

CS 234 Session II

Model-free policy evaluation & control

Recap of lecture

- Model-free policy evaluation
 - Monte Carlo (first-visit, every-visit, incremental)
 - Temporal Difference (TD(0), batch TD \Leftrightarrow certainty equivalence estimate)
- Model-free control
 - Monte Carlo
 - SARSA
 - Q-Learning

Exercise 1


Exercise 4.1. Recall our Mars Rover MDP from last lecture, shown in Figure 3 below. Suppose that our estimate for the value of each state is currently 0. If we experience the history

$$h = (S3, TL, +0, S2, TL, +0, S1, TL, +1, \text{terminal}),$$

then:

1. What is the first-visit Monte Carlo estimate of V at each state?
2. What is the every-visit Monte Carlo estimate of each state?
3. What is the incremental first-visit Monte Carlo estimate of V with $\alpha = \frac{2}{3}$?
4. What is the incremental every-visit Monte Carlo estimate of V with $\alpha = \frac{2}{3}$?

Exercise 1

S1	S2	S3	S4	S5	S6	S7
Okay Field Site R=+1	R=0	R=0	 R=0	R=0	R=0	Fantastic Field Site R=+10

$P(s' s, TL) =$	1	0	0	0	0	0								
	1	0	0	0	0	0		0	1	0	0	0	0	0
	0	1	0	0	0	0		0	0	1	0	0	0	0
	0	0	1	0	0	0		0	0	0	1	0	0	0
	0	0	0	1	0	0	$P(s' s, TR) =$	0	0	0	0	1	0	0
	0	0	0	0	1	0		0	0	0	0	0	1	0
	0	0	0	0	0	1		0	0	0	0	0	0	1
								0	0	0	0	0	0	1

2. $V(S_1) = 1$, $V(S_2) = \gamma$ and $V(S_3) = \gamma^2$.
3. $V(S_1) = \frac{2}{3}$, $V(S_2) = \frac{2}{3}\gamma$ and $V(S_3) = \frac{2}{3}\gamma^2$.

Exercise 2

Exercise 4.3. Consider again the Mars Rover example in Figure 3. Suppose that our estimate for the value of each state is currently 0. If our batch consists of two histories

$$h_1 = (S3, TL, +0, S2, TL, +0, S1, TL, +1, \text{terminal})$$

$$h_2 = (S3, TL, +0, S2, TL, +0, S2, TL, +0, S1, TL, +1, \text{terminal})$$

and our policy is TL , then what is the certainty equivalence estimate?

Exercise 2

Solution. *The certainty equivalence estimate is $V(S_1) = 1$, $V(S_2) = \frac{2\gamma}{3-\gamma}$ and $V(S_3) = \frac{2\gamma^2}{3-\gamma}$. More specifically, S_1 and S_3 , under the certainty equivalence estimate, will transit to left for sure, while S_2 with $2/3$ probability will transit to left and the other $1/3$ probability stay at S_2 .*

Exercise 3

Consider an MDP with $\gamma = 0.5$, $S = \{s_1, s_2, s_3\}$, $A = \{a_1, a_2, a_3\}$, and for any i, j , $P(s_i | s_j, a_i) = 1$, $R(s_i, a_i, s_j) = j$. A Q-learning agent is exploring this MDP with ϵ -greedy strategy ($\epsilon \neq 0$). A random action never happens to pick the greedy action. Ties are broken by choosing the a_i with the smallest i . Suppose the learning rate of Q-learning algorithm is 0.5 and all Q are initialized to zeros. In the following trajectory

$$(s_1, a_1, 1, s_1, a_2, 2, s_2)$$

Is action a_1 and a_2 chosen randomly or greedily? Justify your answer.

Exercise 3

a_1 : greedily, a_2 randomly.

Since all Q are initialized to zeros and ties are broken by choosing smallest i . a_1 is the optimal action at initialization. Also we know that a random action never picks the greedy action, thus a_1 is chosen greedily.

Then the agent performs one step update given the first experience tuple $(s_1, a_1, 1, s_1)$. $Q(s_1, a_1)$ gets updated to 0.5 while $Q(s_1, a_2)$ and $Q(s_1, a_3)$ remains 0. Now the optimal action is a_1 . Therefore the next action a_2 taken by agent must be a random choice.

Exercise 4

Exercise 6.11 Why is Q-learning considered an *off-policy* control method?

