Notes on Reinforcement Learning

Updated: January 15, 2020

Contents

1 Markov Decision Process (MDP)

Markov decision process is one of the most important part in reinforcement learning. In this note, we only consider the fully observable Markov decision process with finite discrete states, with finite discrete actions, and in discrete time steps.

Reinforcement Learning

> “A RL agent interacts with an environment over time. At each time step t, the agent receives a state s_t in a state space $\mathcal{S}$ and selects an action at from an action space $\mathcal{A}$, following a policy $\pi(a_t|s_t)$, which is the agent's behavior, i.e., a mapping from state s_t to actions a_t, receives a scalar reward r_t, and transitions to the next state s_{t+1}, according to the environment dynamics, or model, for reward function $\mathcal{R}(s, a)$ and state transition probability $\mathcal{P}(s_{t+1}|s_t, a_t)$ respectively. In an episodic problem, this process continues until the agent reaches a terminal state and then it restarts. The return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ is the discounted, accumulated reward with the discount factor $\gamma \in (0, 1]$. The agent aims to maximize the expectation of such long term return from each state . . . ”[Li, 2018]

Notations

- State space $\mathcal{S}$. Every element in $\mathcal{S}$ is called a state. The random process $\{S_t\}$ represents the state at time t.
- Action space $\mathcal{A}$. Every element in $\mathcal{A}$ is called an action. The random process $\{A_t\}$ represents the action taken by agent at time t; sometimes, it is also called the control process. We assume $\{A_t\}$ is adapted w.r.t. the filtration generated by $\{S_t\}$.
- Reward $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Reward is a deterministic function used to measure how well agent is doing by taking action a in state s; the random process $\{R_t\}$ represents the reward received by agent at time t. The reward process is stochastic. Moreover, we assume for all t, $R_t \leq R$ for some $R < \infty$.
- History H. The history H_t is defined as all information no later than time t.

Agent The goal of agent is to gather rewards based on the received information from the environment; for example, we expect the behavior of an agent will maximize the expected, discounted, accumulative reward in the future. An RL agent may include one or more of these components: policy, value function, or model.

- A policy fully defines the agent's behavior.

  Definition 1.1 (Policy). A *deterministic policy* π is a map from $\mathcal{S}$ to $\mathcal{A}$,

  $$\pi : s \mapsto \pi(s).$$

  A *stochastic policy* π is a distribution over actions $\mathcal{A}$ given states,

  $$\pi(a|s) = \mathbb{P}(A_t = a \mid S_t = s).$$

- The value function is a prediction of future reward; it is used to evaluate the goodness/badness of states.

  Definition 1.2 (Value function). A *return* from time-step t with the discount $\gamma \in [0, 1]$ is

  $$G_t := R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

  A *state-value function* $v_\pi : \mathcal{S} \to \mathbb{R}$ w.r.t. π is defined as

  $$v_\pi(s) := \mathbb{E}_\pi[G_t \mid S_t = s]$$

A *action-value function* $q_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ w.r.t. π is defined as

$$q_\pi(s,a) := \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

The *optimal state-value function* $v_* : \mathcal{S} \to \mathbb{R}$ is defined as

$$v_*(s) := \max_\pi v_\pi(s).$$

The *optimal action-value function* $q_* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as

$$q_*(s,a) := \max_\pi q_\pi(s,a).$$

Remark. Note that the value function can also be considered as a function of policy π. Therefore, we can define the optimal policy as the policy maximizing the value function.

Definition 1.3 (Optimal Policy)**.** A policy π_* is optimal in $\mathcal{D}$, if for any policy $\pi \in \mathcal{D}$ and for all $s \in \mathcal{S}$,

$$v_{\pi_*}(s) \geq v_\pi(s).$$

- A model predicts what the environment will do next.

  Definition 1.4 (Model)**.** $\mathcal{P}$ is defined as the distribution of next step,

  $$\mathcal{P}^a_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right].$$

  $\mathcal{R}$ is defined as the next expected reward,

  $$\mathcal{R}^a_s = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a].$$

Probability Review A random process $\{S_t\}_{t\in\mathbb{N}}$ with a finite state space $\mathcal{S}$ is called a Markov process if for every $s, s_1, \ldots, s_t \in \mathcal{S}$,

$$\mathbb{P}(S_{t+1} = s \mid S_1 = s_1, \ldots, S_t = s_t) = \mathbb{P}(S_{t+1} = s \mid S_t = s_t);$$

or equivalently, it could be defined relative to a filtration $\mathcal{F}$,

$$\mathbb{E}(f(S_{t+1}) \mid \mathcal{F}_t) = \mathbb{E}(f(S_{t+1}) \mid S_t)$$

for any measurable function $f : \mathcal{S} \to \mathbb{R}$.
Given a Markov process $\{S_t\}_{t\in\mathbb{N}}$, the state transition probability from s to s' is written as

$$\mathcal{P}_{ss'} = \mathbb{P}\left(S_{t+1} = s' \mid S_t = s\right);$$

It forms the state transition matrix

$$\mathcal{P} = \begin{pmatrix} \mathcal{P}_{11} & \ldots & \mathcal{P}_{1n} \\ \vdots & & \\ \mathcal{P}_{n1} & \ldots & \mathcal{P}_{nn} \end{pmatrix}.$$

1.1 Markov Decision Process

Definition 1.5. A *Markov Reward process* is a four tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$:

- A finite state space $\mathcal{S}$.
- A transition matrix $\mathcal{P}$; that is,

  $$\mathcal{P}_{ss'} = \mathbb{P}\left(S_{t+1} = s' \mid S_t = s\right).$$

- A reward function $\mathcal{R} : \mathcal{S} \to \mathbb{R}$ defined as

$$\mathcal{R} : s \mapsto \mathbb{E}[R_{t+1} \mid S_t = s].$$

  where R_{t+1} is the reward at time $t+1$.

- A discount factor $\gamma \in [0, 1]$.

A *Markov decision process* is a five tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$:

- A finite state space $\mathcal{S}$.
- A finite action space $\mathcal{A}$.
- A transition matrix $\mathcal{P}$; that is,

$$\mathcal{P}^a_{ss'} = \mathbb{P}\left(S_{t+1} = s' \mid S_t = s, A_t = a\right).$$

- A reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defined as

$$\mathcal{R} : (s, a) \mapsto \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a].$$

  where R_{t+1} is the reward at time $t+1$.

- A discount factor $\gamma \in [0, 1]$.

Example 1.6 (Policies in MDP)**.** Given a MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ with policy π, the agent's action will be leaded as follow:

1) Start from time t with an initial state $S_t = s$.
2) Take an action based on the policy: $A_t \sim \pi(\cdot | S_t = s)$.
3) Compute the reward: $(s, a) \mapsto \mathcal{R}(s, a)$.
4) Move to the next state based on the transition kernel: $S_{t+1} \sim \mathbb{P}(\cdot \mid S_t = s, A_t = a)$.

1.2 Bellman Equation

Theorem 1.7 (Bellman Expectation Equation)**.** *The state-value function can be decomposed into the sum of immediate reward and discounted value of successor state,*

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s].$$

The action-value function can similarly be decomposed,

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a].$$

Proof. Directly by the definition of $v_\pi(s)$:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma \sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k+1} \mid S_t = s] \\ &= \mathbb{E}_\pi \left[R_{t+1} + \gamma \mathbb{E}_\pi[\sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k+1} \mid S_{t+1}] \middle| S_t = s \right] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \end{aligned}$$

The action-value case is omitted. □

Remark. There are two other equivalent representations of Bellman expectation equation,

1) The Bellman expectation equation can be written as the matrix form as

$$q_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi q_\pi$$

with direct solution

$$q_\pi = (I - \gamma \mathcal{P}^\pi)^{-1} \mathcal{R}^\pi.$$

2) Or we can write it more explicitly,

$$q_\pi(s,a) = \mathcal{R}^\pi(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} q_\pi(s',a).$$

The following result gives the relation between the state and action value function.

Proposition 1.8. *Given a MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ with policy π, we always have*

$$\begin{aligned} v_\pi(s) &= \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s,a); \\ q_\pi(s,a) &= \mathcal{R}^\pi(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} v_\pi(s'). \end{aligned}$$

Proof. By definition,

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi [G_t \mid S_t = s] \\ &= \sum_a \mathbb{E}_\pi [G_t \mid S_t = s, A_t = a] \mathbb{P}(A_t = a \mid S_t = s) \\ &= \sum_a \pi(a|s) q_\pi(s,a) \end{aligned}$$

Then by the Bellman expectation equation,

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma \left(\sum_{s' \in \mathcal{S}} q_\pi(s', A_{t+1}) \mathcal{P}^a_{ss'} \right) \mid S_t = s, A_t = a] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma \left(\sum_{s' \in \mathcal{S}} v_\pi(s') \mathcal{P}^a_{ss'} \right) \mid S_t = s, A_t = a] \\ &= \mathcal{R}^\pi(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} v_\pi(s'). \end{aligned}$$

□

Theorem 1.9 (Bellman Optimality Equation)**.** *Let v_*, q_* be the optimal state-value function and the optimal action-value function, then*

$$\begin{aligned} v_*(s) &= \max_a q_*(s,a) \\ q_*(s,a) &= \mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} v_*(s'). \end{aligned}$$

Remark. We can also write it as below:

$$\begin{aligned} v_*(s) &= \max_a \Big[\mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} v_*(s') \Big]; \\ q_*(s,a) &= \mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} \max_{a'} q_*(s',a'). \end{aligned}$$

This form gives us an anther perspective of the optimal value function; if we define an operator

$$L : v \mapsto \max_a \Big[\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v(s')\Big],$$

then v_* is a stationary point of this operator (that is, $Lv_* = v_*$). The existence of optimal value function will be immediately implied by the contraction of L.

2 Dynamic Programming (DP)

Assume the MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ is known with $\gamma \in (0, 1)$.

Dynamic Programming A method for solving complex problems by breaking them down into subproblems; generally, it requires two properties to apply this method:

- Optimal substructure: an optimal solution can be constructed from optimal solutions of its subproblems.
- Overlapping subproblems: subproblems recur many times; solutions can be cached and reused.

Analysis Review Let X be a vector space. $\|\cdot\| : X \to [0, +\infty)$ is called a *norm* on X if it satisfies

- Subadditivity; $\|u + v\| \leq \|u\| + \|v\|$ for all $u, v \in X$.
- Absolutely homogeneous; $\|av\| = |a|\|v\|$ for all $a \in \mathbb{R}$ and $v \in X$.
- If $\|v\| = 0$, then $v = 0$.

A norm naturally defines a metric on the vector space X. If the topology induced by this metric is complete, then we call X a *Banach space.*

Example. *Given a finite set $\mathcal{S}$, let X be all bounded functional on $\mathcal{S}$. Then*

$$\|v\| := \max_{x \in \mathcal{S}} |v(x)|$$

defines a norm on X; moreover, X is a Banach space with this norm.

An operator $T : X \to X$ is called a γ-*contraction* if there exists $\gamma \in [0, 1)$ such that

$$\|Tu - Tv\| \leq \gamma \|u - v\|.$$

Theorem (Banach Fixed-Point Theorem)**.** *Let X be a Banach space with a contraction $T : X \to X$. Then T admits a unique fixed-point v^* (i.e. $Tv^* = v^*$); moreover, for any $v \in X$, the sequence $\{T^n v\}_{n \in \mathbb{N}}$ converges to v^*.*

Sketch of Proof. Let $v_n := T^n v$. Notice that $\{v_n\}$ is a Cauchy sequence; let $v_n \to v_*$. Then

$$\lim_{n \to \infty} v_n = \lim_{n \to \infty} T(v_{n-1}) = T(v_*)$$

by the continuity of T. □

2.1 Prediction: Policy Evaluation

Recall that given a policy π, we define the state value function as

$$v_\pi(s) := \mathbb{E}_\pi[G_t \mid S_t = s]$$

where $G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$ is the discounted return. The algorithm used to evaluate the map $\pi \mapsto v_\pi$ is called a *policy evaluation.*

In this subsection, we will give an iterative method of policy evaluation. Let's start from the Bellman expectation equation

$$v_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_\pi$$

where v_π is the state value function given a policy π. It naturally defines the Bellman expectation operator

$$T^\pi : v \mapsto \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v.$$

Theorem 2.1. *T^π is a γ-contraction.*

Proof. The norm of $\mathcal{P}^\pi$ is defined as the operator norm.

$$\begin{aligned}\|T^\pi(u) - T^\pi(v)\| &= \|(\mathcal{R}^\pi + \gamma\mathcal{P}^\pi u) - (\mathcal{R}^\pi + \gamma\mathcal{P}^\pi v)\| \\ &= \|\gamma\mathcal{P}^\pi(u-v)\| \\ &\le \gamma\,\|\mathcal{P}^\pi\|\,\|u-v\| \\ &\le \gamma\|u-v\|\end{aligned}$$

□

By Banach fixed-point theorem, an algorithm is naturally given:

1) Start from the value function v_n.
2) Compute $v_{n+1} \leftarrow \mathcal{R}^\pi + \gamma\mathcal{P}^\pi v_n$.
3) Go back to the first step.

