

# Lecture 4: Model Free Control

Emma Brunskill

CS234 Reinforcement Learning.

Winter 2019

- Structure closely follows much of David Silver's Lecture 5. For additional reading please see SB Sections 5.2-5.4, 6.4, 6.5, 6.7

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias

# Class Structure

- Last time: Policy evaluation with no knowledge of how the world works (MDP model not given)
- This time: Control (making decisions) without a model of how the world works
- Next time: Value function approximation

# Evaluation to Control

- Last time: how good is a specific policy?
  - Given no access to the decision process model parameters
  - Instead have to estimate from data / experience
- Today: how can we learn a good policy?

# Recall: Reinforcement Learning Involves

- Optimization ]
- Delayed consequences (planning)
- Exploration ]
- Generalization not yet

# Today: Learning to Control Involves

- Optimization: Goal is to identify a policy with high expected rewards (similar to Lecture 2 on computing an optimal policy given decision process models)
- Delayed consequences: May take many time steps to evaluate whether an earlier decision was good or not
- Exploration: Necessary to try different actions to learn what actions can lead to high rewards

# Today: Model-free Control

- Generalized policy improvement
- Importance of exploration
- Monte Carlo control
- Model-free control with temporal difference (SARSA, Q-learning)
- Maximization bias

# Model-free Control Examples

- Many applications can be modeled as a MDP: Backgammon, Go, Robot locomotion, Helicopter flight, Robocup soccer, Autonomous driving, Customer ad selection, Invasive species management, Patient treatment
- For many of these and other problems either:
  - MDP model is unknown but can be sampled
  - MDP model is known but it is computationally infeasible to use directly, except through sampling



# On and Off-Policy Learning

- On-policy learning
  - Direct experience
  - Learn to estimate and evaluate a policy from experience obtained from following that policy
- Off-policy learning
  - Learn to estimate and evaluate a policy using experience gathered from following a different policy

Off-policy: we can combine experience for trying out different things to try to learn about something we didn't do by itself.

Imagine you have a case where there's only a single state for now. You experienced  $s_1, a_1, s_1, a_2$ ; and  $s_1, a_2, s_1, a_2$ . So you'd like to be able to kind of combine between these experiences so you could learn about doing this:  $s_1, a_1, s_1, a_2$ , even though you've never done that in the world.

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias

# Recall Policy Iteration

- Initialize policy  $\pi$
- Repeat:

- Policy evaluation: compute  $V^\pi$
- Policy improvement: update  $\pi$

$\pi(s) = a \quad \forall s$  *random*

$|A|^{|\mathcal{S}|}$   
monotonic  
policy  
improvement

$$\pi'(s) = \arg \max_a \underbrace{R(s, a)} + \gamma \sum_{s' \in \mathcal{S}} \underbrace{P(s'|s, a)} V^\pi(s') = \arg \max_a Q^\pi(s, a)$$

- Now want to do the above two steps without access to the true dynamics and reward models *How? We're gonna to compute Q.*
- Last lecture introduced methods for model-free policy evaluation

# Model Free Policy Iteration

- Initialize policy  $\pi$
- Repeat:
  - Policy evaluation: compute  $Q^\pi$
  - Policy improvement: update  $\pi$

$$Q(s, a)$$

# MC for On Policy Q Evaluation

Not focus only on state, but  $(s,a)$  tuple

$$V : \mathcal{N}(s) \quad G(s)$$

Initialize  $\underline{N(s,a)} = 0$ ,  $\underline{G(s,a)} = 0$ ,  $\underline{Q^\pi(s,a)} = 0$ ,  $\forall s \in S, \forall a \in A$

Loop

- Using policy  $\pi$  sample episode  $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \dots, s_{i,T_i}$   
 $(s,a)$
- $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \dots + \gamma^{T_i-1} r_{i,T_i}$
- For each **state,action**  $(s,a)$  visited in episode  $i$ 
  - For **first or every** time  $t$  that  $(s,a)$  is visited in episode  $i$ 
    - $N(s,a) = N(s,a) + 1$ ,  $G(s,a) = G(s,a) + G_{i,t}$
    - Update estimate  $\underline{Q^\pi(s,a)} = G(s,a)/N(s,a)$

