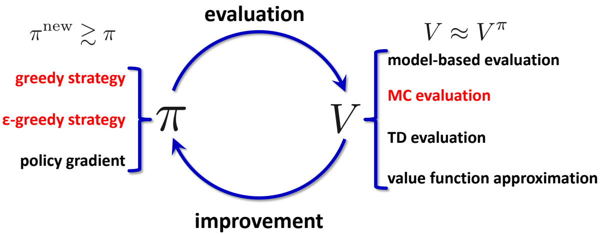
Algorithmic and Theoretical Foundations of RL

Monte Carlo (MC) Learning

Ke Wei
School of Data Science
Fudan University

Policy Iteration Recap



$\pi^{\text{new}} \gtrsim \pi$

greedy strategy

ε-greedy strategy

policy gradient

π

evaluation

improvement

V

$V \approx V^{\pi}$

model-based evaluation

MC evaluation

TD evaluation

value function approximation

Policy Iteration: greedy policy is improved via

$$\pi_{k+1}(s) = \operatorname*{argmax}_{a} \underbrace{\mathbb{E}_{s' \sim P(\cdot|s,a)}[r(s,a,s') + \gamma V^{\pi_k}(s')]}_{Q^{\pi_k}(s,a)},$$

where $V^{\pi_k}(s')$ is evaluated via Bellman equation based on the model.

— *What if system information (P and r) is not available?*

— *Replace model by data (model free).*

— *How to collect data? How to use data?*

— . . . . . .

MC Policy Evaluation (or Prediction)

Basic idea. Given π, estimate $V^\pi(s)$ and $Q^\pi(s,a)$ from sampled trajectories

$$\tau_i = \{(s_0^i, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, \cdots)\}_{i=1}^n \sim \pi.$$

- MC evaluation of $V^\pi(s)$: $s_0^i = s$,

$$V^\pi(s) \approx \frac{1}{n}\sum_{i=1}^n \left(\sum_{t=0}^\infty \gamma^t r_t^i\right).$$

- MC evaluation of $Q^\pi(s,a)$: $s_0^i = s$, $a_0^i = a$,

$$Q^\pi(s,a) \approx \frac{1}{n}\sum_{i=1}^n \left(\sum_{t=0}^\infty \gamma^t r_t^i\right).$$

Action Value Based Policy Iteration

- Policy improvement via state value:

$$\pi_{k+1}(s) = \operatorname*{argmax}_a \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[r(s,a,s') + \gamma V^{\pi_k}(s') \right] .$$

  Given $V^{\pi_k}(s')$, still need to compute the expectation which requires model or needs further evaluation.

- Policy improvement via action value:

$$\pi_{k+1}(s) = \operatorname*{argmax}_a Q^{\pi_k}(s,a).$$

  Ideal for model free RL since we can estimate $Q^{\pi_k}(s,a)$ directly from data.

Primitive MC Learning Algorithm

Algorithm 1: Primitive MC Learning

```
Initialization: $\pi_0$, $n$
for $k = 0, 1, 2, \ldots$ do
    for every $s$ do
        for every $a$ do
            Sample $n$ episodes starting from $(s, a)$, following $\pi_k$:
                $\tau_i = (s_0^i, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, \cdots, s_{T-1}^i, a_{T-1}^i, r_{T-1}^i, s_T^i) \sim \pi_k, \; i = 1, \cdots, n$
            Compute $Q^k(s, a) = \frac{1}{n} \sum_{i=1}^{n} \left( \sum_{t=0}^{T-1} \gamma^t r_t^i \right)$
        end
        $\pi_{k+1}(s) = \operatorname*{argmax}_a Q^k(s, a)$
    end
end
```

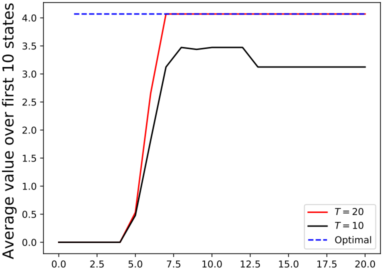
Ideally, T should be ∞ or s_T be a terminal state. In practice, T should be sufficiently large, especially for the sparse reward case.