Note: Reminder that there is a naive method of policy evaluation by directly solving the Bellman expectation equation,

$$v_\pi = (I - \gamma\mathcal{P}^\pi)^{-1}\mathcal{R}^\pi.$$

Which one is better? (The time complexity of computing matrix inverse by Gaussian elimination is about $\mathcal{O}(n^3)$ while $T^n v$ linearly converges to v_π.)

2.2 Control: Policy Iteration

Now we consider the control problem; that is, we aim to find the optimal policy π_*. The following theorem gives a greedy method to find a better policy, such kind of algorithms are called *policy improvement.*

Theorem 2.2. *For a deterministic policy $a = \pi(s)$ and the corresponding action state-value function q_π, define a new policy π' as follow*

$$\pi' : s \mapsto \arg\max_{a\in\mathcal{A}} q_\pi(s,a);$$

then we have $\pi' \ge \pi$.

Proof. It suffices to show that $v_\pi \le v_{\pi'}$.

$$\begin{aligned}v_\pi(s) \le q_\pi(s,\pi'(s)) &= \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \,|S_t = s\right] \\ &\le \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \,|S_t = s\right] \\ &\le \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, \pi'(S_{t+2})) \,|S_t = s\right] \\ &\le \mathbb{E}_{\pi'}\left[R_{t+1} + \gamma R_{t+2} + \ldots \,|S_t = s\right] = v_{\pi'}(s)\end{aligned}$$

□

Combine it with the policy evaluation; then we get an algorithm to find the optimal policy:

```
Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$
Output: Optimal policy $\pi_*$
Set initial policy $\pi$;
while not converged do
    $v \leftarrow v_\pi$ (policy evaluation);
    $q \leftarrow \mathcal{R} + \gamma\mathcal{P}v$;
    $\pi \leftarrow \arg\max_{a\in\mathcal{A}} q_\pi(s,a)$ (policy improvement);
end
```

Algorithm 1: Policy Iteration

If the policy improvement stops (i.e. $\pi' = \pi$), we will have

$$q_\pi(s, \pi'(s)) = q_\pi(s, \pi(s)) = v_\pi(s).$$

And by the definition of π',

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

Then we get

$$v_\pi = \max_{a\in\mathcal{A}} q_\pi(s, a).$$

By the remark part of Theorem 1.9, v_π is the optimal value function.

2.3 Control: Value Iteration

Now we introduce another method to find the optimal policy π_*. Recall that we have the Bellman optimality equation

$$v_*(s) = \max_{a\in\mathcal{A}} \left[\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')\right];$$

it naturally defines the Bellman optimality backup operator

$$T^* : v \mapsto \max_{a\in\mathcal{A}} [\mathcal{R}^a + \gamma\mathcal{P}^a v].$$

Theorem 2.3. *T^* is a γ-contraction.*

Proof. Without loss of generality, fix $s \in \mathcal{S}$ such that $T^*v(s) \geq T^*u(s)$. Define

$$a_*^s := \arg\max_{a\in\mathcal{A}} \left[\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')\right]$$

Then

$$\begin{aligned} T^*v(s) - T^*u(s) &\leq \left[\mathcal{R}(s, a_*^s) + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^{a_*^s} v(s')\right] - \left[\mathcal{R}(s, a_*^s) + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^{a_*^s} u(s')\right] \\ &= \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^{a_*^s} [v(s') - u(s')] \\ &\leq \gamma\|v - u\|. \end{aligned}$$

Therefore, for all $s \in \mathcal{S}$,

$$|T^*v(s) - T^*u(s)| \leq \gamma\|v - u\|;$$

it implies $\|T^*v - T^*u\| \leq \gamma\|v - u\|$. $\square$

Notice that we can apply an iterative method to find the optimal value function v_*. Then we can find the corresponding policy π_* by

$$\pi_*(s) = \arg\max_{a\in\mathcal{A}} q_*(s, a)$$

where $q_*(s, a) = \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$. Now we get the following algorithm:

```
Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$
Output: Optimal policy $\pi_*$
Set initial value function $v$;
while not converged do
    $v \leftarrow \max_{a\in\mathcal{A}}[\mathcal{R}^a + \gamma\mathcal{P}^a v]$;
end
$q \leftarrow \mathcal{R} + \gamma\mathcal{P}v$;
$\pi \leftarrow \arg\max_{a\in\mathcal{A}} q_\pi(s, a)$;
```

Algorithm 2: Value Iteration

3 Model-Free Prediction

Recall that the value function is defined as

$$v_\pi(s) = \mathbb{E}_\pi [G_t \mid S_t = s]$$

We have two methods to find v_π when the policy π is given:

1) Directly solve the Bellman expectation equation: $v_\pi = (I - \gamma \mathcal{P}^\pi)^{-1} \mathcal{R}^\pi$.

2) Iterative policy evaluation: $v_{n+1} \leftarrow \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_n$; finally, $v_n \to v_\pi$.

However, we must know the reward $\mathcal{R}$ and the transition kernel $\mathcal{P}$. In this section, we will introduce two methods to estimate the value function v_π when the MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ is not known.

Probability Review

Theorem (Strong Law of Large Numbers and Its Convergence Rate)**.** *Let* $X_1, \ldots, X_n \overset{iid}{\sim} X$ *with* $\mu := \mathbb{E}|X| < \infty$*, then*

$$\frac{1}{n}(X_1 + X_2 + \cdots + X_n) \xrightarrow{a.s.} \mu$$

as $n \to \infty$*. Moreover, if the variance* $\sigma^2 := \mathrm{Var}X < \infty$*, then for every* $\epsilon > 0$*,*

$$\limsup_{n\to\infty} [\frac{1}{n} \sum_{i=1}^{n} \frac{X_i - \mu}{\sigma}] \cdot \frac{\sqrt{n}}{\sqrt{2 \log \log n}} = 1. \tag{1}$$

Theorem (Central Limit Theorem and Its Convergence Rate)**.** *Let* $X_1, \ldots, X_n \overset{iid}{\sim} X$ *with* $\mu := \mathbb{E}|X| < \infty$ *and* $\sigma^2 = \mathrm{Var}X_i < \infty$*, then*

$$[\frac{1}{n}(X_1 + X_2 + \cdots + X_n) - \mu] \cdot \sqrt{n} \xrightarrow{D} N(0, \sigma^2)$$

as $n \to \infty$*. Moreover, if* $\mathbb{E}|X|^3 < \infty$*, then*

$$\|F_n(x) - \phi(x)\|_\infty \cdot \sqrt{n} \leq \frac{C\mathbb{E}|X|^3}{\sigma^3} \cdot \frac{1}{\sqrt{n}} \tag{2}$$

3.1 Monte-Carlo Policy Evaluation

Key idea: we use the empirical mean instead of the expected return. In Example 1.6, we start from $s \in \mathcal{S}$ and generate an episode $\{S_1, A_1, R_2, S_2, A_2, \ldots, S_k\}$. This method could be used to estimate the value function $v_\pi(s)$ as follow:

```
Input: Policy $\pi$, a state $s \in \mathcal{S}$
Output: State value function $V(s)$
Set the increment counter $N(s) \leftarrow 0$;
Set the increment total return $S(s) \leftarrow 0$;
while $N(s)$ is not large enough do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \ldots, S_k\}$;
    $t \leftarrow \arg\min\{t \mid S_t = s\}$ ;
    % Update only for the First Visit ;
    $G_t \leftarrow$ discounted accumulated reward;
    $N(s) \leftarrow N(s) + 1$;
    $S(s) \leftarrow S(s) + G_t$;
end
$V(s) \leftarrow S(s)/N(s)$.
```

Algorithm 3: First-visit MC Policy Evaluation

Remark. With mild modification, we can use Monte Carlo method to estimate the state-action value function q_π. We denote the estimator as Q.

The following result just says that the algorithm above is correct.

Theorem 3.1. *For the first-visit MC policy evaluation, $V(s)$ converges to $v_\pi(s)$.*

Sketch of Proof. It is easy to notice that (1) Each G_t is iid. (2) $\mathbb{E}|G_t| < \infty$. By the law of large numbers, $V(s) \to v_\pi(s)$ as $N(s)$ tends to infinity. □

Remark. Also, "...the standard deviation of its error falls as $1/\sqrt{n}$..."; here, $1/\sqrt{n}$ is the order of standard deviation in the CLT rather than the convergence rate of SLLN (1) neither the convergence rate of CLT (2). In Monte Carlo, we mainly focus on the standard deviation of our estimates; it is the rate of convergence in probability.

We also have the following every-visit MC algorithm; the most main different part is that the discounted rewards are not independent anymore.

```
Input: Policy $\pi$, a state $s \in \mathcal{S}$
Output: State value function $V(s)$
Set the increment counter $N(s) \leftarrow 0$;
Set the increment total return $S(s) \leftarrow 0$;
while $N(s)$ is not large enough do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$;
    for $t$ in $\{1, 2, \dots, k\}$ do
        if $S_t = s$ then
            % Update for Every Visit at state $s$;
            $G_t \leftarrow$ discounted accumulated reward;
            $N(s) \leftarrow N(s) + 1$;
            $S(s) \leftarrow S(s) + G_t$;
        end
    end
end
$V(s) \leftarrow S(s)/N(s)$.
```

Algorithm 4: Every-visit MC Policy Evaluation

Theorem 3.2. *For the every-visit MC policy evaluation, $V(s)$ converges to $v_\pi(s)$.*

Proof. Omitted. □

3.2 Temporal Difference (TD) Prediction

Temporal difference (TD) learning plays an importance role in reinforcement learning; it includes a large families of algorithms, including SARSA and Q-learning. In this section, we will introduce the most basic TD algorithm, TD(0).

Recall the Bellman expectation equation

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi \left[G_t | S_t = s\right] && \text{(MC)} \\ &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s\right] \\ &= \mathbb{E}_\pi \left[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} | S_t = s\right] \\ &= \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi \left(S_{t+1}\right) | S_t = s\right]. && \text{(DP)} \end{aligned}$$

The Monte Carlo method estimates the expectation in (MC); it is model-free, but computing G_t requires the information of the whole sequence. The dynamic programming method iteratively estimates $v_\pi(S_{t+1})$ in (DP); it doesn't require the episode to be terminated, but requires the MDP elements. The temporal difference (TD) learning can overcome both disadvantages: it is model-free, and it doesn't need to generate a completed sequence. The key approximation is

$$\begin{aligned} G_t &:= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \\ &= R_{t+1} + \gamma G_{t+1} \\ &\approx R_{t+1} + \gamma V(S_{t+1}) \end{aligned}$$

```
Input: Policy $\pi$
Output: State value function $V$
Initialize $V(s)$ for all $s \in \mathcal{S}$;
Initialize state $S$;
for each step of episode do
    Take action $A \sim \pi(\cdot|S)$; observe $R$ and next state $S'$;
    $V(S') \leftarrow V(S) + \alpha(\underset{\sim\sim\sim}{R + \gamma V(S')} - V(S))$;
    $S \leftarrow S'$;
end
```

Algorithm 5: TD Learning, TD(0)

Definition 3.3. In the algorithm above,

- $R + \gamma V(S')$ is called the TD target.
- $R + \gamma V(S') - V(S)$ is called the TD error.

