Reinforcement Learning: Theory and Algorithms

Alekh Agarwal Nan Jiang Sham M. Kakade Wen Sun

October 27, 2020

WORKING DRAFT:

We will be frequently updating the book this fall, 2020. Please email bookrltheory@gmail.com with any typos or errors you find. We appreciate it!

Contents

Notation

The reader might find it helpful to refer back to this notation section.

- We slightly abuse notation and let $[K]$ denote the set $\{0, 1, 2, \dots K-1\}$ for an integer K.
- For a vector v, we let $(v)^2$, $\sqrt{v}$, and $|v|$ be the component-wise square, square root, and absolute value operations.
- Inequalities between vectors are elementwise, e.g. for vectors v, v', we way $v \leq v'$, if the inequality holds elementwise.
- For a vector v, we refer to the j-th component of this vector by either $v(j)$ or $[v]_j$
- Denote the variance of any real valued f under a distribution $\mathcal{D}$ as:

$$\text{Var}_{\mathcal{D}}(f) := E_{x\sim\mathcal{D}}[f(x)^2] - (E_{x\sim\mathcal{D}}[f(x)])^2$$

- We overload notation where, for a distribution μ over $\mathcal{S}$, we write:

$$V^\pi(\mu) = \mathbb{E}_{s\sim\mu}\left[V^\pi(s)\right].$$

- It is helpful to overload notation and let P also refer to a matrix of size $(\mathcal{S}\cdot\mathcal{A}) \times \mathcal{S}$ where the entry $P_{(s,a),s'}$ is equal to $P(s'|s,a)$. We also will define P^π to be the transition matrix on state-action pairs induced by a deterministic policy π. In particular, $P^\pi_{(s,a),(s',a')} = P(s'|s,a)$ if $a' = \pi(s')$ and $P^\pi_{(s,a),(s',a')} = 0$ if $a' \neq \pi(s')$. With this notation,

$$\begin{aligned} Q^\pi &= r + PV^\pi \\ Q^\pi &= r + P^\pi Q^\pi \\ Q^\pi &= (I - \gamma P^\pi)^{-1} r \end{aligned}$$

- For a vector $Q \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, denote the greedy policy and value as:

$$\begin{aligned} \pi_Q(s) &:= \text{argmax}_{a\in\mathcal{A}}\, Q(s,a) \\ V_Q(s) &:= \max_{a\in\mathcal{A}} Q(s,a).. \end{aligned}$$

- For a vector $Q \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$, the *Bellman optimality operator* $\mathcal{T} : \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$ is defined as:

$$\mathcal{T}Q := r + PV_Q\,. \tag{0.1}$$

Part 1

Fundamentals

Chapter 1

Markov Decision Processes and Computational Complexity

1.1 (Discounted) Markov Decision Processes

In reinforcement learning, the interactions between the agent and the environment are often described by a discounted Markov Decision Process (MDP) $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$, specified by:

- A state space $\mathcal{S}$, which may be finite or infinite. For mathematical convenience, we will assume that $\mathcal{S}$ is finite or countable infinite.
- An action space $\mathcal{A}$, which also may be discrete or infinite. For mathematical convenience, we will assume that $\mathcal{A}$ is finite.
- A transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is the space of probability distributions over $\mathcal{S}$ (i.e., the probability simplex). $P(s'|s, a)$ is the probability of transitioning into state s' upon taking action a in state s. We use $P_{s,a}$ to denote the vector $P(\cdot \mid s, a)$.
- A reward function $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$. $r(s, a)$ is the immediate reward associated with taking action a in state s.
- A discount factor $\gamma \in [0, 1)$, which defines a horizon for the problem.
- An initial state distribution $\mu \in \Delta(\mathcal{S})$, which specifies how the initial state s_0 is generated.

In many cases, we will assume that the initial state is fixed at s_0, i.e. μ is a distribution supported only on s_0.

1.1.1 The objective, policies, and values

Policies. In a given MDP $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$, the agent interacts with the environment according to the following protocol: the agent starts at some state $s_0 \sim \mu$; at each time step $t = 0, 1, 2, \ldots$, the agent takes an action $a_t \in \mathcal{A}$, obtains the immediate reward $r_t = r(s_t, a_t)$, and observes the next state s_{t+1} sampled according to $s_{t+1} \sim P(\cdot|s_t, a_t)$. The interaction record at time t,

$$\tau_t = (s_0, a_0, r_1, s_1, \ldots, s_t),$$

is called a *trajectory*, which includes the observed state at time t.

In the most general setting, a policy specifies a decision-making strategy in which the agent chooses actions adaptively based on the history of observations; precisely, a policy is a (possibly randomized) mapping from a trajectory to an action, i.e. $\pi : \mathcal{H} \to \Delta(\mathcal{A})$ where $\mathcal{H}$ is the set of all possible trajectories (of all lengths) and $\Delta(\mathcal{A})$ is the space of probability distributions over $\mathcal{A}$. A *stationary* policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ specifies a decision-making strategy in which the agent chooses actions based only on the current state, i.e. $a_t \sim \pi(\cdot|s_t)$. A deterministic, stationary policy is of the form $\pi : \mathcal{S} \to \mathcal{A}$.

Values. We now define values for (general) policies. For a fixed policy and a starting state $s_0 = s$, we define the value function $V_M^\pi : \mathcal{S} \to \mathbb{R}$ as the discounted sum of future rewards

$$V_M^\pi(s) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \;\Big|\; \pi, s_0 = s\Big].$$

where expectation is with respect to the randomness of the trajectory, that is, the randomness in state transitions and the stochasticity of π. Here, since $r(s, a)$ is bounded between 0 and 1, we have $0 \le V_M^\pi(s) \le 1/(1-\gamma)$.

Similarly, the action-value (or Q-value) function $Q_M^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as

$$Q_M^\pi(s, a) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \;\Big|\; \pi, s_0 = s, a_0 = a\Big].$$

and $Q_M^\pi(s, a)$ is also bounded by $1/(1-\gamma)$.

Goal. Given a state s, the goal of the agent is to find a policy π that maximizes the value, i.e. the optimization problem the agent seeks to solve is:

$$\max_\pi V_M^\pi(s) \tag{0.1}$$

where the max is over all (possibly non-stationary and randomized) policies. As we shall see, there exists a deterministic and stationary policy which is simultaneously optimal for all starting states s.

We drop the dependence on M and write V^π when it is clear from context.

Example 1.1 (Navigation)**.** Navigation is perhaps the simplest to see example of RL. The state of the agent is their current location. The four actions might be moving 1 step along each of east, west, north or south. The transitions in the simplest setting are deterministic. Taking the north action moves the agent one step north of their location, assuming that the size of a step is standardized. The agent might have a goal state g they are trying to reach, and the reward is 0 until the agent reaches the goal, and 1 upon reaching the goal state. Since the discount factor $\gamma < 1$, there is incentive to reach the goal state earlier in the trajectory. As a result, the optimal behavior in this setting corresponds to finding the shortest path from the initial to the goal state, and the value function of a state, given a policy is γ^d, where d is the number of steps required by the policy to reach the goal state.

Example 1.2 (Conversational agent)**.** This is another fairly natural RL problem. The state of an agent can be the current transcript of the conversation so far, along with any additional information about the world, such as the context for the conversation, characteristics of the other agents or humans in the conversation etc. Actions depend on the domain. In the most basic form, we can think of it as the next statement to make in the conversation. Sometimes, conversational agents are designed for task completion, such as travel assistant or tech support or a virtual office receptionist. In these cases, there might be a predefined set of *slots* which the agent needs to fill before they can find a good solution. For instance, in the travel agent case, these might correspond to the dates, source, destination and mode of travel. The actions might correspond to natural language queries to fill these slots.

In task completion settings, reward is naturally defined as a binary outcome on whether the task was completed or not, such as whether the travel was successfully booked or not. Depending on the domain, we could further refine it based

on the quality or the price of the travel package found. In more generic conversational settings, the ultimate reward is whether the conversation was satisfactory to the other agents or humans, or not.

Example 1.3 (Strategic games)**.** This is a popular category of RL applications, where RL has been successful in achieving human level performance in Backgammon, Go, Chess, and various forms of Poker. The usual setting consists of the state being the current game board, actions being the potential next moves and reward being the eventual win/loss outcome or a more detailed score when it is defined in the game. Technically, these are multi-agent RL settings, and, yet, the algorithms used are often non-multi-agent RL algorithms.

1.1.2 Bellman consistency equations for stationary policies

Stationary policies satisfy the following consistency conditions:

Lemma 1.4. *Suppose that π is a stationary policy. Then V^π and Q^π satisfy the following* Bellman consistency equations*: for all $s \in \mathcal{S}, a \in \mathcal{A}$,*

$$V^\pi(s) = Q^\pi(s, \pi(s)).$$
$$Q^\pi(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}\left[V^\pi(s')\right].$$

We leave the proof as an exercise to the reader.

It is helpful to view V^π as vector of length $|\mathcal{S}|$ and Q^π and r as vectors of length $|\mathcal{S}| \cdot |\mathcal{A}|$. We overload notation and let P also refer to a matrix of size $(|\mathcal{S}| \cdot |\mathcal{A}|) \times |\mathcal{S}|$ where the entry $P_{(s,a),s'}$ is equal to $P(s'|s,a)$.

We also will define P^π to be the transition matrix on state-action pairs induced by a stationary policy π, specifically:

$$P^\pi_{(s,a),(s',a')} := P(s'|s,a)\pi(a'|s').$$

In particular, for deterministic policies we have:

$$P^\pi_{(s,a),(s',a')} := \begin{cases} P(s'|s,a) & \text{if } a' = \pi(s') \\ 0 & \text{if } a' \neq \pi(s') \end{cases}$$

With this notation, it is straightforward to verify:

$$Q^\pi = r + \gamma P V^\pi$$
$$Q^\pi = r + \gamma P^\pi Q^\pi .$$

Corollary 1.5. *We have that:*

$$Q^\pi = (I - \gamma P^\pi)^{-1} r \tag{0.2}$$

where I is the identity matrix.

Proof: To see that the $I - \gamma P^\pi$ is invertible, observe that for any non-zero vector $x \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$,

$$\begin{aligned} \|(I - \gamma P^\pi)x\|_\infty &= \|x - \gamma P^\pi x\|_\infty \\ &\geq \|x\|_\infty - \gamma \|P^\pi x\|_\infty && \text{(triangule inequality for norms)} \\ &\geq \|x\|_\infty - \gamma \|x\|_\infty && \text{(each element of } P^\pi x \text{ is an average of } x) \\ &= (1-\gamma)\|x\|_\infty > 0 && (\gamma < 1, x \neq 0) \end{aligned}$$

which implies $I - \gamma P^\pi$ is full rank. ■

The following is also a helpful lemma:

Lemma 1.6. *We have that:*

$$[(1-\gamma)(I-\gamma P^{\pi})^{-1}]_{(s,a),(s',a')} = (1-\gamma)\sum_{h=0}^{\infty}\gamma^t \mathbb{P}_h^{\pi}(s_h = s', a_h = a' | s_0 = s, a_0 = a)$$

so we can view the (s,a)-th row of this matrix as an induced distribution over states and actions when following π after starting with $s_0 = s$ and $a_0 = a$.

We leave the proof as an exercise to the reader.

1.1.3 Bellman optimality equations

A remarkable and convenient property of MDPs is that there exists a stationary and deterministic policy that simultaneously maximizes $V^{\pi}(s)$ for all $s \in \mathcal{S}$. This is formalized in the following theorem:

Theorem 1.7. *Let Π be the set of all non-stationary and randomized policies. Define:*

$$V^{\star}(s) := \sup_{\pi \in \Pi} V^{\pi}(s)$$
$$Q^{\star}(s,a) := \sup_{\pi \in \Pi} Q^{\pi}(s,a).$$

which is finite since $V^{\pi}(s)$ and $Q^{\pi}(s,a)$ are bounded between 0 and $1/(1-\gamma)$.

There exists a stationary and deterministic policy π such that for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$,

$$V^{\pi}(s) = V^{\star}(s)$$
$$Q^{\pi}(s,a) = Q^{\star}(s,a).$$

We refer to such a π as an optimal policy.

Proof: First, let us show that conditioned on $(s_0, a_0, r_0, s_1) = (s, a, r, s')$, the maximum future discounted value, from time 1 onwards, is not a function of s, a, r. Specifically,

$$\sup_{\pi \in \Pi} \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^t r(s_t, a_t) \;\Big|\; \pi, (s_0, a_0, r_0, s_1) = (s, a, r, s')\Big] = \gamma V^{\star}(s')$$

For any policy π, define an "offset" policy $\pi_{(s,a,r)}$, which is the policy that chooses actions on a trajectory τ according to the same distribution that π chooses actions on the trajectory (s, a, r, τ). For example, $\pi_{(s,a,r)}(a_0 = a' | s_0 = s')$ is equal to the probability $\pi(a_1 = a' | (s_0, a_0, r_0, s_1) = (s, a, r, s'))$. By the Markov property, we have that:

$$\mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^t r(s_t, a_t) \;\Big|\; \pi, (s_0, a_0, r_0, s_1) = (s, a, r, s')\Big] = \gamma \mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t r(s_t, a_t) \;\Big|\; \pi_{(s,a,r)}, s_0 = s'\Big] = \gamma V^{\pi_{(s,a,r)}}(s').$$

Hence, due to that $V^{\pi}(s')$ is not a function of (s, a, r), we have

$$\sup_{\pi \in \Pi} \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^t r(s_t, a_t) \;\Big|\; \pi, (s_0, a_0, r_0, s_1) = (s, a, r, s')\Big] = \gamma \cdot \sup_{\pi \in \Pi} V^{\pi_{(s,a,r)}}(s') = \gamma \cdot \sup_{\pi \in \Pi} V^{\pi}(s') = \gamma V^{\star}(s'),$$

thus proving the claim.

We now show the deterministic and stationary policy $\pi(s) = \text{argmax}_{a\in\mathcal{A}} \sup_{\pi'\in\Pi} Q^{\pi'}(s,a)$ satisfies $V^{\pi}(s) = \sup_{\pi'\in\Pi} V^{\pi'}(s)$. For this, we have that:

$$\begin{aligned}
V^{\star}(s_0) &= \sup_{\pi\in\Pi} \mathbb{E}\Big[r(s_0,a_0) + \sum_{t=1}^{\infty}\gamma^t r(s_t,a_t)\Big] \\
&= \sup_{\pi\in\Pi} \mathbb{E}\Big[r(s_0,a_0) + \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^t r(s_t,a_t) \;\Big|\; \pi, (s_0,a_0,r_0,s_1)\Big]\Big] \\
&\leq \sup_{\pi\in\Pi} \mathbb{E}\Big[r(s_0,a_0) + \sup_{\pi'\in\Pi} \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^t r(s_t,a_t) \;\Big|\; \pi', (s_0,a_0,r_0,s_1)\Big]\Big] \\
&= \sup_{\pi\in\Pi} \mathbb{E}\Big[r(s_0,a_0) + \gamma V^{\star}(s_1)\Big] \\
&= \mathbb{E}\Big[r(s_0,a_0) + \gamma V^{\star}(s_1) \;\Big|\; \pi\Big].
\end{aligned}$$

where the second equality is by the tower property of conditional expectations, and the last equality follows from the definition of π. Now, by recursion,

$$V^{\star}(s_0) \leq \mathbb{E}\Big[r(s_0,a_0) + \gamma V^{\star}(s_1) \;\Big|\; \pi\Big] \leq \mathbb{E}\Big[r(s_0,a_0) + \gamma r(s_1,a_1) + \gamma^2 V^{\star}(s_2) \;\Big|\; \pi\Big] \leq \ldots \leq V^{\pi}(s_0).$$

Since $V^{\pi}(s) \leq \sup_{\pi'\in\Pi} V^{\pi'}(s) = V^{\star}(s)$, we have that $V^{\pi} = V^{\star}$, which completes the proof of the first claim.

For the same policy π, an analogous argument can be used prove the second claim. ■

This shows that we may restrict ourselves to using stationary and deterministic policies without any loss in performance. The following theorem, also due to [Bellman, 1956], gives a precise characterization of the optimal value function.

Let us say that a vector $Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ satisfies the *Bellman optimality equations* if:

$$Q(s,a) = r(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[\max_{a'\in\mathcal{A}} Q(s',a')\right].$$

Theorem 1.8 (Bellman Optimality Equations)**.** *For any $Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$, we have that $Q = Q^{\star}$ if and only if Q satisfies the Belman optimality equations. Furthermore, the deterministic policy $\pi(s) \in Q^{\star}(s,a)$ is an optimal policy (where ties are broken in some arbitrary and deterministic manner).*

Before we prove this claim, we will provide a few definitions. Let π_Q denote the greedy policy with respect to a vector $Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$, i.e

$$\pi_Q(s) := \text{argmax}_{a\in\mathcal{A}}\, Q(s,a)\,.$$

where ties are broken in some arbitrary (and deterministic) manner. With this notation, by the above theorem, the optimal policy $\pi^{\star}$ is given by:

$$\pi^{\star} = \pi_{Q^{\star}}\,.$$

Let us also use the following notation to turn a vector $Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ into a vector of length $|\mathcal{S}|$.

$$V_Q(s) := \max_{a\in\mathcal{A}} Q(s,a).$$

The *Bellman optimality operator* $\mathcal{T}_M : \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ is defined as:

$$\mathcal{T}Q := r + \gamma P V_Q\,. \tag{0.3}$$

This allows us to rewrite the Bellman optimality equation in the concise form:

$$Q = \mathcal{T}Q,$$

and, so, the previous theorem states that $Q = Q^\star$ if and only if Q is a fixed point of the operator $\mathcal{T}$.

Proof: We first show sufficiency, i.e. that $Q^\star$ (the state-action value of an optimal policy) satisfies $Q^\star = \mathcal{T}Q^\star$. Let $\pi^\star$ be an optimal stationary and deterministic policy, which exists by Theorem 1.7. First let us show that $V^\star(s) = \max_a Q^\star(s,a)$. We have that $V^\star(s) = V^{\pi^\star}(s) = Q^{\pi^\star}(s, \pi^\star(a)) = Q^\star(s, \pi^\star(a))$, by Lemma 1.4 and Theorem 1.7. Also,

$$\max_a Q^\star(s,a) \geq Q^\star(s, \pi^\star(a)) = V^\star(s) \geq \max_a \max_\pi Q^\pi(s,a) \geq \max_a Q^{\pi^\star}(s,a) = \max_a Q^\star(s,a),$$

which proves the claim. Now for all actions $a \in \mathcal{A}$, we have:

$$\begin{aligned} Q^\star(s,a) &= \max_\pi Q^\pi(s,a) = r(s,a) + \gamma \max_\pi \mathbb{E}_{s' \sim P(\cdot|s,a)}[V^\pi(s')] \\ &\overset{(a)}{=} r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V^\star(s')] \\ &= r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[\max_{a'} Q^\star(s',a')]. \end{aligned}$$

Here the equality (a) follows from Theorem 1.7. This proves sufficiency.

For the converse, suppose $Q = \mathcal{T}Q$ for some Q. We now show that $Q = Q^\star$. Let $\pi = \pi_Q$. That $Q = \mathcal{T}Q$ implies that $Q = r + \gamma P^{\pi_Q} Q$, and so:

$$Q = (I - \gamma P^{\pi_Q})^{-1} r = Q^\pi$$

using Equation 0.2 in the last step. In other words, Q is the action value of the policy π_Q. Now observe for any other deterministic and stationary policy π':

$$\begin{aligned} Q - Q^{\pi'} &= Q^\pi - Q^{\pi'} \\ &= Q^\pi - (I - \gamma P^{\pi'})^{-1} r \\ &= (I - \gamma P^{\pi'})^{-1}((I - \gamma P^{\pi'}) - (I - \gamma P^\pi))Q^\pi \\ &= \gamma (I - \gamma P^{\pi'})^{-1}(P^\pi - P^{\pi'})Q^\pi \,. \end{aligned}$$

The proof is completed by noting that $(P^\pi - P^{\pi'})Q^\pi \geq 0$. To see this, recall that $(1-\gamma)(I - \gamma P^{\pi'})^{-1}$ is a matrix with all positive entries (see Lemma 1.6), and now we can observe that:

$$[(P^\pi - P^{\pi'})Q^\pi]_{s,a} = \mathbb{E}_{s' \sim P(\cdot|s,a)}[Q^\pi(s', \pi(s')) - Q^\pi(s', \pi'(s'))] \geq 0$$

where the last step uses that $\pi = \pi_Q$. Thus we have that $Q \geq Q^{\pi'}$ for all deterministic and stationary π' which shows $Q = Q^\star$, using Theorem 1.7. This completes the proof. ■

1.2 (Episodic) Markov Decision Processes

It will also be natural for us to work with episodic Markov decision Processes. In reinforcement learning, the interactions between the agent and the environment are often described by an episodic time-dependent Markov Decision Process (MDP) $M = (\mathcal{S}, \mathcal{A}, \{P\}_h, \{r\}_h, H, \mu)$, specified by:

- A state space $\mathcal{S}$, which may be finite or infinite.
- An action space $\mathcal{A}$, which also may be discrete or infinite.