# Model-free Generalized Policy Improvement

$$\pi(s) \rightarrow a$$

- Given an estimate  $Q^{\pi_i}(s, a) \forall s, a$
- Update new policy

$$\underline{\pi_{i+1}}(s) = \arg \max_a Q^{\pi_i}(s, a) \quad (1)$$

# Model-free Policy Iteration

- Initialize policy  $\pi$
- Repeat:
  - Policy evaluation: compute  $Q^\pi$  Nothing about convergence
  - Policy improvement: update  $\pi$  given  $Q^\pi$
- May need to modify policy evaluation: need exploration
  - If  $\pi$  is deterministic, can't compute  $Q(s, a)$  for any  $a \neq \pi(s)$
- How to interleave policy evaluation and improvement?
  - Policy improvement is now using an estimated Q

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration**
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias



# Policy Evaluation with Exploration

$$Q^\pi(s, a) \quad \forall s \forall a$$

- Want to compute a model-free estimate of  $Q^\pi$
- In general seems subtle
  - Need to try all  $(s, a)$  pairs but then follow  $\pi$
  - Want to ensure resulting estimate  $Q^\pi$  is good enough so that policy improvement is a monotonic operator
- For certain classes of policies can ensure all  $(s, a)$  pairs are tried such that asymptotically  $Q^\pi$  converges to the true value

How do I make sure to visit all of those states ?

# $\epsilon$ -greedy Policies

- Simple idea to balance exploration and exploitation
- Let  $|A|$  be the number of actions
- Then an  $\epsilon$ -greedy policy w.r.t. a state-action value  $Q^\pi(s, a)$  is

$$\pi(a|s) = \begin{cases} \text{w/prob } 1 - \epsilon & \text{argmax}_a Q^\pi(s, a) \\ \text{else} & a \text{ with prob } \frac{\epsilon}{|A|} \end{cases}$$

*with probability  $1 - \epsilon$  to take the best action*

The nice thing is that it's pretty simple, and still sufficient.

# Check Your Understanding: MC for On Policy Q Evaluation

Initialize  $N(s, a) = 0$ ,  $G(s, a) = 0$ ,  $Q^\pi(s, a) = 0$ ,  $\forall s \in S$ ,  $\forall a \in A$

Loop

- Using policy  $\pi$  sample episode  $i = s_{i,1}, a_{i,1}, r_{i,1}, s_{i,2}, a_{i,2}, r_{i,2}, \dots, s_{i,T_i}$
- $G_{i,t} = r_{i,t} + \gamma r_{i,t+1} + \gamma^2 r_{i,t+2} + \dots + \gamma^{T_i-1} r_{i,T_i}$
- For each **state,action**  $(s, a)$  visited in episode  $i$ 
  - For **first or every** time  $t$  that  $(s, a)$  is visited in episode  $i$ 
    - $N(s, a) = N(s, a) + 1$ ,  $G(s, a) = G(s, a) + G_{i,t}$
    - Update estimate  $Q^\pi(s, a) = G(s, a) / N(s, a)$
- Mars rover with new actions:
  - $r(\underline{-}, a_1) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 + 10]$ ,  $r(\underline{-}, a_2) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 + 5]$ ,  $\gamma = 1$ .
- Assume current greedy  $\pi(s) = a_1 \ \forall s$ ,  $\epsilon = .5$
- Sample trajectory from  $\epsilon$ -greedy policy
- Trajectory =  $(s_3, a_1, 0, s_2, a_2, 0, s_3, a_1, 0, s_2, a_2, 0, s_1, a_1, 1, \text{terminal})$
- First visit MC estimate of  $Q$  of each  $(s, a)$  pair? 1st-visit

$$Q(\underline{-}, a_1) = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad Q(\underline{-}, a_2) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