3.3 Comparison: MC vs. TD

- Online learning.
  - TD can learn without the final outcome; works in continuing (non-terminating) environments.
  - MC can only learn from complete sequences; works for episodic (terminating) environments.
- Variance-bias trade-off.
  - TD target has low variance, some bias.
    Good convergence properties (even with function approximation); not very sensitive to initial value.
  - MD has high variance, zero bias.
    More efficient than MC; TD(0) converges to $v_\pi(s)$ but not always with function approximation; more sensitive to initial value.
- Certainty equivalence.
- Markov environment.
  - TD exploits Markov property; usually more efficient in Markov environments.
  - MC doesn't exploit Markov property; usually more effective in non-Markov environments.

***n*-Step Return** Define the *n-step return* as

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}).$$

Then the n-step TD learning:

$$V\left(S_t\right) \leftarrow V\left(S_t\right) + \alpha\left(G_t^{(n)} - V\left(S_t\right)\right).$$

Note that when $n = \infty$, it becomes the MC method.

TD(λ) Define the *λ-return* as

$$G_t^{\lambda} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}.$$

Then the TD updates:

$$V\left(S_t\right) \leftarrow V\left(S_t\right) + \alpha\left(G_t^{\lambda} - V\left(S_t\right)\right).$$

Note that it can only be computed from complete episodes.

4 Model-Free Control

Recall that we use the policy iteration method in Section 2 to find the optimal policy π_*; the policy iteration repeats the following two steps:

1. Policy evaluation: given a policy π; compute the corresponding value function v_π.
2. Policy improvement: give a value function v_π; find a better policy π'.

However, both of methods above rely on knowing the MDP tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. In this section, we give a model-free method to get the optimal policy by simply changing each model-based algorithm to a model-free one.

Importance Sampling Importance sampling is a variance-reduction technique used to reduce the variance (or relative error) of Monte Carlo estimation. The main difference is that we sample from the importance measure instead of the original one.

On-Policy vs. Off-Policy First, we introduce two concepts: the target policy π which we want to optimize, and the behavioral policy μ which generates the states, actions, and rewards. If π and μ are same, then the task is called the on-policy learning. If π and μ are different, then the task is called the off-policy learning.
In on-policy task, we hope our agent can play the game well by playing this game on itself, while in off-policy learning, we hope our agent can play the game well by observing other player's behavior.

4.1 On-Policy Monte-Carlo Control

In this subsection, we use the Monte Carlo method (see Algorithm 3 and Algorithm 4) to estimate the value function; however, we cannot use the estimate of state value function V, because it is unavoidable to use $\mathcal{R}$ and $\mathcal{P}$ to compute the policy π. Fortunately, we can make a mild modification to get the estimate of state-action value function Q (see the Remark of Algorithm 3). This method has another problem; if we keep using greedy method, some state-action pairs (s, a) may rarely occur. Therefore, we need to introduce the ϵ-greedy method:

Definition 4.1 (ϵ-Greedy)**.** For any $\epsilon \in [0, 1]$, define the ϵ-greedy policy w.r.t. $Q(s, a)$ as

$$\pi(a|s) := \begin{cases} \epsilon/|\mathcal{A}| + 1 - \epsilon & a^* = \arg\max_{a \in \mathcal{A}} Q(s, a) \\ \epsilon/|\mathcal{A}| & \text{o.w.} \end{cases}.$$

Theorem 4.2. *For any ϵ-greedy policy π, the ϵ-greedy policy π' with respect to q_π is an improvement, that is,*

$$v_{\pi'}(s) \geq v_\pi(s).$$

Proof.

$$\begin{aligned} q_\pi(s, \pi'(s)) &= \sum_{a \in \mathcal{A}} \pi'(a|s) q_\pi(s, a) \\ &= \epsilon/|\mathcal{A}| \sum_{a \in \mathcal{A}} q_\pi(s, a) + (1 - \epsilon) \max_{a \in \mathcal{A}} q_\pi(s, a) \\ &\geq \epsilon/|\mathcal{A}| \sum_{a \in \mathcal{A}} q_\pi(s, a) + (1 - \epsilon) \sum_{a \in \mathcal{A}} \frac{\pi(a|s) - \epsilon/|\mathcal{A}|}{1 - \epsilon} q_\pi(s, a) \\ &= \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s, a) = v_\pi(s) \end{aligned}$$

□

Now our policy iteration becomes:

1. Policy evaluation: given any policy π; use Q as an estimator of the state-action value function q_π.

2. Policy improvement: ϵ-greedy policy improvement.

It is natural to ask if this algorithm converges to the optimal policy π_*.

Definition 4.3 (Greedy in the Limit with Infinite Exploration (GLIE))**.** A sequence of policy is called *GLIE* if

- All state-action pairs are explored infinitely many times; that is

$$\lim_{k\to\infty} N_k(s,a) = \infty.$$

- The policy converges on a greedy policy,

$$\lim_{k\to\infty} \pi_k(a|s) = \mathbf{1}(a = \arg\max_{a'\in\mathcal{A}} Q_k(s,a')),$$

  where Q_k is learned action-value function.

Example 4.4. ϵ-greedy policy is GLIE if $\epsilon_k = \frac{1}{k}$.

```
Input:
Output: Optimal policy $\pi_*$
Set the increment counter $N(s,a) \leftarrow 0$ for all $(s,a) \in \mathcal{S}\times\mathcal{A}$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$;
    for $t$ in $\{1,2,\dots,k\}$ do
        $G_t \leftarrow$ discounted accumulated reward;
        $N(S_t,A_t) \leftarrow N(S_t,A_t) + 1$;
        $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \frac{1}{N(S_t,A_t)}(G_t - Q(S_t,A_t))$;
    end
    $\epsilon \leftarrow \frac{1}{n}$;
    $\pi \leftarrow \epsilon$-greedy$(Q)$;
end
```

Algorithm 6: GLIE Monte-Carlo Control

Theorem 4.5. *GLIE Monte-Carlo control converges to the optimal action-value function,*

$$Q(s,a) \to q_*(s,a).$$

4.2 On-Policy TD Control

Sarsa means state-action-reward-state-action. Reminder that the generalized policy iteration is:

1. Policy evaluation: given any policy π; use Q as an estimator of the state-action value function q_π.

2. Policy improvement: ϵ-greedy policy improvement.

Now we use TD prediction in the policy evaluation part.

```
Input: Parameters $\epsilon$ and $\alpha$
Output: Optimal policy $\pi_*$
Initialize $Q(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$;
while not converged do
    $\pi \leftarrow \epsilon$-greedy$(Q)$;
    Initialize $S$ and $A$;
    for each step of episode do
        Take action $A$; observe $R$ and $S'$;
        Generate $A' \sim \pi(\cdot|S')$;
        $Q(S,A) \leftarrow Q(S,A) + \alpha\,[R + \gamma Q(S',A') - Q(S,A)]$;
        $S \leftarrow S'$, $A \leftarrow A'$;
        Update $\alpha$;
        Update $\epsilon$;
    end
end
```

Algorithm 7: Sarsa

Theorem 4.6. *Assume the following conditions are satisfied:*

a) GLIE sequence of policies $\pi_t(a|s)$

b) Robbins-Monro sequence of step-sizes α_t:

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

Then SARSA converges to the optimal action-value function

4.3 Applications of Importance Sampling in Off-Policy Learning

In the off-policy setting, the behavior policy μ is different from the target policy π. Recall that if we fix any $s \in \mathcal{S}$, $\pi(\cdot|s)$ is a measure on $\mathcal{A}$; and we want to maximize the discounted expected reward $\mathbb{E}_\pi G_t$. However, all of our samples are generated by another measure $\mu(\cdot|s)$. The importance sampling is a natural way to estimate the expectation w.r.t. the measure π by sampling from the measure μ. The key relation is

$$\mathbb{E}_\pi X = \int X \mathrm{d}\pi = \int X \frac{\mathrm{d}\pi}{\mathrm{d}\mu} \mathrm{d}\mu = \mathbb{E}_\mu (X \cdot \frac{\mathrm{d}\pi}{\mathrm{d}\mu})$$

where $\frac{\mathrm{d}\pi}{\mathrm{d}\mu}$ is a random variable called Radon-Nikodym derivative (see Appendix A for more details). So we simply apply the relation above in the original algorithms.

Monte-Carlo Notice that the policy π induces a measure on $\mathcal{A} \times \mathcal{A} \times \dots \mathcal{A}$:

$$\begin{aligned}
&\mathbb{P}(A_t = a_t, A_{t+1} = a_{t+1}, \cdots, A_T = a_T \mid S_t = s_t) \\
=&\mathbb{P}(A_t = a_t \mid S_t = s_t)\mathbb{P}(A_{t+1} = a_{t+1}, \cdot, A_T = a_T \mid S_t = s_t, A_t = a_t) \\
=&\pi(a_t|s_t)\mathcal{P}^{a_t}_{s_t,s_{t+1}} \cdot \mathbb{P}(A_{t+1} = a_{t+1}, \cdots, A_T = a_T \mid S_{t+1} = s_{t+1}) \\
=&\dots \\
=&\pi(a_t|s_t) \cdot \pi(a_{t+1}|s_{t+1}) \dots \pi(a_T|s_T) \cdot \prod_t \mathcal{P}
\end{aligned}$$

Similarly, μ induces a measure

$$\begin{aligned}&\mathbb{Q}(A_t = a_t, A_{t+1} = a_{t+1}, \cdots, A_T = a_T \mid S_t = s_t)\\ =&\mu(a_t|s_t) \cdot \mu(a_{t+1}|s_{t+1}) \dots \mu(a_T|s_T) \cdot \prod_t \mathcal{P}\end{aligned}$$

Then the importance ratio is

$$W = \frac{\pi(a_t|s_t) \cdot \pi(a_{t+1}|s_{t+1}) \dots \pi(a_T|s_T)}{\mu(a_t|s_t) \cdot \mu(a_{t+1}|s_{t+1}) \dots \mu(a_T|s_T)}.$$

Temporal Difference Notice that $\pi(\cdot|S)$ and $\mu(\cdot|S)$ are measure on $\mathcal{A}$. The importance ratio is

$$W = \frac{\pi(\cdot|S)}{\mu(\cdot|S)}.$$

We present the off-policy MC algorithm here as an example.

```
Input: Behavioral policy $\mu$ %assume $\mu$ can explore all possibilities
Output: Optimal policy $\pi_*$
Initialize $Q$ as an estimator of action-value function;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_T\}$ using $\mu$;
    $N(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$;
    $S(s,a) \leftarrow 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$;
    for each time $t$ of generated episode do
        $G_t \leftarrow$ cumulative discounted return at time $t$;
        $W_t \leftarrow [\pi(a_t|s_t) \cdot \pi(a_{t+1}|s_{t+1}) \dots \pi(a_T|s_T)] \,/\, [\mu(a_t|s_t) \cdot \mu(a_{t+1}|s_{t+1}) \dots \mu(a_T|s_T)]$;
        $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$;
        $S(S_t, A_t) \leftarrow S(S_t, A_t) + W_t \cdot G_t$;
        $Q(S_t, A_t) \leftarrow S(S_t, A_t)/N(S_t, A_t)$;
    end
end
$\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} Q(s,a)$ for all $s \in \mathcal{S}$;
```

Algorithm 8: Off-Policy Every-Visit MC Control

4.4 Q-learning

Now we introduce an off-policy learning algorithm which doesn't require the importance sampling.

```
Input: Fixed policy $\pi$, the learning rate sequence $\alpha_t$
Output: Optimal policy $\pi_*$
Initialization;
while not converged do
    Take action $A_{t+1} \sim \pi(\cdot|S)$; oberve $R_{t+1}$ and $S_{t+1}$;
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha_t \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right]$;
    $t \leftarrow t + 1$;
end
$\pi_* = \arg\max_{a \in \mathcal{A}} Q$;
```

Algorithm 9: Online Q-Learning

Remark. We will make several comments on the Q-learning:

- The algorithm introduced in this subsection is the most basic form of Q-learning; sometimes, it is called Watkins' Q-learning algorithm. It is first proposed in his PhD thesis [Watkins, 1989].

