CP8319/CPS824
Lecture 10

Instructor: Nariman Farsad

* Some of the slides in this deck are adopted from courses offered David Silver, Emma Brunskill, and Sergey Levine.

Today's Agenda

1. **Review of Previous Lectures**

2. Model-Free Control (Monte Carlo)

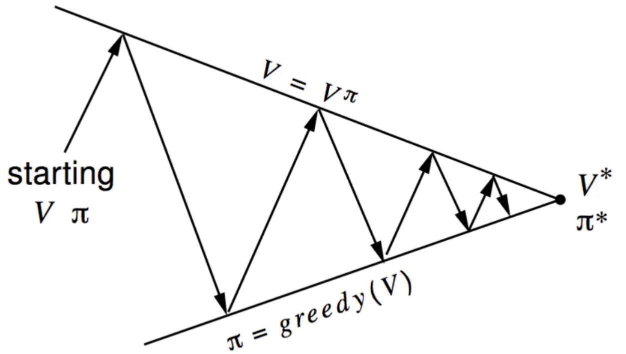
3. Model-Free Control (Temporal Difference)

4. Model-Free Control (Q-Learning)

Policy Iteration: Known Model

Set $i = 0$
Initialize $\pi_0(s)$ randomly for all states s
While $i == 0$ or $\| \pi_i - \pi_{i-1} \|_1 > 0$ (L1-norm, measures if the policy changed for any state):

- $v^{\pi_i} \leftarrow$ MDP value function **policy evaluation** of π_i
- $\pi_{i+1} \leftarrow$ **Policy improvement** on v^{π_i}
- $i = i + 1$



Policy evaluation Estimate v_π
Iterative policy evaluation

Policy improvement Generate $\pi' \geq \pi$
Greedy policy improvement

Model Free Policy Iteration

Set $i = 0$

Initialize $\pi_0(s)$ randomly for all states s

While $i == 0$ or $\| \pi_i - \pi_{i-1} \|_1 > 0$ (L1-norm, measures if the policy changed for any state):

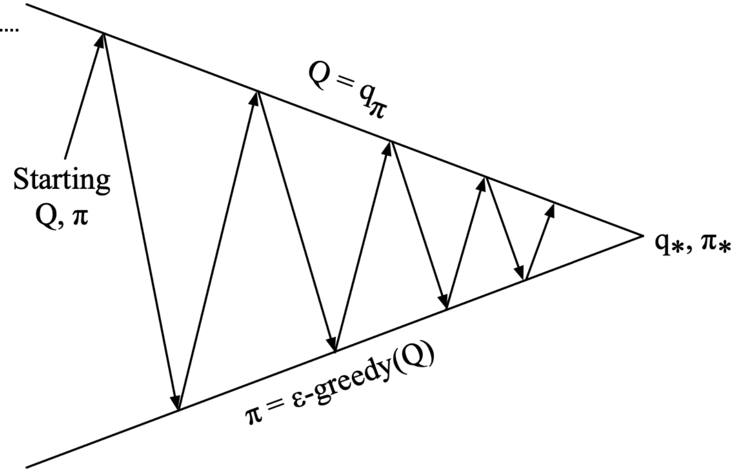
- $Q^{\pi_i} \leftarrow$ MDP value function **MC policy Q evaluation** of π_i
- $\pi_{i+1} \leftarrow$ **ϵ-greedy Policy improvement** on Q^{π_i}
- $i = i + 1$

greedy(Q)

$$\boxed{\pi_{i+1}(a|s) = \begin{cases} 1, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi_i}(s, a') \\ 0, & otherwise \end{cases}}$$

ϵ-greedy(Q)

$$\boxed{\pi_{i+1}(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi_i}(s, a') \\ \epsilon/m, & otherwise \end{cases}}$$



Policy evaluation Monte-Carlo policy evaluation, $Q = q_\pi$

Policy improvement ϵ-greedy policy improvement

ϵ-Greedy and Greedy Example

- Let's say in a state s_1 we can take 3 actions a_1, a_2, a_3.
- Assume that $a_1 = \underset{a' \in \mathcal{A}}{\operatorname{argmax}} Q^{\pi_i}(s, a')$ and $\epsilon = 0.5$.
- What are the ϵ-greedy and greedy policies $\pi(a|s_1)$?