# Monotonic<sup>1</sup> $\epsilon$ -greedy Policy Improvement

## Theorem

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

$$\begin{aligned}
 \underbrace{Q^{\pi_i}(s, \pi_{i+1}(s))}_{\substack{\text{random w/ prob } \epsilon \\ \swarrow \text{greedy}}} &= \sum_{a \in A} \underbrace{\pi_{i+1}(a|s)}_{\substack{\text{random w/ prob } \epsilon \\ \swarrow \text{greedy}}} Q^{\pi_i}(s, a) \\
 &= (\epsilon/|A|) \sum_{a \in A} Q^{\pi_i}(s, a) + (1-\epsilon) \max_a Q^{\pi_i}(s, a) = 1 \\
 &= \frac{\epsilon}{|A|} \sum_a Q^{\pi_i}(s, a) + (1-\epsilon) \max_a Q^{\pi_i}(s, a) \frac{1-\epsilon}{1-\epsilon} \\
 &= \frac{\epsilon}{|A|} \sum_a Q^{\pi_i}(s, a) + (1-\epsilon) \max_a Q^{\pi_i}(s, a) \left[ \frac{\sum_a (\pi_i(a|s) - \epsilon/|A|)}{1-\epsilon} \right] \\
 &\geq \frac{\epsilon}{|A|} \sum_a Q^{\pi_i}(s, a) + \frac{(1-\epsilon)}{1-\epsilon} \sum_a (\pi_i(a|s) - \frac{\epsilon}{|A|}) Q^{\pi_i}(s, a) \\
 &= \frac{\epsilon}{|A|} \sum_a Q^{\pi_i}(s, a) + \sum_a \pi_i(a|s) Q^{\pi_i}(s, a) - \frac{\epsilon}{|A|} \sum_a Q^{\pi_i}(s, a) \\
 &= \sum_a \pi_i(a|s) Q^{\pi_i}(s, a) = V^{\pi_i}
 \end{aligned}$$

- Therefore  $V^{\pi_{i+1}} \geq V^{\pi_i}$  (from the policy improvement theorem)

<sup>1</sup>The theorem assumes that  $Q^{\pi_i}$  has been computed exactly.


# Monotonic<sup>1</sup> $\epsilon$ -greedy Policy Improvement

## Theorem

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

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

- Therefore  $V^{\pi_{i+1}} \geq V^{\pi}$  (from the policy improvement theorem)

<sup>1</sup>The theorem assumes that  $Q^{\pi_i}$  has been computed exactly. 

# Greedy in the Limit of Infinite Exploration (GLIE)

## 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

$$\lim_{i \rightarrow \infty} \pi(a|s) \rightarrow \arg \max_{a \text{ w. prob } s} Q(s, a)$$

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

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control**
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias

# Monte Carlo Online Control / On Policy Improvement

---

```
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:    $G_{k,t} = r_{k,t} + \gamma r_{k,t+1} + \gamma^2 r_{k,t+2} + \dots + \gamma^{T-t} r_{k,T}$   
5:   for  $t = 1, \dots, T$  do  
6:     if First visit to  $(s, a)$  in episode  $k$  then ← could do every visit  
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:   $k = k + 1, \epsilon = \frac{1}{k}$   
12:   $\pi_k = \epsilon\text{-greedy}(Q)$  // Policy improvement  
13: end loop
```

