + \gamma\, V(S^n_{t+1})
\end{aligned}
$$

$$
V(S_t) \leftarrow \frac{1}{n} \sum_i \text{sample}^i
$$



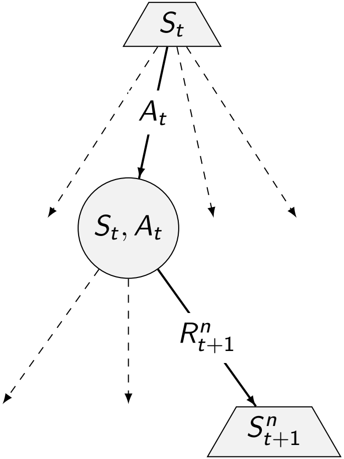
Use samples for evaluating policy?

MDP (p, r known) : Update V estimate by a weighted average:
$V^{\pi}_{k+1}(s) \leftarrow \sum_{s'} p\big(s' \mid s, \pi(s)\big)\big[r(s,\pi(s),s') + \gamma\, V^{\pi}_{k}(s')\big]$

What about stop, try, try, ..., and average?
Samples at time t. $\pi(S_t) \to A_t$, repeat A_t.

$$
\begin{aligned}
\text{sample}^1 &= R^1_{t+1} + \gamma\, V(S^1_{t+1})\\
\text{sample}^2 &= R^2_{t+1} + \gamma\, V(S^2_{t+1})\\
\vdots &= \vdots\\
\text{sample}^n &= R^n_{t+1} + \gamma\, V(S^n_{t+1})
\end{aligned}
$$

$$V(S_t) \leftarrow \frac{1}{n}\sum_i \text{sample}^i$$



Problem: We cannot re-set to S_t easily.

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \; .$$

$\gamma = 1$

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \ .$$

$\gamma = 1$

From first trial (episode): $V(2,3) = \qquad , V(1,3) = \qquad , \ldots$

Temporal-difference value learning

$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$
$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$
$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (2,1)_{\mathbf{-.04}} \rightsquigarrow (3,1)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (4,2)_{\mathbf{-1}} \ .$$

$\gamma = 1$
From first trial (episode): $V(2,3) = 0.92$, $V(1,3) = 0.84, \ldots$

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \ .$$

$\gamma = 1$
From first trial (episode): $V(2,3) = 0.92$, $V(1,3) = 0.84, \ldots$
In second episode, going from $S_t = (1,3)$ to $S_{t+1} = (2,3)$ with reward $R_{t+1} = -0.04$, hence:

$$V(1,3) = R_{t+1} + V(2,3) = -0.04 + 0.92 = 0.88$$

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \ .$$

$\gamma = 1$
From first trial (episode): $V(2,3) = 0.92$, $V(1,3) = 0.84, \ldots$
In second episode, going from $S_t = (1,3)$ to $S_{t+1} = (2,3)$ with reward $R_{t+1} = -0.04$, hence:

$$V(1,3) = R_{t+1} + V(2,3) = -0.04 + 0.92 = 0.88$$

- First estimate 0.84 is a bit lower than 0.88. $V(S_t)$ is different than $R_{t+1} + \gamma V(S_{t+1})$

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \; .$$

$\gamma = 1$
From first trial (episode): $V(2,3) = 0.92$, $V(1,3) = 0.84, \ldots$
In second episode, going from $S_t = (1,3)$ to $S_{t+1} = (2,3)$ with reward $R_{t+1} = -0.04$, hence:

$$V(1,3) = R_{t+1} + V(2,3) = -0.04 + 0.92 = 0.88$$

- First estimate 0.84 is a bit lower than 0.88. $V(S_t)$ is different than $R_{t+1} + \gamma V(S_{t+1})$
- Update ($\alpha\times$ difference): $V(S_t) \leftarrow V(S_t) + \alpha\Big(\big[R_{t+1} + \gamma V(S_{t+1})\big] - V(S_t)\Big)$
- α is the learning rate.

Temporal-difference value learning

$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$$
$$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}} \; .$$

$\gamma = 1$
From first trial (episode): $V(2,3) = 0.92$, $V(1,3) = 0.84, \ldots$
In second episode, going from $S_t = (1,3)$ to $S_{t+1} = (2,3)$ with reward $R_{t+1} = -0.04$, hence:

$$V(1,3) = R_{t+1} + V(2,3) = -0.04 + 0.92 = 0.88$$

- First estimate 0.84 is a bit lower than 0.88. $V(S_t)$ is different than $R_{t+1} + \gamma V(S_{t+1})$
- Update ($\alpha\times$ difference): $V(S_t) \leftarrow V(S_t) + \alpha\Big(\big[R_{t+1} + \gamma V(S_{t+1})\big] - V(S_t)\Big)$
- α is the learning rate.
- $V(S_t) \leftarrow (1-\alpha)V(S_t) + \alpha\,(\text{new sample})$