Exercise 4

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha((r_t + \gamma \max_{a'} Q(s_{t+1}, a')) - Q(s_t, a_t))$$

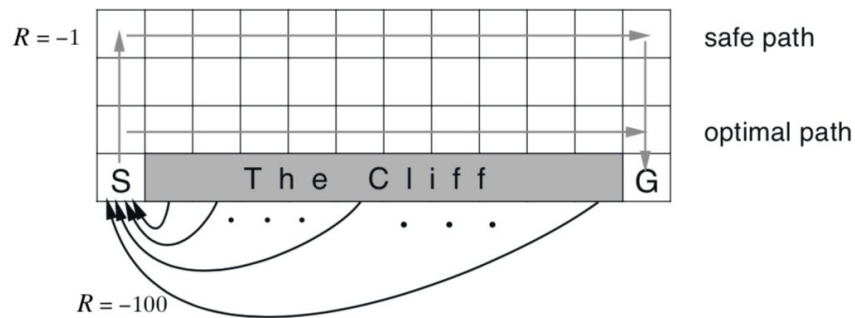
In each iteration of Q-learning, the target value (label for update) is the value of the target policy π' at s_{t+1} . The target policy π' is greedy w.r.t. Q values from last iteration:

$$\pi'(s_{t+1}) = \arg \max_{a'} Q(s_{t+1}, a')$$

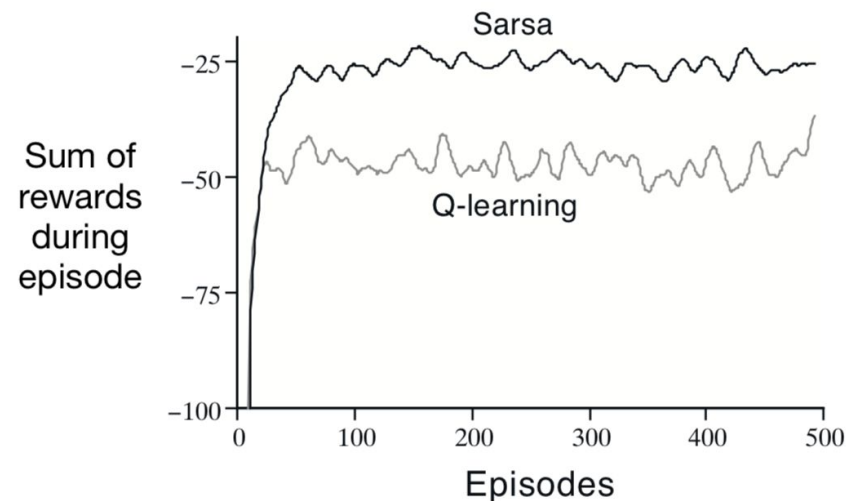
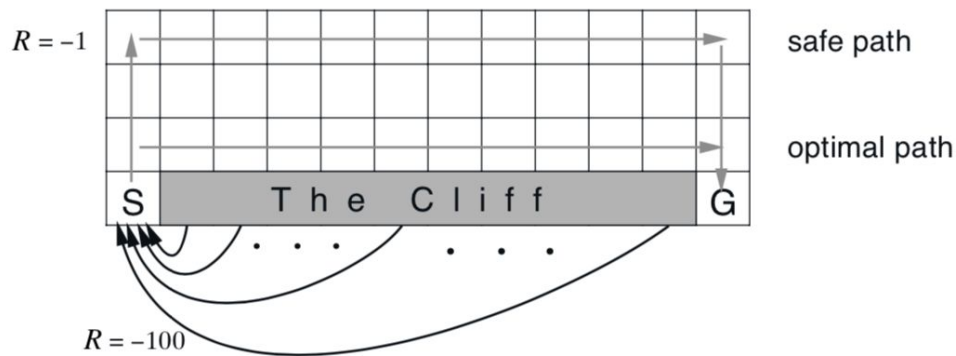
We could learn the optimal policy by improving on a different (greedy) policy. Therefore Q-learning is considered to be an off-policy control method.

Difference between SARSA & Q-Learning

Example 6.6: Cliff Walking This gridworld example compares Sarsa and Q-learning, highlighting the difference between on-policy (Sarsa) and off-policy (Q-learning) methods. Consider the gridworld shown in the upper part of Figure 6.4. This is a standard undiscounted, episodic task, with start and goal states, and the usual actions causing movement up, down, right, and left. Reward is -1 on all transitions except those into the region marked “The Cliff.” Stepping into this region incurs a reward of -100 and sends the agent instantly back to the start.