---



# Check Your Understanding: MC for On Policy Control

- Mars rover with new actions:
    - $r(-, a_1) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 +10]$ ,  $r(-, a_2) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 +5]$ ,  $\gamma = 1$ .
  - Assume current greedy  $\pi(s) = a_1 \ \forall s$ ,  $\epsilon = .5$
  - Sample trajectory from  $\epsilon$ -greedy policy
  - Trajectory =  $(s_3, a_1, 0, s_2, a_2, 0, s_3, a_1, 0, s_2, a_2, 0, s_1, a_1, 1, \text{terminal})$
  - First visit MC estimate of  $Q$  of each  $(s, a)$  pair?
  - $Q^{\epsilon-\pi}(-, a_1) = [1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0]$ ,  $Q^{\epsilon-\pi}(-, a_2) = [0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0]$
  - What is  $\pi(s) = \arg \max_a Q^{\epsilon-\pi}(s, a) \ \forall s$ ?
- 
- What is new  $\epsilon$ -greedy policy, if  $k = 3$ ,  $\epsilon = 1/k$

## Theorem

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

# Model-free Policy Iteration

- Initialize policy  $\pi$
- Repeat:
  - Policy evaluation: compute  $Q^\pi$
  - Policy improvement: update  $\pi$  given  $Q^\pi$

bootstrapping



- What about TD methods?

$$V^\pi(s) = V^\pi(s) + \alpha \left( \underbrace{r + \gamma V^\pi(s')}_{\text{sampling expectation}} - V^\pi(s) \right)$$

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control**
- 5 Maximization Bias
- 6 Maximization Bias

# Model-free Policy Iteration with TD Methods

- Use temporal difference methods for policy evaluation step
- Initialize policy  $\pi$
- Repeat:
  - Policy evaluation: compute  $Q^\pi$  using temporal difference updating with  $\epsilon$ -greedy policy
  - Policy improvement: Same as Monte carlo policy improvement, set  $\pi$  to  $\epsilon$ -greedy ( $Q^\pi$ )

# General Form of SARSA Algorithm

- 
- 1: Set initial  $\epsilon$ -greedy policy  $\pi$  randomly,  $t = 0$ , initial state  $s_t = s_0$
  - 2: Take  $a_t \sim \pi(s_t)$  // Sample action from policy
  - 3: Observe  $(r_t, s_{t+1})$
  - 4: **loop**
  - 5:   Take action  $a_{t+1} \sim \pi(s_{t+1})$
  - 6:   Observe  $(r_{t+1}, s_{t+2})$
  - 7:   Update Q given  $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$ :  
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + (1 - \alpha) (r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

$\downarrow$
  - 8:   Perform policy improvement:  
$$\pi(s_t) = \arg \max_a Q(s_t, a) \quad \epsilon\text{-greedy}$$
  - 9:    $t = t + 1$
  - 10: **end loop**
-

# General Form of SARSA Algorithm

$$(1-\alpha)V + \alpha(r + \gamma Q) \\ V \leftarrow \alpha(r + \gamma Q - V)$$

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

What are the benefits to improving the policy after each step?

What are the benefits to updating the policy less frequently?

# Convergence Properties of SARSA

## Theorem

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

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

*learning  
rate  
parameter*

$$\left. \begin{aligned} \sum_{t=1}^{\infty} \alpha_t &= \infty \\ \sum_{t=1}^{\infty} \alpha_t^2 &< \infty \end{aligned} \right\}$$

$\alpha_t = 1/t$

*empirically  
you don't  
use this*



# Convergence Properties of SARSA

## Theorem

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

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

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

Would one want to use a step size choice that satisfies the above in practice? Likely not.

# Q-Learning: Learning the Optimal State-Action Value

- Can we estimate the value of the optimal policy  $\pi^*$  without knowledge of what  $\pi^*$  is?
- Yes! Q-learning
- Key idea: Maintain state-action  $Q$  estimates and use to bootstrap—use the value of the best future action
- Recall SARSA

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha((r_t + \gamma Q(s_{t+1}, \underline{a_{t+1}})) - Q(s_t, a_t)) \quad (2)$$

- Q-learning:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha((r_t + \gamma \max_{a'} \underline{Q(s_{t+1}, a')}) - Q(s_t, a_t)) \quad (3)$$