- A time-dependent transition function $P_h : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is the space of probability distributions over $\mathcal{S}$ (i.e., the probability simplex). $P_h(s'|s,a)$ is the probability of transitioning into state s' upon taking action a in state s at time step h. Note that time-dependent setting generalizes the stationary setting where all steps share a same transition .
- A reward function $r_h : \mathcal{S} \times \mathcal{A} \to [0,1]$. $r_h(s,a)$ is the immediate reward associated with taking action a in state s at time step h.
- A integer H which defines the horizon of the problem.
- An initial state distribution $\mu \in \Delta(\mathcal{S})$, which species how the initial state s_0 is generated.

Here, for a policy π, a state s, and $h \in \{0, \ldots H-1\}$, we define the value function $V_h^\pi : \mathcal{S} \to \mathbb{R}$ as

$$V_h^\pi(s) = \mathbb{E}\Big[\sum_{t=h}^{H-1} r_h(s_t, a_t) \;\Big|\; \pi, s_h = s \Big] .$$

where again expectation is with respect to the randomness of the trajectory, that is, the randomness in state transitions and the stochasticity of π. Similarly, the action-value (or Q-value) function $Q_h^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as

$$Q_h^\pi(s,a) = \mathbb{E}\Big[\sum_{t=h}^{H-1} r_h(s_t, a_t) \;\Big|\; \pi, s_h = s, a_h = a \Big].$$

We also use the notation $V^\pi(s) = V_0^\pi(s)$.

Again, given a state s, the goal of the agent is to find a policy π that maximizes the value, i.e. the optimization problem the agent seeks to solve is:

$$\max_\pi V^\pi(s) \tag{0.4}$$

where recall that $V^\pi(s) = V_0^\pi(s)$.

Theorem 1.9. *(Bellman optimality equations) Define*

$$Q_h^\star(s,a) = \sup_{\pi \in \Pi} Q_h^\pi(s,a)$$

where the sup *is over all non-stationary and randomized policies. We have that $Q_h = Q_h^\star$ for all $h \in [H]$ if and only if for all $h \in [H]$,*

$$Q_h(s,a) = r_h(s,a) + \gamma \mathbb{E}_{s' \sim P_h(\cdot|s,a)} \left[\max_{a' \in \mathcal{A}} Q_{h+1}(s', a') \right] .$$

Furthermore, $\pi(s,h) = \mathrm{argmax}_{a \in \mathcal{A}} Q_h^\star(s,a)$ is an optimal policy.

We leave the proof as an exercise to the reader.

1.3 Computational Complexity

The remainder of this section will be concerned with computing an optimal policy when given knowledge of the MDP $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$. While much of this book is concerned with statistical limits, understanding the computational limits can be informative. We will consider algorithms which give both exact and approximately optimal policies. In particular, we will be interested in polynomial time (and strongly polynomial time) algorithms.

	Value Iteration	Policy Iteration	LP-Algorithms
Poly?	$\lvert\mathcal{S}\rvert^2\lvert\mathcal{A}\rvert \frac{L(P,r,\gamma)\log\frac{1}{1-\gamma}}{1-\gamma}$	$(\lvert\mathcal{S}\rvert^3 + \lvert\mathcal{S}\rvert^2\lvert\mathcal{A}\rvert)\frac{L(P,r,\gamma)\log\frac{1}{1-\gamma}}{1-\gamma}$	$\lvert\mathcal{S}\rvert^3\lvert\mathcal{A}\rvert L(P,r,\gamma)$
Strongly Poly?	✗	$(\lvert\mathcal{S}\rvert^3 + \lvert\mathcal{S}\rvert^2\lvert\mathcal{A}\rvert)\cdot\min\left\{\frac{\lvert\mathcal{A}\rvert^{\lvert\mathcal{S}\rvert}}{\lvert\mathcal{S}\rvert}, \frac{\lvert\mathcal{S}\rvert^2\lvert\mathcal{A}\rvert\log\frac{\lvert\mathcal{S}\rvert^2}{1-\gamma}}{1-\gamma}\right\}$	$\lvert\mathcal{S}\rvert^4\lvert\mathcal{A}\rvert^4\log\frac{\lvert\mathcal{S}\rvert}{1-\gamma}$

Table 0.1: Computational complexities of various approaches (we drop universal constants). Polynomial time algorithms depend on the bit complexity, $L(P, r, \gamma)$, while strongly polynomial algorithms do not. Note that only for a fixed value of γ are value and policy iteration polynomial time algorithms; otherwise, they are not polynomial time algorithms. Similarly, only for a fixed value of γ is policy iteration a strongly polynomial time algorithm. In contrast, the LP-approach leads to both polynomial time and strongly polynomial time algorithms; for the latter, the approach is an interior point algorithm. See text for further discussion, and Section 1.7 for references. Here, $|\mathcal{S}|^2|\mathcal{A}|$ is the assumed runtime per iteration of value iteration, and $|\mathcal{S}|^3 + |\mathcal{S}|^2|\mathcal{A}|$ is the assumed runtime per iteration of policy iteration (note that for this complexity we would directly update the values V rather than Q values, as described in the text); these runtimes are consistent with assuming cubic complexity for linear system solving.

Suppose that (P, r, γ) in our MDP M is specified with rational entries. Let $L(P, r, \gamma)$ denote the total bit-size required to specify M, and assume that basic arithmetic operations $+, -, \times, \div$ take unit time. Here, we may hope for an algorithm which (exactly) returns an optimal policy whose runtime is polynomial in $L(P, r, \gamma)$ and the number of states and actions.

More generally, it may also be helpful to understand which algorithms are *strongly* polynomial. Here, we do not want to explicitly restrict (P, r, γ) to be specified by rationals. An algorithm is said to be strongly polynomial if it returns an optimal policy with runtime that is polynomial in only the number of states and actions (with no dependence on $L(P, r, \gamma)$).

1.4 Iterative Methods

Planning refers to the problem of computing $\pi^\star_M$ given the MDP specification $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$. This section reviews classical planning algorithms that compute $Q^\star$.

1.4.1 Value Iteration

A simple algorithm is to iteratively apply the fixed point mapping: starting at some Q, we iteratively apply $\mathcal{T}$:

$$Q \leftarrow \mathcal{T}Q\,,$$

This is algorithm is referred to as Q*-value iteration*.

Lemma 1.10. *(contraction) For any two vectors $Q, Q' \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$,*

$$\|\mathcal{T}Q - \mathcal{T}Q'\|_\infty \leq \gamma\|Q - Q'\|_\infty$$

Proof: First, let us show that for all $s \in \mathcal{S}$, $|V_Q(s) - V_{Q'}(s)| \leq \max_{a\in\mathcal{A}} |Q(s,a) - Q'(s,a)|$. Assume $V_Q(s) > V_{Q'}(s)$ (the other direction is symmetric), and let a be the greedy action for Q at s. Then

$$|V_Q(s) - V_{Q'}(s)| = Q(s,a) - \max_{a'\in\mathcal{A}} Q'(s,a') \leq Q(s,a) - Q'(s,a) \leq \max_{a\in\mathcal{A}} |Q(s,a) - Q'(s,a)|.$$

Using this,

$$\begin{aligned}\|\mathcal{T}Q - \mathcal{T}Q'\|_\infty &= \gamma\|PV_Q - PV_{Q'}\|_\infty \\ &= \gamma\|P(V_Q - V_{Q'})\|_\infty \\ &\leq \gamma\|V_Q - V_{Q'}\|_\infty \\ &= \gamma \max_s |V_Q(s) - V_{Q'}(s)| \\ &\leq \gamma \max_s \max_a |Q(s,a) - Q'(s,a)| \\ &= \gamma\|Q - Q'\|_\infty\end{aligned}$$

where the first inequality uses that each element of $P(V_Q - V_{Q'})$ is a convex average of $V_Q - V_{Q'}$ and the second inequality uses our claim above. ■

The following result bounds the sub-optimality of the greedy policy itself, based on the error in Q-value function.

Lemma 1.11. *(Q-Error Amplification) For any vector $Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$,*

$$V^{\pi_Q} \geq V^\star - \frac{2\|Q - Q^\star\|_\infty}{1-\gamma}\mathbb{1}.$$

where $\mathbb{1}$ denotes the vector of all ones.

Proof: Fix state s and let $a = \pi_Q(s)$. We have:

$$\begin{aligned}V^\star(s) - V^{\pi_Q}(s) =& Q^\star(s,\pi^\star(s)) - Q^{\pi_Q}(s,a) \\ =& Q^\star(s,\pi^\star(s)) - Q^\star(s,a) + Q^\star(s,a) - Q^{\pi_Q}(s,a) \\ =& Q^\star(s,\pi^\star(s)) - Q^\star(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[V^\star(s') - V^{\pi_Q}(s')] \\ \leq& Q^\star(s,\pi^\star(s)) - Q(s,\pi^\star(s)) + Q(s,a) - Q^\star(s,a) \\ &+ \gamma\mathbb{E}_{s'\sim P(s,a)}[V^\star(s') - V^{\pi_Q}(s')] \\ \leq& 2\|Q - Q^\star\|_\infty + \gamma\|V^\star - V^{\pi_Q}\|_\infty.\end{aligned}$$

where the first inequality uses $Q(s,\pi^\star(s)) \leq Q(s,\pi_Q(s)) = Q(s,a)$ due to the definition of π_Q. ■

Theorem 1.12. *(Q-value iteration convergence). Set $Q^{(0)} = 0$. For $k = 0, 1, \ldots$, suppose:*

$$Q^{(k+1)} = \mathcal{T}Q^{(k)}$$

Let $\pi^{(k)} = \pi_{Q^{(k)}}$. For $k \geq \frac{\log\frac{2}{(1-\gamma)^2\epsilon}}{1-\gamma}$,

$$V^{\pi^{(k)}} \geq V^\star - \epsilon\mathbb{1}\,.$$

Proof: Since $\|Q^\star\|_\infty \leq 1/(1-\gamma)$, $Q^{(k)} = \mathcal{T}^k Q^{(0)}$ and $Q^\star = \mathcal{T}Q^\star$, Lemma 1.10 gives

$$\|Q^{(k)} - Q^\star\|_\infty = \|\mathcal{T}^k Q^{(0)} - \mathcal{T}^k Q^\star\|_\infty \leq \gamma^k\|Q^{(0)} - Q^\star\|_\infty = (1-(1-\gamma))^k\|Q^\star\|_\infty \leq \frac{\exp(-(1-\gamma)k)}{1-\gamma}\,.$$

The proof is completed with our choice of γ and using Lemma 1.11. ■

Iteration complexity for an exact solution. With regards to computing an exact optimal policy, when the gap between the current objective value and the optimal objective value is smaller than $2^{-L(P,r,\gamma)}$, then the greedy policy will be optimal. This leads to claimed complexity in Table 0.1. Value iteration is not strongly polynomial algorithm due to that, in finite time, it may never return the optimal policy.

1.4.2 Policy Iteration

The policy iteration algorithm starts from an arbitrary policy π_0, and repeat the following iterative procedure: for $k = 0, 1, 2, \ldots$

1. *Policy evaluation.* Compute Q^{π_k}
2. *Policy improvement.* Update the policy:

$$\pi_{k+1} = \pi_{Q^{\pi_k}}$$

In each iteration, we compute the Q-value function of π_k, using the analytical form given in Equation 0.2, and update the policy to be greedy with respect to this new Q-value. The first step is often called *policy evaluation*, and the second step is often called *policy improvement*.

Lemma 1.13. *We have that:*

1. $Q^{\pi_{k+1}} \geq \mathcal{T} Q^{\pi_k} \geq Q^{\pi_k}$
2. $\|Q^{\pi_{k+1}} - Q^\star\|_\infty \leq \gamma \|Q^{\pi_k} - Q^\star\|_\infty$

Proof: First let us show that $\mathcal{T} Q^{\pi_k} \geq Q^{\pi_k}$. Note that the policies produced in policy iteration are always deterministic, so $V^{\pi_k}(s) = Q^{\pi_k}(s, \pi_k(s))$ for all iterations k and states s. Hence,

$$\begin{aligned}\mathcal{T} Q^{\pi_k}(s,a) &= r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[\max_{a'} Q^{\pi_k}(s', a')] \\ &\geq r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[Q^{\pi_k}(s', \pi_k(s'))] = Q^{\pi_k}(s,a).\end{aligned}$$

Now let us prove that $Q^{\pi_{k+1}} \geq \mathcal{T} Q^{\pi_k}$. First, let use see that $Q^{\pi_{k+1}} \geq Q^{\pi_k}$:

$$Q^{\pi_k} = r + \gamma P^{\pi_k} Q^{\pi_k} \leq r + \gamma P^{\pi_{k+1}} Q^{\pi_k} \leq \sum_{t=0}^{\infty} \gamma^t (P^{\pi_{k+1}})^t r = Q^{\pi_{k+1}}.$$

where we have used that π_{k+1} is the greedy policy in the first inequality and recursion in the second inequality. Using this,

$$\begin{aligned}Q^{\pi_{k+1}}(s,a) &= r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[Q^{\pi_{k+1}}(s', \pi_{k+1}(s'))] \\ &\geq r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[Q^{\pi_k}(s', \pi_{k+1}(s'))] \\ &= r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[\max_{a'} Q^{\pi_k}(s', a')] = \mathcal{T} Q^{\pi_k}(s,a)\end{aligned}$$

which completes the proof of the first claim.

For the second claim,

$$\|Q^\star - Q^{\pi_{k+1}}\|_\infty \leq \|Q^\star - \mathcal{T} Q^{\pi_k}\|_\infty = \|\mathcal{T} Q^\star - \mathcal{T} Q^{\pi_{k+1}}\|_\infty \leq \gamma \|Q^\star - Q^{\pi_k}\|_\infty$$

where we have used that $Q^\star \geq Q^{\pi_{k+1}} \geq Q^{\pi_k}$ in second step and the contraction property of $\mathcal{T}(\cdot)$ (see Lemma 1.10 in the last step. ■

With this lemma, a convergence rate for the policy iteration algorithm immediately follows.

Theorem 1.14. *(Policy iteration convergence). Let π_0 be any initial policy. For $k \geq \frac{\log \frac{1}{(1-\gamma)\epsilon}}{1-\gamma}$, the k-th policy in policy iteration has the following performance bound:*

$$Q^{\pi^{(k)}} \geq Q^\star - \epsilon \mathbb{1}\,.$$

Iteration complexity for an exact solution. With regards to computing an exact optimal policy, it clear from the previous results that policy iteration is no worse than value iteration. However, with regards to obtaining an exact solution MDP that is independent of the bit complexity, $L(P, r, \gamma)$, improvements are possible (and where we assume basic arithmetic operations on real numbers are order one cost). Naively, the number of iterations of policy iterations is bounded by the number of policies, namely $|\mathcal{A}|^{|\mathcal{S}|}$; here, a small improvement is possible, where the number of iterations of policy iteration can be bounded by $\frac{|\mathcal{A}|^{|\mathcal{S}|}}{|\mathcal{S}|}$. Remarkably, for a fixed value of γ, policy iteration can be show to be a strongly polynomial time algorithm, where policy iteration finds an exact policy in at most $\frac{|\mathcal{S}|^2|\mathcal{A}|\log\frac{|\mathcal{S}|^2}{1-\gamma}}{1-\gamma}$ iterations. See Table 0.1 for a summary, and Section 1.7 for references.

1.5 The Linear Programming Approach

It is helpful to understand an alternative approach to finding an optimal policy for a known MDP. With regards to computation, consider the setting where our MDP $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$ is known and P, r, and γ are all specified by rational numbers. Here, from a computational perspective, the previous iterative algorithms are, strictly speaking, not polynomial time algorithms, due to that they depend polynomially on $1/(1-\gamma)$, which is not polynomial in the description length of the MDP . In particular, note that any rational value of $1-\gamma$ may be specified with only $O(\log\frac{1}{1-\gamma})$ bits of precision. In this context, we may hope for a fully polynomial time algorithm, when given knowledge of the MDP, which would have a computation time which would depend polynomially on the description length of the MDP M, when the parameters are specified as rational numbers. We now see that the LP approach provides a polynomial time algorithm.

1.5.1 The Primal LP and A Polynomial Time Algorithm

Consider the following optimization problem with variables $V \in \mathbb{R}^{|\mathcal{S}|}$:

$$\begin{aligned} \min \quad & \sum_s \mu(s)V(s) \\ \text{subject to} \quad & V(s) \geq r(s,a) + \gamma \sum_{s'} P(s'|s,a)V(s') \quad \forall a \in \mathcal{A},\ s \in \mathcal{S} \end{aligned}$$

Here, the optimal value function $V^\star(s)$ is the unique solution to this linear program. With regards to computation time, linear programming approaches only depend on the description length of the coefficients in the program, due to that this determines the computational complexity of basic additions and multiplications. Thus, this approach will only depend on the bit length description of the MDP, when the MDP is specified by rational numbers.

Computational complexity for an exact solution. Table 0.1 shows the runtime complexity for the LP approach, where we assume a standard runtime for solving a linear program. The strongly polynomial algorithm is an interior point algorithm. See Section 1.7 for references.

Policy iteration and the simplex algorithm. It turns out that the policy iteration algorithm is actually the simplex method with block pivot. While the simplex method, in general, is not a strongly polynomial time algorithm, the policy iteration algorithm is a strongly polynomial time algorithm, provided we keep the discount factor fixed. See [Ye, 2011].

1.5.2 The Dual LP and the State-Action Polytope

For a fixed (possibly stochastic) policy π, let us define the state-action visitation distribution ν_μ^π as:

$$\nu_\mu^\pi(s,a) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr^\pi(s_t = s, a_t = a)$$

where $\Pr^\pi(s_t = s, a_t = a)$ is the state-action visitation probability, where we execute π in M starting at state $s_0 \sim \mu$.

Recall Lemma 1.6 which provides a way to easily compute $\nu_\mu^\pi(s,a)$ through an appropriate vector-matrix multiplication.

It is straightforward to verify that ν_μ^π satisfies, for all states $s \in \mathcal{S}$:

$$\sum_a \nu_\mu^\pi(s,a) = (1-\gamma)\mu(s) + \gamma\sum_{s',a'} P(s|s',a')\nu_\mu^\pi(s',a').$$

Let us define the state-action polytope as follows:

$$\mathcal{K} := \{\nu |\, \nu \geq 0 \text{ and } \sum_a \nu(s,a) = (1-\gamma)\mu(s) + \gamma\sum_{s',a'} P(s|s',a')\nu(s',a')\}$$

We now see that this set precisely characterizes all state-action visitation distributions.

Proposition 1.15. We have that $\mathcal{K}$ is equal to the set of all feasible state-action distributions, i.e. $\nu \in \mathcal{K}$ if and only if there exists a stationary (and possibly randomized) policy π such that $\nu_\mu^\pi = \nu$.

With respect the variables $\nu \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$, the dual LP formulation is as follows:

$$\begin{aligned} \max \quad & \frac{1}{1-\gamma}\sum_{s,a}\nu(s,a)r(s,a) \\ \text{subject to} \quad & \nu \in \mathcal{K} \end{aligned}$$

Note that $\mathcal{K}$ is itself a polytope, and one can verify that this is indeed the dual of the aforementioned LP. This approach provides an alternative approach to finding an optimal solution.

If $\nu^\star$ is the solution to this LP, then we have that:

$$\pi^\star(a|s) = \frac{\nu^\star(s,a)}{\sum_{a'}\nu^\star(s,a')}.$$

An alternative optimal policy is $\operatorname{argmax}_a \nu^\star(s,a)$ (and these policies are identical if the optimal policy is unique).

1.6 Advantages and The Performance Difference Lemma

Throughout, we will overload notation where, for a distribution μ over $\mathcal{S}$, we write:

$$V^\pi(\mu) = \mathbb{E}_{s\sim\mu}\left[V^\pi(s)\right].$$

The *advantage* $A^\pi(s,a)$ of a policy π is defined as

$$A^\pi(s,a) := Q^\pi(s,a) - V^\pi(s).$$

Note that:

$$A^*(s,a) := A^{\pi^*}(s,a) \leq 0$$

for all state-action pairs.

Analogous to the state-action visitation distribution, define the discounted state visitation distribution $d^\pi_{s_0}$ as:

$$d^\pi_{s_0}(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr^\pi(s_t = s|s_0) \tag{0.5}$$

where $\Pr^\pi(s_t = s|s_0)$ is the state visitation probability, under π starting at state s_0. We also write:

$$d^\pi_\mu(s) = \mathbb{E}_{s_0\sim\mu}\left[d^\pi_{s_0}(s)\right].$$

for a distribution μ over $\mathcal{S}$.