Exponential moving average

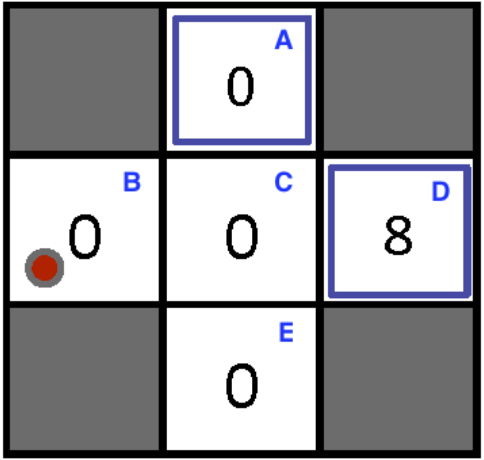
$$\overline{x}_n = (1-\alpha)\overline{x}_{n-1} + \alpha x_n$$

What does it remember about the past? Try to derive:

$$\overline{x}_n = f(\alpha, x_n, x_{n-1}, x_{n-2}, x_{n-3}, \dots)$$

Example: TD Value learning

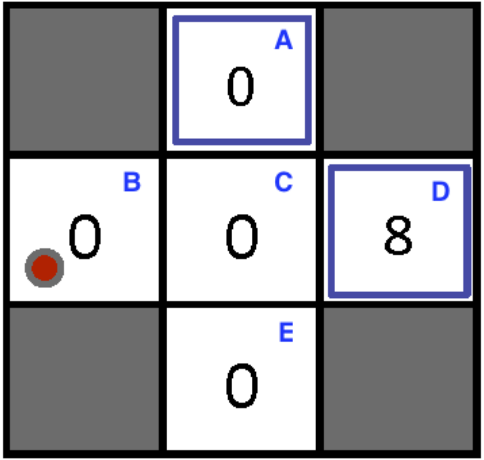
$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$



- Values represent initial $V(s)$
- Assume: $\gamma = 1, \alpha = 0.5, \pi(s) = \rightarrow$

Example: TD Value learning

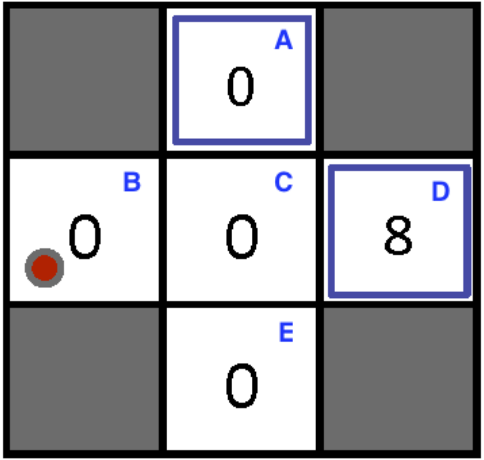
$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$



- Values represent initial $V(s)$
- Assume: $\gamma = 1, \alpha = 0.5, \pi(s) = \rightarrow$
- $(B, \rightarrow, C), -2, \Rightarrow V(B)$?

Example: TD Value learning

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$$



- Values represent initial $V(s)$
- Assume: $\gamma = 1, \alpha = 0.5, \pi(s) = \rightarrow$
- $(B, \rightarrow, C), -2, \Rightarrow V(B)?$
- $(C, \rightarrow, D), -2, \Rightarrow V(C)?$

Temporal difference value learning: algorithm

Input: the policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\mathit{terminal}) = 0$

Loop for each episode:
    Initialize S
    Loop 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

What is wrong with the temporal difference Value learning?

The Good: Model-free value learning by mimicking Bellman updates.

What is wrong with the temporal difference Value learning?

The Good: Model-free value learning by mimicking Bellman updates.

The Bad: How to turn values into a (new) policy?

- $\pi(s) = \arg\max_a \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma V(s') \right]$

What is wrong with the temporal difference Value learning?

The Good: Model-free value learning by mimicking Bellman updates.

The Bad: How to turn values into a (new) policy?