greedy(Q)

$$\pi_{i+1}(a|s) = \begin{cases} 1, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}} Q^{\pi_i}(s, a') \\ 0, & otherwise \end{cases}$$

ϵ-greedy(Q)

$$\pi_{i+1}(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon\,, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}} Q^{\pi_i}(s, a') \\ \epsilon/m \qquad\quad , & otherwise \end{cases}$$

Today's Agenda

1. Review of Previous Lectures
2. **Model-Free Control (Monte Carlo)**
3. Model-Free Control (Temporal Difference)
4. Model-Free Control (Q-Learning)

Model Free Policy Iteration

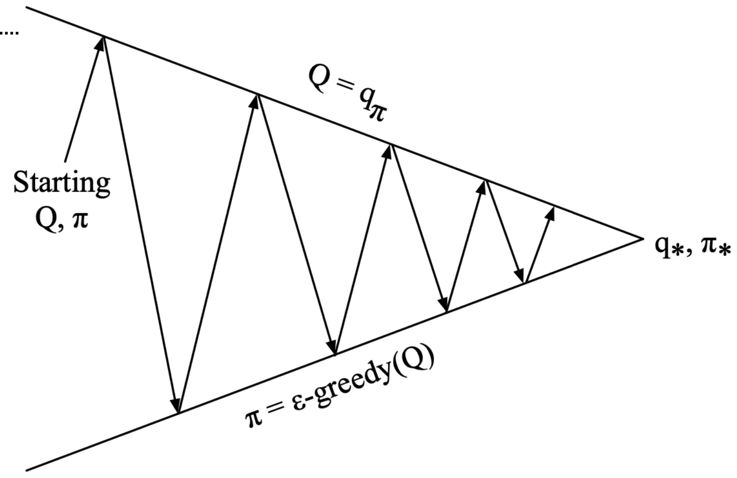
Set $i = 0$
Initialize $\pi_0(s)$ randomly for all states s
While $i \;==\; 0$ or $\| \pi_i - \pi_{i-1} \|_1 > 0$ (L1-norm, measures if the policy changed for any state):

- $Q^{\pi_i} \leftarrow$ MDP value function **MC policy Q evaluation** of π_i
- $\pi_{i+1} \leftarrow$ **ϵ-greedy Policy improvement** on Q^{π_i}
- $i = i + 1$

ϵ-greedy(Q)

$$\boxed{\pi_{i+1}(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon\,, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi_i}(s, a') \\ \epsilon/m\,, & otherwise \end{cases}}$$

Does ϵ-greedy step provably improve policy?



Policy evaluation Monte-Carlo policy evaluation, $Q = q_\pi$
Policy improvement ϵ-greedy policy improvement

Monotonic ϵ-Greedy Policy Improvement

Theorem

For any ϵ-greedy policy π_i, the ϵ-greedy policy w.r.t. Q^{π_i}, π_{i+1} is a monotonic improvement $V^{\pi_{i+1}} \geq V^{\pi_i}$

$$
\begin{aligned}
Q^{\pi_i}(s, \pi_{i+1}(s)) &= \sum_{a \in A} \pi_{i+1}(a|s) Q^{\pi_i}(s, a) \\
&= (\epsilon/|A|) \left[\sum_{a \in A} Q^{\pi_i}(s, a) \right] + (1 - \epsilon) \max_a Q^{\pi_i}(s, a) \\
&= (\epsilon/|A|) \left[\sum_{a \in A} Q^{\pi_i}(s, a) \right] + (1 - \epsilon) \max_a Q^{\pi_i}(s, a) \frac{1 - \epsilon}{1 - \epsilon} \\
&= (\epsilon/|A|) \left[\sum_{a \in A} Q^{\pi_i}(s, a) \right] + (1 - \epsilon) \max_a Q^{\pi_i}(s, a) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1 - \epsilon} \\
&\geq \frac{\epsilon}{|A|} \left[\sum_{a \in A} Q^{\pi_i}(s, a) \right] + (1 - \epsilon) \sum_{a \in A} \frac{\pi_i(a|s) - \frac{\epsilon}{|A|}}{1 - \epsilon} Q^{\pi_i}(s, a) \\
&= \sum_{a \in A} \pi_i(a|s) Q^{\pi_i}(s, a) = V^{\pi_i}(s)
\end{aligned}
$$