The following lemma is a helpful in a number of our analyses.

Lemma 1.16. *(The performance difference lemma) For all policies π, π' and distributions μ over $\mathcal{S}$,*

$$V^\pi(\mu) - V^{\pi'}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s'\sim d^\pi_\mu}\mathbb{E}_{a'\sim\pi(\cdot|s')}\left[A^{\pi'}(s',a')\right].$$

Proof: Let $\Pr^\pi(\tau|s_0 = s)$ denote the probability of observing a trajectory τ when starting in state s and following the policy π. By definition of $d^{\pi_\theta}_{s_0}$, observe that for any function $f : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$,

$$\mathbb{E}_{\tau\sim\Pr^\pi}\left[\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\right] = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}_{s_0}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\big[f(s,a)\big]. \tag{0.6}$$

Using a telescoping argument, we have:

$$\begin{aligned}
V^\pi(s) - V^{\pi'}(s) &= \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right] - V^{\pi'}(s)\\
&= \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t\Big(r(s_t,a_t) + V^{\pi'}(s_t) - V^{\pi'}(s_t)\Big)\right] - V^{\pi'}(s)\\
&\overset{(a)}{=} \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t\Big(r(s_t,a_t) + \gamma V^{\pi'}(s_{t+1}) - V^{\pi'}(s_t)\Big)\right]\\
&\overset{(b)}{=} \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t\Big(r(s_t,a_t) + \gamma\mathbb{E}[V^{\pi'}(s_{t+1})|s_t,a_t] - V^{\pi'}(s_t)\Big)\right]\\
&\overset{(c)}{=} \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t\Big(Q^{\pi'}(s_t,a_t) - V^{\pi'}(s_t)\Big)\right]\\
&= \mathbb{E}_{\tau\sim\Pr^\pi(\tau|s_0=s)}\left[\sum_{t=0}^{\infty}\gamma^t A^{\pi'}(s_t,a_t)\right] = \frac{1}{1-\gamma}\mathbb{E}_{s'\sim d^\pi_s}\,\mathbb{E}_{a\sim\pi(\cdot|s)}\gamma^t A^{\pi'}(s',a),
\end{aligned}$$

where (a) rearranges terms in the summation via telescoping; (b) uses the tower property of conditional expectations; (c) follows by definition; and the final equality follows from Equation 0.6. ■

1.7 Bibliographic Remarks and Further Reading

We refer the reader to [Puterman, 1994] for a more detailed treatment of dynamic programming and MDPs. [Puterman, 1994] also contains a thorough treatment of the dual LP, along with a proof of Lemma 1.15

With regards to the computational complexity of policy iteration, [Ye, 2011] showed that policy iteration is a strongly polynomial time algorithm for a fixed discount rate [1]. Also, see [Ye, 2011] for a good summary of the computational complexities of various approaches. [Mansour and Singh, 1999] showed that the number of iterations of policy iteration can be bounded by $\frac{|\mathcal{A}|^{|\mathcal{S}|}}{|\mathcal{S}|}$.

With regards to a strongly polynomial algorithm, the CIPA algorithm [Ye, 2005] is an interior point algorithm with the claimed runtime in Table 0.1.

Lemma 1.11 is due to Singh and Yee [1994].

The performance difference lemma is due to [Kakade and Langford, 2002, Kakade, 2003], though the lemma was implicit in the analysis of a number of prior works.

[1] The stated strongly polynomial runtime in Table 0.1 for policy iteration differs from that in [Ye, 2011] due to we assume that the runtime per iteration of policy iteration is $|\mathcal{S}|^3 + |\mathcal{S}|^2|\mathcal{A}|$.

Chapter 2

Sample Complexity

Let us now look at the statistical complexity of learning a near optimal policy. Here, we look at a more abstract sampling model, a generative model, which allows us study the minimum number of transitions we need to observe. This chapter characterizes the minimax optimal sample complexity of estimating $Q^\star$ and learning a near optimal policy.

In this chapter, we will assume that the reward function is known (and deterministic). This is often a mild assumption, particularly due to that much of the difficulty in RL is due to the uncertainty in the transition model P. This will also not effect the minimax sample complexity.

This chapter follows the results due to [Azar et al., 2013], along with some improved rates due to [Agarwal et al., 2020b],

Generative models. A *generative model* provides us with a sample $s' \sim P(\cdot|s,a)$ upon input of a state action pair (s,a). Let us consider the most naive approach to learning (when we have access to a generative model): suppose we call our simulator N times at each state action pair. Let $\widehat{P}$ be our empirical model, defined as follows:

$$\widehat{P}(s'|s,a) = \frac{\text{count}(s',s,a)}{N}$$

where count(s',s,a) is the number of times the state-action pair (s,a) transitions to state s'. As the N is the number of calls for each state action pair, the total number of calls to our generative model is $|\mathcal{S}||\mathcal{A}|N$.

The generative model setting is a reasonable abstraction to understand the statistical limit, without having to directly address exploration.

We define $\widehat{M}$ to be the empirical MDP that is identical to the original M, except that it uses $\widehat{P}$ instead of P for the transition model. When clear from context, we drop the subscript on M on the values, action values (and one-step variances and variances which we define later). We let $\widehat{V}^\pi$, $\widehat{Q}^\pi$, $\widehat{Q}^\star$ $\widehat{\pi}^\star$ denote the value function, action value function, and optimal policy in $\widehat{M}$.

A key question here is:

Do we require an accurate model of the world in order to find a near optimal policy?

Let's us first start by looking at the naive approach where we build an accurate model of world, which will be sufficient for learning a near optimal policy. In particular, as we shall see $O(|\mathcal{S}|^2|\mathcal{A}|)$ is sufficient to provide us with an accurate

model [1] The question is if we can improve upon this and find a near optimal policy with a number of samples that is *sub-linear* in the model size, i.e. use a number of samples that is smaller than $O(|\mathcal{S}|^2|\mathcal{A}|)$. Furthermore, we also wish to characterize the minimax dependence on the effective horizon, i.e. on the dependence on $1/(1-\gamma)$.

2.1 Warmup: a naive model-based approach

Note that since P has a $|\mathcal{S}|^2|\mathcal{A}|$ parameters, a naive approach would be to estimate P accurately and then use our accurate model $\widehat{P}$ for planning.

Proposition 2.1. There exists an absolute constant c such that the following holds. Suppose $\epsilon \in \left(0, \frac{1}{1-\gamma}\right)$ and that we obtain

$$\text{\# samples from generative model } \geq \frac{\gamma}{(1-\gamma)^4}\frac{|\mathcal{S}|^2|\mathcal{A}|\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{\epsilon^2}$$

where we uniformly sample every state action pair. Then, with probability greater than $1-\delta$, we have:

- (Model accuracy) The transition model is ϵ has error bounded as:

$$\max_{s,a} \|P(\cdot|s,a) - \widehat{P}(\cdot|s,a)\|_1 \leq (1-\gamma)^2\epsilon/2\,.$$

- (Uniform value accuracy) For all policies π,

$$\|Q^\pi - \widehat{Q}^\pi\|_\infty \leq \epsilon/2$$

- (Near optimal planning) Suppose that $\widehat{\pi}$ is the optimal policy in $\widehat{M}$. We have that:

$$\|\widehat{Q}^{\widehat{\pi}} - Q^\star\|_\infty \leq \epsilon$$

Before we provide the proof, the following lemmas will be helpful throughout:

Lemma 2.2. *(Simulation Lemma) For all π we have that:*

$$Q^\pi - \widehat{Q}^\pi \quad = \quad \gamma(I - \gamma\widehat{P}^\pi)^{-1}(P - \widehat{P})V^\pi$$

Proof: Using our matrix equality for Q^π (see Equation 0.2), we have:

$$\begin{aligned} Q^\pi - \widehat{Q}^\pi &= (I-\gamma P^\pi)^{-1}r - (I-\gamma\widehat{P}^\pi)^{-1}r \\ &= (I-\gamma\widehat{P}^\pi)^{-1}((I-\gamma\widehat{P}^\pi) - (I-\gamma P^\pi))Q^\pi \\ &= \gamma(I-\gamma\widehat{P}^\pi)^{-1}(P^\pi - \widehat{P}^\pi)Q^\pi \\ &= \gamma(I-\gamma\widehat{P}^\pi)^{-1}(P - \widehat{P})V^\pi \end{aligned}$$

which proves the claim. ■

Lemma 2.3. *For any policy π, MDP M and vector $v \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$, we have $\left\|(I-\gamma P^\pi)^{-1}v\right\|_\infty \leq \|v\|_\infty/(1-\gamma)$.*

Proof: Note that $v = (I-\gamma P^\pi)(I-\gamma P^\pi)^{-1}v = (I-\gamma P^\pi)w$, where $w = (I-\gamma P^\pi)^{-1}v$. By triangle inequality, we have

$$\|v\| = \|(I-\gamma P^\pi)w\| \geq \|w\|_\infty - \gamma\|P^\pi w\|_\infty \geq \|w\|_\infty - \gamma\|w\|_\infty\,,$$

[1] Note that this is consistent with parameter counting since P is specified by $O|\mathcal{S}|^2|\mathcal{A}|$ parameters.

where the final inequality follows since $P^\pi w$ is an average of the elements of w by the definition of P^π so that $\|P^\pi w\|_\infty \leq \|w\|_\infty$. Rearranging terms completes the proof. ■

Now we are ready to complete the proof of our proposition.

Proof: Using the concentration of a distribution in the ℓ_1 norm (Lemma A.4), we have that for a fixed s, a that, with probability greater than $1 - \delta$, we have:

$$\|P(\cdot|s,a) - \widehat{P}(\cdot|s,a)\|_1 \leq c\sqrt{\frac{|\mathcal{S}|\log(1/\delta)}{m}}$$

where m is the number of samples used to estimate $\widehat{P}(\cdot|s,a)$. The first claim now follows by the union bound (and redefining δ and c appropriately).

For the second claim, we have that:

$$\begin{aligned}\|Q^\pi - \widehat{Q}^\pi\|_\infty &= \|\gamma(I - \gamma\widehat{P}^\pi)^{-1}(P - \widehat{P})V^\pi\|_\infty \leq \frac{\gamma}{1-\gamma}\|(P - \widehat{P})V^\pi\|_\infty \\ &\leq \frac{\gamma}{1-\gamma}\left(\max_{s,a}\|P(\cdot|s,a) - \widehat{P}(\cdot|s,a)\|_1\right)\|V^\pi\|_\infty \leq \frac{\gamma}{(1-\gamma)^2}\max_{s,a}\|P(\cdot|s,a) - \widehat{P}(\cdot|s,a)\|_1\end{aligned}$$

where the penultimate step uses Holder's inequality. The second claim now follows.

The proof for the final claim immediately follows from the second claim. ■

2.2 Sublinear Sample Complexity

In the previous approach, we are able to accurately estimate the value of *every* policy in the unknown MDP M. However, with regards to planning, we only need an accurate estimate $\widehat{Q}^\star$ of $Q^\star$, which we may hope would require less samples. Let us now see that the model based approach can be refined to obtain minimax optimal sample complexity, which we will see is sublinear in the model size.

We wills state our results in terms of N, and recall that N is the # of call to the generative models per state actin pair, so that:

$$\text{\# samples from generative model } = |\mathcal{S}||\mathcal{A}|N.$$

Let us start with a crude bound on the optimal action-values, which provides a sublinear rate. In the next section, we will improve upon this to obtain the minimax optimal rate.

Proposition 2.4. (Crude Value Bounds) Let $\delta \geq 0$. With probability greater than $1 - \delta$,

$$\begin{aligned}\|Q^\star - \widehat{Q}^\star\|_\infty &\leq \Delta_{\delta,N} \\ \|Q^\star - \widehat{Q}^{\pi^\star}\|_\infty &\leq \Delta_{\delta,N},\end{aligned}$$

where:

$$\Delta_{\delta,N} := \frac{\gamma}{(1-\gamma)^2}\sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}$$

Note that the first inequality above shows a sublinear rate on estimating the value function. Ultimately, we are interested in the value $V^{\widehat{\pi}^\star}$ when we execute $\widehat{\pi}^\star$, not just an estimate $\widehat{Q}^\star$ of $Q^\star$. Here, by Lemma 1.11, we loose an additional horizon factor and have:

$$\|Q^\star - \widehat{Q}^{\widehat{\pi}^\star}\|_\infty \leq \frac{1}{1-\gamma}\Delta_{\delta,N}$$

We return to this point in Corollary 2.7 and Theorem 2.8.

Before we provide the proof, the following lemma will be helpful throughout.

Lemma 2.5. *(Component-wise Bounds) We have that:*

$$\begin{aligned} Q^\star - \widehat{Q}^\star &\leq \gamma(I - \gamma \widehat{P}^{\pi^\star})^{-1}(P - \widehat{P})V^\star \\ Q^\star - \widehat{Q}^\star &\geq \gamma(I - \gamma \widehat{P}^{\widehat{\pi}^\star})^{-1}(P - \widehat{P})V^\star \end{aligned}$$

Proof: For the first claim, the optimality of $\pi^\star$ in M implies:

$$Q^\star - \widehat{Q}^\star = Q^{\pi^\star} - \widehat{Q}^{\widehat{\pi}^\star} \leq Q^{\pi^\star} - \widehat{Q}^{\pi^\star} = \gamma(I - \gamma \widehat{P}^{\pi^\star})^{-1}(P - \widehat{P})V^\star,$$

where we have used Lemma 2.2 in the final step. This proves the first claim.

For the second claim,

$$\begin{aligned} Q^\star - \widehat{Q}^\star &= Q^{\pi^\star} - \widehat{Q}^{\widehat{\pi}^\star} \\ &= (1-\gamma)\left((I - \gamma P^{\pi^\star})^{-1} r - (I - \gamma \widehat{P}^{\widehat{\pi}^\star})^{-1} r\right) \\ &= (I - \gamma \widehat{P}^{\pi^\star})^{-1}((I - \gamma \widehat{P}^{\widehat{\pi}^\star}) - (I - \gamma P^{\pi^\star}))Q^\star \\ &= \gamma(I - \gamma \widehat{P}^{\pi^\star})^{-1}(P^{\pi^\star} - \widehat{P}^{\widehat{\pi}^\star})Q^\star \\ &\leq \gamma(I - \gamma \widehat{P}^{\pi^\star})^{-1}(P^{\pi^\star} - \widehat{P}^{\pi^\star})Q^\star \\ &= \gamma(I - \gamma \widehat{P}^{\pi^\star})^{-1}(P - \widehat{P})V^\star, \end{aligned}$$

where the inequality follows from $\widehat{P}^{\widehat{\pi}^\star} Q^\star \leq \widehat{P}^{\pi^\star} Q^\star$, due to the optimality of $\pi^\star$. This proves the second claim. ■

Proof: Following from the simulation lemma (Lemma 2.2) and Lemma 2.3, we have:

$$\|Q^\star - \widehat{Q}^{\pi^\star}\|_\infty \leq \frac{\gamma}{1-\gamma} \|(P - \widehat{P})V^\star\|_\infty.$$

Also, the previous lemma, implies that:

$$\|Q^\star - \widehat{Q}^\star\|_\infty \leq \frac{\gamma}{1-\gamma} \|(P - \widehat{P})V^\star\|_\infty$$

By applying Hoeffding's inequality and the union bound,

$$\|(P - \widehat{P})V^\star\|_\infty = \max_{s,a} |\mathbb{E}_{s' \sim P(\cdot|s,a)}[V^\star(s')] - \mathbb{E}_{s' \sim \widehat{P}(\cdot|s,a)}[V^\star(s')]| \leq \frac{1}{1-\gamma}\sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}$$

which holds with probability greater than $1 - \delta$. This completes the proof. ■

2.3 Minimax Optimal Sample Complexity with the Model Based Approach

We now refine the crude bound on $\widehat{Q}^\star$ to be optimal:

Theorem 2.6. *(Value estimation) For $\delta \geq 0$ and with probability greater than $1 - \delta$,*

$$\|Q^\star - \widehat{Q}^\star\|_\infty \leq \gamma\sqrt{\frac{c}{(1-\gamma)^3}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + \frac{c\gamma}{(1-\gamma)^3}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N},$$

where c is an absolute constant.

Corollary 2.7. *Provided that $\epsilon \leq 1$, we have that if*

$$N \geq \frac{c}{(1-\gamma)^3} \frac{|\mathcal{S}||\mathcal{A}| \log(c|\mathcal{S}||\mathcal{A}|/\delta)}{\epsilon^2},$$

then with probability greater than $1-\delta$, then

$$\|Q^\star - \widehat{Q}^\star\|_\infty \leq \epsilon.$$

Note that this implies $\|Q^\star - Q^{\widehat{\pi}^\star}\|_\infty \leq \epsilon/(1-\gamma)$.

Ultimately, we are interested in the value $V^{\widehat{\pi}^\star}$ when we execute $\widehat{\pi}^\star$, not just an estimate $\widehat{Q}^\star$ of $Q^\star$. The above corollary shows is not sharp with regards to finding a near optimal policy. The following Theorem shows that in fact both value estimation and policy estimation have the same rate.

Theorem 2.8. *Provided that $\epsilon \leq \sqrt{\frac{1}{1-\gamma}}$, we have that if*

$$N \geq \frac{c}{(1-\gamma)^3} \frac{|\mathcal{S}||\mathcal{A}| \log(c|\mathcal{S}||\mathcal{A}|/\delta)}{\epsilon^2},$$

then with probability greater than $1-\delta$, then

$$\|Q^\star - Q^{\widehat{\pi}^\star}\|_\infty \leq \epsilon, \text{ and } \|Q^\star - \widehat{Q}^\star\|_\infty \leq \epsilon.$$

We state this improved theorem without proof due to it being more involved, and only prove Theorem 2.6. See Section 2.5 for further discussion.

2.3.1 Lower Bounds

Let us say that an estimation algorithm $\mathcal{A}$, which is a map from samples to an estimate $\widehat{Q}^\star$, is (ϵ, δ)-good on MDP M if $\|Q^\star - \widehat{Q}^\star\|_\infty \leq \epsilon$ holds with probability greater than $1-\delta$.

Theorem 2.9. *There exists ϵ_0, δ_0, c and a set of MDPs $\mathcal{M}$ such that for $\epsilon \in (0, \epsilon_0)$ and $\delta \in (0, \delta_0)$ if algorithm $\mathcal{A}$ is (ϵ, δ)-good on all $M \in \mathcal{M}$, then $\mathcal{A}$ must use a number of samples that is lower bounded as follows*

$$\text{\# samples from generative model } \geq \frac{c}{1-\gamma} \frac{|\mathcal{S}||\mathcal{A}| \log(c|\mathcal{S}||\mathcal{A}|/\delta)}{\epsilon^2},$$

2.3.2 Variance Lemmas

The key to the shaper analysis is to more sharply characterize the variance in our estimates.

Denote the variance of any real valued f under a distribution $\mathcal{D}$ as:

$$\mathrm{Var}_{\mathcal{D}}(f) := E_{x\sim\mathcal{D}}[f(x)^2] - (E_{x\sim\mathcal{D}}[f(x)])^2$$

Slightly abusing the notation, for $V \in R^{|\mathcal{S}|}$, we define the vector $\mathrm{Var}_P(V) \in R^{|\mathcal{S}||\mathcal{A}|}$ as:

$$\mathrm{Var}_P(V)(s,a) := \mathrm{Var}_{P(\cdot|s,a)}(V)$$

Equivalently,

$$\mathrm{Var}_P(V) = P(V)^2 - (PV)^2\,.$$

Now we characterize a relevant deviation in terms of the its variance.

Lemma 2.10. *Let $\delta \geq 0$. With probability greater than $1-\delta$,*

$$|(P-\widehat{P})V^\star| \leq \sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}\sqrt{\mathrm{Var}_P(V^\star)} + \frac{1}{1-\gamma}\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{3N}\mathbb{1}\,.$$

Proof: The claims follows from Bernstein's inequality along with a union bound over all state-action pairs. ■

The key ideas in the proof are in how we bound $\|(I-\gamma\widehat{P}^{\pi^\star})^{-1}\sqrt{\mathrm{Var}_P(V^\star)}\|_\infty$ and $\|(I-\gamma\widehat{P}^{\widehat{\pi}^\star})^{-1}\sqrt{\mathrm{Var}_P(V^\star)}\|_\infty$. It is helpful to define Σ_M^π as the variance of the discounted reward, i.e.

$$\Sigma_M^\pi(s,a) := \mathbb{E}\left[\left(\sum_{t=0}^\infty \gamma^t r(s_t,a_t) - Q_M^\pi(s,a)\right)^2 \middle| s_0=s, a_0=a\right]$$

where the expectation is induced under the trajectories induced by π in M. It is straightforward to verify that $\|\Sigma_M^\pi\|_\infty \leq \gamma^2/(1-\gamma)^2$.

ex The following lemma shows that Σ_M^π satisfies a Bellman consistency condition.