- When this algorithm converges, we can get the optimal Q-function. Because when converged,

$$Q_*(S_t, A_t) = Q_*(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_{a'} Q_*(S_{t+1}, a') - Q_*(S_t, A_t)\right]$$

  where $S_{t+1} \sim \mathcal{P}^{A_t}_{S_t, S_{t+1}}$. Conditional on the past information:

$$\begin{aligned} \mathbb{E}_\pi \left[Q_*(S_t, A_t) \mid S_t = s, A_t = a\right] &= \mathcal{R}^a_s + \gamma \mathbb{E}_\pi \left[\max_{a'} Q_*(S_{t+1}, a') \mid S_t = s, A_t = a\right] \\ &= \mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} \max_{a'} Q_*(s', a') \end{aligned}$$

  It implies that

$$Q_* = \mathcal{R} + \gamma \mathcal{P} \max_a Q_*.$$

  It means Q_* is the solution of Bellman optimality equation.

Theorem 4.7. *Suppose the policy π satisfies the Assumption 5.3 and the learning rate sequence satisfies the Robbins-Monro conditions. Then Q-learning control converges to the optimal action-value function.*

Remark. Since the action-value function can be represented as the linear combination of $\{\mathbf{1}_{(s,a)}\}_{(s,a) \in \mathcal{S} \times \mathcal{A}}$, it would be a special case of linear approximation. We put off the proof to Section 5.

5 Function Approximation

Recall that in the dynamic programming, we use iterative policy evaluation to compute the value function v_π; in the model-free setting, we apply the Monte Carlo or temporal difference method. However, there are several problems when dealing with large MDPs:

- we cannot save the whole Q-function in memory;
- it would take a long time to visit all of state-action pairs.

In this section, we will introduce a solution for large MDPs. It is assumed that the value function could be approximated by a parametric family of functions (of course, the parameter space is much smaller than the state-action space). We mainly focus on the linear approximation.

Without loss of generality, we only present the linear approximation for the action-value function. It is easy to generalize this case to the state-value function. Let $\phi_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for $i \in \{1, 2, \ldots, M\}$ be linearly independent. Then their linear combination

$$\Phi := \overline{\mathrm{Span}\{\phi_i\}_{i=1}^M}$$

forms a closed convex set, so the projection operator Proj_Φ is well-defined. Moreover, every point in the space Φ with coordinate θ is denoted by

$$Q_\theta = \sum_{i=1}^M \phi_i \cdot \theta(i) = \phi^T \theta.$$

5.1 Prediction with Linear Approximation

First, we assume the value function Q is known. We want to solve the following optimization problem:

$$\min_{\theta \in \mathbb{R}^M} J(\theta) := \mathbb{E}_\pi \|Q_\theta(S, A) - Q(S, A)\|^2 \tag{3}$$

We can apply the stochastic gradient method to find its minima.

$$\theta \leftarrow \theta - \frac{\alpha}{2} \nabla J(\theta)$$

where

$$\begin{aligned}
\nabla J(\theta) &= \mathbb{E}_\pi \nabla \|Q_\theta(S, A) - Q(S, A)\|^2 \\
&= \mathbb{E}_\pi \left[2(Q_\theta - Q)\nabla Q_\theta\right] \\
&= 2\mathbb{E}_\pi \left[(Q_\theta - Q)\phi\right]
\end{aligned}$$

Therefore, our update becomes

$$\theta \leftarrow \theta - \alpha \mathbb{E}_\pi \left[(Q_\theta - Q)\phi\right] \tag{4}$$

Unfortunately, usually the value function is not given. We need to estimate the value function in the SGD iteration above:

- **Monte-Carlo**. We estimate the value function using the discounted future return G_t. Then

$$\theta \leftarrow \theta - \alpha \left[(Q_\theta - G_t)\phi(S_t, A_t)\right] \tag{5}$$

```
**Input:** Policy $\pi$, learning rate $\alpha$
**Output:** Value function $Q_\theta$
Initialize $\theta$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$ using $\pi$;
    for time step $t$ in the generated episode do
        $\theta \leftarrow \theta - \alpha\left[(Q_\theta(S_t, A_t) - G_t)\phi(S_t, A_t)\right]$;
    end
end
```
Algorithm 10: MC Policy Evaluation with Linear Approximation

- **Temporal difference**. We estimate the value function using the TD target:

$$\theta \leftarrow \theta - \alpha\left[(Q_\theta - (R_{t+1} + \gamma Q_\theta(S_{t+1}, A_{t+1})))\phi(S_t, A_t)\right] \tag{6}$$

```
**Input:** Policy $\pi$, learning rate $\alpha$
**Output:** Value function $Q_\theta$
Initialize $\theta$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$ using $\pi$;
    for time step $t$ in the generated episode do
        $\theta \leftarrow \theta - \alpha\left[(Q_\theta - R_{t+1} - \gamma Q_\theta(S_{t+1}, A_{t+1}))\phi(S_t, A_t)\right]$;
    end
end
```
Algorithm 11: TD(0) Policy Evaluation with Linear Approximation

Convergence Now let's consider the asymptotic property of the algorithms above. The following theorem gives the convergence of the Algorithm 10:

Theorem 5.1. *Assume the stepsize sequence satisfies Robbins-Monro conditions. Then the Algorithm 10 converges to the global minima of optimization problem (3).*

Proof. Since the objective function (3) is strongly convex. It suffices to show the SGD update is unbiased.

$$\begin{aligned}\mathbb{E}_\pi\left[(Q_\theta(S_t, A_t) - G_t)\phi(S_t, A_t) \mid S_t, A_t\right] &= (Q_\theta(S_t, A_t) - \mathbb{E}_\pi\left[G_t \mid S_t, A_t\right])\phi(S_t, A_t) \\ &= [Q_\theta(S_t, A_t) - Q(S_t, A_t)]\phi(S_t, A_t)\end{aligned}$$

Then we conclude the convergence by the traditional result in SGD theory. □

Then we consider the convergence of TD method. The proof is more complicated since it is not directly implied by the general result of SGD.

Theorem 5.2. *Assume the stepsize sequence satisfies Robbins-Monro conditions. Then the Algorithm 11 converges to the global minima of optimization problem (3).*

Proof. TOBE ADDED. □

5.2 Control with Linear Approximation

On-policy control As in Section 2, we can use the policy iteration method to find the optimal policy:

1. **Policy evaluation**: Given π, compute the approximated value function $Q \approx q_\pi$ with value function approximation (e.g. Algorithm 10 or Algorithm 11).
2. **Policy improvement**: ϵ-greedy policy improvement.

Off-policy control Now we consider the online Watkins' Q-learning algorithm with linear approximation. We have seen the original version in Section 4 and the proof is deferred to here. The iteration is

$$\theta_{t+1} = \theta_t + \alpha_t \cdot \left[R_{t+1} + \gamma \max_{a \in \mathcal{A}} \phi^T \theta_t(S_{t+1}, a) - \phi^T \theta_t(S_t, A_t) \right] \cdot \phi(S_t, A_t)$$

```
Input: Fixed policy $\pi$, learning rate sequence $\alpha_t$
Output: Optimal policy $\pi^*$
Initialization;
while not converged do
    Take action $A_{t+1} \sim \pi(\cdot|S)$; oberve $R_{t+1}$ and $S_{t+1}$;
    $\theta \leftarrow \theta - \alpha_t \left[ (Q_\theta - R_{t+1} - \gamma \max_{a \in \mathcal{A}} Q_\theta(S_{t+1}, a)) \phi(S_t, A_t) \right]$;
    $t \leftarrow t + 1$;
end
$\pi_* \leftarrow \arg\max_{\mathcal{A}} Q_\theta$;
```

Algorithm 12: Online Q-learning with Linear Approximation

Now we show the convergence of Q-learning control sequence. First, we list several notations we will use later:

- Let $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ be the MDP tuple. Here we consider a more general case than our usual definition. $\mathcal{S} \subset \mathbb{R}^d$ is a compact set; $\mathcal{A}$ is still a finite set. $\{S_t\}_{t \in \mathbb{N}}$ is generated by a fixed policy π.
- Let $\mathcal{Q}$ be the space of all action-value functions (or Q-functions).
- $\{\phi_i\}_{i=1}^M$ is linear independent and their spanned space is a ϵ-dense set for the space $\mathcal{Q}$. Let

$$\phi := \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_M \end{pmatrix}$$

be the $M \times 1$ vector. Moreover, we define

$$\Sigma_\pi = \mathbb{E}_\pi \left[\phi \phi^T \right];$$

it is a $M \times M$ matrix.

- Let $a^* = \arg\max_{\mathcal{A}} Q_\theta$. Define

$$\Sigma_\pi^*(\theta) = \mathbb{E}_\pi \left[\phi(S, a^*) \phi^T(S, a^*) \right];$$

it is a $M \times M$ matrix.

Assumption 5.3. *The following assumptions are used to make sure every state-action pair will be visited infinite times:*

- *The state process $\{S_t\}$ is uniformly ergodic with invariant measure μ.*
- *The fixed policy π satisfies $\pi(s, a) > 0$ for all $a \in \mathcal{A}$ and μ-almost all $s \in \mathcal{S}$.*

The following proof is adapted to our notations from [Melo et al., 2008].

Theorem 5.4. *Suppose the policy π satisfies the Assumption 5.3; and for all $\theta \in \mathbb{R}^M$,*

$$\Sigma_\pi > \gamma^2 \Sigma_\pi^*(\theta); \tag{7}$$

and the step-size sequence satisfies Robbins-Monro conditions. Then the Q-learning iteration defined in Algorithm 12 converges almost surely.

Proof. **#TODO**

- #Lyapunov theorem: consider the non-autonomous case
- #Uniqueness: delete this part

The proof will be divided into three parts:

- The limit point of the Q-learning control iteration is the equilibrium point the following ODE:
$$\dot{\theta} = \mathbb{E}_\pi \left[[R + \gamma \max_{\mathcal{A}} \phi^T \theta - \phi^T \theta] \cdot \phi \right].$$
Fix two trajectories of ODE with different initial conditions: $\theta_1(t)$ and $\theta_2(t)$. Define
$$\tilde{\theta}_t := \theta_1(t) - \theta_2(t).$$
- We choose $(x_1, x_2) \mapsto \|x_1 - x_2\|^2$ as the **Lyapunov function candidate**. It is obvious that it is positive definite. And its derivative along the trajectory (θ_1, θ_2) is
$$\begin{aligned}
\frac{d}{dt}\|\tilde{\theta}_t\|^2 &= 2\tilde{\theta}_t^T \cdot \frac{d}{dt}\Big(\theta_1(t) - \theta_2(t)\Big) \\
&= 2\tilde{\theta}_t^T \cdot \mathbb{E}_\pi \left[\left(\gamma \max_{\mathcal{A}} \phi^T \theta_1 - \phi^T \theta_1 - \gamma \max_{\mathcal{A}} \phi^T \theta_2 + \phi^T \theta_2\right) \cdot \phi \right] \\
&= 2\tilde{\theta}_t^T \cdot \mathbb{E}_\pi \left[\left(\gamma (\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) - \phi^T \tilde{\theta}_t\right) \cdot \phi \right] \\
&= -2\tilde{\theta}_t^T \left[\mathbb{E}_\pi \phi\phi^T\right] \tilde{\theta}_t + 2\gamma \cdot \tilde{\theta}_t^T \mathbb{E}_\pi \left[(\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) \cdot \phi \right] \\
&= -2\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t + 2\gamma \cdot \mathbb{E}_\pi \left[\tilde{\theta}_t^T \phi \cdot (\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) \right]
\end{aligned}$$
- It remains to show that $\frac{d}{dt}\|\tilde{\theta}\|^2 < 0$ for all t (then $x \mapsto \|x\|^2$ is the **Lyapunov function**).
    - Let $a_1^* = \arg\max_{a \in \mathcal{A}} \phi^T \theta_1$ and $a_2^* = \arg\max_{a \in \mathcal{A}} \phi^T \theta_2$. Then
    $$\begin{aligned}
    \max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2 &= \phi^T(s, a_1^*)\theta_1 - \phi^T(s, a_2^*)\theta_2 \\
    &= \phi^T(s, a_1^*)\tilde{\theta} + \phi^T(s, a_1^*)\theta_2 - \phi^T(s, a_2^*)\theta_2 \\
    &\le \phi^T(s, a_1^*)\tilde{\theta}
    \end{aligned}$$
    where in the last step we notice that $\phi^T(s, a_1^*)\theta_2 \le \phi^T(s, a_2^*)\theta_2$ by the definition of a_2^*. Similarly, we have
    $$\max_{\mathcal{A}} \phi^T \theta_2 - \max_{\mathcal{A}} \phi^T \theta_1 \le \phi^T(s, a_2^*)\tilde{\theta}.$$
    - Define $S_t^+ := \{(s, a) \in \mathcal{S} \times \mathcal{A} \colon \tilde{\theta}_t^T \phi > 0\}$ and $S_t^- := \{(s, a) \in \mathcal{S} \times \mathcal{A} \colon \tilde{\theta}_t^T \phi < 0\}$. Then we have
    $$\begin{aligned}
    &\mathbb{E}_\pi \left[\tilde{\theta}_t^T \phi \cdot (\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) \right] \\
    =&\mathbb{E}_\pi \left[\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi \cdot (\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) \right] + \mathbb{E}_\pi \left[\mathbf{1}_{S_t^-} \cdot \tilde{\theta}_t^T \phi \cdot (\max_{\mathcal{A}} \phi^T \theta_1 - \max_{\mathcal{A}} \phi^T \theta_2) \right] \\
    \le&\mathbb{E}_\pi \left[\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi \cdot \tilde{\theta}_t^T \phi(s, a_1^*) \right] + \mathbb{E}_\pi \left[\mathbf{1}_{S_t^-} \cdot \tilde{\theta}_t^T \phi \cdot \tilde{\theta}_t^T \phi(s, a_2^*) \right]
    \end{aligned}$$
    - Then by Cauchy–Schwarz inequality,
    $$\begin{aligned}
    \mathbb{E}_\pi \left[\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi \cdot \tilde{\theta}_t^T \phi(s, a_1^*) \right] &= \mathbb{E}_\pi \left[\left(\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi\right) \cdot \left(\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi(s, a_1^*)\right) \right] \\
    &\le \sqrt{\mathbb{E}_\pi \left(\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi\right)^2 \cdot \mathbb{E}_\pi \left(\mathbf{1}_{S_t^+} \cdot \tilde{\theta}_t^T \phi(s, a_1^*)\right)^2} \\
    &\le \sqrt{\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t \cdot \tilde{\theta}_t^T \Sigma_\pi^*(\theta_1) \tilde{\theta}_t}
    \end{aligned}$$
    Similarly,
    $$\mathbb{E}_\pi \left[\mathbf{1}_{S_t^-} \cdot \tilde{\theta}_t^T \phi \cdot \tilde{\theta}_t^T \phi(s, a_2^*) \right] \le \sqrt{\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t \cdot \tilde{\theta}_t^T \Sigma_\pi^*(\theta_2) \tilde{\theta}_t}$$