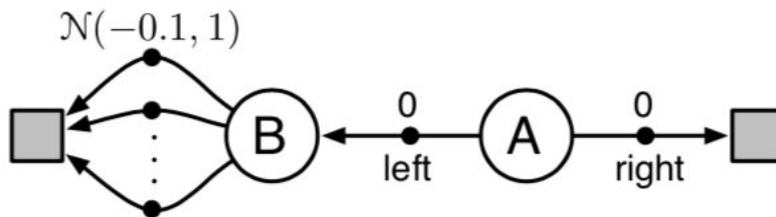


The lower part of Figure 6.4 shows the performance of the Sarsa and Q-learning methods with ε -greedy action selection, $\varepsilon = 0.1$. After an initial transient, Q-learning learns values for the optimal policy, that which travels right along the edge of the cliff. Unfortunately, this results in its occasionally falling off the cliff because of the ε -greedy action selection. Sarsa, on the other hand, takes the action selection into account and learns the longer but safer path through the upper part of the grid. Although Q-learning actually learns the values of the optimal policy, its on-line performance is worse than that of Sarsa, which learns the roundabout policy. Of course, if ε were gradually reduced, then both methods would asymptotically converge to the optimal policy.

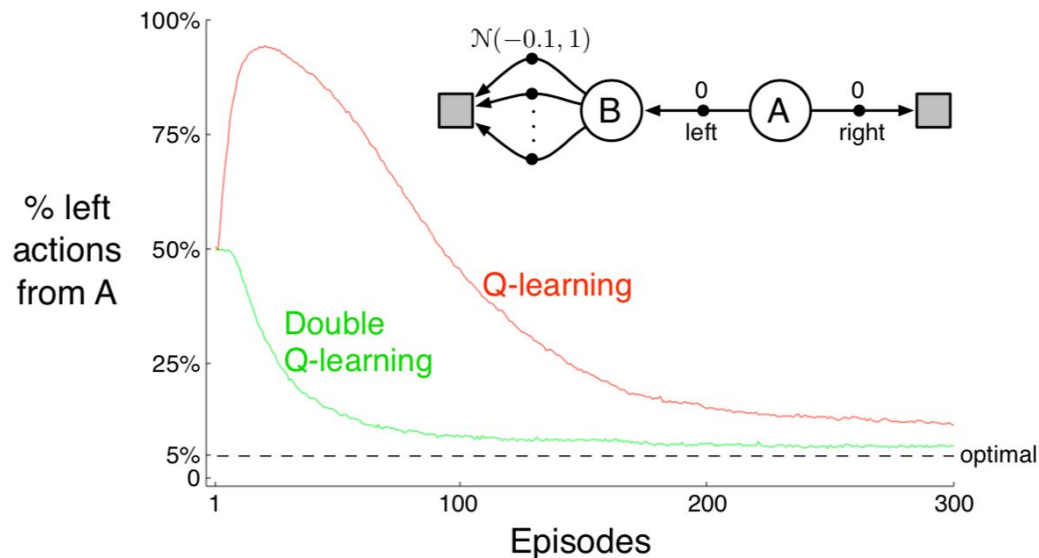


Effects of Maximization Bias*

Consider a MDP with two non-terminal states A and B. Episodes always start in A with a choice between two actions, **left** and **right**. The **right** action transitions immediately to the terminal state with a reward and return of zero. The **left** action transitions to B, also with a reward and return of zero, from which there are many possible actions all of which cause immediate termination with a reward drawn from a normal distribution with mean -0.1 and variance 1.0 . Thus, the expected return for any trajectory starting with **left** is -0.1 , and thus taking **left** in state A is always a mistake. Nevertheless, our control methods may favor **left** because of maximization bias making B appear to have a positive value.



* Adapted from Sutton & Barto Example 6.7



Q-learning with ϵ -greedy action selection initially learns to strongly favor the left action on this example. Even at asymptote, Q-learning takes the left action about 5% more often than is optimal at our parameter settings ($\epsilon = 0.1$, $\alpha = 0.1$, and $\gamma = 1$).