Illustrative Example

								O	
	O								
			O					O	
						G			
						O			

Goal: +10, obstacle: -10; goal is terminal state.

Illustrative Example



The learned policy is evaluated exactly using model.

Inefficiency of Primitive MC Learning

- A trajectory is only used for estimating one state-action value;
- Wait until all trajectories have been collected before policy update;
- Old state-action values are not reused and thus wasted (next lecture).

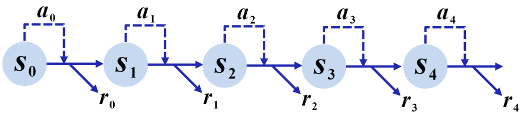
Table of Contents

Sample Efficient MC Policy Evaluation

MC Learning (or Control)

Off-Policy MC Learning

Use Trajectory More Efficiently



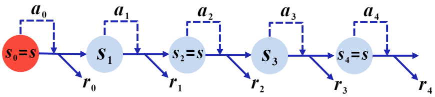
Trajectory $(s_0, a_0, r_0, s_1, a_1, r_1, \cdots) \sim \pi$ starting from s contains sub-trajectories $(s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots)$ that starts from other states (e.g. $s_t = s'$). Thus, return from the sub-trajectory

$$G_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}$$

can be used to build an estimator of $V^\pi(s')$. Namely, one trajectory can be used to estimate different $V^\pi(s)$.

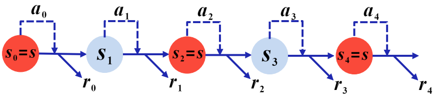
There is no essential difference in the MC evaluations of state value and action value in methodology. Thus discussion in this section will be mainly based on state value.

First-Visit and Every Visit



First Visit

- Only sub-trajectory that starts from the first visit of s is used in the estimation of $V^\pi(s)$; One trajectory is only used once in the evaluation of $V^\pi(s)$.



Every Visit

- All sub-trajectories that start from of s is used in the estimation of $V^\pi(s)$; One trajectory might be used many times in the evaluation of $V^\pi(s)$.

First-Visit MC Policy Evaluation

Algorithm 2: First-Visit Monte Carlo Policy Evaluation

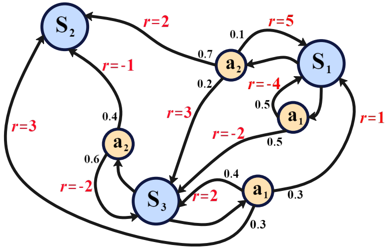
```
Initialization: Counter of visited numbers $N(s) = 0$, the total return $G(s) = 0$, $\forall s \in \mathcal{S}$
for $k = 0, 1, 2, \ldots$ do
    Initialize $s_0$ and sample an episode following $\pi$:
        $(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim \pi$
    $G \leftarrow 0$
    for $t = T-1, T-2, \ldots, 0$ do
        $G \leftarrow \gamma G + r_t$
        if $s_t$ does not appear in $(s_0, s_1, \cdots, s_{t-1})$ then
            $N(s_t) \leftarrow N(s_t) + 1$
            $G(s_t) \leftarrow G(s_t) + G$
            $V^{first}(s_t) \leftarrow G(s_t)/N(s_t)$
        end
    end
end
```

Every-Visit MC Policy Evaluation

Algorithm 3: Every-Visit Monte Carlo Policy Evaluation

```
Initialization: Counter of visited numbers $N(s) = 0$, the total return $G(s) = 0$, $\forall s \in \mathcal{S}$
for $k = 0, 1, 2, \dots$ do
    Initialize $s_0$ and sample an episode following $\pi$:
        $$(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim \pi$$
    $G \leftarrow 0$
    for $t = T-1, T-2, \dots, 0$ do
        $G \leftarrow \gamma G + r_t$
        $N(s_t) \leftarrow N(s_t) + 1$
        $G(s_t) \leftarrow G(s_t) + G$
        $V^{every}(s_t) \leftarrow G(s_t)/N(s_t)$
    end
end
```

Illustrative Example

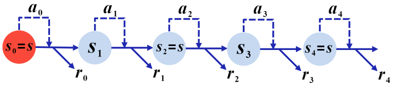


Consider policy $\pi(a|s) = 0.5$ for each state s and each action a and $\gamma = 0.9$. Recall that $V^{\pi} = [-0.21, 0, 0.31]^T$.