Lemma 2.11. *(Bellman consistency of Σ) For any MDP M,*

$$\Sigma_M^\pi = \gamma^2\mathrm{Var}_P(V_M^\pi) + \gamma^2 P^\pi \Sigma_M^\pi \tag{0.1}$$

where P is the transition model in MDP M.

The proof is left as an exercise to the reader.

Lemma 2.12. *(Weighted Sum of Deviations) For any policy π and MDP M,*

$$\left\|(I-\gamma P^\pi)^{-1}\sqrt{\mathrm{Var}_P(V_M^\pi)}\right\|_\infty \leq \sqrt{\frac{2}{(1-\gamma)^3}},$$

where P is the transition model of M.

Proof: Note that $(1-\gamma)(I-\gamma P^\pi)^{-1}$ is matrix whose rows are a probability distribution. For a positive vector v and a distribution ν (where ν is vector of the same dimension of v), Jensen's inequality implies that $\nu\cdot\sqrt{v} \leq \sqrt{\nu\cdot v}$. This implies:

$$\begin{aligned}
\|(I-\gamma P^\pi)^{-1}\sqrt{v}\|_\infty &= \frac{1}{1-\gamma}\|(1-\gamma)(I-\gamma P^\pi)^{-1}\sqrt{v}\|_\infty \\
&\leq \sqrt{\left\|\frac{1}{1-\gamma}(I-\gamma P^\pi)^{-1}v\right\|_\infty} \\
&\leq \sqrt{\left\|\frac{2}{1-\gamma}(I-\gamma^2 P^\pi)^{-1}v\right\|_\infty}\,.
\end{aligned}$$

where we have used that $\|(I-\gamma P^\pi)^{-1}v\|_\infty \leq 2\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty$ (which we will prove shortly). The proof is completed as follows: by Equation 0.1, $\Sigma_M^\pi = \gamma^2(I-\gamma^2 P^\pi)^{-1}\mathrm{Var}_P(V_M^\pi)$, so taking $v = \mathrm{Var}_P(V_M^\pi)$ and using that $\|\Sigma_M^\pi\|_\infty \leq \gamma^2/(1-\gamma)^2$ completes the proof.

Finally, to see that $\|(I-\gamma P^\pi)^{-1}v\|_\infty \leq 2\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty$, observe:

$$\begin{aligned}
\|(I-\gamma P^\pi)^{-1}v\|_\infty &= \|(I-\gamma P^\pi)^{-1}(I-\gamma^2 P^\pi)(I-\gamma^2 P^\pi)^{-1}v\|_\infty \\
&= \|(I-\gamma P^\pi)^{-1}\Big((1-\gamma)I+\gamma(I-\gamma P^\pi)\Big)(I-\gamma^2 P^\pi)^{-1}v\|_\infty \\
&= \|\Big((1-\gamma)(I-\gamma P^\pi)^{-1}+\gamma I\Big)(I-\gamma^2 P^\pi)^{-1}v\|_\infty \\
&\leq (1-\gamma)\|(I-\gamma P^\pi)^{-1}(I-\gamma^2 P^\pi)^{-1}v\|_\infty + \gamma\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty \\
&\leq \frac{1-\gamma}{1-\gamma}\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty + \gamma\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty \\
&\leq 2\|(I-\gamma^2 P^\pi)^{-1}v\|_\infty
\end{aligned}$$

which proves the claim. ■

2.3.3 Completing the proof

Lemma 2.13. *Let $\delta \geq 0$. With probability greater than $1-\delta$, we have:*

$$\begin{aligned}
\mathrm{Var}_P(V^\star) &\leq 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star}) + \Delta'_{\delta,N}\mathbb{1} \\
\mathrm{Var}_P(V^\star) &\leq 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^\star) + \Delta'_{\delta,N}\mathbb{1}
\end{aligned}$$

where

$$\Delta'_{\delta,N} := \frac{1}{(1-\gamma)^2}\sqrt{\frac{18\log(6|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + \frac{1}{(1-\gamma)^4}\frac{4\log(6|\mathcal{S}||\mathcal{A}|/\delta)}{N}\,.$$

Proof: By definition,

$$\begin{aligned}
\mathrm{Var}_P(V^\star) &= \mathrm{Var}_P(V^\star) - \mathrm{Var}_{\widehat{P}}(V^\star) + \mathrm{Var}_{\widehat{P}}(V^\star) \\
&= P(V^\star)^2 - (PV^\star)^2 - \widehat{P}(V^\star)^2 + (\widehat{P}V^\star)^2 + \mathrm{Var}_{\widehat{P}}(V^\star) \\
&= (P-\widehat{P})(V^\star)^2 - \Big((PV^\star)^2 - (\widehat{P}V^\star)^2\Big) + \mathrm{Var}_{\widehat{P}}(V^\star)
\end{aligned}$$

Now we bound each of these terms with Hoeffding's inequality and the union bound. For the first term, with probability greater than $1-\delta$,

$$\|(P-\widehat{P})(V^\star)^2\|_\infty \leq \frac{1}{(1-\gamma)^2}\sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}\,.$$

For the second term, again with probability greater than $1-\delta$,

$$\begin{aligned}
\|(PV^\star)^2 - (\widehat{P}V^\star)^2\|_\infty &\leq \|PV^\star + \widehat{P}V^\star\|_\infty \|PV^\star - \widehat{P}V^\star\|_\infty \\
&\leq \frac{2}{1-\gamma}\|(P-\widehat{P})V^\star\|_\infty \leq \frac{2}{(1-\gamma)^2}\sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}\,.
\end{aligned}$$

where we have used that $(\cdot)^2$ is a component-wise operation in the second step. For the last term:

$$\begin{aligned}
\mathrm{Var}_{\widehat{P}}(V^\star) &= \mathrm{Var}_{\widehat{P}}(V^\star - \widehat{V}^{\pi^\star} + \widehat{V}^{\pi^\star}) \\
&\leq 2\mathrm{Var}_{\widehat{P}}(V^\star - \widehat{V}^{\pi^\star}) + 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star}) \\
&\leq 2\|V^\star - \widehat{V}^{\pi^\star}\|_\infty^2 + 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star}) \\
&= 2\Delta_{\delta,N}^2 + 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star})\,.
\end{aligned}$$

where $\Delta_{\delta,N}$ is defined in Proposition 2.4. To obtain a cumulative probability of error less than δ, we replace δ in the above claims with $\delta/3$. Combining these bounds completes the proof of the first claim. The argument in the above display also implies that $\mathrm{Var}_{\widehat{P}}(V^\star) \le 2\Delta_{\delta,N}^2 + 2\mathrm{Var}_{\widehat{P}}(\widehat{V}^\star)$ which proves the second claim. ■

Using Lemma 2.10 and 2.13, we have the following corollary.

Corollary 2.14. *Let $\delta \ge 0$. With probability greater than $1-\delta$, we have:*

$$|(P-\widehat{P})V^\star| \le c\sqrt{\frac{\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star})\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + \Delta''_{\delta,N}\mathbb{1}$$
$$|(P-\widehat{P})V^\star| \le c\sqrt{\frac{\mathrm{Var}_{\widehat{P}}(\widehat{V}^\star)\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + \Delta''_{\delta,N}\mathbb{1}\,,$$

where

$$\Delta''_{\delta,N} := c\frac{1}{1-\gamma}\left(\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}\right)^{3/4} + \frac{c}{(1-\gamma)^2}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}\,,$$

and where c is an absolute constant.

Proof:(of Theorem 2.6) The proof consists of bounding the terms in Lemma 2.5. We have:

$$\begin{aligned}
& \gamma\|(I-\gamma\widehat{P}^{\pi^\star})^{-1}(P-\widehat{P})V^\star\|_\infty \\
\le\ & c\gamma\sqrt{\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}}\|(I-\gamma\widehat{P}^{\pi^\star})^{-1}\sqrt{\mathrm{Var}_{\widehat{P}}(\widehat{V}^{\pi^\star})}\|_\infty + \frac{c\gamma}{(1-\gamma)^2}\left(\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}\right)^{3/4} \\
& + \frac{c\gamma}{(1-\gamma)^3}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N} \\
\le\ & \gamma\sqrt{\frac{2}{(1-\gamma)^3}}\sqrt{\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + \frac{c\gamma}{(1-\gamma)^2}\left(\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}\right)^{3/4} + \frac{c\gamma}{(1-\gamma)^3}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N} \\
\le\ & 3\gamma\sqrt{\frac{1}{(1-\gamma)^3}}c\sqrt{\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}} + 2\frac{c\gamma}{(1-\gamma)^3}\frac{\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{N}\,,
\end{aligned}$$

where the first step uses Corollary 2.14; the second uses Lemma 2.12; and the last step uses that $2ab \le a^2+b^2$ (and choosing a, b appropriately). The proof of the lower bound is analogous. Taking a different absolute constant completes the proof. ■

2.4 Scalings and Effective Horizon Dependencies

It will be helpful to more intuitively understand why $1/(1-\gamma)^3$ is the effective horizon dependency one might hope to expect, from a dimensional analysis viewpoint. Due to that $Q^\star$ is a quantity that is as large as $1/(1-\gamma)$, to account for this scaling, it is natural to look at obtaining relative accuracy.

In particular, if

$$N \ge \frac{c}{1-\gamma}\frac{|\mathcal{S}||\mathcal{A}|\log(c|\mathcal{S}||\mathcal{A}|/\delta)}{\epsilon^2},$$

then with probability greater than $1-\delta$, then

$$\|Q^\star - Q^{\widehat{\pi}^\star}\|_\infty \le \frac{\epsilon}{1-\gamma}, \text{ and } \|Q^\star - \widehat{Q}^\star\|_\infty \le \frac{\epsilon}{1-\gamma}.$$

(provided that $\epsilon \leq \sqrt{1-\gamma}$ using Theorem 2.8). In other words, if we had normalized the value functions [2], then for additive accuracy (on our normalized value functions) our sample size would scale linearly with the effective horizon.

2.5 Bibliographic Remarks and Further Readings

The notion of a generative model was first introduced in [Kearns and Singh, 1999], which made the argument that, up to horizon factors and logarithmic factors, both model based methods and model free methods are comparable. [Kakade, 2003] gave an improved version of this rate (analogous to the crude bounds seen here).

Theorem 2.6 is due to [Azar et al., 2013], and the proof in this section largely follows this work. Improvements are possible with regards to bounding the quality of $\widehat{\pi}^\star$; here, Theorem 2.8 shows that the model based approach is near optimal even for policy itself; showing that the quality of $\widehat{\pi}^\star$ does suffer any amplification factor of $1/(1-\gamma)$. [Sidford et al., 2018] provides the first proof of this improvement using a variance reduction algorithm with value iteration. The improvement in Theorem 2.8 is due to [Agarwal et al., 2020b], which shows that the naive model based approach is sufficient.

Finally, we remark that we may hope for the bounds on our value estimation to hold up to $\epsilon \leq 1/(1-\gamma)$, which would be consistent with the lower bounds. Here, the work in [Li et al., 2020] shows this limit is achievable, albeit with a slightly different algorithm where they introduce perturbations. It is an open question if the naive model based approach also achieves the non-asymptotic statistical limit.

[2] Rescaling the value functions by multiplying by $(1-\gamma)$, i.e. $Q^\pi \leftarrow (1-\gamma)Q^\pi$, would keep the values bounded between 0 and 1. Throughout, this book it is helpful to understand sample size with regards to normalized quantities.

Chapter 3

Approximate Value Function Methods

For large MDPs, when the underlying MDPs are unknown and we do not have a generative model, we cannot directly perform policy iteration. This chapter will consider a simple approach where we learn an approximate Q function and then update our policy greedily with respect to the estimated Q function.

This chapter focuses on obtaining of *average case function approximation error* bounds, provided we have a somewhat stringent condition on how the underlying MDP behaves, quantified by the *concentrability coefficient*. This notion was introduced in [Munos, 2003, 2005]. While the notion is somewhat stringent, we will see that there is reason to believe it is not avoidable. Chapters 11 and 12 seek to relax this notion.

3.1 Setting

We consider infinite discounted MDPs $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$ in this chapter. Here the MDP might have large or even continuous state space. We are given a policy class $\Pi = \{\pi : \mathcal{S} \mapsto \mathcal{A}\} \subset \mathcal{S} \mapsto \mathcal{A}$. Note that the policy class is a restricted policy class which is a subset of the class of all mappings from $\mathcal{S}$ to $\mathcal{A}$. We denote the best policy in policy class as $\pi^\star$, which is the policy that maximizes the expected total reward with μ as the initial state distribution:

$$\pi^\star \in \mathrm{argmax}_{\pi \in \Pi} \mathbb{E}\left[\sum_{h=0}^{\infty} \gamma^h r(s_h, a_h) | a_h = \pi(s_h)\right].$$

Note that $\pi^\star$ is the best policy in policy class that maximizes the objective function and it is not necessarily true that $\pi^\star$ will be the optimal policy of the MDP M which maximizes total reward starting from any state simultaneously (i.e., the policy class might not be rich enough to contain the optimal policy of M).

3.2 Approximate Greedy Policy Selector

Given a policy π^0, one intuitive approach we attempt to do is to act greedily with respect π at every state. However due to the unknown MDP and large state space, we will not be able to have $A^\pi(s, a)$ at every state-action pair. Instead, we can act greedily in the average sense:

$$\widehat{\pi} \in \mathrm{argmax}_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^\pi}\left[A^{\pi^0}(s, \pi(s))\right].$$

We call the above procedure as greedy policy selector. We aim to pick a policy that acts greedily with respect to π^0 under the states visited by π^0.

Implement the exact greedy policy selector is not possible due to the fact that we do not know the exact A^π. In this section, we explain how to achieve an ε-approximate greedy policy selector, which is defined in the definition below.

Definition 3.1 (ε-approximate Greedy Policy Selector)**.** Given a policy π, we denote $\mathcal{G}_\varepsilon(\pi, \Pi, \mu)$ as the oracle that returns a policy $\widehat{\pi} \in \Pi$, such that:

$$\mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \widehat{\pi}(s)) \geq \max_{\widetilde{\pi}\in\Pi} \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \widetilde{\pi}(s)) - \varepsilon.$$

Below we study two approaches to implement the above selector: one is via a reduction to classification with the policy class Π and the other one is via a reduction to regression using value function approximation.

3.2.1 Implementing Approximate Greedy Policy Selector using Classification

Below we explain that we can implement such approximate Greedy Policy Selector via reduction to a classic supervised learning oracle—weighted multi-class classification. We first define a weighted classification oracle as follows.

Definition 3.2 (Weighted Classification Oracle)**.** Given a dataset $\mathcal{D} = \{s_i, c_i\}_{i=1}^N$ where $c_i \in \mathbb{R}^A$, and a policy class Π, the weight classification oracle returns the best classifier:

$$\text{CO}(\mathcal{D}, \Pi) = \text{argmax}_{\pi\in\Pi} \sum_{i=1}^N c_i[\pi(s)],$$

where $c[a]$ denotes the value in the entry in c that corresponds to action a.

Weighted classification oracle is a standard oracle in supervised learning setting, and weighted classification oracle can be further reduced to a regular classification oracle or a regression oracle. We will assume the existence of CO.

Now we can implement an approximate greedy policy selector via the CO oracle using data from d_μ^π up to statistical error. We draw a dataset $\mathcal{D} = \{s_i, a_i, \widetilde{A}_i\}$, where $s_i \sim d_\mu^\pi$, $a_i \sim U(\mathcal{A})$, and $\widetilde{A}_i$ is an unbiased estimate of $A^\pi(s_i, a_i)$ computed from a single rollout. We can perform the policy selection procedure using the CO oracle as follows:

$$\widehat{\pi} := \text{argmax}_{\pi\in\Pi} \sum_{i=1}^N \widetilde{c}_i[\pi(s_i)], \tag{0.1}$$

where $\widetilde{c}_i \in \mathbb{R}^A$ is a one-hot vector with zeros everywhere, except the entry that corresponds to a_i contains $\frac{\widetilde{A}_i}{1/A}$. Essentially we are performing importance weighting here so that an $\widetilde{c}_i$ is indeed an unbiased estimate of the vector $[A^\pi(s_i, a)]_{a\in\mathcal{A}}^\top \in \mathbb{R}^A$, given s_i.

Theorem 3.3 (Approximate Greedy Policy Selector)**.** *Given a dataset $\mathcal{D} = \{s_i, a_i, \widetilde{A}_i\}$, where $s_i \sim d_\mu^\pi$, $a_i \sim U(\mathcal{A})$, and $\widetilde{A}_i$ is an unbiased estimate of $A^\pi(s_i, a_i)$ computed from a single rollout, denote $\widehat{\pi}$ as the return in Eq. 0.1. We have that with probability at least $1-\delta$:*

$$\mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \widehat{\pi}(s)) \geq \max_{\widetilde{\pi}\in\Pi} \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \widetilde{\pi}(s)) - \frac{4A}{1-\gamma}\sqrt{\frac{\ln(|\Pi|/\delta)}{N}}.$$

Proof: We can apply Hoeffding's inequality for a fixed policy $\pi \in \Pi$ and then a union bound over all $\pi \in \Pi$. With probability at least $1-\delta$, we have that for all $\pi \in \Pi$

$$\left| \sum_{i=1}^{N} \widetilde{c}_i[\pi(s_i)]/N - \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \pi(s)) \right| \leq \frac{2A}{1-\gamma} \sqrt{\frac{\ln\left(|\Pi|/\delta\right)}{N}} := \varepsilon_{stat}$$

To see this, note that first of all, we have:

$$\mathbb{E}_{s_i}\left[\mathbb{E}\left[\widetilde{c}_i[\pi(s_i)]\middle| s_i\right]\right] = \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \pi(s)),$$

as $\mathbb{E}[\widetilde{c}_i|s_i] = [A^\pi(s_i, a)]_{a\in\mathcal{A}}^\top$, due to the importance weighting in $\widetilde{c}_i$. Second, note that we have:

$$|\widetilde{c}_i[a]| \leq \frac{A}{1-\gamma}.$$

With the uniform convergence result, we can conclude that:

$$\mathbb{E}_{s\sim d_\mu^\pi} A^\pi\left(s, \widehat{\pi}(s)\right) \geq \frac{1}{N}\sum_{i=1}^{N} \widetilde{c}_i[\widehat{\pi}(s_i)] - \varepsilon_{stat} \leq \frac{1}{N}\sum_{i=1}^{N} \widetilde{c}_i[\pi(s_i)] - \varepsilon_{stat} \geq \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \pi(s)) - 2\varepsilon_{stat},$$

for any π including $\operatorname{argmax}_\pi \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, a)$. ■

Note that the above analysis shows that we can approximately optimize $\max_{\widetilde{\pi}\in\Pi} \mathbb{E}_{s\sim d_\mu^\pi} A^\pi(s, \widetilde{\pi})$ up to statistical error. We can set $\frac{A}{1-\gamma}\sqrt{\ln(|\Pi|/\delta)/N} = \varepsilon$ and solve for N which is the total number of i.i.d samples we need to draw in order to get an ε-approximate policy selector with probability at least $1-\delta$.

3.2.2 Implementing Approximate Greedy Policy Selector using Regression

Here we present an implementation based on value function approximation. Specifically, instead of starting directly from a restrict policy class Π and a reduction to classification, we start from a restricted value function class $\mathcal{F} = \{f : \mathcal{S}\times\mathcal{A} \mapsto [1, 1/(1-\gamma)]\}$. In this case, one can think about the policies class Π consisting of all greedy policies with respect to $f \in \mathcal{F}$, i.e., $\Pi = \{\pi(s) = \operatorname{argmax}_a f(s, a) : f \in \mathcal{F}\}$.

We perform a reduction to regression. Consider the following least square regression problem. Given the dataset $\{s_i, a_i, \widetilde{A}_i\}$ with $s_i \sim d_\mu^\pi, a_i \sim U(\mathcal{A})$ and $\widetilde{A}_i$ is an unbiased estimate of $A^\pi(s_i, a_i)$, we perform the following regression:

$$\widehat{f} \in \operatorname{argmax}_{f\in\mathcal{F}} \sum_{i=1}^{N} \left(f(s_i, a_i) - \widetilde{A}^i\right)^2.$$

With $\widehat{f}$, the approximate greedy policy is set as:

$$\widehat{\pi}(s) = \operatorname{argmax}_{a\in\mathcal{A}} \widehat{f}(s, a), \forall s.$$

Using a similar uniform convergence argument as in the proof of Theorem 3.3, it is not hard to get a similar generalization bound as in Theorem 3.3.