- $\pi(s) = \arg\max_a \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma V(s') \right]$
- $\pi(s) = \arg\max_a Q(s, a)$

Q-learning

Reminder: V, Q-value iteration for MDPs

Value/Utility iteration (depth limited evaluation):

- Start: $V_0(s) = 0$
- In each step update V by looking one step ahead:
  $V_{k+1}(s) \leftarrow \max_a \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma V_k(s') \right]$

Q values more useful (think about updating π)

- Start: $Q_0(s, a) = 0$
- In each step update Q by looking one step ahead:
  $Q_{k+1}(s, a) \leftarrow \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma \max_{a'} Q_k(s', a') \right]$

Q-learning (one episode of)

MDP update: $Q_{k+1}(s,a) \leftarrow \sum_{s'} p(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$

Q-learning (one episode of)

MDP update: $Q_{k+1}(s, a) \leftarrow \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma \max_{a'} Q_k(s', a') \right]$

Learn Q values as the robot/agent goes (temporal difference)

- Drive the robot (take action a in state s) and fetch rewards (s, a, s', R)

Q-learning (one episode of)

MDP update: $Q_{k+1}(s, a) \leftarrow \sum_{s'} p(s' \mid s, a) \left[r(s, a, s') + \gamma \max_{a'} Q_k(s', a') \right]$

Learn Q values as the robot/agent goes (temporal difference)

- Drive the robot (take action a in state s) and fetch rewards (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.

Q-learning (one episode of)

MDP update: $Q_{k+1}(s,a) \leftarrow \sum_{s'} p(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$

Learn Q values as the robot/agent goes (temporal difference)

- Drive the robot (take action a in state s) and fetch rewards (s, a, s', R)
- We know old estimates $Q(s,a)$ (and $Q(s',a')$), if not, initialize.

- A new sample estimate (of $Q(s,a)$) at time t

  $\text{sample} = R_{t+1} + \gamma \max_{a} Q(S_{t+1}, a)$

Q-learning (one episode of)

MDP update: $Q_{k+1}(s,a) \leftarrow \sum_{s'} p(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$

Learn Q values as the robot/agent goes (temporal difference)

- Drive the robot (take action a in state s) and fetch rewards (s, a, s', R)
- We know old estimates $Q(s,a)$ (and $Q(s',a')$), if not, initialize.

- A new sample estimate (of $Q(s,a)$) at time t
  $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update
  $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$
  or (the same)
  $Q(S_t, A_t) \leftarrow (1-\alpha) Q(S_t, A_t) + \alpha\,\text{sample}$

Q-learning (one episode of)

MDP update: $Q_{k+1}(s,a) \leftarrow \sum_{s'} p(s' \mid s,a)\left[r(s,a,s') + \gamma \max_{a'} Q_k(s',a')\right]$

Learn Q values as the robot/agent goes (temporal difference)

- Drive the robot (take action a in state s) and fetch rewards (s, a, s', R)
- We know old estimates $Q(s,a)$ (and $Q(s',a')$), if not, initialize.

- A new sample estimate (of $Q(s,a)$) at time t
  $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update
  $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$
  or (the same)
  $Q(S_t, A_t) \leftarrow (1-\alpha)Q(S_t, A_t) + \alpha\,\text{sample}$

In each step Q approximates the optimal q^* function.

Q-learning: algorithm (repeating episodes, until terminal or exhausted)

step size $0 < \alpha \leq 1$
initialize $Q(s, a)$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
repeat episodes:
    initialize S
    for for each step of episode: **do**
        choose A from $\mathcal{A}(S)$
        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
until Time is up, . . .

From Q-learning to Q-learning agent

- Drive the robot and fetch rewards. (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.
- A new sample estimate: $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$

From Q-learning to Q-learning agent

- Drive the robot and fetch rewards. (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.
- A new sample estimate: $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$

Technicalities for the Q-learning agent

From Q-learning to Q-learning agent

- Drive the robot and fetch rewards. (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.
- A new sample estimate: sample $= R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$

Technicalities for the Q-learning agent

- How to represent the Q-function?

From Q-learning to Q-learning agent

- Drive the robot and fetch rewards. (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.
- A new sample estimate: $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$

Technicalities for the Q-learning agent

- How to represent the Q-function?
- What is the value for terminal? $Q(s, \text{Exit})$ or $Q(s, \text{None})$

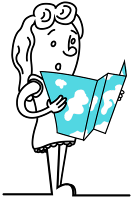
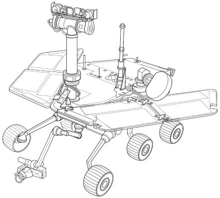
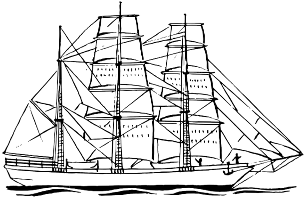
From Q-learning to Q-learning agent

- Drive the robot and fetch rewards. (s, a, s', R)
- We know old estimates $Q(s, a)$ (and $Q(s', a')$), if not, initialize.
- A new sample estimate: $\text{sample} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$
- α update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(\text{sample} - Q(S_t, A_t))$

Technicalities for the Q-learning agent

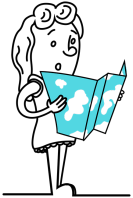
- How to represent the Q-function?
- What is the value for terminal? $Q(s, \text{Exit})$ or $Q(s, \text{None})$
- How to drive? Where to drive next? Does it change over the course?