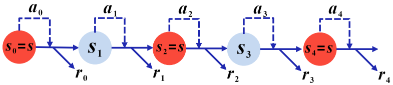
Consider a sampled trajectory: $(s_1, a_1, -2, s_3, a_1, 1, s_1, a_2, 3, s_3, a_2, -1, s_2)$.

- First-visit policy evaluation for state s_3:
  $N(s_3) = 1, V^{first}(s_3) = (1 + 0.9 \times 3 + 0.9^2 \times (-1)) = 2.89.$
- Every-visit policy evaluation for state s_3:
  $N(s_3) = 2, V^{every}(s_3) = (1 + 0.9 \times 3 + 0.9^2 \times (-1) - 1)/2 = 0.945.$

First-Visit vs Every-Visit



First Visit



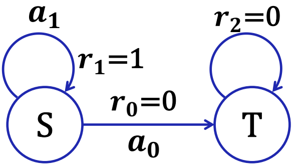
Every Visit

MSE = bias^2+variance

	Un-biased	Short MSE	Long MSE
First visit	Yes	Higher	Lower
Every visit	No	Lower	Higher

“Reinforcement learning with replacing eligibility traces” by Singh and Sutton, 1996.

Illustrative Example



$\pi(a_1|s) = p, \quad \pi(a_0|s) = 1 - p$. Set $\gamma = 1$.

State value of π at s is $V^\pi(s) = \frac{p}{1-p}$.

- Single trajectory

$$\mathbb{E}\left[V^{first}(s)\right] = \frac{p}{1-p}, \quad \text{MSE}\left[V^{first}\right] = \text{Var}\left[V^{first}\right] = \frac{p}{(1-p)^2};$$

$$\mathbb{E}\left[V^{every}\right](s) = \frac{p}{2(1-p)}, \quad \text{MSE}\left[V^{every}\right] \leq \frac{p}{2(1-p)^2}.$$

- As the number of trajectories increases, it can be shown that

$$V^{every}(s) \to \frac{p}{1-p}.$$

Incremental Monte Carlo Policy Evaluation

As already seen, mean evaluation can be conducted in an incremental way:

$$N(s_t) \leftarrow N(s_t) + 1, \quad V(s_t) \leftarrow V(s_t) + \frac{1}{N(s_t)}(G - V(s_t)).$$

Algorithm 4: First-Visit Monte Carlo Policy Evaluation (Incremental Version)

```
Initialization: Visited numbers $N(s) = 0$ and initialize $V(s)\ \forall s \in \mathcal{S}$.
for $k = 0, 1, 2, \ldots$ do
    Initialize $s_0$ and sample an episode following $\pi$:
        $(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim \pi$
    $G \leftarrow 0$
    for $t = T-1, T-2, \ldots, 0$ do
        $G \leftarrow \gamma G + r_t$
        if $s_t$ does not appear in $(s_0, s_1, \cdots, s_{t-1})$ then
            $N(s_t) \leftarrow N(s_t) + 1$
            $V(s_t) \leftarrow V(s_t) + \frac{1}{N(s_t)}(G - V(s_t))$
        end
    end
end
```

Without further specification, discussion in the rest of this lecture will focus on first visit, and the superscript “first” will be omitted.