Each step of policy improvement improves policy or keeps it the same

Model Free Policy Iteration

Set $i = 0$

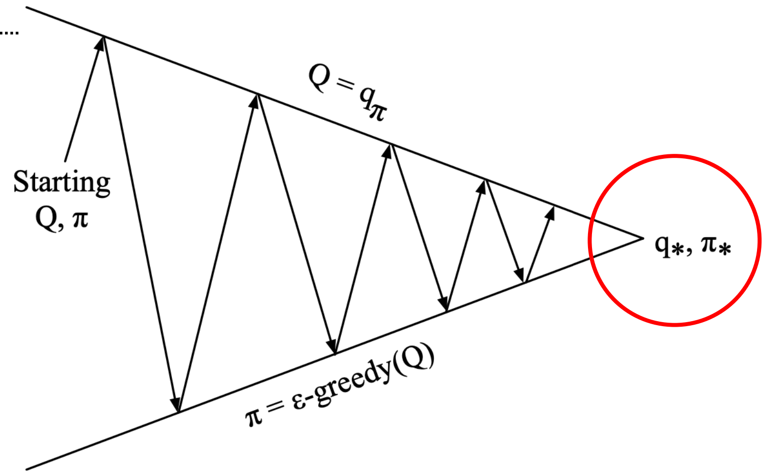
Initialize $\pi_0(s)$ randomly for all states s

While $i == 0$ or $\| \pi_i - \pi_{i-1} \|_1 > 0$ (L1-norm, measures if the policy changed for any state):

- $Q^{\pi_i} \leftarrow$ MDP value function **MC policy Q evaluation** of π_i
- $\pi_{i+1} \leftarrow$ **ϵ-greedy Policy improvement** on Q^{π_i}
- $i = i + 1$

ϵ-greedy(Q)

$$\boxed{\pi_{i+1}(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon\,, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi_i}(s, a') \\ \epsilon/m\,, & otherwise \end{cases}}$$



Policy evaluation Monte-Carlo policy evaluation, $Q = q_\pi$

Policy improvement ϵ-greedy policy improvement

Although ϵ-greedy step provably improve policy, does it converge to the optimal policy?

Greedy in the Limit of Infinite Exploration (GLIE)

Definition of GLIE

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

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

- Behavior policy (policy used to act in the world) converges to greedy policy
  $\lim_{i \to \infty} \pi(a|s) \to \arg\max_a Q(s, a)$ with probability 1

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

GLIE Monte Carlo Control

Theorem

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

Monte Carlo Online Control/On Policy Improvement

Initialize $Q(s,a)=0, N(s,a)=0\ \forall(s,a)$, Set $\epsilon=1,\ k=1$
$\pi_k = \epsilon$-greedy(Q) // Create initial ϵ-greedy policy
loop
    Sample k-th episode $(s_{k,1}, a_{k,1}, r_{k,1}, s_{k,2}, \ldots, s_{k,T})$ given π_k
    $G_{k,t} = r_{k,t} + \gamma r_{k,t+1} + \gamma^2 r_{k,t+2} + \cdots \gamma^{T_i - 1} r_{k,T_i}$
    for $t = 1, \ldots, T$ **do**
        if First visit to (s,a) in episode k **then**
            $N(s,a) = N(s,a) + 1$
            $Q(s_t,a_t) = Q(s_t,a_t) + \frac{1}{N(s,a)}(G_{k,t} - Q(s_t,a_t))$
        end if
    end for
    $k = k+1,\ \epsilon = 1/k$
    $\pi_k = \epsilon$-greedy(Q) // Policy improvement
end loop

MC for On Policy Control Example

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

- Robot with two actions
  - R(-, a_1) = [1 0 0 0 0 0 +10] and R(-, a_2) = [0 0 0 0 0 0 +5]
- $\pi(s) = a_1 \; \forall s, \gamma = 1, \epsilon = 0.5$. Any action from s1 and s7 terminates episode
- Sample episode = $(s_3, a_1, 0, s_2, a_2, 0, s_3, a_1, 0, s_2, a_2, 0, s_1, a_1, 1$ terminal)
- First visit MC estimate of Q of each (s, a) pair?
  - $Q^\pi(-, a_1) = [1\ 0\ 1\ 0\ 0\ 0\ 0], \;\; Q^\pi(-, a_2) = [0\ 1\ 0\ 0\ 0\ 0\ 0]$
- What is $\pi(s) = \arg\max_a Q^\pi(s, a) \;\; \forall \mathcal{S}$?
- What is new ϵ-greedy policy, if $k = 3, \;\; \epsilon = 1/k$? Give an example for $\pi(s_1)$.