* Adapted from Sutton & Barto Example 6.7

Model-free Evaluation (Monte Carlo)

Algorithm 2 First-Visit Monte Carlo Policy Evaluation

```
1: procedure FIRST-VISIT-MONTE-CARLO( $h_1, \dots, h_j$ )
2:   For all states  $s$ ,  $N(s) \leftarrow 0$ ,  $S(s) \leftarrow 0$ ,  $V(s) \leftarrow 0$ 
3:   for each episode  $h_j$  do
4:     for  $t = 1, \dots, L_j$  do
5:       if  $s_{j,t} \neq s_{j,u}$  for  $u < t$  then
6:          $N(s_{j,t}) \leftarrow N(s_{j,t}) + 1$ 
7:          $S(s_{j,t}) \leftarrow S(s_{j,t}) + G_{j,t}$ 
8:          $V^\pi(s_{j,t}) \leftarrow S(s_{j,t})/N(s_{j,t})$ 
9:   return  $V^\pi$ 
```

Algorithm 3 Every-Visit Monte Carlo Policy Evaluation

```
1: procedure EVERY-VISIT-MONTE-CARLO( $h_1, \dots, h_j$ )
2:   For all states  $s$ ,  $N(s) \leftarrow 0$ ,  $S(s) \leftarrow 0$ ,  $V(s) \leftarrow 0$ 
3:   for each episode  $h_j$  do
4:     for  $t = 1, \dots, L_j$  do
5:        $N(s_{j,t}) \leftarrow N(s_{j,t}) + 1$ 
6:        $S(s_{j,t}) \leftarrow S(s_{j,t}) + G_{j,t}$ 
7:        $V^\pi(s_{j,t}) \leftarrow S(s_{j,t})/N(s_{j,t})$ 
8:   return  $V^\pi$ 
```

Algorithm 4 Incremental First-Visit Monte Carlo Policy Evaluation

```
1: procedure INCREMENTAL-FIRST-VISIT-MONTE-CARLO( $\alpha, h_1, \dots, h_j$ )
2:   For all states  $s$ ,  $N(s) \leftarrow 0$ ,  $V(s) \leftarrow 0$ 
3:   for each episode  $h_j$  do
4:     for  $t = 1, \dots, \text{terminal}$  do
5:       if  $s_{j,t} \neq s_{j,u}$  for  $u < t$  then
6:          $N(s_{j,t}) \leftarrow N(s_{j,t}) + 1$ 
7:          $V^\pi(s_{j,t}) \leftarrow V^\pi(s) + \alpha(G_{j,t} - V^\pi(s))$ 
8:   return  $V^\pi$ 
```

Algorithm 5 Incremental Every-Visit Monte Carlo Policy Evaluation

```
1: procedure INCREMENTAL-EVERY-VISIT-MONTE-CARLO( $\alpha, h_1, \dots, h_j$ )
2:   For all states  $s$ ,  $N(s) \leftarrow 0$ ,  $V(s) \leftarrow 0$ 
3:   for each episode  $h_j$  do
4:     for  $t = 1, \dots, \text{terminal}$  do
5:        $N(s_{j,t}) \leftarrow N(s_{j,t}) + 1$ 
6:        $V^\pi(s_{j,t}) \leftarrow V^\pi(s) + \alpha(G_{j,t} - V^\pi(s))$ 
7:   return  $V^\pi$ 
```

Model-free Evaluation (Temporal Difference)

Algorithm 5 Incremental Every-Visit Monte Carlo Policy Evaluation

```
1: procedure INCREMENTAL-EVERY-VISIT-MONTE-CARLO( $\alpha, h_1, \dots, h_j$ )
2:   For all states  $s$ ,  $N(s) \leftarrow 0$ ,  $V(s) \leftarrow 0$ 
3:   for each episode  $h_j$  do
4:     for  $t = 1, \dots, \text{terminal}$  do
5:        $N(s_{j,t}) \leftarrow N(s_{j,t}) + 1$ 
6:        $V^\pi(s_{j,t}) \leftarrow V^\pi(s) + \alpha(G_{j,t} - V^\pi(s))$ 
7:   return  $V^\pi$ 
```

Algorithm 6 TD Learning to evaluate policy π

```
1: procedure TDLEARNING(step size  $\alpha$ , number of trajectories  $n$ )
2:   For all states  $s$ ,  $V^\pi(s) \leftarrow 0$ 
3:   while  $n > 0$  do
4:     Begin episode  $E$  at state  $s$ 
5:     while  $n > 0$  and episode  $E$  has not terminated do
6:        $a \leftarrow$  action at state  $s$  under policy  $\pi$ 
7:       Take action  $a$  in  $E$  and observe reward  $r$ , next state  $s'$ 
8:        $V^\pi(s) \leftarrow V^\pi(s) + \alpha(R + \gamma V^\pi(s') - V^\pi(s))$ 
9:        $s \leftarrow s'$ 
10:   return  $V^\pi$ 
```