- Finally, we apply our assumption (7), $\Sigma_\pi > \gamma^2 \Sigma_\pi^*(\theta)$:

$$\begin{aligned}\frac{d}{dt}\|\tilde{\theta}_t\|^2 &\le -2\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t + 2\gamma\sqrt{\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t \cdot \tilde{\theta}_t^T \Sigma_\pi^*(\theta_1)\tilde{\theta}_t} + 2\gamma\sqrt{\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t \cdot \tilde{\theta}_t^T \Sigma_\pi^*(\theta_2)\tilde{\theta}_t} \\ &< -2\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t + 2\tilde{\theta}_t^T \Sigma_\pi \tilde{\theta}_t = 0.\end{aligned}$$

Then the proof is completed. We just show that the Q-learning sequence has the unique limit point. It is easy to see that when the algorithm terminates,

$$\phi\phi^T\theta^* = [R + \gamma \max_{\mathcal{A}} \phi^T\theta^*] \cdot \phi.$$

Conditional on the current action-state pair (S, A),

$$\phi\phi^T\theta^* = \phi \cdot [\mathcal{R}^\pi(S, A) + \gamma \mathcal{P}^\pi \max_{\mathcal{A}} \phi^T\theta^*].$$

Since $\phi\phi^T$ is full-rank (by linear independence), we have

$$\theta^* = (\phi\phi^T)^{-1}\phi \cdot [\mathcal{R}^\pi(S, A) + \gamma \mathcal{P}^\pi \max_{\mathcal{A}} \phi^T\theta^*]$$

Let $T^* : Q \mapsto \mathcal{R}^\pi + \gamma\mathcal{P}^\pi \max_{a\in\mathcal{A}} Q$ be the Bellman optimality operator (see Theorem 1.9). Multiple ϕ on both side:

$$Q_{\theta^*} = \phi^T(\phi\phi^T)^{-1}\phi \cdot T^*(Q_{\theta^*}).$$

Define $\text{Proj}_\Phi := \phi^T(\phi\phi^T)^{-1}\phi$. Note that $\text{Proj}_\Phi : \mathcal{Q} \to \mathcal{Q}$ is the projection operator; it maps any $Q \in \mathcal{Q}$ to the nearest point on $\Phi := \overline{\text{Span}\{\phi_i\}_{i=1}^M}$. Particularly, if $\Phi = \mathcal{Q}$ and $\{\phi_i\}$ forms a unit orthogonal basis, then Q_{θ^*} is the solution of Bellman equation; so it induces the optimal policy. □

Now let's re-consider the omitted proof of Theorem 4.7.

Corollary 5.5. *Suppose the policy π satisfies the Assumption 5.3 and the step-size sequence satisfies Robbins-Monro conditions. Then the Q-learning control sequence given in Algorithm 9 converges to the optimal action-value function.*

Proof. It suffices to show that the look-up table basis satisfies the condition (7). Notice that Σ_π is an identity matrix and $\Sigma_\pi^*(\theta)$ is similar to the identity matrix but some terms at the diagonal are zeros. So, for every $\gamma < 1$, the equation (7) holds.
Then because the look-up table basis $\{\mathbf{1}_{(s,a)}\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ is a unit orthogonal basis and it spans the whole space $\mathcal{Q}$. By Theorem 5.4, the unique limit point satisfies the Bellman optimality equation. Proof completed. □

6 Policy Gradient

We have learned two methods to find the optimal policy: policy iteration and value iteration. Both methods are required to compute the value function (value-based). In this section, we will introduce a policy-based approach to find the optimal policy. Its main idea is similar to the function approximation: assume the policy is in the parametric family $\{\pi_\theta\}$; and its performance can be represented as $J(\theta)$ (we call it *policy objective functions*); we aim to maximize its performance by stochastic gradient ascent:

$$\theta_{t+1} \leftarrow \theta_t + \alpha \widehat{\nabla J(\theta_t)}$$

where $\widehat{\nabla J(\theta_t)}$ is a stochastic estimate of $\nabla J(\theta_t)$. Here listed several choices of parametric family of policies:

(1) Softmax policy:
$$\pi_\theta(a|s) \propto e^{\phi^T(s,a)\theta}.$$

(2) Gaussian policy:
$$a \sim N(\phi^T(s)\theta, \sigma^2).$$

(3) Direct parameterization:
$$\pi_\theta(a|s) \in \Delta(\mathcal{A})^{|\mathcal{S}|}$$
where $\Delta(\mathcal{A})^{|\mathcal{S}|}$ is the simplex with length $|\mathcal{S}|$ and elements indexed by the set $\mathcal{A}$.

Note that (1) and (3) contains all of stochastic policies so they are complete; (2) belongs to the restricted parameterization family. And generally, $J(\theta)$ is not concave for the direct parameterization and softmax parameterization. See Lemma 3.1 in [Agarwal et al., 2019] for an example.
Lastly, besides REINFORCE introduced in this section, there are many other policy gradient algorithms in this field. See [Weng, 2018][1]; it introduces 15+ related algorithms and their implementations.

6.1 Policy Gradient Theorem

Assume the distribution of initial state S_0 is μ. The expected return of a policy π is defined as

$$\begin{aligned}\mathbb{E}_\pi[\sum_{t=1}^{\infty}\gamma^t R_t] &= \mathbb{E}_\pi[\sum_{t=1}^{\infty}\gamma^t \mathbb{E}_\pi[R_t|A_{t-1},S_{t-1}]] \\ &= \mathbb{E}_{S_0\sim\mu}\mathbb{E}_\pi[\sum_{t=0}^{\infty}\gamma^t\mathcal{R}(S_t,A_t) \mid S_0].\end{aligned}$$

We can use this as a measure to evaluate the performance of a policy π; define

$$J(\theta) = \mathbb{E}_{S_0\sim\mu}\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty}\gamma^t\mathcal{R}(S_t,A_t) \mid S_0]$$

where a policy is determined by its parameter θ, denoted by π_θ. We maximize its performance using

$$\theta_{t+1} \leftarrow \theta_t + \alpha\ \nabla J(\theta_t).$$

The following theorem gives the explicit expression of $\nabla J(\theta)$.

Theorem 6.1 (Policy Gradient Theorem)**.** *Let $J(\theta)$ be given as above. Then*

$$\nabla J(\theta) = \frac{1}{1-\gamma}\mathbb{E}_{S_0\sim\mu}\mathbb{E}_{s\sim d_{S_0}^{\pi_\theta}}\mathbb{E}_{a\sim\pi(\cdot|s)}[\nabla_\theta \log\pi_\theta(a|s)q_\pi(s,a) \mid S_0].$$

[1] https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html

Proof. [2] Let $\tau_t := \{S_0, A_0, \dots, S_t, A_t\}$ be the trajectory up to time t. Then

$$\mathbb{E}_\pi \mathcal{R}(S_t, A_t) = \sum_{\tau_t} \mathbb{P}_\pi(\tau_t)\mathcal{R}(s_t, a_t). \tag{8}$$

where $\mathbb{P}_\pi$ is a possibility measure on $(\mathcal{S} \times \mathcal{A})^t$ induced by π (we have seen this trick in Section 4.3) and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward in MDP. Fix the initial state S_0. Now we compute the gradient of $J(\theta)$:

$$\begin{aligned}
\nabla J(\theta) &= \nabla \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(S_t, A_t) \mid S_0] \\
&= \nabla \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi_\theta}[\mathcal{R}(S_t, A_t) \mid S_0] \\
(8) \quad &= \nabla \sum_{t=0}^{\infty} \gamma^t \sum_{\tau_t} \mathbb{P}_{\pi_\theta}(\tau_t)\mathcal{R}(s_t, a_t) \\
&= \sum_{t=0}^{\infty} \gamma^t \sum_{\tau_t} \Big(\nabla \mathbb{P}_{\pi_\theta}(\tau_t)\Big)\mathcal{R}(s_t, a_t) \\
&= \sum_{t=0}^{\infty} \gamma^t \sum_{\tau_t} \Big(\nabla \log \mathbb{P}_{\pi_\theta}(\tau_t)\Big)\mathcal{R}(s_t, a_t)\mathbb{P}_{\pi_\theta}(\tau_t) \\
&= \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi_\theta}[\nabla \log \mathbb{P}_\pi(\tau_t)\mathcal{R}(S_t, A_t) \mid S_0] \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t \nabla \log \mathbb{P}_\pi(\tau_t)\mathcal{R}(S_t, A_t) \mid S_0]
\end{aligned}$$

Now we compute $\mathbb{P}_\pi(\tau_t)$:

$$\begin{aligned}
\mathbb{P}_\pi(\tau_t) &:= \mathbb{P}_\pi(S_0 = s_0, A_0 = a_0, \dots, S_t = s_t, A_t = a_t) \\
&= \prod_{t'=0}^{t-1} \Big(\pi_\theta(a_{t'}|s_{t'})\mathcal{P}^{a_{t'}}_{s_{t'}, s_{t'+1}}\Big) \cdot \pi_\theta(a_t|s_t)
\end{aligned}$$

Then

$$\nabla_\theta \log \mathbb{P}_\pi(\tau_t) = \sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'})$$

where all $\mathcal{P}$ are gone since they do not rely on θ.

[2] The proof is from this note (link: http://katselis.web.engr.illinois.edu/ECE586/Lecture14.pdf). Our result is different from David Silver's slides but fortunately it is same as the setting in [Agarwal et al., 2019].