# Off-Policy Control Using Q-learning

- In the prior slide assumed there was some  $\pi_b$  used to act
- $\pi_b$  determines the actual rewards received
- Now consider how to improve the behavior policy (policy improvement)
- Let behavior policy  $\pi_b$  be  $\epsilon$ -greedy with respect to (w.r.t.) current estimate of the optimal  $Q(s, a)$

# Q-Learning with $\epsilon$ -greedy Exploration

---

1: Initialize  $Q(s, a), \forall s \in S, a \in A$   $t = 0$ , initial state  $s_t = s_0$

2: Set  $\pi_b$  to be  $\epsilon$ -greedy w.r.t.  $Q$

3: **loop**

4: Take  $a_t \sim \pi_b(s_t)$  // Sample action from policy

5: Observe  $(r_t, s_{t+1})$

6: Update  $Q$  given  $(s_t, a_t, r_t, s_{t+1})$ :

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha (r_t + \max_{a'} Q(s_t, a') - Q(s_t, a_t))$$

7: Perform policy improvement: set  $\pi_b$  to be  $\epsilon$ -greedy w.r.t.  $Q$

8:  $t = t + 1$

$\pi_b$  for  $s_t$

9: **end loop**

---

# Q-Learning with $\epsilon$ -greedy Exploration

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

Does how  $Q$  is initialized matter?

# Check Your Understanding: Q-learning

- Mars rover with new actions:
  - $r(-, a_1) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 +10]$ ,  $r(-, a_2) = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 +5]$ ,  $\gamma = 1$ .
- Assume current greedy  $\pi(s) = a_1 \ \forall s$ ,  $\epsilon = .5$
- Sample trajectory from  $\epsilon$ -greedy policy
- Trajectory =  $(s_3, a_1, 0, s_2, a_2, 0, s_3, a_1, 0, s_2, a_2, 0, s_1, a_1, 1, \text{terminal})$
- New  $\epsilon$ -greedy policy under MC, if  $k = 3$ ,  $\epsilon = 1/k$ : with probability  $2/3$  choose  $\pi = [1 \ 2 \ 1 \ \text{tie} \ \text{tie} \ \text{tie} \ \text{tie}]$ , else choose randomly
- Q-learning updates? Initialize  $\epsilon = 1/k$ ,  $k = 1$ , and  $\alpha = 0.5$
- $\pi$  is random with probability  $\epsilon$ , else  $\pi = [1 \ 1 \ 1 \ 2 \ 1 \ 2 \ 1]$
- First tuple:  $(s_3, a_1, 0, s_2)$ .
- Q-learning:
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \arg \max_a Q(s_{t_1}, a) - Q(s_t, a_t))$$

# Q-Learning with $\epsilon$ -greedy Exploration

- What conditions are sufficient to ensure that Q-learning with  $\epsilon$ -greedy exploration converges to optimal  $Q^*$ ?

*s.a.  $\infty$  often  
conditions on  $\alpha \leftarrow$  see SARSA*

- What conditions are sufficient to ensure that Q-learning with  $\epsilon$ -greedy exploration converges to optimal  $\pi^*$ ?

*GLIE*

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias**
- 6 Maximization Bias



# Maximization Bias<sup>1</sup>

- Consider single-state MDP ( $|S| = 1$ ) with 2 actions, and both actions have 0-mean random rewards, ( $\mathbb{E}(r|a = a_1) = \mathbb{E}(r|a = a_2) = 0$ ).  $\gamma = 0$
- Then  $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$  optimal
- Assume there are prior samples of taking action  $a_1$  and  $a_2$
- Let  $\hat{Q}(s, a_1), \hat{Q}(s, a_2)$  be the finite sample estimate of  $Q$
- Use an unbiased estimator for  $Q$ : e.g.  $\hat{Q}(s, a_1) = \frac{1}{n(s, a_1)} \sum_{i=1}^{n(s, a_1)} r_i(s, a_1)$
- Let  $\hat{\pi} = \arg \max_a \hat{Q}(s, a)$  be the greedy policy w.r.t. the estimated  $\hat{Q}$
- *Even though each estimate of the state-action values is unbiased, the estimate of  $\hat{\pi}$ 's value  $\hat{V}^{\hat{\pi}}$  can be biased:*

$$\begin{aligned} \hat{V}^{\hat{\pi}} &= E[\max(\hat{Q}(a_1), \hat{Q}(a_2))] \\ &\stackrel{\square}{=} \max[E(Q(a_1)), E(Q(a_2))] \\ &= \max[0, 0] \\ &= 0 \\ &= V^{\pi} \end{aligned}$$