Table of Contents

Sample Efficient MC Policy Evaluation

MC Learning (or Control)

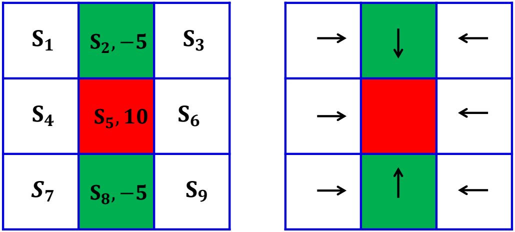
Off-Policy MC Learning

Simply Combine MC Policy Evaluation with Greedy Policy

Algorithm 5: MC Learning with Greedy Policy

```
Initialization: $Q(s, a) = 0, N(s, a) = 0, \forall s, a$; Initialize $\pi_0$.
for $k = 0, 1, 2, \dots$ do
    Initialize $s_0$ and sample an episode following $\pi_k$:
        $$(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim \pi_k$$
    $G \leftarrow 0$
    for $t = T-1, T-2, \dots, 0$ do
        $G \leftarrow \gamma G + r_t$
        if $(s_t, a_t)$ does not appear in $(s_0, a_0, s_1, a_1, \dots, s_{t-1}, a_{t-1})$ then
            $N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$
            $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)}(G - Q(s_t, a_t))$
            $\pi_{k+1}(a|s_t) = \begin{cases} 1 & \text{if } a = \operatorname*{argmax}_a Q(s_t, a) \\ 0 & \text{otherwise} \end{cases}$
        end
    end
end
```

An Example Algorithm 5 Fails to Work



S_1	$S_2, -5$	S_3
S_4	$S_5, 10$	S_6
S_7	$S_8, -5$	S_9

→	↓	←
→		←
→	↑	←

Consider the gridworld problem (left) where $\gamma = 0.9$. Assume $Q(s, a) = 0$ for all s, a and π_0 is given in the right plot. It can be verified that π_0 does not change for Algorithm 5.

Recall: Exploitation vs Exploration

ALL learning methods face the exploitation vs exploration dilemma: seek to make a best decision at each time step based on the current/local information, but need to act suboptimally in order to explore all possibilities. More precisely, if the local information can be computed accurately (e.g., each action value can be computed exactly given a policy), then a series of local decisions can lead to global optimal. However, when the local information is not accurate, it may mislead. In this situation, one should be allowed to explore the non-optimal decision when making the decision so that it is possible to achieve the global, long term optimum.

- How to encourage exploration?
  - Explore state-action pairs when sampling episodes.
  - ϵ-greedy policy
  - Off-policy learning

ϵ-Greedy Policy

With small probability ϵ randomly choose an action to ensure exploration:

$$\pi'(a|s) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{A}|} & \text{if } a = \operatorname*{argmax}_a Q^\pi(s, a'), \\ \frac{\epsilon}{|\mathcal{A}|} & \text{otherwise.} \end{cases}$$

Theorem 1

For any policy π, the ϵ-greedy policy π' with respect to Q^π is an improvement, i.e., $V^{\pi'}(s) \geq V^\pi(s)$, provided $\pi(a|s) > 0$, $\forall a$ and ϵ is sufficiently small.

Proof of Theorem 1

It suffices to show the one-step improvement of π' over π: $\mathcal{T}^{\pi'} V^\pi \geq V^\pi$, which is equivalent to

$$\sum_a \pi'(a|s) Q^\pi(s,a) \geq \sum_a \pi(a|s) Q^\pi(s,a) = V^\pi(s).$$

This follows directly from

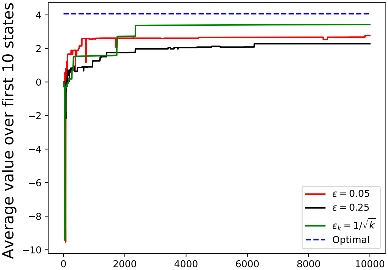
$$\begin{aligned}
\sum_a \pi'(a|s) Q^\pi(s,a) &= \frac{\epsilon}{|\mathcal{A}|} \sum_a Q^\pi(s,a) + (1-\epsilon) \max_a Q^\pi(s,a) \\
&= \frac{\epsilon}{|\mathcal{A}|} \sum_a Q^\pi(s,a) + \left(\sum_a \left(\pi(a|s) - \frac{\epsilon}{|\mathcal{A}|} \right) \right) \max_a Q^\pi(s,a) \\
&\geq \frac{\epsilon}{|\mathcal{A}|} \sum_a Q^\pi(s,a) + \sum_a \left(\pi(a|s) - \frac{\epsilon}{|\mathcal{A}|} \right) Q^\pi(s,a) \\
&= \sum_a \pi(a|s) Q^\pi(s,a).
\end{aligned}$$