Today’s Agenda

1. Review of Previous Lectures
2. Model-Free Control (Monte Carlo)
3. **Model-Free Control (Temporal Difference)**
4. Model-Free Control (Q-Learning)

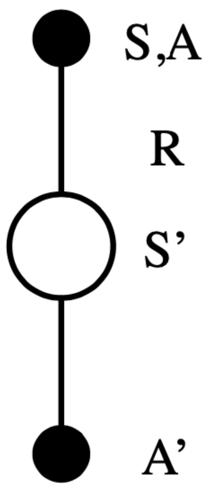
Why TD?

- Temporal-difference (TD) learning has several advantages over Monte-Carlo (MC)
  - Lower variance
  - Online learning
  - Incomplete sequences

- Natural idea: use TD instead of MC in our control loop
  - Apply TD to $Q(s, a)$
  - Use ϵ-greedy policy improvement
  - Update every time-step

Updating Action-Value Functions with SARSA

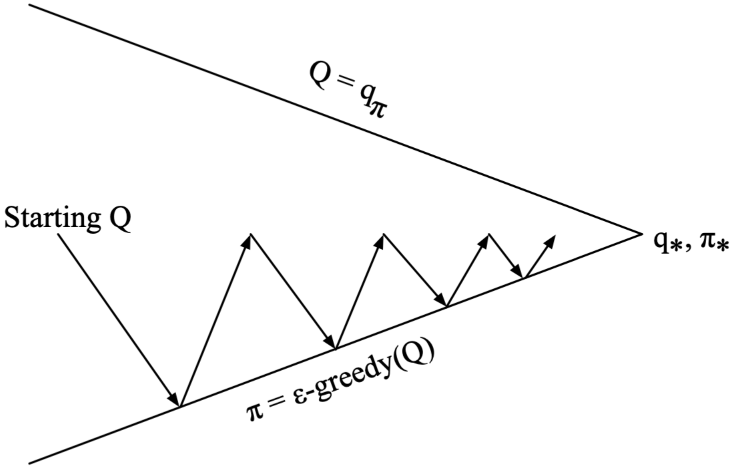
- Stat, Action, Reward, Next State, Next Action (SARSA) is the TD methods that can be used to evaluate the Q-value



$$Q(S,A) \leftarrow Q(S,A) + \alpha \left(R + \gamma Q(S',A') - Q(S,A) \right)$$

SARSA For On-Policy Control

Set initial ϵ-greedy policy π, $t = 0$, initial state $s_t = s_0$
Take $a_t \sim \pi(s_t)$ // Sample action from policy
Observe (r_t, s_{t+1})
loop
  Take action $a_{t+1} \sim \pi(s_{t+1})$
  Observe (r_{t+1}, s_{t+2})
  $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
  $\pi(s_t) = \arg\max_a Q(s_t, a)$ w.prob $1 - \epsilon$, else random
  $t = t + 1$
end loop



Every time-step:

Policy evaluation Sarsa, $Q \approx q_\pi$

Policy improvement ϵ-greedy policy improvement

Does SARSA converge? Does it converge for any choice of α?

Convergence Properties of SARSA

Theorem

SARSA for finite-state and finite-action MDPs converges to the optimal action-value, $Q(s,a) \to Q^*(s,a)$, under the following conditions:

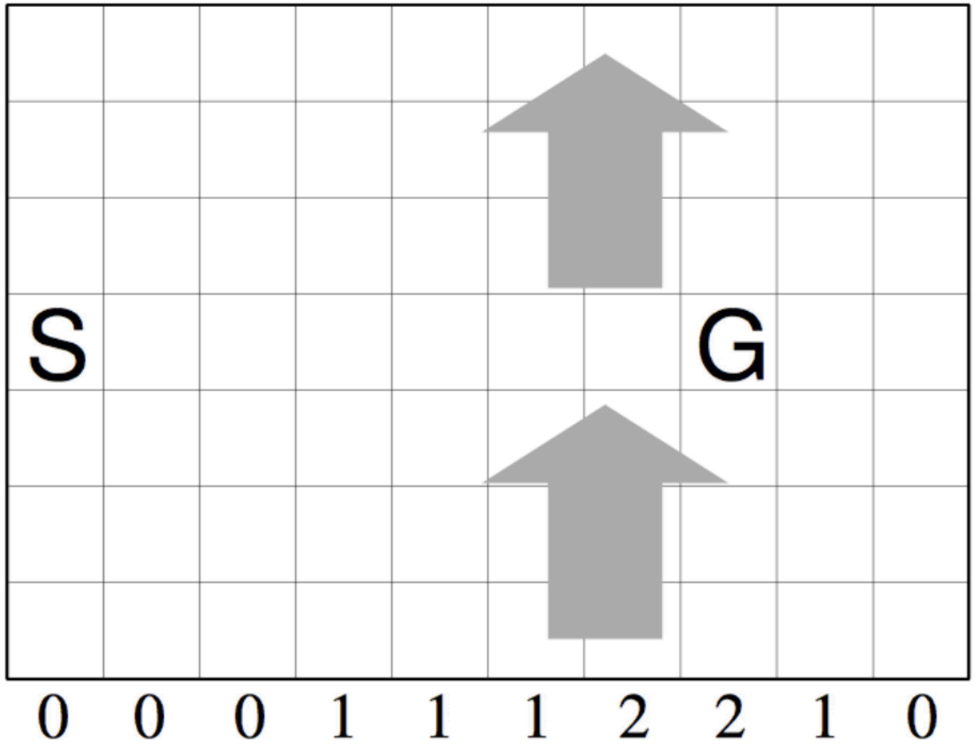
1. The policy sequence $\pi_t(a|s)$ satisfies the condition of GLIE
2. The step-sizes α_t satisfy the Robbins-Munro sequence such that

$$\sum_{t=1}^{\infty} \alpha_t = \infty$$

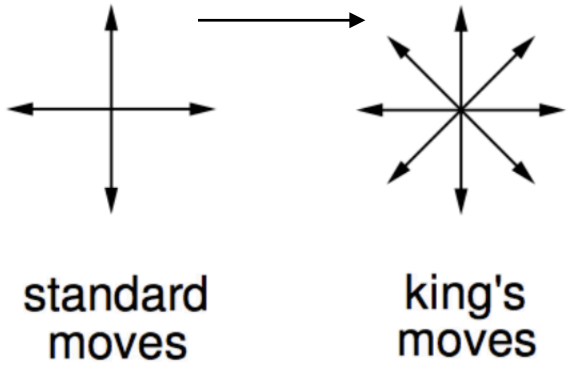
$$\sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

- For ex. $\alpha_t = \frac{1}{t}$ satisfies the above condition.
- Would one want to use a step size choice that satisfies the above in practice? Likely not.