Now plug it into $\nabla J(\theta)$:

$$\begin{aligned}
\nabla J(\theta) &= \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty}\gamma^t[\sum_{t'=0}^{t}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})]\mathcal{R}(S_t,A_t)\mid S_0] \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty}\sum_{t':t\geq t'}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\gamma^t\mathcal{R}(S_t,A_t)\mid S_0] \\
\text{(change order)} \quad &= \mathbb{E}_{\pi}[\sum_{t'=0}^{\infty}\sum_{t:t\geq t'}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\gamma^t\mathcal{R}(S_t,A_t)\mid S_0] \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t'=0}^{\infty}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\underbrace{\sum_{t\geq t'}\gamma^t\mathcal{R}(S_t,A_t)}_{\text{Cond. on }(S_{t'},A_{t'})...}\mid S_0] \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t'=0}^{\infty}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\underbrace{\sum_{(s,a)}\mathbb{E}[\sum_{t\geq t'}\gamma^t\mathcal{R}(S_t,A_t)\mid S_{t'}=s,A_{t'}=a]\cdot\mathbb{P}(S_{t'}=s,A_{t'}=a\mid S_0=s)\mid S_0]}_{\text{...get this.}} \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t'=0}^{\infty}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\sum_{(s,a)}\gamma^{t'}q_\pi(s,a)\cdot\mathbb{P}(S_{t'}=s,A_{t'}=a\mid S_0=s)\mid S_0] \\
&= \mathbb{E}_{\pi_\theta}[\sum_{t'=0}^{\infty}\nabla_\theta \log\pi_\theta(A_{t'}|S_{t'})\gamma^{t'}\mathbb{E}_\pi[q_\pi(S_{t'},A_{t'})]\mid S_0] \\
&= \sum_{t=0}^{\infty}\mathbb{E}_\pi\left[\gamma^t\nabla_\theta\log\pi_\theta(A_t|S_t)q_\pi(S_t,A_t)\mid S_0\right]
\end{aligned}$$

We keep simplifying this formula.

$$\begin{aligned}
&\sum_{t=0}^{\infty}\mathbb{E}_{\pi_\theta}\left[\gamma^t\nabla_\theta\log\pi_\theta(A_t|S_t)q_\pi(S_t,A_t)\mid S_0\right] \\
=&\sum_{t=0}^{\infty}\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\gamma^t\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\cdot\pi_\theta(a|s)\cdot\mathbb{P}(S_t=s\mid S_0) \\
=&\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\sum_{t=0}^{\infty}\gamma^t\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\cdot\pi_\theta(a|s)\cdot\mathbb{P}(S_t=s\mid S_0) \\
=&\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\cdot\pi_\theta(a|s)\sum_{t=0}^{\infty}\gamma^t\cdot\mathbb{P}(S_t=s\mid S_0) \\
=&\sum_{s\in\mathcal{S}}\sum_{t=0}^{\infty}\gamma^t\cdot\mathbb{P}(S_t=s\mid S_0)\sum_{a\in\mathcal{A}}\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\cdot\pi_\theta(a|s)
\end{aligned}$$

Now we turn the fixed initial state S_0 to be random. Then we get

$$\begin{aligned}
\nabla J(\theta) &= \mathbb{E}_{S_0\sim\mu}\sum_s\left(\sum_{t=0}^{\infty}\gamma^t\cdot\mathbb{P}(S_t=s\mid S_0)\right)\mathbb{E}_{a\sim\pi(\cdot|s)}[\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\mid S_0] \\
&= \frac{1}{1-\gamma}\mathbb{E}_{S_0\sim\mu}\mathbb{E}_{s\sim d_{S_0}^{\pi_\theta}}\mathbb{E}_{a\sim\pi(\cdot|s)}[\nabla_\theta\log\pi_\theta(a|s)q_\pi(s,a)\mid S_0]
\end{aligned}$$

where we normalize $\sum_{t=0}^{\infty}\gamma^t\cdot\mathbb{P}(S_t=s\mid S_0)$ as a distribution on $\mathcal{S}$, $d_{S_0}^{\pi_\theta}$. $\square$

It should be mentioned that there are other choices of policy objective functions (see David Silver's slides). Since the result is similar to the discounted reward case, we omit the calculation of their gradients and just summary the result below:

Theorem 6.2. *(Policy Gradient Theorem) Let $J(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi(a|s)\mathcal{R}_s^a$. Then*

$$\nabla_\theta J(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi(a|s) \nabla_\theta \log \pi_\theta(s,a) q_{\pi_\theta}(s,a).$$

6.2 REINFORCE: Monte Carlo Policy Gradient

Based on the policy gradient theorem, we immediately get an algorithm. For each update, we use Monte Carlo method to estimate $\nabla J(\theta)$. The following pseudo-code from [Sutton and Barto, 2018] is written for the case where d^{π_θ} is an atomic measure at the initial state.

```
Output: Optimal policy $\pi^*$
Initialize $\theta$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$ using $\pi_\theta$;
    for time step $t$ in the generated episode do
        $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(S_t, A_t) Q_t^{\pi_\theta}$;
    end
end
```

Algorithm 13: MC Policy Gradient (REINFORCE)

Remark. Notice that the action value function Q^{π_θ} is not known. Therefore, we cannot directly use the algorithm above. Generally, we need to use a unbiased estimate of Q^{π_θ} take replacement of Q^{π_θ}.

Because the expectation of gradient update is exactly the true gradient. The convergence of this algorithm is guaranteed with diminishing learning rate by the SGD theory; so the proof is omitted here.

Theorem 6.3. *Assume the stepsize sequence satisfies Robbins-Monro conditions. Then the iteration sequence in Algorithm 13 converges to a stationary point.*

REINFORCE with Baseline Now we consider a more general algorithm, REINFORCE with baseline. Let

$$J(\theta) = \sum_{t=0}^{H} \mathbb{E}_\pi \left[\gamma^t \nabla_\theta \log \pi_\theta(A_t|S_t) \left(q_\pi(S_t, A_t) - b(S_t) \right) \mid S_0 \right]$$

where $b(s)$ is an arbitrary baseline (it could be any function, even a random variable). Then the parameter update (in the Algorithm 13) becomes

$$\theta_{t+1} \leftarrow \theta_t + \alpha \left(G_t - b(S_t) \right) \frac{\nabla_\theta \pi_\theta(A_t, S_t)}{\pi_\theta(S_t|S_t)}.$$

```
Output: Optimal policy $\pi^*$
Initialize $\theta$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$ using $\pi_\theta$;
    for time step $t$ in the generated episode do
        $\theta \leftarrow \theta + \alpha \left( G_t - b(S_t) \right) \nabla_\theta \log \pi_\theta(A_t, S_t).$;
    end
end
```

Algorithm 14: REINFORCE with Baseline

6.3 Actor-Critic Policy Gradient

We use a *critic* to estimate the action-value function (Q-function) Q^{π_θ}. Assume the Q-function can be represented using function approximation

$$Q_w \approx Q^{\pi_\theta}.$$

Then our algorithm should include two steps:

- Critic: update the Q-function parameter w.
- Actor: update the policy parameter θ.

Now we have the actor-critic policy gradient algorithm:

```
Output: Optimal policy $\pi^*$
Initialize $\theta$ and $w$;
while not converged do
    Generate an episode $\{S_1, A_1, R_2, S_2, A_2, \dots, S_k\}$ using $\pi_\theta$;
    for time step $t$ in the generated episode do
        $\theta \leftarrow \theta + \alpha\left(G_t - b(S_t)\right)\nabla_\theta \log \pi_\theta(A_t, S_t)$.;
        $w \leftarrow w + \beta[R_t + \gamma Q_w(S_{t+1}, A_{t+1}) - Q_w(S_t, A_t)]\phi(S_t, A_t)$;
    end
end
```

Algorithm 15: Q Actor-Critic

Note that in the actor-critic algorithm, generally the update would be a biased estimate of the true gradient. Fortunately, we have the following **compatible function approximation theorem**. It says when the function approximation is chosen appropriately, our update is still unbiased.

Theorem 6.4 (Compatible Function Approximation Theorem)**.** *If the following two conditions are satisfied*

*1) Value function approximation is **compatible** to the policy; that is,*

$$\nabla_w Q_w(s,a) = \nabla_\theta \log \pi_\theta(s,a).$$

2) Value function parameters w minimize the mean-squared error; that is,

$$\epsilon = \mathbb{E}_{\pi_\theta}(Q^{\pi_\theta}(s,a) - Q_w(s,a))^2.$$

Then the policy gradient is exact; that is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)].$$

Proof. [3] By condition (2), we have

$$\nabla_w \epsilon = 0.$$

Then by the definition of ϵ and condition (1),

$$\mathbb{E}_{\pi_\theta}(Q^{\pi_\theta}(s,a) - Q_w(s,a))\nabla_w Q_w(s,a) = 0$$
$$\mathbb{E}_{\pi_\theta}(Q^{\pi_\theta}(s,a) - Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a) = 0$$

Re-arrange it:

$$\mathbb{E}_{\pi_\theta}[Q^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(s,a)] = \mathbb{E}_{\pi_\theta}[Q_w(s,a))\nabla_\theta \log \pi_\theta(s,a)]$$

Then the proof is completed. □

Remark. The condition (1) is a super strong condition. [Sutton et al., 2000] mentions that "...Q_w being linear in the features given on the righthand side may be the only way to satisfy this condition".

[3] This proof is from David Sliver's lecture slides, Lecture 7.

A Importance Sampling

Edited from my note for PSTAT 221B - ADV PROBILTY THEORY. Mainly included: the definition, examples, and applications of importance sampling.

Notation Let $\mathbf{P}$ be any probability measure on the measurable space $(\Omega, \mathcal{F})$ and $X : \Omega \to \mathbb{R}$ be any random variable. $\mathbf{P}(X)$ represents the expectation of X with respective to $\mathbf{P}$.

Why do we need importance sampling? Let's consider the traditional **Monte Carlo method** used to estimate the expectation. Let V be a random variable. If $y = \mathbf{P}(V)$, then we call V is the $\mathbf{P}$-estimator of y. And define

$$\hat{V}_M = \frac{1}{M}\sum_{j=1}^{M} V^{(j)}$$

where $V^{(j)}$ are iid samples of V. Then $\hat{V}_M$ is called the estimate of y. Obviously, it is unbiased; that is, $\mathbf{P}(\hat{V}_M) = y$. We consider a very rare event A and define $V = \mathbf{1}_A$. Then the variance of estimator is $\mathrm{Var}(\hat{V}) = \frac{1}{M}\mathbf{P}(A)(1-\mathbf{P}(A))$. But its relative error

$$\sqrt{\frac{\mathbf{P}(A)(1-\mathbf{P}(A))}{M\mathbf{P}(A)^2}}$$

could be very large.

Definition A.1. Let $\mathbf{P}$ and $\mathbf{Q}$ be two measures on the measurable space $(\Omega, \mathcal{F})$. If for some non-negative random variable Z with $\mathbf{P}(Z) = 1$ such that

$$\mathbf{Q} = Z\mathbf{P},$$

then it is called $\mathbf{Q} \ll \mathbf{P}$ with Radon-Nikodym derivative Z.

Definition A.2 (Importance measures)**.** Let $\mathbf{Q} \ll \mathbf{P}$ with Radon-Nikodym derivative Z. If $Z > 0$ when $V \neq 0$, then $\mathbf{Q}$ is called an importance measure.

Now let $Y = LV$ be the $\mathbf{Q}$-estimator of y for $L = \begin{cases} \frac{1}{Z} & \text{if } Z > 0 \\ 0 & \text{o.w.} \end{cases}$.

$$\hat{y}_M = \frac{1}{M}\sum_{j=1}^{M} Y^{(j)}$$

where $\{Y^{(j)}\}$ are iid samples of Y.