3.3 Approximate Policy Iteration (API)

With the above approximate greedy policy selector, now we introduce the Approximate Policy Iteration (API) algorithm, which is described in the following iteration:

$$\pi^{t+1} := \mathcal{G}_\varepsilon(\pi^t, \Pi, \mu). \tag{0.2}$$

Note that API does not guarantee policy improvement nor convergence without additional assumption. We will give an example in the next section where we show that even with the exact approximate greedy policy selection, i.e., $\varepsilon = 0$, API cannot make any policy improvement and could oscillate between two suboptimal policies forever.

To have meaningful guarantees of policy improvement and convergence for API, we introduce the following concentration assumption on the initial distribution μ:

Assumption 3.4 (Bounded Concentration Coefficient)**.** We assume that $C := \max_{\pi \in \Pi} \sup_{s \in \mathcal{S}} \frac{d_\mu^\pi(s)}{\mu(s)} < \infty$.

With this assumption, we can show that API has monotonic improvement as long as there is local improvement, i.e., $\max_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi(s))$ is reasonably big.

Theorem 3.5 (Monotonic Policy Improvement)**.** *For any t, we have:*

$$V^{\pi^{t+1}} - V^{\pi^t} \geq \frac{1}{C} \left[\max_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^{\pi^t}} \left[A^{\pi^t}(s, \pi(s)) \right] - \varepsilon \right].$$

Proof: We start with Performance Difference Lemma.

$$\begin{aligned}
(1-\gamma)\left(V^{\pi^{t+1}} - V^{\pi^t}\right) &= \mathbb{E}_{s \sim d_\mu^{\pi^{t+1}}} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right] \\
&= \mathbb{E}_{s \sim d_\mu^{\pi^t}} \frac{d_\mu^{\pi^{t+1}}(s)}{d_\mu^{\pi^t}(s)} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right] \geq \mathbb{E}_{s \sim d_\mu^{\pi^t}} \frac{(1-\gamma)\mu(s)}{d_\mu^{\pi^t}(s)} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right] \\
&\geq (1-\gamma) \mathbb{E}_{s \sim d_\mu^{\pi^t}} \inf_s \frac{\mu(s)}{d_\mu^{\pi^t}(s)} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right] \geq \frac{1-\gamma}{C} \mathbb{E}_{s \sim d_\mu^{\pi^t}} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right],
\end{aligned}$$

where the last inequality uses the definition of C in Assumption 3.4.

This implies that:

$$V^{\pi^{t+1}} - V^{\pi^t} \geq \frac{1}{C} \mathbb{E}_{s \sim d_\mu^{\pi^t}} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right] \geq \frac{1}{C} \max_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^{\pi^t}} \left[A^{\pi^t}(s, \pi(s)) \right] - \frac{\varepsilon}{C}.$$

■

The above theorem implies that when $\max_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi(s)) > \varepsilon$ and $C < \infty$, then we make monotonic improvement every iteration.

3.4 Failure Case of API Without Assumption 3.4

In this section, we show that API indeed will fail to provide policy improvement if $C < \infty$. To illustrate this phenomena, we simply consider the exact greedy policy selector, i.e., we assume that for $\mathcal{G}_\varepsilon(\pi, \Pi, \mu)$, we have $\varepsilon = 0$.

Claim 3.6. *There exists a policy class Π, an MDP, a μ restart distribution where $C = \infty$, and two policies π' and π'', such that if one start API with $\pi^0 \in \{\pi', \pi''\}$, π^t and π^{t+1} will oscillate between π' and π'' which are both γ away from the optimal policy. Namely API with $\pi^{t+1} = \mathcal{G}_0(\pi^t, \Pi, \mu)$ will not be able to make any policy improvement nor will it converge.*

Proof: The MDP is shown in Fig. 0.1 where the transition is deterministic and $\mu(s_1) = 1$. We consider Π that contains all stationary policies. We consider the two policies π' and π'' as follows:

$$\pi'(s_1) = a_1, \pi'(s_2) = a_2, \pi'(s_3) = a_1; \quad \pi''(s_1) = a_2, \pi''(s_2) = a_1, \pi''(s_3) = a_2.$$

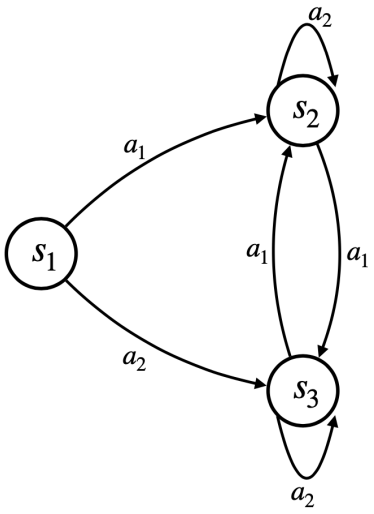

Figure 0.1: The example MDP. The MDP has deterministic transition and μ has probability mass on s_1. We have reward zero every where except $r(s_2, a_1) = r(s_3, a_1) = 1$.

Hence for π', $d_\mu^{\pi'}(s_3) = 0$ and $d_\mu^{\pi'}(s) > 0$ for $s \neq s_3$. Similarly for π'', we have $d_\mu^{\pi''}(s_2) = 0$ and $d_\mu^{\pi''}(s) > 0$ for $s \neq s_2$.

Consider the greedy policy selection under π':

$$\pi \in \mathrm{argmax}_{\pi \in \Pi} \mathbb{E}_{s \sim d_\mu^{\pi'}} A^{\pi'}(s, \pi(s)).$$

We claim that π'' is one of the maximizers of the above procedure. This is because $d_\mu^{\pi'}(s_3) = 0$ and thus $\pi(s_3)$ does not affect the objective function at all. For s_1, note that $Q^{\pi'}(s_1, a_1) = 0$ while $Q^{\pi'}(s_1, a_2) > 0$. Thus a greedy policy will pick a_2 which is consistent to the choice of π''. For s_2, we have $Q^{\pi'}(s_2, a_1) > 0$ while $Q^{\pi'}(s_2, a_2) = 0$. Thus a greedy policy will pick a_1 at s_2 which again is consistent with the choice of π'' at s_2. This concludes that π'' is one of the greedy policies under π'.

Similarly, one can argue that $\pi' \in \mathrm{argmax}_{\pi \in \Pi} \mathbb{E}_{s \sim d_h^{\pi''}} A^{\pi''}(s, \pi(s))$, i.e., at π'', API will switch back to π' in the next iteration.

Thus, we have proven that when running API with either π' or π'' as initialization, API can oscillate between π' and π'' forever. Note that π' and π'' have the same value and are γ away from the optimal policy's value. ■

The above phenomena really comes from the fact that API is making abrupt policy update, i.e., there is no way we can guarantee π^{t+1} is close to π^t, for instance, in terms of their resulting state distribution. Thus, for states s that have really small probability under $d_\mu^{\pi^t}$, $\pi^{t+1}(\cdot|s)$ and $\pi^t(\cdot|s)$ could be different. Intuitively, if we look at the Performance Difference between π^{t+1} and π^t, we see that:

$$V^{\pi^{t+1}} - V^{\pi^t} = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d_\mu^{\pi^{t+1}}} \left[A^{\pi^t}(s, \pi^{t+1}(s)) \right],$$

which says that in order to make policy improvement, we need the new policy π^{t+1} to be greedy with respect to π^t under $d_\mu^{\pi^{t+1}}$—the state distribution of the new policy. However, the greedy policy selector only selects a policy that is greedy with respect to π^t under $d_\mu^{\pi^t}$. Hence, unless $d_\mu^{\pi^t}$ and $d_\mu^{\pi^{t+1}}$ are close, we will not be able to directly transfer the potential local one-step improvement $\mathbb{E}_{s \sim d^{\pi^t}} A^{\pi^t}(s, \pi^{t+1}(s))$ to policy improvement $V^{\pi^{t+1}} - V^{\pi^t}$.

3.5 Can we relax the concentrability notion?

There is another family of policy optimization algorithm which use incremental policy updates, i.e., when we perform policy update, we ensure that $d_\mu^{\pi^{t+1}}$ is not all that different from $d_\mu^{\pi^t}$. Incremental policy update will the core of Part 3. For instance, the algorithm Conservative Policy Iteration, which we study in Chapter 12, uses a conservative policy update $\pi^{t+1}(\cdot|s) := (1-\alpha)\pi^t(\cdot|s) + \alpha\mathcal{G}_\varepsilon(\pi^t, \Pi, \mu)$ for all s, which, for small α, ensures that $d_\mu^{\pi^{t+1}}$ and $d_\mu^{\pi^t}$ are not too far apart. Properly setting the step size α, we can show that CPI makes monotonic policy improvement and converges to a local optimal policy with a more relaxed condition over Assumption 3.4. We will explain the benefit of incremental policy updating in more detail in Part III.

3.6 Bibliographic Remarks and Further Readings

The notion of concentrability was developed in [Munos, 2003, 2005] in order to permitting sharper bounds in terms of average case function approximation error, provided that the concentrability coefficient is bounded. These methods also permit sample based fitting methods, with sample size and error bounds, provided there is a data collection policy that induces a bounded concentrability coefficient [Munos, 2005, Szepesvári and Munos, 2005, Antos et al., 2008, Lazaric et al., 2016]. Chen and Jiang [2019] provide a more detailed discussion on this quantity.

Chapter 4

Generalization

Up to now we have focussed on "tabular" MDPs. While studying this setting is theoretically important, we ultimately seek to have learnability results which are applicable to cases where number of states is large (or, possibly, countably or uncountably infinite). This is a question of generalization.

A fundamental question here is:

To what extent is generalization in RL similar to (or different from) that in supervised learning?

This is the focus of this chapter. Understanding this question is crucial in how we study (and design) scalable algorithms. These insights will also help us to motivate the various more refined assumptions (and settings) that we will consider in subsequent chapters.

In supervised learning (and binary classification in particular), it is helpful to distinguish between two different objectives: First, it is not difficult to see that, in general, it is not possible to learn the Bayes optimal classifier in a sample efficient manner without strong underlying assumptions on the data generating process.[1] Alternatively, given some restricted set of classifiers (our hypothesis class $\mathcal{H}$, which may not contain the Bayes optimal classifier), we may hope to do as well as the best classifier in this set, i.e. we seek low (statistical) *regret*. This objective is referred to as agnostic learning; here, obtaining low regret is possible, provided some measure of the complexity of our hypothesis set is not too large.

With regards to reinforcement learning, we may ask a similar question. It is not difficult to see that in order to provably learn the truly optimal policy in a sample efficient manner (say that does not depend on the number of states $|\mathcal{S}|$), then we must rely on quite strong assumptions. Analogous to the agnostic learning question in supervised learning, we may ask the following question: given some restricted (and low complexity) policy class Π (which may not contain the optimal policy $\pi^\star$), what is the sample complexity of doing nearly as well as the best policy in this class?

This chapter follows the reduction from reinforcement learning to supervised learning that was first introduced in [Kearns et al., 2000], which used a different algorithm (the "trajectory tree" algorithm), and our discussion here largely follows the motivation discussed in [Kearns et al., 2000, Kakade, 2003].

Before we address this question, a few remarks are in order.

Binary classification as a $\gamma = 0$ RL problem. Let us observe that the problem of binary classification can be thought of as learning in an MDP: take $\gamma = 0$ (i.e. the effective horizon is 1); suppose we have a distribution of

[1] Such impossibility results are often referred to as a "No free lunch theorem" theorems.

starting states $s_0 \sim \mu$; suppose $|\mathcal{A}| = 2$; and the reward function is $r(s,a) = \mathbf{1}(\text{label}(s) = a)$. In other words, we equate our action with the prediction of the binary class, and the reward function is 1 or 0, determined by if our prediction is correct.

Sampling model In this chapter, we consider a weaker (and more realistic) sampling model where we have a starting state distribution μ over states. We assume sampling access to the MDP where we start at a state $s_0 \sim \mu$; we can rollout a policy π of our choosing; and we can terminate the trajectory at will. We are interested in learning with a small number of observed trajectories.

4.1 Review: Binary Classification and Generalization

One of the most important concepts for learning binary classifiers is that it is possible to *generalize* even when the state space is infinite. Here note that the domain of our classifiers, often denoted by $\mathcal{X}$, is analogous to the state space $\mathcal{S}$. We now briefly review some basics of supervised learning before we turn to the question of generalization in reinforcement learning.

Consider the problem of binary classification with N labeled examples of the form $(x_i, y_i)_{i=1}^N$, with $x_i \in \mathcal{X}$ and $y_i \in \{0,1\}$. Suppose we have a (finite or infinte) set $\mathcal{H}$ of binary classifiers where each $h \in \mathcal{H}$ is a mapping of the form $h : \mathcal{X} \to \{0,1\}$. Let $\mathbf{1}(h(x) \neq y)$ be an indicator which takes the value 0 if $h(x) = y$ and 1 otherwise. We assume that our samples are drawn i.i.d. according to a fixed joint distribution D over (x, y).

Define the empirical error and the true error as:

$$\widehat{\text{err}}(h) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(h(x_i) \neq y_i), \quad \text{err}(h) = \mathbb{E}_{(X,Y)\sim D}\mathbf{1}(h(X) \neq Y).$$

For a given $h \in \mathcal{H}$, Hoeffding's inequality implies that with probability at least $1 - \delta$:

$$|\text{err}(h) - \widehat{\text{err}}(h)| \leq \sqrt{\frac{1}{2N}\log\frac{2}{\delta}}.$$

This and the union bound give rise to what is often referred to as the "Occam's razor" bound:

Proposition 4.1. (The "Occam's razor" bound) Suppose $\mathcal{H}$ is finite. Let $\widehat{h} = \arg\min_{h\in\mathcal{H}} \widehat{\text{err}}(h)$ and $h^\star = \arg\min_{h\in\mathcal{H}} \text{err}(h)$. With probability at least $1 - \delta$:

$$\text{err}(\widehat{h}) - \text{err}(h^\star) \leq \sqrt{\frac{2}{N}\log\frac{2|\mathcal{H}|}{\delta}}.$$

Hence, provided that

$$N \geq \frac{c\log\frac{2|\mathcal{H}|}{\delta}}{\epsilon^2},$$

then with probability at least $1 - \delta$, we have that:

$$\text{err}(\widehat{h}) - \text{err}(h^\star) \leq \epsilon.$$

A key observation here is that the our regret — the regret is the left hand side of the above inequality — has *no dependence* on the size of $\mathcal{X}$ (i.e. $\mathcal{S}$) which may be infinite and is only logarithmic in the number of hypothesis in our class.

In the supervised learning setting, a crucial observation is that even though a hypothesis set $\mathcal{H}$ may be infinite, the number of possible behaviors of on a finite set of states is not necessarily exhaustive. Let us review the definition of

the VC dimension for a hypothesis set of boolean functions. We say that the set $\{x_1, x_2, \ldots x_d\}$ is shattered if there exists an $h \in \mathcal{H}$ that can realize any of the possible 2^d labellings. The *Vapnik–Chervonenkis* (VC) dimension is the size of the largest shattered set. If $d = VC(\mathcal{H})$, then the Sauer–Shelah lemma states the number of possible labellings on a set of N points by functions in $\mathcal{H}$ is at most $\left(\frac{eN}{d}\right)^d$. For $d << N$, this is much less than 2^N.

The following classical bound highlights how generalization is possible on infinite hypothesis classes with VC dimension.

Proposition 4.2. (VC dimension and generalization) Let $\widehat{h} = \arg\min_{h \in \mathcal{H}} \widehat{\text{err}}(h)$ and $h^\star = \arg\min_{h \in \mathcal{H}} \text{err}(h)$. Suppose $\mathcal{H}$ has a bounded VC dimension. For $m \geq \text{VC}(\mathcal{H})$, we have that with probability at least $1 - \delta$:

$$\text{err}(\widehat{h}) - \text{err}(h^\star) \leq \sqrt{\frac{c}{N}\left(\text{VC}(\mathcal{H}) \log \frac{2N}{\text{VC}(\mathcal{H})} + \log \frac{2}{\delta}\right)},$$

where c is an absolute constant

4.2 Generalization and Agnostic Learning in RL

Now consider the case where we have a set of policies Π (either finite or infinite). For example, Π could be a parametric set. Alternatively, we could have a set of parametric value functions $\mathcal{V} = \{f_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R} | \, \theta \in \mathbb{R}^d\}$, and Π could be the set of policies that are greedy with respect to values in $\mathcal{V}$.

The goal of agnostic learning can be formulated by the following optimization problem:

$$\max_{\pi \in \Pi} \mathbb{E}_{s_0 \sim \mu} V^\pi(s_0)$$

As before, we only hope to perform favorably against the best policy in Π. Recall that in our aforementioned sampling model we have the ability to obtain trajectories from $s_0 \sim \mu$ under policies of our choosing. As we have seen, agnostic learning is possible in the supervised learning setting, with regret bounds that have no dependence on the size of the domain — the size of domain is analogous to the size the state space $|\mathcal{S}|$.

4.2.1 Upper Bounds: Data Reuse and Importance Sampling

We now provide a reduction of RL to the supervised learning problem. The key issue is how to efficiently reuse data. Here, we will simply collect N trajectories by executing a policy which chooses samples uniformly at random; let π_{uar} denote this policy. For simplicity, we only consider deterministic policies.

The following shows how we can obtain a nearly unbiased estimate of the reward with this uniform policy:

Lemma 4.3. *(Near unbiased estimation of $V^\pi(s_0)$) We have that:*

$$|\mathcal{A}|^H \, \mathbb{E}_{\pi_{\text{uar}}}\left[\mathbf{1}\Big(\pi(s_0) = a_0, \, \ldots, \pi(s_H) = a_H\Big) \sum_{t=0}^{H} \gamma^t r(s_t, a_t) \Big| s_0\right] = \mathbb{E}_\pi\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) \Big| s_0\right].$$

(Truncation) We also have that:

$$|V^\pi(s_0) - \mathbb{E}_\pi\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t)\right]| \leq \gamma^H/(1-\gamma),$$

which implies that for $H = \frac{\log\left(1/\left(\epsilon(1-\gamma)\right)\right)}{1-\gamma}$ we will have an ϵ approximation to $V^\pi(s_0)$.

In other words, the estimated reward of π on a trajectory is nonzero only when π takes exactly identical actions to those taken by π_{uar} on the trajectory, in which case the estimated value of π is $|\mathcal{A}|^H$ times that of π_{uar}. Note the factor of $|\mathcal{A}|^H$, due to importance sampling, leads this being a high variance estimate. We will return to this point in the next section.

Proof: To be added... ■

Denote the n-th sampled trajectory by $(s_0^n, a_0^n, r_1^n, s_1^n, \ldots, s_H^n)$, where H is the cutoff time where the trajectory ends. We can then use following to estimate the γ-discounted reward of any given policy π:

$$\widehat{V}^\pi(s_0) = \frac{|\mathcal{A}|^H}{N} \sum_{n=1}^{N} \mathbf{1}\Big(\pi(s_0^n) = a_0^n, \ldots \pi(s_H^n) = a_H^n\Big) \sum_{t=0}^{H} \gamma^t r(s_t^n, a_t^n).$$

Proposition 4.4. (Generalization in RL) Suppose Π is a finite set of policies. Let $\widehat{\pi} = \arg\max_{\pi\in\Pi} \widehat{V}^\pi(s_0)$. Using $H = \frac{\log\left(2/\left(\epsilon(1-\gamma)\right)\right)}{1-\gamma}$ we have that with probability at least $1-\delta$:

$$V^{\widehat{\pi}}(s_0) \geq \arg\max_{\pi\in\Pi} V^\pi(s_0) - \frac{\epsilon}{2} - |\mathcal{A}|^H \sqrt{\frac{2}{N} \log \frac{2|\Pi|}{\delta}}.$$

Hence, provided that

$$N \geq |\mathcal{A}|^H \frac{c \log(2|\Pi|/\delta)}{\epsilon^2},$$

then with probability at least $1-\delta$, we have that:

$$V^{\widehat{\pi}}(s_0) \geq \arg\max_{\pi\in\Pi} V^\pi(s_0) - \epsilon.$$

This is the analogue of the Occam's razor bound for RL.

Importantly, the above shows that we can avoid dependence on the size of the state space, though this comes at the price of an exponential dependence on the horizon. As we see in the next section, this dependence is unavoidable (without making further assumptions).

With regards to infinite hypothesis classes of policies, extending the Occam's razor bound can be done with standard approaches from statistical learning theory. For example, consider the case where $|\mathcal{A}| = 2$, where Π is class of deterministic policies. Here, as each $\pi \in \Pi$ can be viewed as Boolean function, $\text{VC}(\Pi)$ is defined in the usual manner. Here, we have:

Proposition 4.5. (Bounded VC dimension) Suppose $|\mathcal{A}| = 2$ and that suppose Π has a bounded VC dimension. Let $\widehat{\pi} = \arg\max_{\pi\in\Pi} \widehat{V}^\pi(s_0)$. Using $H = \frac{\log\left(2/\left(\epsilon(1-\gamma)\right)\right)}{1-\gamma}$ and for $N \geq \text{VC}(\Pi)$, we have that with probability at least $1-\delta$:

$$V^{\widehat{\pi}}(s_0) \geq \arg\max_{\pi\in\Pi} V^\pi(s_0) - \frac{\epsilon}{2} - 2^H \sqrt{\frac{c}{N}\left(\text{VC}(\Pi) \log \frac{2N}{\text{VC}(\Pi)} + \log \frac{2}{\delta}\right)},$$

where c is an absolute constant.