MC Learning with ϵ-Greedy Policy

Algorithm 6: MC Learning with ϵ-Greedy Exploration

```
Initialization: $N(s, a) = 0, Q(s, a) = 0, \forall s, a, \pi_0$
for $k = 0, 1, 2, \dots$ do
    Initialize $s_0$ and sample an episode following $\pi_k$:
        $$(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim \pi_k$$
    $G \leftarrow 0$
    for $t = T-1, T-2, \dots, 0$ do
        $G \leftarrow \gamma G + r_t$
        if $(s_t, a_t)$ does not appear in $(s_0, a_0, s_1, a_1, \dots, s_{t-1}, a_{t-1})$ then
            $N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$
            $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)}(G - Q(s_t, a_t))$
            Update policy of visited state via $\epsilon_k$-greedy:
                $$\pi_{k+1}(a|s_t) = \begin{cases} 1 - \epsilon_k + \frac{\epsilon_k}{|\mathcal{A}|} & \text{if } a = \operatorname*{argmax}_{a'} Q(s_t, a') \\ \frac{\epsilon_k}{|\mathcal{A}|} & \text{otherwise} \end{cases}$$
        end
    end
end
```

Illustrative Example



For the previously mentioned 10×10 gridworld problem.

Table of Contents

Sample Efficient MC Policy Evaluation

MC Learning (or Control)

Off-Policy MC Learning

Off-Policy Monte Carlo Evaluation

- On-policy learning vs off-policy learning
  - On-policy: Learn target policy π from experience sampled from π;
  - Off-policy: Learn target policy π from experience sampled from b.
- On-policy ϵ-greedy method which is not deterministic needs to behave non-optimally in order to explore all actions.
- Off-policy method attempts to learn a deterministic optimal policy from data generated by another exploratory policy.

Importance Sampling for Off-Policy MC Evaluation

In order to evaluate action value $Q^{\pi}(s,a)$ from data sampled from a behavior policy b, we need to express $Q^{\pi}(s,a)$ in terms of the expectation with respect to b. Given a subtrajectory $\tau_t = \{s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots\}$, let $(s_t, a_t) = (s, a)$ and P_t^{π} be the distribution of τ_t under policy π (similarly for P_t^{b}). We have,

$$\begin{aligned} Q^{\pi}(s,a) &= \mathbb{E}_{\tau_t \sim P_t^{\pi}}\left[G_t\right] \\ &= \mathbb{E}_{\tau_t \sim P_t^{b}}\left[\frac{P_t^{\pi}(\tau_t)}{P_t^{b}(\tau_t)} G_t\right], \end{aligned}$$

where $G_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}$ and

$$\frac{P_t^{\pi}(\tau_t)}{P_t^{b}(\tau_t)} = \frac{P(s_{t+1}|s_t,a_t)\prod_{k=t+1}^{\infty} P(s_{k+1}|s_k,a_k)\,\pi(a_k|s_k)}{P(s_{t+1}|s_t,a_t)\prod_{k=t+1}^{\infty} P(s_{k+1}|s_k,a_k)\,b(a_k|s_k)} = \prod_{k=t+1}^{\infty} \frac{\pi(a_k|s_k)}{b(a_k|s_k)}$$

is known as importance-sampling ratio.

Overall, it requires coverage for the behavior policy, i.e., $b(a|s) > 0$ if $\pi(a|s) > 0$.