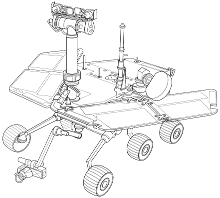
Exploration vs. Exploitation

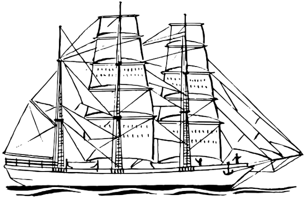




- Drive the known road or try a new one?

Exploration vs. Exploitation

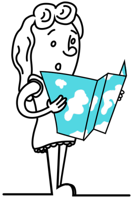


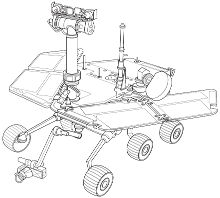


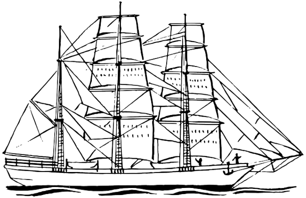


- Drive the known road or try a new one?
- Go to the university menza or try a nearby restaurant?

Exploration vs. Exploitation

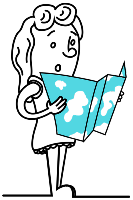
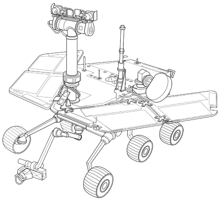
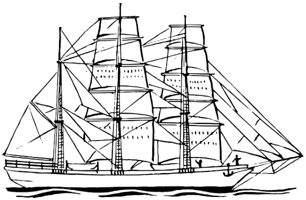






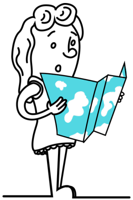
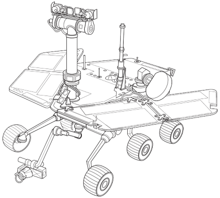
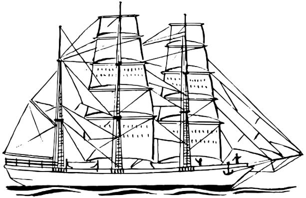
- Drive the known road or try a new one?
- Go to the university menza or try a nearby restaurant?
- Use the SW (operating system) I know or try a new one?

Exploration vs. Exploitation

- Drive the known road or try a new one?
- Go to the university menza or try a nearby restaurant?
- Use the SW (operating system) I know or try a new one?
- Go to bussiness or study a demanding program?

Exploration vs. Exploitation

- Drive the known road or try a new one?
- Go to the university menza or try a nearby restaurant?
- Use the SW (operating system) I know or try a new one?
- Go to bussiness or study a demanding program?
- ...

How to explore?

Random (ϵ-greedy):

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.
- With probability $1 - \epsilon$, use the policy.

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.
- With probability $1 - \epsilon$, use the policy.

Problems with randomness?

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.
- With probability $1 - \epsilon$, use the policy.

Problems with randomness?

- Keeps exploring forever.

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.
- With probability $1 - \epsilon$, use the policy.

Problems with randomness?

- Keeps exploring forever.
- Should we keep ϵ fixed (over learning)?

How to explore?

Random (ϵ-greedy):

- Flip a coin every step.
- With probability ϵ, act randomly.
- With probability $1 - \epsilon$, use the policy.

Problems with randomness?

- Keeps exploring forever.
- Should we keep ϵ fixed (over learning)?
- ϵ same everywhere?

What we have learned

- Agent/robot may learn by acting an getting rewards
- Model based vs. model-free methods
- Direct learning vs. temporal-difference learning
- From learning state values to Q-learning

References I

Further reading: Chapter 21 of [2] (chapter 23 of [3]). More detailed discussion in [4], chapters 2, 5, and 6.

[1] Dan Klein and Pieter Abbeel.
UC Berkeley CS188 Intro to AI – course materials.
http://ai.berkeley.edu/.
Used with permission of Pieter Abbeel.

[2] Stuart Russell and Peter Norvig.
Artificial Intelligence: A Modern Approach.
Prentice Hall, 3rd edition, 2010.
http://aima.cs.berkeley.edu/.

[3] Stuart Russell and Peter Norvig.
Artificial Intelligence: A Modern Approach.
Prentice Hall, 4th edition, 2021.

References II

[4] Richard S. Sutton and Andrew G. Barto.
Reinforcement Learning; an Introduction.
MIT Press, 2nd edition, 2018.
http://www.incompleteideas.net/book/the-book-2nd.html.