We do not prove this result here, which follows a standard argument using results in statistical learning theory. The key observation here is that, the Sauer–Shelah lemma bounds the number of possible labellings on a set of N trajectories (each of length H) by $\left(\frac{eNH}{d}\right)^d$, where $d = \text{VC}(\Pi)$.

See Section 4.5.

4.2.2 Lower Bounds

Clearly, the drawback of these bounds are that they are exponential in the problem horizon. We now see that if we desire a sample complexity that scales with $O(\log |\Pi|)$, then an exponential dependence on the effective horizon is unavoidable, without making further assumptions.

An algorithm is a procedure which sequentially samples trajectories and then returns some policy π (we often say the algorithm is *proper* if it returns a $\pi \in \Pi$). An algorithm is deterministic if it executes a policy (to obtain a trajectory) in manner that is a deterministic function of the data that it has collected. We only consider deterministic algorithms in this section, which does not quantitatively change the conclusions.

First, let us present the following simple observation, which already shows that avoiding an $\exp(1/(1-\gamma))$ dependence is not possible.

Proposition 4.6. (Lower Bound for The Complete Policy Class) Suppose $|\mathcal{A}| = 2$ and $|\mathcal{S}| = 2^H$, where $H = \lfloor \frac{\log(2)}{1-\gamma} \rfloor$. Let Π be the set of all 2^H policies. There exists a family of MDPs such that if a deterministic algorithm $\mathcal{A}$ is guaranteed to find a policy π such that:

$$V^{\widehat{\pi}}(s_0) \geq \arg\max_{\pi \in \Pi} V^{\pi}(s_0) - 1/4.$$

then $\mathcal{A}$ must use $N \geq 2^H$ trajectories.

Observe that $\log |\Pi| = H \log(2)$, so this already rules out the possibility of logarithmic dependence on the size of the policy class, without having an exponential dependence on H. The proof is straightforward, where we consider a family of binary trees where the rewards are at one of the terminal leaf nodes.

Proof: Consider a family of deterministic MDPs, where each in each MDP the dynamics are specified by a binary tree of depth H, with $H = \lfloor \frac{\log(2)}{1-\gamma} \rfloor$ and where there is a reward at one of the terminal leaf nodes. Note that for setting of H, $\gamma^H \leq \exp(-(1-\gamma)H) \geq 1/2$. Since Π is the set of all 2^H policies, then we must check every leaf, in the worst case (due that our algorithm is deterministic). This completes the proof. ■

In the previous proposition, our policy class was the complete class. Often, we are dealing with policies class which are far more restrictive. Even in this case, the following proposition strengthens this lower bound to be applicable to *arbitrary* policy classes, showing that even here (if we seek no dependence on $|\mathcal{S}|$), we must either have exponential dependence on the effective horizon or we must exhaustively try what is the effective size of all our policies.

Proposition 4.7. (Lower Bound for an Arbitrary Policy Class) Define $H = \lfloor \frac{\log(2)}{1-\gamma} \rfloor$. Suppose $|\mathcal{A}| = 2$ and let Π be an arbitrary policy class. There exists a family of MDPs such that if a deterministic algorithm $\mathcal{A}$ is guaranteed to find a policy $\widehat{\pi}$ such that:

$$\mathbb{E}\left[V^{\widehat{\pi}}(s_0)\right] \geq \arg\max_{\pi \in \Pi} V^{\pi}(s_0) - \epsilon.$$

(where the expectation is with respect to the trajectories the algorithm observes) then $\mathcal{A}$ must use an expected number of trajectories N where

$$N \geq c \frac{\min\{2^H, 2^{\mathrm{VC}(\Pi)}\}}{\epsilon^2},$$

where c is a universal constant.

We can interpret $2^{\mathrm{VC}(\Pi)}$ is the effective the number of policies in our policy class (by the definition of the VC dimension, it is number of different behaviors in our policy set). Thus, requiring $O(2^{\mathrm{VC}(\Pi)})$ samples shows that, in the worst case, we are not able to effectively reuse data (as was the case in supervised learning), unless have an exponential dependence on the horizon.

Proof: We will only prove this result for $\epsilon = 1/4$, where we will see that we need

$$N \geq \min\{2^H, 2^{\mathrm{VC}(\Pi)}\}$$

By definition of the VC dimension, our policy class can exhibit $2^{\mathrm{VC}(\Pi)}$ distinct action sequences on $\mathrm{VC}(\Pi)$ states. Suppose $\mathrm{VC}(\Pi) \leq H$. Here, we can construct a binary tree where the set of distinct leaves visited by Π will be precisely equal to $2^{\mathrm{VC}(\Pi)}$. By placing a unit reward at one of these leaves, the algorithm will be forced to explore all of the leaves. If $\mathrm{VC}(\Pi) \leq H$, then exploring the full binary tree is necessary.

We leave the general case as an exercise for the reader. As a hint, consider two different types of leaf nodes: for all but one of the leaf nodes, we obtain unit reward with $1/2$ probability, and, if the remaining leaf node is reached, we obtain unit reward with $1/2 + \epsilon$ probability. ■

4.3 Interpretation: How should we study generalization in RL?

The above clearly shows that, without further assumptions, agnostic learning (in the standard supervised learning sense) is not possible in RL, unless we can tolerate an exponential dependence on the horizon $1/(1-\gamma)$. Note that agnostic learning is not about being (unconditionally) optimal, but only being competitive among some restricted (hopefully lower complexity) set of models. Regardless, even with this weaker success criterion, avoiding the exponential dependence on the effective horizon is simply not possible.

This motivates the study of RL to consider either stronger assumptions or means in which the agent can obtain side information. Three examples of approaches that we will consider in this book are:

- Structural (and Modelling) Assumptions: By making stronger assumptions about the world, we can move away from agnostic learning and escape the curse of dimensionality. We will see examples of this in Part 2.
- Distribution Dependent Results (and Distribution Shift): When we move to policy gradient methods (in Part 3), we will consider results which depend on given distribution of how we obtain samples. Here, we will make connections to transfer learning.
- Imitation learning and behavior cloning: here will consider models where the agent has input from, effectively, a teacher, and we will see how this alleviates the problem of curse of dimensionality.

4.4 Approximation Limits with Linearity Assumptions

Given our previous lower bounds and discussion, it is natural to consider making assumptions. A common assumption is that the Q-function (or value function) is a (nearly) linear function of some given features (our representation); this is a natural assumption to begin our study of *function approximation*. In practice, suche features are either hand-crafted or a pre-trained neural network that transforms a state-action pair to a d-dimensional embedding [2].

We now see that, even when we make such linearity assumptions, there are hard thresholds, on the worst case approximation error of our representation, that have to be satisfied in order for our linearity assumption to be helpful.

We now provide a lower bound on the approximation limits for value-based learning, when we have a approximate linear representation. Formally, the agent is given a feature extractor $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, which can be hand-crafted or a pre-trained neural network that transforms a state-action pair to a d-dimensional embedding. The following assumption states that the given feature extractor can be used to predict the Q-function (of *any* policy) with approximation error at most ϵ_{approx} linear function.

In this section, *we assume we are in the finite horizon (undiscounted) setting.*

[2] The more challenging question is to *learn* the features

Assumption 4.8 (Linear Value Function Approximation)**.** There exists $\epsilon_{\text{approx}} > 0$, such that for any $h \in [H]$ and any policy π, there exists $\theta_h^\pi \in \mathbb{R}^d$ such that for any $(s,a) \in \mathcal{S} \times \mathcal{A}$, $|Q_h^\pi(s,a) - \langle \theta_h, \phi(s,a) \rangle| \leq \epsilon_{\text{approx}}$.

Here ϵ_{approx} is the approximation error, which indicates the quality of the representation. If $\epsilon_{\text{approx}} = 0$, then all Q-functions can be perfectly represented by a linear function of $\phi(\cdot,\cdot)$. In general, as we increase the dimension of ϕ we expect that ϵ_{approx} becomes smaller , since larger dimension usually has more expressive power.

Later on, we will see that if ϵ_{approx} is 0, then sample efficient learning is possible (with a polynomial dependence on H and d, but no dependence on $|\mathcal{S}|$ and $|\mathcal{A}|$). The following theorem shows that such assumptions, necessarily, need ϵ_{approx} close to 0, else sample efficient learning is not possible, which is consistent with our agnostic learning lower bounds in this chapter. In particular, The following theorem shows when $\epsilon_{\text{approx}} = \Omega\left(\sqrt{\frac{H}{d}}\right)$, the agent needs to sample exponential number of trajectories to find a near-optimal policy.

Theorem 4.9 (Exponential Lower Bound for Value-based Learning)**.** *There exists a family of MDPs with $|\mathcal{A}| = 2$ and a feature extractor ϕ that satisfy Assumption 4.8, such that any algorithm that returns a $1/2$-optimal policy with probability 0.9 needs to sample $\Omega\left(\min\{|\mathcal{S}|, 2^H, \exp(d\epsilon_{approx}^2/16)\}\right)$ trajectories.*

We state the theorem without proof. The lower bound is again based on a the deterministic binary tree hard instance, with only one rewarding node (i.e. state) at a leaf. With no further assumptions, as before to find a $1/2$-optimal policy for such MDPs, the agent must enumerate all possible states in level $H-1$ to find the state with reward $R = 1$. Doing so intrinsically induces a sample complexity of $\Omega(2^H)$.

Th key idea of the proof is that we can construct a set of features so that Assumption 4.8 holds, and, yet, these features reveal no additional information to the learner (and, so, the previous lower bound still applies). The main idea in the construction uses the following fact regarding the ϵ-approximate rank of the identity matrix of size 2^H: this (large) identity matrix can be approximated to ϵ- accuracy (in the spectral norm) with a matrix of rank only $O(H\epsilon^2)$ [3]. In our context, this fact can be used to construct a set of features ϕ, all of which live in an $O(H\epsilon^2)$ dimensional subspace, where these features well approximate all 2^H value function; the crucial property here is that the features can be constructed with *no knowledge* of the actual reward function.

4.5 Bibliographic Remarks and Further Readings

The reduction from reinforcement learning to supervised learning was first introduced in [Kearns et al., 2000], which used a different algorithm (the "trajectory tree" algorithm), as opposed to the importance sampling approach presented here. [Kearns et al., 2000] made the connection to the VC dimension of the policy. The fundamental sample complexity tradeoff — between polynomial dependence on the size of the state space and exponential dependence on the horizon — was discussed in depth in [Kakade, 2003].

The approximation limits with linear function approximation are results from [Du et al., 2019].

[3] Such a result can be proven with the e Johnson-Lindenstrauss Lemma

Part 2

Strategic Exploration

Chapter 5

Multi-armed & Linear Bandits

For the case, where $\gamma = 0$ (or $H = 1$ in the undiscounted case), the problem of learning in an unknown MDP reduce to the multi-armed bandit problem. The basic algorithms and proof methodologies here are important to understand in their own right, due to that we will have to extend these with more sophisticated variants to handle the exploration-exploitation tradeoff in the more challenging reinforcement learning problem.

This chapter follows analysis of the LinUCB algorithm from the original proof in [Dani et al., 2008], with a simplified concentration analysis due to [Abbasi-Yadkori et al., 2011].

5.1 The K-Armed Bandit Problem

The setting is where we have K decisions (the "arms"), where when we play arm $i \in \{1, 2, \dots K\}$ we obtain a random reward r_i which has mean reward:

$$E[r_i] = \mu_i$$

where we assume $\mu_i \in [-1, 1]$.

Every iteration t, the learner will pick an arm $I_t \in [1, 2, \dots K]$. Our cumulative regret is defined as:

$$R_T = T \cdot \max_i \mu_i - \sum_{t=0}^{T-1} \mu_{I_t}$$

We denote $a^\star = \text{argmax}_i \, \mu_i$ as the optimal arm. We define gap $\Delta_a = \mu_{a^\star} - \mu(a)$ for any arm a.

Theorem 5.1. *There exists an algorithm such that with probability at least $1 - \delta$, we have:*

$$R_T = O\left(\min \left\{ \sqrt{KT \cdot \ln(TK/\delta)}, \sum_{a \neq a^\star} \frac{\ln(TK/\delta)}{\Delta_a} \right\} + K \right).$$

5.1.1 The Upper Confidence Bound (UCB) Algorithm

We summarize the upper confidence bound (UCB) algorithm in Alg. 1. For simplicity, we allocate the first K rounds to pull each arm once.

Algorithm 1 UCB

1: Play each arm once and denote received reward as r_a for all $a \in \{1, 2, \dots K\}$
2: **for** $t = 0 \to T - 1 - K$ **do**
3: Execute arm $I_t = \arg\max_{i \in [K]} \left(\hat{\mu}^t(i) + \sqrt{\frac{\log(TK/\delta)}{N^t(i)}} \right)$
4: Observe r_{I_t}
5: **end for**

where every iteration t, we main counts of each arm:

$$N^t(a) = 1 + \sum_{i=0}^{t-1} \mathbf{1}\{I_i = a\},$$

where I_t is the index of the arm that is picked by the algorithm at iteration t. We main the empirical mean for each arm as follows:

$$\widehat{\mu}^t(a) = \frac{1}{N^t(a)} \left(r_a + \sum_{i=0}^{t-1} \mathbf{1}\{I_i = a\} r_i \right).$$

Recall that r_a is the reward of arm a we got during the first K rounds.

We also main the upper confidence bound for each arm as follows:

$$\widehat{\mu}^t(a) + 2\sqrt{\frac{\ln(TK/\delta)}{N^t(a)}}.$$

The following lemma shows that this is a valid upper confidence bound with high probability.

Lemma 5.2 (Upper Confidence Bound)**.** *For all $t \in [0, \dots, T-1]$ and $a \in [1, 2, \dots K]$, we have that with probability at least $1 - \delta$,*

$$\left|\widehat{\mu}^t(a) - \mu_a\right| \leq 2\sqrt{\frac{\ln(TK/\delta)}{N^t(a)}}. \tag{0.1}$$

The proof of the above lemma uses Azuma-Hoeffding's inequality (Theorem A.2) for each arm a and iteration t and then apply a union bound over all T iterations and K arms.

Now we can conclude the proof of the main theorem.

Proof: Below we conditioned on the above Inequality 0.1 holds. This gives us the following optimism:

$$\mu_a \leq \widehat{\mu}^t(a) + 2\sqrt{\frac{\ln(TK/\delta)}{N^t(a)}}, \forall a, t.$$

Thus, we can upper bound the regret as follows:

$$\mu^\star - \mu_{I_t} \leq \widehat{\mu}^t(I_t) + 2\sqrt{\frac{\ln(TK/\delta)}{N^t(I_t)}} - \mu_{I_t} \leq 4\sqrt{\frac{\ln(TK/\delta)}{N^t(I_t)}}.$$

Sum over all iterations, we get:

$$\sum_{t=0}^{T-1} \mu^\star - \mu_{I_t} \leq 4\sqrt{\ln(TK/\delta)} \sum_{t=0}^{T-1} \sqrt{\frac{1}{N^t(I_t)}}$$
$$= 4\sqrt{\ln(TK/\delta)} \sum_a \sum_{i=1}^{N^T(a)} \frac{1}{\sqrt{i}} \leq 8\sqrt{\ln(TK/\delta)} \sum_a \sqrt{N^T(a)} \leq 8\sqrt{\ln(TK/\delta)} \sqrt{K \sum_a N^T(a)}$$
$$\leq 8\sqrt{\ln(TK/\delta)}\sqrt{KT}.$$

Note that our algorithm has regret K at the first K rounds.

On the other hand, if for each arm a, the gap $\Delta_a > 0$, then, we must have:

$$N^T(a) \leq \frac{4\ln(TK/\delta)}{\Delta_a^2}.$$

which is because after the UCB of an arm a is below $\mu^\star$, UCB algorithm will never pull this arm a again (the UCB of the $\mu^\star$ is no smaller than $\mu^\star$).

Thus for the regret calculation, we get:

$$\sum_{t=0}^{T-1} \mu^\star - \mu_{I_t} \leq \sum_{a \neq a^\star} N_k^T(a)\Delta_a = \sum_{a \neq a^\star} \frac{4\ln(TK/\delta)}{\Delta_a}.$$

Together with the fact that Inequality 0.1 holds with probability at least $1 - \delta$, we conclude the proof. ■

5.2 Linear Bandits: Handling Large Action Spaces

Let $D \subset \mathbb{R}^d$ be a compact (but otherwise arbitrary) set of decisions. On each round, we must choose a decision $x_t \in D$. Each such choice results in a reward $r_t \in [-1, 1]$.

We assume that, regardless of the history $\mathcal{H}$ of decisions and observed rewards, the conditional expectation of r_t is a fixed linear function, i.e. for all $x \in D$,

$$\mathbb{E}[r_t | x_t = x] = \mu^\star \cdot x \in [-1, 1],$$

where $x \in D$ is arbitrary. Here, observe that we have assumed the mean reward for any decision is bounded in $[-1, 1]$. Under these assumptions, the *noise sequence*,

$$\eta_t = r_t - \mu^\star \cdot x_t$$

is a martingale difference sequence.

The is problem is essentially a bandit version of a fundamental geometric optimization problem, in which the agent's feedback on each round t is only the observed reward r_t and where the agent does not know $\mu^\star$ apriori.

If $x_0, \ldots x_{T-1}$ are the decisions made in the game, then define the *cumulative regret* by

$$R_T = \mu^\star \cdot x^\star - \sum_{t=0}^{T-1} \mu^\star \cdot x_t$$

Algorithm 2 The Linear UCB algorithm

Input: λ, β_t

1: **for** $t = 0, 1 \ldots$ **do**

2: Execute

$$x_t = \text{argmax}_{x \in D} \max_{\mu \in \text{BALL}_t} \mu \cdot x$$

and observe the reward r_t.

3: Update BALL_{t+1} (as specified in Equation 0.2).

4: **end for**

where $x^\star \in D$ is an optimal decision for $\mu^\star$, i.e.

$$x^\star \in \text{argmax}_{x \in D} \, \mu^\star \cdot x$$

which exists since D is compact. Observe that if the mean $\mu^\star$ were known, then the optimal strategy would be to play $x^\star$ every round. Since the expected loss for each decision x equals $\mu^\star \cdot x$, the cumulative regret is just the difference between the expected loss of the optimal algorithm and the expected loss for the actual decisions x_t. By the Hoeffding-Azuma inequality (see Lemma A.2), the observed reward $\sum_{t=0}^{T-1} r_t$ will be close to their (conditional) expectations $\sum_{t=0}^{T-1} \mu^\star \cdot x_t$.

Since the sequence of decisions $x_1, \ldots, x_{T-1}$ may depend on the particular sequence of random noise encountered, R_T is a random variable. Our goal in designing an algorithm is to keep R_T as small as possible.

5.2.1 The LinUCB algorithm

LinUCB is based on "optimism in the face of uncertainty," which is described in Algorithm 2. At episode t, we use all previous experience to define an uncertainty region (an ellipse) BALL_t. The center of this region, $\widehat{\mu}_t$, is the solution of the following regularized least squares problem:

$$\begin{aligned} \widehat{\mu}_t &= \arg\min_{\mu} \sum_{\tau=0}^{t-1} \|\mu \cdot x_\tau - r_\tau\|_2^2 + \lambda \|\mu\|_2^2 \\ &= \Sigma_t^{-1} \sum_{\tau=0}^{t-1} r_\tau x_\tau, \end{aligned}$$

where λ is a parameter and where

$$\Sigma_t = \lambda I + \sum_{\tau=0}^{t-1} x_\tau x_\tau^\top, \text{ with } \Sigma_0 = \lambda I.$$

The shape of the region BALL_t is defined through the feature covariance Σ_t.

Precisely, the uncertainty region, or confidence ball, is defined as:

$$\text{BALL}_t = \left\{ (\widehat{\mu}_t - \mu^\star)^\top \Sigma_t (\widehat{\mu}_t - \mu^\star) \leq \beta_t \right\}, \tag{0.2}$$

where β_t is a parameter of the algorithm.

Computation. Suppose that we have an efficient linear optimization oracle, i.e. that we can efficiently solve the problem:

$$\max_{x \in D} \nu \cdot x$$

for any ν. Even with this, Step 2 of LinUCB may not be computationally tractable. For example, suppose that D is provided to us as a polytope, then the above oracle can be efficiently computed using linear programming, while LinUCB is an NP-hard optimization. Here, we can actually use a wider confidence region, where we can keep track of ℓ_1 ball which contains BALL_t. See Section 5.4 for further reading.