<sup>1</sup>Example from Mannor, Simester, Sun and Tsitsiklis. Bias and Variance

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias**

# Maximization Bias<sup>2</sup>

- Consider single-state MDP ( $|S| = 1$ ) with 2 actions, and both actions have 0-mean random rewards, ( $\mathbb{E}(r|a = a_1) = \mathbb{E}(r|a = a_2) = 0$ ).
- Then  $Q(s, a_1) = Q(s, a_2) = 0 = V(s)$
- Assume there are prior samples of taking action  $a_1$  and  $a_2$
- Let  $\hat{Q}(s, a_1), \hat{Q}(s, a_2)$  be the finite sample estimate of  $Q$
- Use an unbiased estimator for  $Q$ : e.g.  $\hat{Q}(s, a_1) = \frac{1}{n(s, a_1)} \sum_{i=1}^{n(s, a_1)} r_i(s, a_1)$
- Let  $\hat{\pi} = \arg \max_a \hat{Q}(s, a)$  be the greedy policy w.r.t. the estimated  $\hat{Q}$
- *Even though each estimate of the state-action values is unbiased, the estimate of  $\hat{\pi}$ 's value  $\hat{V}^{\hat{\pi}}$  can be biased:*

---

<sup>2</sup>Example from Mannor, Simester, Sun and Tsitsiklis. Bias and Variance Approximation in Value Function Estimates. Management Science 2007

# Double Learning

- The greedy policy w.r.t. estimated  $Q$  values can yield a maximization bias during finite-sample learning
- Avoid using max of estimates as estimate of max of true values
- Instead split samples and use to create two independent unbiased estimates of  $Q_1(s_1, a_i)$  and  $Q_2(s_1, a_i) \forall a$ .
  - Use one estimate to select max action:  $a^* = \arg \max_a Q_1(s_1, a)$
  - Use other estimate to estimate value of  $a^*$ :  $Q_2(s, a^*)$
  - Yields unbiased estimate:  $\mathbb{E}(Q_2(s, a^*)) = Q(s, a^*)$
- Why does this yield an unbiased estimate of the max state-action value?
- If acting online, can alternate samples used to update  $Q_1$  and  $Q_2$ , using the other to select the action chosen
- Next slides extend to full MDP case (with more than 1 state)

# Double Q-Learning

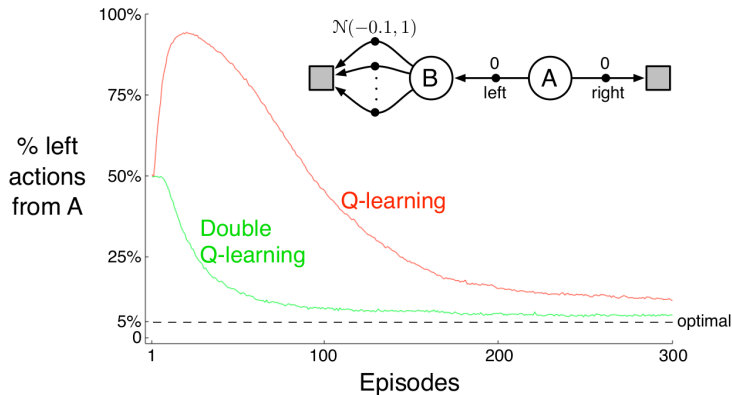
---

```
1: Initialize  $Q_1(s, a)$  and  $Q_2(s, a), \forall s \in S, a \in A$   $t = 0$ , initial state  $s_t = s_0$ 
2: loop
3:   Select  $a_t$  using  $\epsilon$ -greedy  $\pi(s) = \arg \max_a Q_1(s_t, a) + Q_2(s_t, a)$ 
4:   Observe  $(r_t, s_{t+1})$ 
5:   if (with 0.5 probability) then
6:      $Q_1(s_t, a_t) \leftarrow Q_1(s_t, a_t) + \alpha ($ 
7:   else
8:      $Q_2(s_t, a_t) \leftarrow Q_2(s_t, a_t) + \alpha ($ 
9:   end if
10:   $t = t + 1$ 
11: end loop
```