Importance Sampling for Off-Policy MC Evaluation

Given an

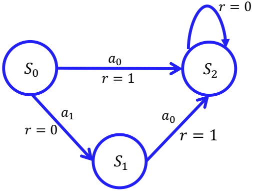
$$\{s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T\} \sim b,$$

off-policy MC evaluation has the following form:

$$\begin{aligned} N(s_t, a_t) &\leftarrow N(s_t, a_t) + 1 \\ Q(s_t, a_t) &\leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)} \left(G_t \frac{P_t^{\pi}}{P_t^{b}} - Q(s_t, a_t) \right) \end{aligned}$$

Weight for Initial Pair Should Not Be Included

Note when defining $Q^{\pi}(s, a)$, action a is independent of policy π. Thus, when computing importance sampling weight for (s_t, a_t), $\frac{\pi(a_t|s_t)}{b(a_t|s_t)}$ is excluded.



Suppose $\gamma < 1$. Optimal policy for s_0 is $\pi^*(s_0) = a_0$. Set $Q(s, a) = 0$ for all (s, a), $\pi_0(s_0) = a_1$ and $\pi_0(s_1) = a_0$. Two possible episodes for an exploratory behavior policy b:

$$(s_0, a_0, 1, s_2) \quad \text{and} \quad (s_0, a_1, 0, s_1, a_0, 1, s_2).$$

It is easy to verify that π_0 will not be updated if $\frac{\pi_0(a_0|s_0)}{b(a_0|s_0)} = 0$ is included in the computation of importance sampling weight. In contrast, π_0 will be updated if $\frac{\pi_0(a_0|s_0)}{b(a_0|s_0)} = 0$ is not included.

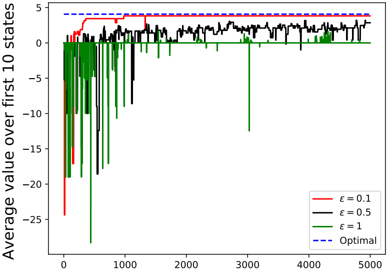
Example is from Wenye Li.

Off-Policy MC Learning

```
Algorithm 7: Off-policy MC Learning
Initialization: $\forall s, a$, initialize $Q(s,a)$, $\pi_0(s) = \mathrm{argmax}_a Q(s,a)$, $N(s,a) = 0$.
for $k = 0, 1, 2, \ldots$ do
    $b_k \leftarrow$ any soft policy, i.e., $b_k(a|s) > 0, \forall s, a$
    Initialize $s_0$ and sample an episode following $b_k$:
        $$(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_{T-1}, a_{T-1}, r_{T-1}, s_T) \sim b_k$$
    $G \leftarrow 0, W \leftarrow 1$
    for $t = T-1, T-2, \ldots, 0$ do
        $G \leftarrow r_t + \gamma G$
        if $(s_t, a_t)$ does not appear in $(s_0, a_0, s_1, a_1, \ldots, s_{t-1}, a_{t-1})$ then
            $N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$
            $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \frac{1}{N(s_t, a_t)}(W \cdot G - Q(s_t, a_t))$
            $\pi_{k+1}(s_t) \leftarrow \mathrm{argmax}_a Q(s_t, a)$
        end
        $W \leftarrow W \frac{\pi_k(a_t|s_t)}{b_k(a_t|s_t)}$
    end
end
```

To handle the potential high variance incurred by importance sampling, one may consider weighted importance sampling. See "Reinforcement learning: An introduction" by Sutton and Barto, 2018.

Illustrative Example



ϵ-greedy policy is used as behavior policy.

For the previously mentioned 10×10 gridworld problem.

Remark

- Policy evaluation and policy improvement is a general and fundamental framework for RL algorithms. Different evaluation methods may require different improvement methods, and vice versa. As already presented, if we use data sampled from target policy for evaluation, we should use ϵ-greedy policy for improvement to encourage exploration. In contrast, if using greedy policy for improvement, we may need to use data sampled from a more exploratory behavior policy for evaluation based on importance sampling.
- Most algorithms presented in this lecture and the next one admit certain convergence guarantees under mild conditions, details of which are omitted.

Questions?