Lemma A.3. *Let $\mathbf{Q}_* = Z_*\mathbf{P}$ where $Z_* = \frac{|V|}{\mathbf{P}(|V|)}$ provided $\mathbf{P}(|V|) < \infty$. Then $\mathbf{Q}_*$ minimizes the variance over all importance measures $\mathbf{Q}$.*

Proof. Because $\mathbf{Q}(Y) = y$ for every importance measure Q, it suffices to prove $\mathbf{Q}_*(Y_*^2) \leq \mathbf{Q}(Y^2)$:

$$\begin{aligned}
\mathbf{Q}_*(Y_*^2) &= \mathbf{Q}_*((L_*V)^2) \\
&= \mathbf{P}(Z_*(L_*V)^2) \\
&= \mathbf{P}(V^2/Z_*)
\end{aligned}$$

where $L_* = \frac{1}{Z_*}\mathbf{1}_{\{Z_*>0\}}$. Now consider $\mathbf{Q} = Z\mathbf{P}$ for any Z such that $\mathbf{Q}$ is an importance measure. Then

$$\begin{aligned}
\mathbf{Q}_*(Y_*^2) &= \mathbf{P}(V^2/Z_*) \\
&= \mathbf{Q}(\frac{V^2}{Z_*}L) \\
&= \mathbf{Q}(\frac{V^2}{|V|}L)\mathbf{P}(|V|) \\
&= (\mathbf{Q}(|V|L))^2 \\
&\leq \mathbf{Q}(L^2V^2)
\end{aligned}$$

where we use Jensen's inequality for the last inequality. □

Corollary A.4. *If $V > 0$, then $\mathbf{Q}_*$-variance is* 0.

Proof. Let $Z = V/\mathbf{P}(V)$. By the previous lemma, we have

$$\mathbf{Q}_*(Y^2) - y^2 \leq \mathbf{Q}(V^2/Z^2) - y^2 = 0;$$

therefore, the $\mathbf{Q}_*$-variance is 0. □

Example A.5. Let $Z_\theta = e^{\theta V - \Lambda(\theta)}$ where $\Lambda(\theta) = \log \mathbf{P}(e^{\theta V})$. Then we can define the importance measure

$$\mathbf{Q}_\theta = Z_\theta \mathbf{P}.$$

Take $V \sim \text{Exp}(1)$ under $\mathbf{P}$. Then $y = \mathbf{P}(V) = 1$ and $\mathbf{P}(V^2) = 2$. **First**, we compute the distribution of V under $\mathbf{Q}_\theta$. Notice that for all $\theta < 1$, $\Lambda(\theta) = -\log(1-\theta)$.
We want to prove $V \sim \text{Exp}(1-\theta)$ for $\theta < 1$ under $\mathbf{Q}_\theta$. First, we notice

$$M_t = \mathbf{1}_{\{V \leq t\}} - t \wedge V.$$

is a martingale under $\mathbf{P}$. Recall that for every Poisson process N with rate 1, $N_t - t$ is a martingale; and the first jumping time $\tau = \inf_t\{t > 0 : N_t > 0\}$ is a stopping time with the distribution $\text{Exp}(1)$. Therefore, the $\mathbf{P}$-martingale M is exactly the stopped martingale of $N_t - t$ at τ.
Second, we apply the Girsanov theorem to this $\mathbf{P}$-martingale M. We just learn that $M_t = \mathbf{1}_{\{V \leq t\}} - t \wedge V$ is a martingale with respect to its natural filtration $\mathcal{F}_t$. And $Z = (1-\theta)e^{\theta V}$ is the Radon-Nikodym density. It is easy to see that

$$Z_t = \frac{e^{-\theta \mathbf{1}_{V \leq t} + \theta t \wedge V}}{\mathbf{P}\left(e^{-\theta \mathbf{1}_{V \leq t} + \theta t \wedge V}\right)}.$$

Because Z_t is a bounded martingale. Its a.s. limit $Z_\infty = \frac{e^{\theta V}}{\mathbf{P}(e^{\theta V})} = Z$. So by uniqueness, $Z_t = \mathbf{P}(Z \,|\, \mathcal{F}_t)$.
And notice that Z_t solves

$$dZ_t = -\theta Z_{t-} dM_t$$

with $Z_0 = 1$ by applying the Itô's formula. Then by Girsanov-Meyer theorem,

$$\tilde{M}_t = M_t + \theta \langle M, M \rangle_t$$

is a martingale under $\mathbf{Q}$.
Third, we want to find the specific form of $\tilde{M}$. It suffices to compute $\langle M, M \rangle_t$. Using Theorem 28 in Protter's book, we find

$$\begin{aligned} [M, M]_t &= \sum_{s \leq t} (\Delta M_s)^2 \\ &= \mathbf{1}_{\{V \leq t\}}; \\ \langle M, M \rangle_t &= t \wedge V. \end{aligned}$$

Finally, we get

$$\tilde{M}_t = \mathbf{1}_{\{V \leq t\}} - t \wedge V + \theta t \wedge V;$$

and from the equation above, the compensator of $\mathbf{1}_{\{V \leq t\}}$ under $\mathbf{Q}$ is $(1-\theta) t \wedge V$. It implies that $V \sim \text{Exp}(1-\theta)$.
Now we get the distribution of V under $\mathbf{Q}_\theta$. Then it is easy to compute the $\mathbf{Q}_\theta$-variance of Y.

$$\begin{aligned} \mathbf{Q}_\theta(Y^2) &= \mathbf{Q}_\theta(e^{-2\theta V + 2\Lambda(\theta)} V^2) \\ &= \begin{cases} \frac{2}{(1-\theta)(1+\theta)^3} & -1 < \theta < 1 \\ \infty & \text{o.w.} \end{cases}. \end{aligned}$$

Simulation. First, for each θ from -0.99 to 0.99, we sample 1000 random numbers from V under $\mathbf{Q}_\theta$. Second, we compute the variance of each estimator of $Y = V/Z$; and plot it. We can notice that the variance is minimized at around 0.5. Moreover, the importance sample has lower variance.

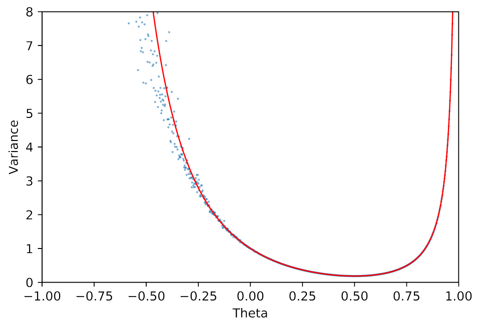

Figure 1: Var. is minimized around 0.5

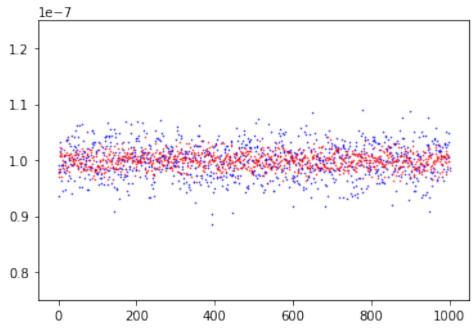

Figure 2: IS reduces the variance

Example A.6. Take $V \sim \text{Exp}(n)$ under $\mathbf{P}$. Now $\Lambda(\theta) = \log(\frac{n}{n-\theta})$ for $\theta < n$ and

$$\begin{aligned}\mathbf{Q}_\theta(V \leq t) &= \mathbf{P}(\mathbf{1}_{\{V\leq t\}} Z) \\ &= \int \mathbf{1}_{\{V \leq t\}} \cdot e^{\theta V - \Lambda(\theta)} d\mathbf{P} \\ &= \int_0^t (n-\theta) e^{-(n-\theta)x} dx\end{aligned}$$

so $V \sim \text{Exp}(n-\theta)$ under $\mathbf{Q}_\theta$. And

$$\mathbf{Q}_\theta(Y^2) = \frac{2n^2}{(n-\theta)(n+\theta)^3}.$$

It is easy to see that it is minimized by $\theta = \frac{n}{2}$.

Example A.7. Now we consider the rare event $\{V_n \geq t\}$ for a fixed t. Under $\mathbf{P}$ we have $\mathbf{P}(V_n \geq t) = e^{-nt} \to 0$ as $n \uparrow \infty$. Let $X_t = \mathbf{1}_{\{V_n \geq t\}}$. Then

$$\mathbf{P}(X^2) = \mathbf{P}(X) = e^{-nt}.$$

Using the same Radon-Nikodym density Z_θ defined in the previous example, the distribution of V_n under $\mathbf{Q}_\theta$ is $\text{Exp}(n-\theta)$. Then we sample X under $\mathbf{Q}_\theta$ and use $Y = X/Z_\theta$ as the $\mathbf{Q}_\theta$-estimator of $\mathbf{P}(X)$. It is easy to see that

$$\mathbf{Q}_\theta(Y^2) = \mathbf{Q}_\theta(\mathbf{1}_{\{V_n \geq t\}} \cdot e^{-2\theta V_n + 2\Lambda(\theta)}) = \frac{2n}{n+\theta} e^{-(n+\theta)t}.$$

Moreover, $\text{Var}_{\mathbf{Q}_\theta}(Y)$ is a decreasing function in θ on $(0, n)$. And we can notice that if $\theta = kn$ for some $k \in (0, 1)$,

$$\frac{\mathbf{Q}_\theta(Y^2)}{\mathbf{P}(X^2)} = \frac{2n}{n+\theta} e^{-\theta t} \to 0$$

as $n \uparrow \infty$. It means when n is very large, the importance sampling method will perform much better than the traditional Monte Carlo estimate.

Let $\{X_n\}$ be iid sequence of real-valued random variables, and $S_n = \sum_{i=1}^n X_i$. Set

$$Z = e^{\theta S_n - n\Lambda(\theta)}$$

where $\Lambda(\theta) = \log \mathbf{P}(e^{\theta X_1})$. Then we have the importance measure $\mathbf{Q}_\theta = Z\mathbf{P}$. Let Y_n be the $\mathbf{Q}$-estimator of y_n ($Y_n \to 0$ as $n \uparrow \infty$).

Definition A.8. Y_n has vanishing relative error (VRE) if

$$\lim_{n\to\infty} \frac{\mathbf{Q}(Y_n^2)}{y_n^2} \to 1.$$

Definition A.9. Y_n has bounded relative error (BRE) if

$$\limsup_{n\uparrow\infty} \frac{\mathbf{Q}(Y_n^2)}{y_n^2} < \infty.$$

Definition A.10. Y_n is logarithmically efficient (LE) if

$$\liminf_{n\uparrow\infty} \frac{\log \mathbf{Q}(Y_n^2)}{\log(y_n^2)} = 1.$$

Example A.11. Let M_n be the number of trails. And for per fixed precision it is of order y_n^2; that is $M_n = oy_n^2$ for some constant o. Assume

$$\delta^2 = \frac{\mathrm{Var}_{\mathbf{Q}}(Y_n)}{M_n}.$$

Take log on both sides:

$$\log(\delta^2 + \frac{1}{o}) = \log \mathbf{Q}(Y_n^2) - \log M_n.$$

Then we divide $\log y_n^2$:

$$\frac{\log(o\delta^2 + 1)}{\log y_n^2} = \frac{\log \mathbf{Q}(Y_n^2)}{\log y_n^2} - 1.$$

Finally, we get

$$\lim_{n\uparrow\infty} \frac{\log \mathbf{Q}(Y_n^2)}{\log y_n^2} = 1;$$

it implies Y_n has logarithmical efficiency.

Lemma A.12. *If there exists $\gamma > 0$ such that the following two conditions hold*

$$\limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}(Y_n^2) \le -2\gamma \tag{UB}$$

$$\liminf_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}(Y_n) \ge -\gamma \tag{LB}$$

Then Y_n is logarithmically efficient.

Proof. It is easy to see that $\frac{\log \mathbf{Q}(Y_n^2)}{\log(y_n^2)} \le 1$ by using Jensen's inequality for the convex function $x \mapsto x^2$. And applying UB and LB conditions, we have

$$\begin{aligned}\liminf_{n\uparrow\infty} \frac{\log \mathbf{Q}(Y_n^2)}{2\log(y_n)} &\le \frac{\limsup_{n\uparrow\infty} \log \mathbf{Q}(Y_n^2)}{2\liminf_{n\uparrow\infty} \log \mathbf{Q}(Y_n)} \\ &\le \frac{-2\gamma}{-2\gamma} = 1.\end{aligned}$$

□

Lemma A.13. *1) Λ and Λ^* are convex functions.*

2) $\Lambda^*(x) = 0$ if $x := \mathbf{P}(X_1)$ is finite.

3) If $\Lambda(\lambda) < \infty$ for any $\lambda > 0$, then for all $z \ge x$,

$$\Lambda^*(z) = \sup_{\lambda\ge 0}\{\lambda z - \Lambda(\lambda)\}$$

is a non-decreasing function in z.