5.2.2 Upper and Lower Bounds

Our main result here is that we have sublinear regret with only a polynomial dependence on the dimension d and, importantly, no dependence on the cardinality of the decision space D, i.e. on $|D|$.

Theorem 5.3. *Suppose that the noise η_t is σ^2 sub-Gaussian [1], that $\|\mu^\star\| \leq W$, and that $\|x\| \leq B$ for all $x \in D$. Set $\lambda = \sigma^2/W^2$ and*

$$\beta_t := \sigma^2\Big(2 + 4d\log\left(1 + \frac{TB^2W^2}{d}\right) + 8\log(4/\delta)\Big).$$

We have that with probability greater than $1-\delta$, that (simultaneously) for all $T \geq 0$,

$$R_T \leq c\sigma\sqrt{T}\left(d\log\left(1 + \frac{TB^2W^2}{d\sigma^2}\right) + \log(4/\delta)\right)$$

where c is an absolute constant. In other words, we have that R_T is $O^\star(d\sqrt{T})$ with high probability.

The following shows that no algorithm can do better.

Theorem 5.4. *(Lower bound) There exists a distribution over linear bandit problems (i.e. a distribution over μ) with rewards the rewards being bounded by 1 in magnitude and $\sigma^2 \leq 1$, such that for every (randomized) algorithm, we have for $n \geq \max\{256, d^2/16\}$,*

$$\mathbb{E}_\mu \mathbb{E} R_T \geq \frac{1}{2500} d\sqrt{T}.$$

where the inner expectation is with respect to randomness in the problem and the algorithm.

5.3 LinUCB Analysis

In establishing the upper bounds there are two main propositions from which the upper bounds follow. The first is in showing that the confidence region is appropriate.

Proposition 5.5. (Confidence) Let $\delta > 0$. We have that

$$\Pr(\forall t,\ \mu^\star \in \text{BALL}_t) \geq 1 - \delta.$$

Section 5.3.2 is devoted to establishing this confidence bound. In essence, the proof seeks to understand the growth of the quantity $(\widehat{\mu}_t - \mu^\star)^\top \Sigma_t (\widehat{\mu}_t - \mu^\star)$.

The second main step in analyzing LinUCB is to show that as long as the aforementioned high-probability event holds, we have some control on the growth of the regret. Let us define

$$\text{regret}_t = \mu^\star \cdot x^* - \mu^\star \cdot x_t$$

which denotes the instantaneous regret.

The following bounds the sum of the squares of instantaneous regret.

[1] Roughly speaking, this say that tail probabilities of η_t decay no more slowly than a Gaussian distribution. If the noise is bounded, i.e. $|\eta_t| \leq B$

Proposition 5.6. (Sum of Squares Regret Bound) Suppose that $\|x\| \leq B$ for $x \in D$. Suppose β_t is increasing and larger than 1. For LinUCB, if $\mu^\star \in \text{BALL}_t$ for all t, then

$$\sum_{t=0}^{T-1} \text{regret}_t^2 \leq 4\beta_T d \log\left(1 + \frac{TB^2}{d\lambda}\right)$$

This is proven in Section 5.3.1. The idea of the proof involves a potential function argument on the log volume (i.e. the log determinant) of the "precision matrix" Σ_t (which tracks how accurate our estimates of $\mu^\star$ are in each direction). The proof involves relating the growth of this volume to the regret.

Using these two results we are able to prove our upper bound as follows:

Proof:[Proof of Theorem 5.3] By Propositions 5.5 and 5.6 along with the Cauchy-Schwarz inequality, we have, with probability at least $1 - \delta$,

$$R_T = \sum_{t=0}^{T-1} \text{regret}_t \leq \sqrt{T \sum_{t=0}^{T-1} \text{regret}_t^2} \leq \sqrt{4T\beta_T d \log\left(1 + \frac{TB^2}{d\lambda}\right)}.$$

The remainder of the proof follows from using our chosen value of β_T and algebraic manipulations (that $2ab \leq a^2 + b^2$). ■

We now provide the proofs of these two propositions.

5.3.1 Regret Analysis

In this section, we prove Proposition 5.6, which says that the sum of the squares of the instantaneous regrets of the algorithm is small, assuming the evolving confidence balls always contain the true mean $\mu^\star$. An important observation is that on any round t in which $\mu^\star \in \text{BALL}_t$, the instantaneous regret is at most the "width" of the ellipsoid in the direction of the chosen decision. Moreover, the algorithm's choice of decisions forces the ellipsoids to shrink at a rate that ensures that the sum of the squares of the widths is small. We now formalize this.

Unless explicitly stated, all norms refer to the ℓ_2 norm.

Lemma 5.7. *Let $x \in D$. If $\mu \in \text{BALL}_t$ and $x \in D$. Then*

$$|(\mu - \widehat{\mu}_t)^\top x| \leq \sqrt{\beta_t x^\top \Sigma_t^{-1} x}$$

Proof: By Cauchy-Schwarz, we have:

$$\begin{aligned}|(\mu - \widehat{\mu}_t)^\top x| &= |(\mu - \widehat{\mu}_t)^\top \Sigma_t^{1/2} \Sigma_t^{-1/2} x| = |(\Sigma_t^{1/2}(\mu - \widehat{\mu}_t))^\top \Sigma_t^{-1/2} x| \\ &\leq \|\Sigma_t^{1/2}(\mu - \widehat{\mu}_t)\| \|\Sigma_t^{-1/2} x\| = \|\Sigma_t^{1/2}(\mu - \widehat{\mu}_t)\| \sqrt{x^\top \Sigma_t^{-1} x} \leq \sqrt{\beta_t x^\top \Sigma_t^{-1} x}\end{aligned}$$

where the last inequality holds since $\mu \in \text{BALL}_t$. ■

Define

$$w_t := \sqrt{x_t^\top \Sigma_t^{-1} x_t}$$

which we interpret as the "normalized width" at time t in the direction of the chosen decision. We now see that the width, $2\sqrt{\beta_t} w_t$, is an upper bound for the instantaneous regret.

Lemma 5.8. *Fix $t \le T$. If $\mu^\star \in \text{BALL}_t$, then*

$$\text{regret}_t \le 2\min\left(\sqrt{\beta_t} w_t, 1\right) \le 2\sqrt{\beta_T}\min\left(w_t, 1\right)$$

Proof: Let $\widetilde{\mu} \in \text{BALL}_t$ denote the vector which minimizes the dot product $\widetilde{\mu}^\top x_t$. By choice of x_t, we have

$$\widetilde{\mu}^\top x_t = \max_{\mu \in \text{BALL}_t} \max_{x \in D} \mu^\top x \ge (\mu^\star)^\top x^*,$$

where the inequality used the hypothesis $\mu^\star \in \text{BALL}_t$. Hence,

$$\begin{aligned}\text{regret}_t &= (\mu^\star)^\top x^* - (\mu^\star)^\top x_t \le (\widetilde{\mu} - \mu^\star)^\top x_t \\ &= (\widetilde{\mu} - \widehat{\mu}_t)^\top x_t + (\widehat{\mu}_t - \mu^\star)^\top x_t \le 2\sqrt{\beta_t} w_t\end{aligned}$$

where the last step follows from Lemma 5.7 since $\widetilde{\mu}$ and $\mu^\star$ are in BALL_t. Since $r_t \in [-1, 1]$, regret_t is always at most 2 and the first inequality follows. The final inequality is due to that β_t is increasing and larger than 1. ■

The following two lemmas prove useful in showing that we can treat the log determinant as a potential function, where can bound the sum of widths independently of the choices made by the algorithm.

Lemma 5.9. *We have:*

$$\det \Sigma_T = \det \Sigma_0 \prod_{t=0}^{T-1} (1 + w_t^2).$$

Proof: By the definition of Σ_{t+1}, we have

$$\begin{aligned}\det \Sigma_{t+1} &= \det(\Sigma_t + x_t x_t^\top) = \det(\Sigma_t^{1/2}(I + \Sigma_t^{-1/2} x_t x_t^\top \Sigma_t^{-1/2})\Sigma_t^{1/2}) \\ &= \det(\Sigma_t)\det(I + \Sigma_t^{-1/2} x_t (\Sigma_t^{-1/2} x_t)^\top) = \det(\Sigma_t)\det(I + v_t v_t^\top),\end{aligned}$$

where $v_t := \Sigma_t^{-1/2} x_t$. Now observe that $v_t^\top v_t = w_t^2$ and

$$(I + v_t v_t^\top) v_t = v_t + v_t(v_t^\top v_t) = (1 + w_t^2) v_t$$

Hence $(1 + w_t^2)$ is an eigenvalue of $I + v_t v_t^\top$. Since $v_t v_t^\top$ is a rank one matrix, all other eigenvalues of $I + v_t v_t^\top$ equal 1. Hence, $\det(I + v_t v_t^\top)$ is $(1 + w_t^2)$, implying $\det \Sigma_{t+1} = (1 + w_t^2)\det \Sigma_t$. The result follows by induction. ■

Lemma 5.10. *("Potential Function" Bound) For any sequence $x_0, \ldots x_{T-1}$ such that, for $t < T$, $\|x_t\|_2 \le B$, we have:*

$$\log\left(\det \Sigma_{T-1} / \det \Sigma_0\right) = \log\det\left(I + \frac{1}{\lambda}\sum_{t=0}^{T-1} x_t x_t^\top\right) \le d\log\left(1 + \frac{TB^2}{d\lambda}\right).$$

Proof: Denote the eigenvalues of $\sum_{t=0}^{T-1} x_t x_t^\top$ as $\sigma_1, \ldots \sigma_d$, and note:

$$\sum_{i=1}^{d} \sigma_i = \text{Trace}\left(\sum_{t=0}^{T-1} x_t x_t^\top\right) = \sum_{t=0}^{T-1} \|x_t\|^2 \le TB^2.$$

Using the AM-GM inequality,

$$\begin{aligned}&\log\det\left(I + \frac{1}{\lambda}\sum_{t=0}^{T-1} x_t x_t^\top\right) = \log\left(\prod_{i=1}^{d}(1 + \sigma_i/\lambda)\right) \\ &= d\log\left(\prod_{i=1}^{d}(1 + \sigma_i/\lambda)\right)^{1/d} \le d\log\left(\frac{1}{d}\sum_{i=1}^{d}(1 + \sigma_i/\lambda)\right) \le d\log\left(1 + \frac{TB^2}{d\lambda}\right),\end{aligned}$$

which concludes the proof. ■

Finally, we are ready to prove that if $\mu^\star$ always stays within the evolving confidence region, then our regret is under control.

Proof:[Proof of Proposition 5.6] Assume that $\mu^\star \in \textsc{Ball}_t$ for all t. We have that:

$$\sum_{t=0}^{T-1} \text{regret}_t^2 \leq \sum_{t=0}^{T-1} 4\beta_t \min(w_t^2, 1) \leq 4\beta_T \sum_{t=0}^{T-1} \min(w_t^2, 1)$$
$$\leq 4\beta_T \sum_{t=0}^{T-1} \ln(1 + w_t^2) \leq 4\beta_T \log\Big(\det \Sigma_{T-1} / \det \Sigma_0\Big) = 4\beta_T d \log\left(1 + \frac{TB^2}{d\lambda}\right)$$

where the first inequality follow from By Lemma 5.8; the second from that β_t is an increasing function of t; the third uses that for $0 \leq y \leq 1$, $\ln(1+y) \geq y/2$; the final two inequalities follow by Lemmas 5.9 and 5.10. ■

5.3.2 Confidence Analysis

Proof:[Proof of Proposition 5.5] Since $r_\tau = x_\tau \cdot \mu^\star + \eta_\tau$, we have:

$$\widehat{\mu}_t - \mu^\star = \Sigma_t^{-1} \sum_{\tau=0}^{t-1} r_\tau x_\tau - \mu^\star = \Sigma_t^{-1} \sum_{\tau=0}^{t-1} x_\tau (x_\tau \cdot \mu^\star + \eta_\tau) - \mu^\star$$
$$= \Sigma_t^{-1} \left(\sum_{\tau=0}^{t-1} x_\tau (x_\tau)^\top\right) \mu^\star - \mu^\star + \Sigma_t^{-1} \sum_{\tau=0}^{t-1} \eta_\tau x_\tau = \lambda \Sigma_t^{-1} \mu^\star + \Sigma_t^{-1} \sum_{\tau=0}^{t-1} \eta_\tau x_\tau$$

For any $0 < \delta_t < 1$, using Lemma A.5, it holds with probability at least $1 - \delta_t$,

$$\sqrt{(\widehat{\mu}_t - \mu^\star)^\top \Sigma_t (\widehat{\mu}_t - \mu^\star)} = \|(\Sigma_t)^{1/2}(\widehat{\mu}_t - \mu^\star)\|$$
$$\leq \left\|\lambda \Sigma_t^{-1/2} \mu^\star\right\| + \left\|\Sigma_t^{-1/2} \sum_{\tau=0}^{t-1} \eta_\tau x_\tau\right\|$$
$$\leq \sqrt{\lambda}\|\mu^\star\| + \sqrt{2\sigma^2 \log\left(\det(\Sigma_t) \det(\Sigma^0)^{-1} / \delta_t\right)}.$$

where we have also used the triangle inequality and that $\|\Sigma_t^{-1}\| \leq 1/\lambda$.

We seek to lower bound $\Pr(\forall t,\, \mu^\star \in \textsc{Ball}_t)$. Note that at $t = 0$, by our choice of λ, we have that $\textsc{Ball}_0$ contains $W^\star$, so $\Pr(\mu^\star \notin \textsc{Ball}_0) = 0$. For $t \geq 1$, let us assign failure probability $\delta_t = (3/\pi^2)/t^2$ for the t-th event, which, using the above, gives us an upper bound on the sum failure probability as

$$1 - \Pr(\forall t,\, \mu^\star \in \textsc{Ball}_t) = \Pr(\exists t,\, \mu^\star \notin \textsc{Ball}_t) \leq \sum_{t=1}^{\infty} \Pr(\mu^\star \notin \textsc{Ball}_t) < \sum_{t=1}^{\infty} (1/t^2)(3/\pi^2) = 1/2.$$

This along with Lemma 5.10 completes the proof. ■

5.4 Bibliographic Remarks and Further Readings

The orignal multi-armed bandit model goes to back to [Robbins, 1952]. The linear bandit model was first introduced in [Abe and Long, 1999]. Our analysis of the LinUCB algorithm follows from the original proof in [Dani et al., 2008],

with a simplified concentration analysis due to [Abbasi-Yadkori et al., 2011]. The first sub-linear regret bound here was due to [Auer et al., 2002], which used a more complicated algorithm.

The lower bound we present is also due to [Dani et al., 2008], which also shows that LinUCB is minimax optimal.

with probability one, then we can take $\sigma^2 = B^2$.

Chapter 6

Strategic Exploration in Tabular MDPs

We now turn to how an agent acting in an unknown MDP can obtain a near-optimal reward over time. Compared with the previous setting with access to a generative model, we no longer have easy access to transitions at each state, but only have the ability to execute trajectories in the MDP. The main complexity this adds to the learning process is that the agent has to engage in exploration, that is, plan to reach new states where enough samples have not been seen yet, so that optimal behavior in those states can be learned.

Learning is in an episodic setting, where in every episode k, the learner acts for H step starting from a fixed starting state $s_0 \sim \mu$ and, at the end of the H-length episode, the state is reset. It is straightforward to extend this setting where the starting state is sampled from a distribution, i.e. $s_0 \sim \mu$. In particular, we will follow the episodic MDP model in Section 1.2

The goal of the agent is to minimize her expected cumulative regret over K episodes:

$$\text{Regret} := \mathbb{E}\left[KV^\star(s_0) - \sum_{k=0}^{K-1}\sum_{h=0}^{H-1} r(s_h^k, a_h^k)\right],$$

where the expectation is with respect to the randomness of the MDP environment and, possibly, any randomness of the agent's strategy.

In this chapter, we consider tabular MDPs where $\mathcal{S}$ and $\mathcal{A}$ are discrete. We denote $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$.

We will now present a sub-linear regret algorithm, UCB-Value Iteration. This chapter follows the proof in [Azar et al., 2017], with a number of simplifications, albeit with a worse sample complexity.

We denote V^π as the expected total reward of π, i.e., $V^\pi := \mathbb{E}_{s_0\sim\mu} V^\pi(s_0)$.

6.1 The UCB-VI algorithm

If the learner then executes π^k in the underlying MDP to generate a single trajectory $\tau^k = \{s_h^k, a_h^k\}_{h=0}^{H-1}$ with $a_h = \pi_h^k(s_h^k)$ and $s_{h+1}^k \sim P_h(\cdot|s_h^k, a_h^k)$. We first define some notations below. Consider the very beginning of episode k. We use the history information up to the end of episode $k-1$ (denoted as $\mathcal{H}_{<k}$) to form some statistics. Specifically,

Algorithm 3 UCBVI

Input: reward function r (assumed to be known), confidence parameters

1: **for** $k = 0 \dots K$ **do**
2: Compute $\widehat{P}_h^k$ as the empirical estimates, for all h (Eq. 0.1)
3: Compute reward bonus b_h^k for all h (Eq. 0.2)
4: Run Value-Iteration on $\{\widehat{P}_h^k, r + b_h^k\}_{h=0}^{H-1}$ (Eq. 0.3)
5: Set π^k as the returned policy of VI.
6: **end for**

we define:

$$N_h^k(s,a,s') = \sum_{i=1}^{k-1} \mathbf{1}\{(s_h^i, a_h^i, s_{h+1}^i) = (s,a,s')\},$$
$$N_h^k(s,a) = \sum_{i=1}^{k-1} \mathbf{1}\{(s_h^i, a_h^i) = (s,a)\}, \forall h, s, a.$$

Namely, we maintain counts of how many times s, a, s' and s, a are visited at time step h from the beginning of the learning process to the end of the episode $k-1$. We use these statistics to form an empirical model:

$$\widehat{P}_h^k(s'|s,a) = \frac{N_h^k(s,a,s')}{N_h^k(s,a)}, \forall h, s, a, s'. \tag{0.1}$$

We will also use the counts to define a *reward bonus*, denoted as $b_h(s,a)$ for all h, s, a. Denote $L := \ln(SAHK/\delta)$ (δ as usual represents the failure probability which we will define later). We define reward bonus as follows:

$$b_h^k(s,a) = H\sqrt{\frac{L}{N_h^k(s,a)}}. \tag{0.2}$$

With reward bonus and the empirical model, the learner uses *Value Iteration* on the empirical transition $\widehat{P}_h^k$ and the combined reward $r_h + b_h^k$. Starting at H (note that H is a fictitious extra step as an episode terminates at $H-1$), we perform dynamic programming all the way to $h = 0$:

$$\begin{aligned}
&\widehat{V}_H^k(s) = 0, \forall s, \\
&\widehat{Q}_h^k(s,a) = \min\left\{r_h(s,a) + b_h^k(s,a) + \widehat{P}_h^k(\cdot|s,a) \cdot \widehat{V}_{h+1}^k,\ H\right\}, \\
&\widehat{V}_h^k(s) = \max_a \widehat{Q}_h^k(s,a), \pi_h^k(s) = \operatorname{argmax}_a \widehat{Q}_h^k(s,a), \forall h, s, a.
\end{aligned} \tag{0.3}$$

Note that when using $\widehat{V}_{h+1}^k$ to compute $\widehat{Q}_h^k$, we truncate the value by H. This is because we know that due to the assumption that $r(s,a) \in [0,1]$, no policy's Q value will ever be larger than H.

Denote $\pi^k = \{\pi_0^k, \dots, \pi_{H-1}^k\}$. Learner then executes π^k in the MDP to get a new trajectory τ^k.

UCBVI repeats the above procedure for K episodes.

6.2 Analysis

We will prove the following theorem.

Theorem 6.1 (Regret Bound of UCBVI)**.** *UCBVI achieves the following regret bound:*

$$\textit{Regret} := \mathbb{E}\left[\sum_{k=0}^{K-1}\left(V^\star - V^{\pi^k}\right)\right] \leq 2H^2S\sqrt{AK\cdot\ln(SAH^2K^2)} = \widetilde{O}\left(H^2S\sqrt{AK}\right)$$

Remark While the above regret is sub-optimal, the algorithm we presented here indeed achieves a sharper bound in the leading term $\widetilde{O}(H^2\sqrt{SAK})$ [Azar et al., 2017], which gives the tight dependency bound on S, A, K. The dependency on H is not tight and tightening the dependency on H requires modifications to the reward bonus (use Bernstein inequality rather than Hoeffding's inequality for reward bonus design).

We prove the above theorem in this section.

We start with bounding the error from the learned model $\widehat{P}_h^k$.