Windy Gridworld Example

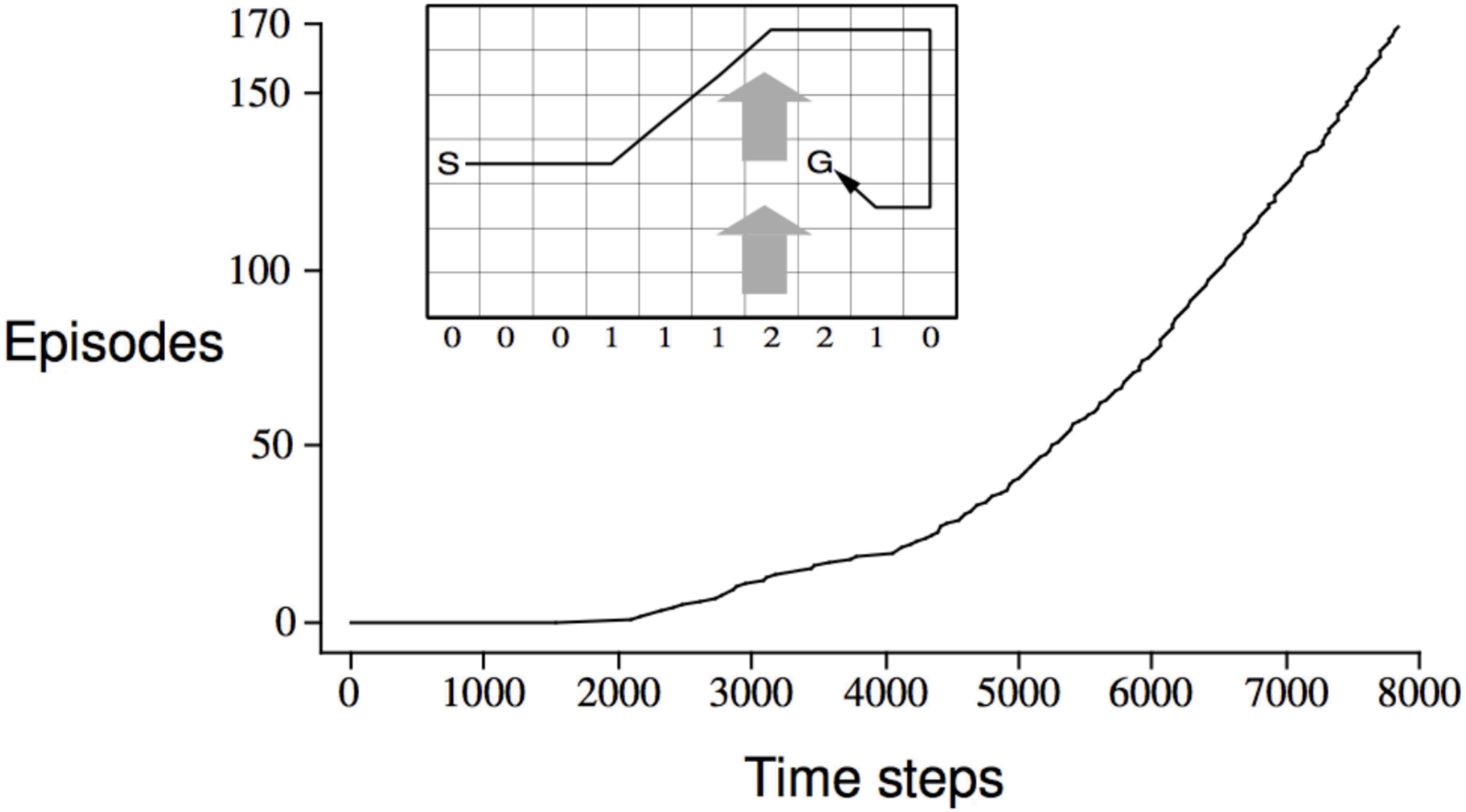


Because of the wind



- Reward = -1 per time-step until reaching goal
- Undiscounted $\gamma = 1$

SARSA on the Windy Gridworld



SARSA Example

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Set initial ϵ-greedy policy π, $t = 0$, initial state $s_t = s_0$
Take $a_t \sim \pi(s_t)$ // Sample action from policy
Observe (r_t, s_{t+1})
loop
    Take action $a_{t+1} \sim \pi(s_{t+1})$
    Observe (r_{t+1}, s_{t+2})
    $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
    $\pi(s_t) = \arg\max_a Q(s_t, a)$ w.prob $1 - \epsilon$, else random
    $t = t + 1$
end loop

- Initialize $\gamma = 1, \epsilon = 1/k, k = 1$, and $\alpha = 0.5$
- Initialize $Q(-, a_1) = [1, 0, 0, 0, 0, 0, 10]$, $Q(-, a_2) = [1, 0, 0, 0, 0, 0, 5]$
- Assume we observe tuple by acting in the word according to SARSA:

  $(s_6, a_1, 0, s_7, a_2)$
- What is $Q(s_6, a_1)$ after a SARSA update?

Today’s Agenda

1. Review of Previous Lectures

2. Model-Free Control (Monte Carlo)

3. Model-Free Control (Temporal Difference)

4. **Model-Free Control (Q-Learning)**

Off-Policy Learning

- Evaluate *target policy* $\pi(a|s)$ to compute $v^{\pi}(s)$ or $q^{\pi}(s,a)$
- While following *behavior policy* $\mu(a|s)$

$$\{s_1, a_1, r_2, \dots, s_T\} \sim \mu$$

- Why is this important?
  - Learn from observing humans or other agents
  - Re-use experience generated from old policies $\pi_1, \pi_2, \dots, \pi_{t-1}$
  - Learn about *optimal* policy, while following *exploratory* policy
  - Learn about multiple policies while following one policy