---

- Compared to Q-learning, how does this change the: memory requirements, computation requirements per step, amount of data required?

# Double Q-Learning (Figure 6.7 in Sutton and Barto 2018)



Due to the maximization bias, Q-learning spends much more time selecting suboptimal actions than double Q-learning.

# Table of Contents

- 1 Generalized Policy Iteration
- 2 Importance of Exploration
- 3 Monte Carlo Control
- 4 Temporal Difference Methods for Control
- 5 Maximization Bias
- 6 Maximization Bias

# What You Should Know

- Be able to implement MC on policy control and SARSA and Q-learning
- Compare them according to properties of how quickly they update, (informally) bias and variance, computational cost
- Define conditions for these algorithms to converge to the optimal Q and optimal  $\pi$  and give at least one way to guarantee such conditions are met.



# Class Structure

- Last time: Policy evaluation with no knowledge of how the world works (MDP model not given)
- This time: Control (making decisions) without a model of how the world works
- **Next time: Value function approximation**

# Backup Material, Not Expected to Cover in This Lecture

# Recall: Off Policy, Policy Evaluation

- Given data from following a behavior policy  $\pi_b$  can we estimate the value  $V^{\pi_e}$  of an alternate policy  $\pi_e$ ?
- Neat idea: can we learn about other ways to do things different than what we actually did?
- Discussed how to do this for Monte Carlo evaluation
- Used Importance Sampling
- First see how to do off policy evaluation with TD

# Importance Sampling for Off Policy TD (Policy Evaluation)

- Recall the Temporal Difference (TD) algorithm which is used to incremental model-free evaluation of a policy  $\pi_b$ . Precisely, given a state  $s_t$ , an action  $a_t$  sampled from  $\pi_b(s_t)$  and the observed reward  $r_t$  and next state  $s_{t+1}$ , TD performs the following update:

$$V^{\pi_b}(s_t) = V^{\pi_b}(s_t) + \alpha(r_t + \gamma V^{\pi_b}(s_{t+1}) - V^{\pi_b}(s_t)) \quad (4)$$

- Now want to use data generated from following  $\pi_b$  to estimate the value of different policy  $\pi_e$ ,  $V^{\pi_e}$
- Change TD target  $r_t + \gamma V(s_{t+1})$  to weight target by single importance sample ratio
- New update:

$$V^{\pi_e}(s_t) = V^{\pi_e}(s_t) + \alpha \left[ \frac{\pi_e(a_t|s_t)}{\pi_b(a_t|s_t)} (r_t + \gamma V^{\pi_e}(s_{t+1}) - V^{\pi_e}(s_t)) \right] \quad (5)$$

# Importance Sampling for Off Policy TD Cont.

- Off Policy TD Update:

$$V^{\pi_e}(s_t) = V^{\pi_e}(s_t) + \alpha \left[ \frac{\pi_e(a_t|s_t)}{\pi_b(a_t|s_t)} (r_t + \gamma V^{\pi_e}(s_{t+1}) - V^{\pi_e}(s_t)) \right] \quad (6)$$

- Significantly lower variance than MC IS. (Why?)
- Does  $\pi_b$  need to be the same at each time step?
- What conditions on  $\pi_b$  and  $\pi_e$  are needed for off policy TD to converge to  $V^{\pi_e}$ ?