Proof. 1) Recall that $\Lambda(\theta) := \log \mathbf{P}(e^{\theta X})$. For $\alpha \in [0,1]$, by Holder's inequality,

$$\begin{aligned}\Lambda(\alpha\theta_1 + (1-\alpha)\theta_2) &= \log \mathbf{P}(e^{\alpha\theta_1} \cdot e^{(1-\alpha)\theta_2}) \\ &\leq \log \left(\mathbf{P}(e^{\theta_1})^{\alpha} \cdot \mathbf{P}(e^{(1-\alpha)\theta_2})^{(1-\alpha)} \right) \\ &= \alpha\Lambda(\theta_1) + (1-\alpha)\Lambda(\theta_2).\end{aligned}$$

And every Legendre transform of convex function is also convex:

$$\begin{aligned}\Lambda^*(\alpha z_1 + (1-\alpha)z_2) &= \sup_{\lambda \geq 0}\{\lambda\alpha z_1 - \alpha\Lambda(\lambda) + \lambda(1-\alpha)z_2 - (1-\alpha)\Lambda(\lambda)\} \\ &\leq \alpha\Lambda^*(z_1) + (1-\alpha)\Lambda^*(z_2)\end{aligned}$$

2) First, we notice that

$$\Lambda'(\lambda) = \frac{1}{\mathbf{P}(E^{\lambda X})}\mathbf{P}(Xe^{\lambda X});$$

so $x - \Lambda'(0) = 0$. It is equivalent to say $\lambda = 0$ maximize the concave function $\lambda \mapsto \lambda x - \Lambda(\lambda)$. Therefore,

$$\Lambda^*(x) = \sup_{\lambda} \left(\lambda x - \Lambda(\lambda) \right) = 0.$$

3) Let $z \geq x$. Because $\lambda > 0$,

$$\lambda x - \Lambda(\lambda) \leq \lambda z - \Lambda(\lambda).$$

Then we take sup on both sides:

$$\Lambda^*(x) \leq \Lambda^*(z).$$

□

Now we consider the rare event $\{S_n/n > \beta\}$ for $\beta > \mathbf{P}(X_1)$. When $n \to \infty$, by the central limit theorem, its probability will tends to 0. The following theorem says, if we apply importance sampling method for its indicator, we will have logarithmic efficiency.

Theorem A.14. *$Y = \mathbf{1}_{\{S_n/n > \beta\}} e^{-\theta S_n + n\Lambda(\theta)}$ is logarithmically efficient.*

Proof. To prove LE, it suffices to check (UB) and (LB) conditions:

$$\limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}(Y_n^2) \leq -2\gamma$$

$$\liminf_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}(Y_n) \geq -\gamma$$

For the (UB) condition, use Jensen's inequality:

$$\begin{aligned}\limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}_\theta(Y_n^2) &= \limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}_\theta(e^{-2\theta S_n + 2n\Lambda(\theta)} \mathbf{1}_{\{S_n/n > \beta\}}) \\ &\leq -2\theta\beta + 2\Lambda(\theta) + \limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}_\theta(S_n/n > \beta) \\ &\leq -2\theta\beta + 2\Lambda(\theta)\end{aligned}$$

Then we take $\sup_{\theta \geq 0}$ on both sides:

$$\limsup_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}_\theta(Y_n^2) \leq -2\sup_{\theta \geq 0}\{\theta\beta - \Lambda(\theta)\} = -2\Lambda^*(\beta)$$

Recall that by the Cramer's theorem, $\{S_n/n\}$ obeys the large deviation principle in $\mathbb{R}$ with the good rate function Λ^*. That is, for any open set $G \subset \mathbb{R}$

$$\liminf_{n\uparrow\infty} \frac{1}{n} \log \mathbf{P}\left(S_n/n \in G\right) \geq -\inf_{z\in G} \Lambda^*(z).$$

Therefore, for (LB) condition,

$$\begin{aligned}
\liminf_{n\uparrow\infty} \frac{1}{n} \log \mathbf{Q}\left(Y_n\right) &= \liminf_{n\uparrow\infty} \frac{1}{n} \log \mathbf{P}\left(S_n/n > \beta\right) \\
&\geq -\inf_{z>\beta} \Lambda^*\left(z\right) \\
&= -\Lambda^*\left(\beta\right)
\end{aligned}$$

For the last equality, we use the monotonicity of Λ^* proven in the previous lemma. □

B Dynamical Systems

See [Brin and Stuck, 2002] and this slides for more details. For any set X and a family of maps $\{f\} := \{f(t) : X \to X\}_{t\in[0,+\infty)}$, if $\{f\}$ forms a semi-group w.r.t. the operation $\circ$ (i.e. $f(t) \circ f(s) = f(t+s)$ and $f(0) = \mathbf{1}_X$), then $(X, \{f\})$ is called a (continuous-time) *dynamical system*; sometimes, it is called a *semi-flow.*

Now let's consider the connection between the dynamical systems and differential equations.

- **From ODE to the dynamical system.**

  Suppose we have a smooth and compact-supported function $F : \mathbb{R}^n \to \mathbb{R}^n$ and $x : [0, +\infty) \to \mathbb{R}^n$, then
  $$\frac{d}{dt}x(t) = F(x(t))$$
  defines a differential equation. For convenience, we write it as
  $$\dot{x} = F(x).$$
  Solutions for this equation defines an action of $[0, +\infty)$ on $\mathbb{R}^d$ as following: Let $\varphi(x_0, t)$ be the unique solution for this equation with initial point x_0 (Picard–Lindelöf Theorem). For $t = 0$,
  $$\begin{aligned}\varphi(\cdot, 0) : \mathbb{R}^d &\to \mathbb{R}^d\\ x_0 &\mapsto x_0\end{aligned}$$
  is an identity map on $\mathbb{R}^n$; for $t, s \in [0, +\infty)$,
  $$\begin{aligned}\varphi(\varphi(\cdot, s), t) : \mathbb{R}^d &\to \mathbb{R}^d\\ x_0 &\mapsto \varphi(x_0, s + t).\end{aligned}$$
  Then define $f(t) := \varphi(\cdot, t)$; it forms a semi-group action on $\mathbb{R}^n$.

- **From the dynamical system to ODE.**

  Now we have a semi-flow $\{f(t) : \mathbb{R}^d \to \mathbb{R}^d\}_{t\in[0,+\infty)}$. Assume $(x, t) \mapsto \varphi(x, t) := f(t)(x)$ defines a smooth map from $\mathbb{R}^d \times [0, +\infty) \to \mathbb{R}^d$. Then φ is the solution to the following ODE
  $$\dot{x} = \frac{d}{dt}\varphi(x, 0).$$

Due to the deep connection between the dynamical systems and differential equations, we simply call any ODE of form $\dot{x} = F(x)$ as a (time-invariant) system in this note. And an orbit (or a trajectory) of this system is the orbit of semi-group action,
$$\text{Orbit}(x) := \{f(t)(x) \colon t \in [0, +\infty)\};$$
for convenience, we denote is by $x(t)$.

Definition B.1. Let $\dot{x} = F(x)$ be any time-invariant system.

- $x_e \in \mathbb{R}^n$ is an *equilibrium point* of this system if $F(x_e) = 0$.
- The system $\dot{x} = F(x)$ is called *Lyapunov stable* at x_e if for every $\epsilon > 0$ there exists $\delta > 0$ such that if $\|x(0) - x_e\| < \delta$ then
  $$\|x(t) - x_e\| < \epsilon$$
  holds for all $t > 0$.
- The system $\dot{x} = F(x)$ is called *asymptotically stable* at x_e if if it is Lyapunov stable at x_e and there exists $\delta > 0$ such that if $\|x(0) - x_e\| < \delta$, then
  $$x(t) \to x_e$$
  as $t \to \infty$.

- The system $\dot{x} = F(x)$ is called *exponentially stable* at x_e if it is asymptotically stable at x_e and there exist $\alpha > 0$, $\beta > 0$ and $\delta > 0$ such that if $\|x(0) - x_e\| < \delta$ then

$$\|x(t) - x_e\| < \alpha \|x(0) - x_e\| e^{-\beta t}$$

holds for all $t > 0$.

For convenience, we take a shift $x \mapsto x - x_e$ such that 0 is the equilibrium point.

The following theorem is usually called the Lyapunov global asymptotic stability theorem; it could be extended to the time-varying systems.

Theorem B.2. *Given a time-invariant system $\dot{x} = F(x)$. Suppose there is a continuously differentiable function $V : \mathbb{R}^n \to \mathbb{R}$ such that*

- *V is positive definite; that is,*
  - *$V(z) \geq 0$ for all z.*
  - *$V(z) = 0$ if and only if $z = 0$.*
  - *V is coercive; that is all sub-level sets of V are bounded.*
- *$\dot{V}(t) < 0$ for all $t > 0$, where $\dot{V} := \frac{d}{dt}V(x(t))$.*

Then every trajectory of $\dot{x} = F(x)$ converges to 0 as $t \to \infty$.

Proof. Suppose there exists $x(t) \not\to 0$ as $t \to \infty$. Because $\dot{V} < 0$ for $t \neq 0$, $V(x(t))$ is decreasing and V is non-negative (bounded below by 0), it converges to $\epsilon > 0$ as $t \to \infty$. So we have

$$0 < \epsilon \leq V(x(t)) \leq V(x(0))$$

holds for all $t > 0$. By the coercivity of V,

$$C := \{z \in \mathbb{R}^n : \epsilon \leq V(z) \leq V(x(0))\}$$

is compact (in general, closed bounded set may not be compact (see here); fortunately, the domain of V is the Euclidean space $\mathbb{R}^d$). Because we have assumed everything is smooth,

$$\sup_{z \in C} \dot{V} = \sup_{z \in C} \langle \nabla V(z), F(z) \rangle = -a < 0.$$

Then since the trajectory $x(t)$ is contained in C, we have

$$\dot{V}(x(t)) \leq -a.$$

Then because

$$V(x(T)) - V(x(0)) = \int_0^T \dot{V}(x(t))dt \leq -aT,$$

we have for $T > \frac{V(x(0))}{a}$, $V(x(T)) < 0$. Contradiction. □

The following result doesn't require $\dot{V}$ strictly less than 0, but it cannot be generalized to time-varying systems. Its proof is beyond this note so omitted.

Theorem B.3 (Lasalle)**.** *Given a time-invariant system $\dot{x} = F(x)$. Suppose there is a function $V : \mathbb{R}^n \to \mathbb{R}$ such that*

- *V is positive definite.*
- *$\dot{V}(t) \leq 0$.*
- *The solution to $\dot{x} = F(x)$ with $\dot{V} = 0$ is $x(t) = 0$ for all t.*

Then every trajectory of $\dot{x} = F(x)$ converges to 0 *as* $t \to \infty$.

We have the following strategies to analyze the properties of trajectories:

a) Select a positive definite function $V : \mathbb{R}^d \to \mathbb{R}$ as the **Lyapunov function candidate**.

b) Evaluate $\dot{V}$ for every trajectory of the given system.

c) Check the stability theorem conditions; pass means V is the desired **Lyapunov function**.

d) If fails, go back to the step (a).

It could be tricky to choose an appropriate V in some cases.

References

[Agarwal et al., 2019] Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. (2019). Optimality and approximation with policy gradient methods in markov decision processes. arXiv preprint arXiv:1908.00261.

[Brin and Stuck, 2002] Brin, M. and Stuck, G. (2002). Introduction to dynamical systems. Cambridge university press.

[Li, 2018] Li, Y. (2018). Deep reinforcement learning. arXiv preprint arXiv:1810.06339.

[Melo et al., 2008] Melo, F. S., Meyn, S. P., and Ribeiro, M. I. (2008). An analysis of reinforcement learning with function approximation. In Proceedings of the 25th international conference on Machine learning, pages 664–671. ACM.

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

[Sutton et al., 2000] Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In Advances in neural information processing systems, pages 1057–1063.

[Watkins, 1989] Watkins, C. J. C. H. (1989). Learning from delayed rewards.

[Weng, 2018] Weng, L. (2018). Policy gradient algorithms. lilianweng.github.io/lil-log.