Q-Learning: Learning the Optimal State-Action Value

- SARSA is an on-policy learning algorithm
  - SARSA estimates the value of the current behavior policy (policy used to take actions in the world)
  - i.e., both *target policy* and *behavior policy* are the same

- For MDP:
  - We know that optimal policy π^* (i.e., *target policy*) is deterministic (i.e., greedy)
  - But we need the *behavior policy* to be stochastic (i.e., ϵ-greedy) to explore

- Can we directly estimate the value of the *greedy target policy*, while acting with an *ϵ-greedy behavior policy*?
  - Yes! Q-learning, an off-policy RL algorithm

Q-Learning: Learning the Optimal State-Action Value

Q-learning: off-policy learning of action-values $Q(s, a)$

- Next action is chosen using *ϵ-greedy behavior policy* $a_{t+1} \sim \mu(\cdot\,|s_t)$ where

$$\mu(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon\,, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q(s, a') \\ \epsilon/m\,, & otherwise \end{cases}$$

- But when updating Q-values consider alternative successor action according to *greedy target policy* $a'' \sim \pi(\cdot\,|s_t)$ where

$$\pi(a|s) = \begin{cases} 1, & \text{if } a = \underset{a' \in \mathcal{A}}{\operatorname{argmax}}\, Q(s, a') \\ 0, & otherwise \end{cases}$$

or since this is deterministic:

$$a'' = \pi(s) = \arg \max_{a' \in \mathcal{A}} Q(s, a')$$

Q-Learning: Learning the Optimal State-Action Value

$a_{t+1} \sim \mu(\cdot\,|s_t)$

$a'' \sim \pi(\cdot\,|s_t)$ or since deterministic $a'' = \pi(s_t) = \arg\max_{a' \in \mathcal{A}} Q(s_t, a')$

SARSA Update:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

Q-Learning Update:

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

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

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

Q-Learning Algorithm

Initialize $Q(s, a), \forall s \in S, a \in A$ $t = 0$, initial state $s_t = s_0$
Set π_b to be ϵ-greedy w.r.t. Q
loop
    Take $a_t \sim \pi_b(s_t)$ // Sample action from policy
    Observe (r_t, s_{t+1})
    $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t))$
    $\pi(s_t) = \arg\max_a Q(s_t, a)$ w.prob $1 - \epsilon$, else random
    $t = t + 1$
end loop

Q-Learning Convergence

- What conditions are sufficient to ensure that Q-learning with ε-greedy exploration converges to optimal Q^*?
  - Visit all (s, a) pairs infinitely often, and the step-sizes α_t satisfy the Robbins-Munro sequence. Note: the algorithm does not have to be greedy in the limit of infinite exploration (GLIE) to satisfy this (could keep ϵ large).

- What conditions are sufficient to ensure that Q-learning with ε-greedy exploration converges to optimal π^*?
  - The algorithm is GLIE, along with the above requirement to ensure the Q value estimates converge to the optimal Q.

Q-learning Example

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Initialize $Q(s,a), \forall s \in S, a \in A$ $t = 0$, initial state $s_t = s_0$
Set π_b to be ϵ-greedy w.r.t. Q
loop
  Take $a_t \sim \pi_b(s_t)$ // Sample action from policy
  Observe (r_t, s_{t+1})
  $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t))$
  $\pi(s_t) = \arg\max_a Q(s_t, a)$ w.prob $1 - \epsilon$, else random
  $t = t + 1$
end loop

- Initialize $\gamma = 1, \epsilon = 1/k, k = 1$, and $\alpha = 0.5$
- Initialize $Q(-, a_1) = [1, 0, 0, 0, 0, 0, 10]$, $Q(-, a_2) = [1, 0, 0, 0, 0, 0, 5]$
- Assume we observe tuple by acting in the word: $(s_6, a_1, 0, s_7)$
- What is $Q(s_6, a_1)$ after a Q-Learning update? How does this compare to SARSA?

Cliff Walking Example

- Q-learning is more optimistic than SARSA by taking the max action

- SARSA is preferred in environments where you must be more cautious
  - Environments with large negative rewards
  - A real robot acting in real world, which might break or cause damage

- Q-Learning is better in environments where you do not need to be cautious
  - Where we do not have many negative results
  - Training in simulation environment

- Both algorithms eventually converge