MC Control

```
1: Initialize  $Q(s, a) = 0, N(s, a) = 0 \forall (s, a)$ , Set  $\epsilon = 1, k = 1$ 
2:  $\pi_k = \epsilon\text{-greedy}(Q)$  // Create initial  $\epsilon$ -greedy policy
3: loop
4:   Sample  $k$ -th episode  $(s_{k,1}, a_{k,1}, r_{k,1}, s_{k,2}, \dots, s_{k,T})$  given  $\pi_k$ 
4:    $G_{k,t} = r_{k,t} + \gamma r_{k,t+1} + \gamma^2 r_{k,t+2} + \dots \gamma^{T_i-1} r_{k,T_i}$ 
5:   for  $t = 1, \dots, T$  do
6:     if First visit to  $(s, a)$  in episode  $k$  then
7:        $N(s, a) = N(s, a) + 1$ 
8:        $Q(s_t, a_t) = Q(s_t, a_t) + \frac{1}{N(s,a)}(G_{k,t} - Q(s_t, a_t))$ 
9:     end if
10:  end for
11:   $k = k + 1, \epsilon = 1/k$ 
12:   $\pi_k = \epsilon\text{-greedy}(Q)$  // Policy improvement
13: end loop
```

Convergence of MC Control

Theorem

GLIE Monte-Carlo control converges to the optimal state-action value function $Q(s, a) \rightarrow Q^*(s, a)$

Definition of GLIE

- All state-action pairs are visited an infinite number of times

$$\lim_{i \rightarrow \infty} N_i(s, a) \rightarrow \infty$$

- Behavior policy converges to greedy policy

- A simple GLIE strategy is ϵ -greedy where ϵ is reduced to 0 with the following rate: $\epsilon_i = 1/i$

Proof of Monotonic ϵ -greedy Policy Improvement

$$\begin{aligned} Q^{\pi_i}(s, \pi_{i+1}(s)) &= \sum_{a \in A} \pi_{i+1}(a|s) Q^{\pi_i}(s, a) \\ &= \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \max_{a'} Q^{\pi_i}(s, a') \\ &= \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \max_{a'} Q^{\pi_i}(s, a') \frac{1 - \epsilon}{1 - \epsilon} \\ &= \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \max_{a'} Q^{\pi_i}(s, a') \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1 - \epsilon} \\ &= \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1 - \epsilon} \max_{a'} Q^{\pi_i}(s, a') \\ &\geq \frac{\epsilon}{|A|} \sum_{a \in A} Q^{\pi_i}(s, a) + (1 - \epsilon) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1 - \epsilon} Q^{\pi_i}(s, a) \\ &= \sum_{a \in A} \pi_i(a|s) Q^{\pi_i}(s, a) \\ &= V^{\pi_i}(s) \end{aligned}$$

SARSA

```
1: procedure SARSA( $\epsilon, \alpha_t$ )
2:   Initialize  $Q(s, a)$  for all  $s \in S, a \in A$  arbitrarily except  $Q(\text{terminal}, \cdot) = 0$ 
3:    $\pi \leftarrow \epsilon$ -greedy policy with respect to  $Q$ 
4:   for each episode do
5:     Set  $s_1$  as the starting state
6:     Choose action  $a_1$  from policy  $\pi(s_1)$ 
7:     loop until episode terminates
8:       Take action  $a_t$  and observe reward  $r_t$  and next state  $s_{t+1}$ 
9:       Choose action  $a_{t+1}$  from policy  $\pi(s_{t+1})$ 
10:       $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$ 
11:       $\pi \leftarrow \epsilon$ -greedy with respect to  $Q$  (policy improvement)
12:       $t \leftarrow t + 1$ 
13:   Return  $Q, \pi$ 
```

Q-learning

```
1: procedure Q-LEARNING( $\epsilon, \alpha, \gamma$ )
2:   Initialize  $Q(s, a)$  for all  $s \in S, a \in A$  arbitrarily except  $Q(\text{terminal}, \cdot) = 0$ 
3:    $\pi \leftarrow \epsilon$ -greedy policy with respect to  $Q$ 
4:   for each episode do
5:     Set  $s_1$  as the starting state
6:      $t \leftarrow 1$ 
7:     loop until episode terminates
8:       Sample action  $a_t$  from policy  $\pi(s_t)$ 
9:       Take action  $a_t$  and observe reward  $r_t$  and next state  $s_{t+1}$ 
10:       $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))$ 
11:       $\pi \leftarrow \epsilon$ -greedy policy with respect to  $Q$  (policy improvement)
12:       $t \leftarrow t + 1$ 
13:   return  $Q, \pi$ 
```