Lemma 6.2 (State-action wise ℓ_1 model error)**.** *Fix $\delta \in (0,1)$. For all $k \in [0,\ldots,K-1], s \in \mathcal{S}, a \in \mathcal{A}, h \in [0,\ldots,H-1]$, with probability at least $1-\delta$, we have:*

$$\left\|\widehat{P}_h^k(\cdot|s,a) - P_h^\star(\cdot|s,a)\right\|_1 \leq \sqrt{\frac{S\ln(SAHK/\delta)}{N_h^k(s,a)}}.$$

The proof of the above lemma uses Proposition A.4 and a union bound over all s, a, n, h.

The following lemma is still about model error, but this time we consider an average model error.

Lemma 6.3 (State-action wise average model error)**.** *Fix $\delta \in (0,1)$. For all $k \in [1,\ldots,K-1], s \in \mathcal{S}, a \in \mathcal{A}, h \in [0,\ldots,H-1]$, and consider $V_h^\star : \mathcal{S} \to [0,H]$, with probability at least $1-\delta$, we have:*

$$\left|\widehat{P}_h^k(\cdot|s,a)\cdot V_{h+1}^\star - P_h^\star(\cdot|s,a)\cdot V_{h+1}^\star\right| \leq H\sqrt{\frac{\ln(SAHN/\delta)}{N_h^k(s,a)}}.$$

Proof: We provide a proof sketch. Consider a fixed s, a, k, h. We have:

$$\widehat{P}_h^k(\cdot|s,a)\cdot V_{h+1}^\star = \frac{1}{N_h^k(s,a)}\sum_{i=1}^{k-1}\mathbf{1}\{(s_h^i,a_h^k) = (s,a)\}V_{h+1}^\star(s_{h+1}^i).$$

Note that for any $(s_h^i, a_h^i) = (s,a)$, we have $\mathbb{E}\left[V_{h+1}^\star(s_{h+1}^i)|s_h^i,a_h^i\right] = P_h(\cdot|s,a)\cdot V_{h+1}^\star$. Thus, we can apply Hoeffding's inequality here to bound $\left|\widehat{P}_h^k(\cdot|s,a)\cdot V_{h+1}^\star - P_h(\cdot|s,a)\cdot V_{h+1}^\star\right|$. With a union bound over all s, a, k, h, we conclude the proof. ■

We denote the two inequalities in Lemma 6.2 and Lemma 6.3 as event $\mathcal{E}_{model}$. Note that the failure probability of $\mathcal{E}_{model}$ is at most 2δ. Below we condition on $\mathcal{E}_{model}$ being true (we deal with failure event at the very end).

Now we study the effect of reward bonus. Similar to the idea in multi-armed bandits, we want to pick a policy π^k, such that the value of π^k in under the combined reward $r_h + b_h^k$ and the empirical model $\widehat{P}_h^k$ is optimistic, i.e., we want $\widehat{V}_0^k(s_0) \geq V_0^\star(s_0)$ for all s_0. The following lemma shows that via reward bonus, we are able to achieve this optimism.

Lemma 6.4 (Optimism)**.** *Assume $\mathcal{E}_{model}$ is true. For all episode k, we have:*

$$\widehat{V}_0^k(s_0) \geq V_0^\star(s_0), \forall s_0 \in \mathcal{S};$$

where $\widehat{V}_h^k$ is computed based on VI in Eq. 0.3.

Proof: We prove via induction. At the additional time step H we have $\widehat{V}_H^k(s) = V_H^\star(s) = 0$ for all s.

Starting at $h+1$, and assuming we have $\widehat{V}_{h+1}^k(s) \geq V_{h+1}^\star(s)$ for all s, we move to h below.

Consider any $s, a \in \mathcal{S} \times \mathcal{A}$. First, if $Q_h^k(s,a) = H$, then we have $Q_h^k(s,a) \geq Q_h^\star(s,a)$.

$$\begin{aligned}
\widehat{Q}_h^k(s,a) - Q_h^\star(s,a) &= b_h^k(s,a) + \widehat{P}_h^k(\cdot|s,a) \cdot \widehat{V}_{h+1}^k - P_h^\star(\cdot|s,a) \cdot V_{h+1}^\star \\
&\geq b_h^k(s,a) + \widehat{P}_h^k(\cdot|s,a) \cdot V_{h+1}^\star - P_h^\star(\cdot|s,a) \cdot V_{h+1}^\star \\
&= b_h^k(s,a) + \left(\widehat{P}_h^k(\cdot|s,a) - P_h^\star(\cdot|s,a)\right) \cdot V_{h+1}^\star \\
&\geq b_h^k(s,a) - H\sqrt{\frac{\ln(SAHK/\delta)}{N_h^k(s,a)}} \geq 0.
\end{aligned}$$

where the first inequality is from the inductive hypothesis, and the last inequality uses Lemma 6.3.

From $\widehat{Q}_{h+1}^k$, one can finish the proof by showing $\widehat{V}_h^n(s) \geq V_h^\star(s), \forall s$. ■

Now we are ready to prove the main theorem.

Proof:[Proof of Theorem 7.9]

Let us consider episode k and denote $\mathcal{H}_{<k}$ as the history up to the end of episode $k-1$. We consider bounding $V^\star - V^{\pi^k}$. Using Optimism and the simulation lemma, we can get the following result:

$$V^\star - V^{\pi^k} \leq \widehat{V}_0^k(s_0) - V_0^{\pi^k}(s_0) \leq \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h \sim d_h^{\pi^k}} \left[b_h^k(s_h,a_h) + \left(\widehat{P}_h^k(\cdot|s_h,a_h) - P^\star(\cdot|s_h,a_h)\right) \cdot \widehat{V}_{h+1}^{\pi^k} \right] \quad (0.4)$$

We prove the above two inequalities in the lecture. We leave the proof of the above inequality (Eq 0.4 as an exercise for readers. Note that this is slightly different from the usual simulation lemma, as here we truncate $\widehat{V}$ by H during VI.

Under $\mathcal{E}_{model}$, we can bound $\left(\widehat{P}_h^k(\cdot|s_h,a_h) - P^\star(\cdot|s_h,a_h)\right) \cdot \widehat{V}_{h+1}^{\pi^k}$ (recall Lemma 6.2) with a Holder's inequality:

$$\begin{aligned}
\left|\left(\widehat{P}_h^k(\cdot|s_h,a_h) - P^\star(\cdot|s_h,a_h)\right) \cdot \widehat{V}_{h+1}^{\pi^k}\right| &\leq \left\|\widehat{P}_h^k(\cdot|s_h,a_h) - P^\star(\cdot|s_h,a_h)\right\|_1 \left\|\widehat{V}_{h+1}^{\pi^k}\right\|_\infty \\
&\leq H\sqrt{\frac{S\ln(SAKH/\delta)}{N_h^k(s,a)}}.
\end{aligned}$$

Hence, back to per-episode regret $V^\star - V^{\pi^k}$, we get:

$$\begin{aligned}
V^\star - V^{\pi^k} &\leq \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h \sim d_h^{\pi^k}} \left[b_h^k(s_h,a_h) + H\sqrt{S\ln(SAHK/\delta)/N_h^k(s_h,a_h)} \right] \\
&\leq \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h \sim d_h^{\pi^k}} \left[2H\sqrt{S\ln(SAHK/\delta)/N_h^k(s_h,a_h)} \right] \\
&= 2H\sqrt{\ln(SAHK/\delta)}\mathbb{E}\left[\sum_{h=0}^{H-1} \frac{1}{\sqrt{N_h^k(s_h^k,a_h^k)}} \Big| \mathcal{H}_{<k} \right],
\end{aligned}$$

where in the last term the expectation is taken with respect to the trajectory $\{s_h^k, a_h^k\}$ (which is generated from π^k) while conditioning on all history $\mathcal{H}_{<k}$ up to and including the end of episode $k-1$.

Now we sum all episodes together and take the failure event into consideration.

$$\begin{aligned}\mathbb{E}\left[\sum_{k=0}^{K-1} V^\star - V^{\pi^k}\right] &= \mathbb{E}\left[\mathbf{1}\{\mathcal{E}_{model}\}\left(\sum_{k=0}^{K-1} V^\star - V^{\pi^k}\right)\right] + \mathbb{E}\left[\mathbf{1}\{\overline{\mathcal{E}}_{model}\}\left(\sum_{k=0}^{K-1} V^\star - V^{\pi^k}\right)\right] \\ &\leq \mathbb{E}\left[\mathbf{1}\{\mathcal{E}_{model}\}\left(\sum_{k=0}^{K-1} V^\star - V^{\pi^k}\right)\right] + 2\delta KH \\ &\leq 2H\sqrt{S\ln(SAHK/\delta)}\mathbb{E}\left[\sum_{k=0}^{K-1}\sum_{h=0}^{H-1}\frac{1}{\sqrt{N_h^k(s_h^k,a_h^k)}}\right] + 2\delta KH\end{aligned}$$

We can bound the double summation term above using lemma 6.5. We can conclude that:

$$\mathbb{E}\left[\sum_{n=1}^{N} V^\star - V^{\pi^n}\right] \leq 4H^2S\sqrt{AN\ln(SAHN/\delta)} + 2\delta NH.$$

Now set $\delta = 1/NH$, we get:

$$\mathbb{E}\left[\sum_{n=1}^{N} V^\star - V^{\pi^n}\right] \leq 4H^2S\sqrt{AN\ln(SAH^2N^2)} + 2 = O\left(H^2S\sqrt{AN\ln(SAH^2N^2)}\right).$$

This concludes the proof of Theorem 7.9. ■

Lemma 6.5. *Consider arbitrary K sequence of trajectories $\tau^k = \{s_h^k, a_h^k\}_{h=0}^{H-1}$ for $k = 0, \ldots, K-1$. We have*

$$\sum_{k=0}^{K-1}\sum_{h=0}^{H-1}\frac{1}{\sqrt{N_h^k(s_h^k,a_h^k)}} \leq 2H\sqrt{SAK}.$$

Proof: We swap the order of the two summation above:

$$\begin{aligned}&\sum_{k=0}^{K-1}\sum_{h=0}^{H-1}\frac{1}{\sqrt{N_h^k(s_h^k,a_h^k)}} = \sum_{h=0}^{H-1}\sum_{k=0}^{K-1}\frac{1}{\sqrt{N_h^k(s_h^k,a_h^k)}} = \sum_{h=0}^{H-1}\sum_{s,a\in\mathcal{S}\times\mathcal{A}}\sum_{i=1}^{N_h^K(s,a)}\frac{1}{\sqrt{i}} \\ &\leq 2\sum_{h=0}^{H-1}\sum_{s,a\in\mathcal{S}\times\mathcal{A}}\sqrt{N_h^K(s,a)} \leq \sum_{h=0}^{H-1}\sqrt{SA\sum_{s,a}N_h^K(s,a)} = H\sqrt{SAK},\end{aligned}$$

where in the first inequality we use the fact that $\sum_{i=1}^{N} 1/\sqrt{i} \leq 2\sqrt{N}$, and in the second inequality we use CS inequality. ■

6.3 Bibliographic Remarks and Further Readings

The first provably correct PAC algorithm for reinformcent learning (which finds a near optimal policy) was due to Kearns and Singh [2002], which provided the E^3 algorithm; it achieves polynomial sample complexity in tabular MDPs. Brafman and Tennenholtz [2002] presents the Rmax algorithm which provides a refined PAC analysis over E^3.

Both are model based approaches [Kakade, 2003] improves the sample complexity to be $O(S^2A)$. Both E^3 and Rmax uses the concept of absorbing MDPs to achieve optimism and balance exploration and exploitation.

Jaksch et al. [2010] provides the first $O(\sqrt(T))$ regret bound, where T is the number of timesteps in the MDP (T is proportiona to K in our setting); this dependence on T is optimal. Subsequently, Azar et al. [2017], Dann et al. [2017] provide algorithms that, asymptotically, achieve minimax regret bound in tabular MDPs. By this, we mean that for sufficienlty large T (for $T \geq \Omega(|\mathcal{S}|^2)$), the results in Azar et al. [2017], Dann et al. [2017] obtain optimal dependencies on $|\mathcal{S}|$ and $|\mathcal{A}|$. The requirement that $T \geq \Omega(|\mathcal{S}|^2)$ before these bounds hold is essentially the requirement that non-trivial model accuracy is required. It is an opent question to remove this dependence.

Lower bounds are provided in [Dann and Brunskill, 2015, Osband and Van Roy, 2016, Azar et al., 2017].

Further exploration strategies. Refs and discussion for Q-learning and thompson sampling to be added...

Chapter 7

Linearly Parameterized MDPs

In this chapter, we consider learning and exploration in linearly parameterized MDPs—the linear MDP. Linear MDP generalizes tabular MDPs into MDPs with potentially infinitely many state and action pairs.

This chapter follows largely follows the model and analysis first provided in [Jin et al., 2020].

7.1 Setting

We consider episodic finite horizon MDP with horizon H, $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \{r_h\}_h, \{P_h\}_h, H, s_0\}$, where s_0 is a fixed initial state, $r_h : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ and $P_h : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$ are time-dependent reward function and transition kernel. Note that for time-dependent finite horizon MDP, the optimal policy will be time-dependent as well. For simplicity, we overload notations a bit and denote $\pi = \{\pi_0, \ldots, \pi_{H-1}\}$, where each $\pi_h : \mathcal{S} \mapsto \mathcal{A}$. We also denote $V^\pi := V_0^\pi(s_0)$, i.e., the expected total reward of π starting at $h = 0$ and s_0.

We define the learning protocol below. Learning happens in an episodic setting. Every episode k, learner first proposes a policy π^k based on all the history information up to the end of episode $k-1$. The learner then executes π^k in the underlying MDP to generate a single trajectory $\tau^k = \{s_h^k, a_h^k\}_{h=0}^{H-1}$ with $a_h = \pi_h^k(s_h^k)$ and $s_{h+1}^k \sim P_h(\cdot|s_h^k, a_h^k)$. The goal of the learner is to minimize the following cumulative regret over N episodes:

$$\text{Regret} := \mathbb{E}\left[\sum_{k=0}^{K-1}\left(V^\star - V^{\pi^k}\right)\right],$$

where the expectation is with respect to the randomness of the MDP environment and potentially the randomness of the learner (i.e., the learner might make decisions in a randomized fashion).

7.1.1 Low-Rank MDPs and Linear MDPs

Note that here we do not assume $\mathcal{S}$ and $\mathcal{A}$ are finite anymore. Indeed in this note, both of them could be continuous. Without any further structural assumption, the lower bounds we saw in the Generalization Lecture forbid us to get a polynomially regret bound.

The structural assumption we make in this note is a linear structure in both reward and the transition.

Definition 7.1 (Linear MDPs)**.** Consider transition $\{P_h\}$ and $\{r_h\}_h$. A linear MDP has the following structures on r_h

and P_h:

$$r_h(s,a) = \theta_h^\star \cdot \phi(s,a), \quad P_h(\cdot|s,a) = \mu_h^\star \phi(s,a), \forall h$$

where ϕ is a known state-action feature map $\phi : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}^d$, and $\mu_h^\star \in \mathbb{R}^{|\mathcal{S}| \times d}$. Here $\phi, \theta_h^\star$ are **known** to the learner, while $\mu^\star$ is unknown. We further assume the following norm bound on the parameters: (1) $\sup_{s,a} \|\phi(s,a)\|_2 \leq 1$, (2) $\|v^\top \mu_h^\star\|_2 \leq \sqrt{d}$ for any v such that $\|v\|_\infty \leq 1$, and all h, and (3) $\|\theta_h^\star\|_2 \leq W$ for all h. We assume $r_h(s,a) \in [0,1]$ for all h and s, a.

The model essentially says that the transition matrix $P_h \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}||\mathcal{A}|}$ has rank at most d, and $P_h = \mu_h^\star \Phi$. where $\Phi \in \mathbb{R}^{d \times |\mathcal{S}||\mathcal{A}|}$ and each column of Φ corresponds to $\phi(s,a)$ for a pair $s, a \in \mathcal{S} \times \mathcal{A}$.

Linear Algebra Notations For real-valued matrix A, we denote $\|A\|_2 = \sup_{x:\|x\|_2=1} \|Ax\|_2$ which denotes the maximum singular value of A. We denote $\|A\|_F$ as the Frobenius norm $\|A\|_F^2 = \sum_{i,j} A_{i,j}^2$ where $A_{i,j}$ denotes the i,j'th entry of A. For any Positive Definite matrix Λ, we denote $x^\top \Lambda x = \|x\|_\Lambda^2$. We denote $\det(A)$ as the determinant of the matrix A. For a PD matrix Λ, we note that $\det(\Lambda) = \prod_{i=1}^d \sigma_i$ where σ_i is the eigenvalues of Λ. For notation simplicity, during inequality derivation, we will use $\lesssim, \eqsim$ to suppress all absolute constants. We will use $\widetilde{O}$ to suppress all absolute constants and log terms.

7.2 Planning in Linear MDPs

We first study how to do value iteration in linear MDP if μ is given.

We start from $Q_{H-1}^\star(s,a) = \theta_{H-1}^\star \cdot \phi(s,a)$, and $\pi_{H-1}^\star(s) = \operatorname{argmax}_a Q_{H-1}^\star(s,a) = \operatorname{argmax}_a \theta_{H-1}^\star \cdot \phi(s,a)$, and $V_{H-1}^\star(s) = \operatorname{argmax}_a Q_{H-1}^\star(s,a)$.

Now we do dynamic programming from $h+1$ to h:

$$Q_h^\star(s,a) = \theta_h^\star \cdot \phi(s,a) + \mathbb{E}_{s \sim P_h(\cdot|s,a)} V_{h+1}^\star(s') = \theta_h^\star \cdot \phi(s,a) + P_h(\cdot|s,a) \cdot V_{h+1}^\star = \theta_h^\star \cdot \phi(s,a) + (\mu_h^\star \phi(s,a))^\top V_{h+1}^\star \tag{0.1}$$

$$= \phi(s,a) \cdot \left(\theta_h^\star + (\mu_h^\star)^\top V_{h+1}^\star\right) = \phi(s,a) \cdot w_h, \tag{0.2}$$

where we denote $w_h := \theta_h^\star + (\mu_h^\star)^\top V_{h+1}^\star$. Namely we see that $Q_h^\star(s,a)$ is a linear function with respect to $\phi(s,a)$! We can continue by defining $\pi_h^\star(s) = \operatorname{argmax}_a Q_h^\star(s,a)$ and $V_h^\star(s) = \max_a Q_h^\star(s,a)$.

At the end, we get a sequence of linear $Q^\star$, i.e., $Q_h^\star(s,a) = w_h \cdot \phi(s,a)$, and the optimal policy is also simple, $\pi_h^\star(s) = \operatorname{argmax}_a w_h \cdot \phi(s,a)$, for all $h = 0, \ldots, H-1$.

One key property of linear MDP is that a Bellman Backup of any function $f : \mathcal{S} \mapsto \mathbb{R}$ is a linear function with respect to $\phi(s,a)$. We summarize the key property in the following claim.

Claim 7.2. *Consider any arbitrary function $f : \mathcal{S} \mapsto [0,H]$. At any time step $h \in [0, \ldots H-1]$, there must exist a $w \in \mathbb{R}^d$, such that, for all $s, a \in \mathcal{S} \times \mathcal{A}$:*

$$r_h(s,a) + P_h(\cdot|s,a) \cdot f = w^\top \phi(s,a).$$

The proof of the above claim is essentially the Eq. 0.1.

7.3 Learning Transition using Ridge Linear Regression

In this section, we consider the following simple question: given a dataset of state-action-next state tuples, how can we learn the transition P_h for all h?

Note that $\mu^\star \in \mathbb{R}^{|\mathcal{S}| \times d}$. Hence explicitly writing down and storing the parameterization $\mu^\star$ takes time at least $|\mathcal{S}|$. We show that we can represent the model in a non-parametric way.

We consider a particular episode n. Similar to Tabular-UCBVI, we learn a model at the very beginning of the episode n using all data from the previous episodes (episode 1 to the end of the episode $n-1$). We denote such dataset as:

$$\mathcal{D}_h^n = \left\{ s_h^i, a_h^i, s_{h+1}^i \right\}_{i=0}^{n-1}.$$

We maintain the following statistics using $\mathcal{D}_h^n$:

$$\Lambda_h^n = \sum_{i=0}^{n-1} \phi(s_h^i, a_h^i) \phi(s_h^i, a_h^i)^\top + \lambda I,$$

where $\lambda \in \mathbb{R}^+$ (it will be set to 1 eventually, but we keep it here for generality).

To get intuition of Λ^n, think about the tabular setting where $\phi(s,a)$ is a one-hot vector (zeros everywhere except that the entry corresponding to (s,a) is one). Then Λ_h^n is a diagonal matrix and the diagonal entry contains $N^n(s,a)$—the number of times (s,a) has been visited.

We consider the following multi-variate linear regression problem. Denote $\delta(s)$ as a one-hot vector that has zero everywhere except that the entry corresponding to s is one. Denote $\epsilon_h^i = P(\cdot|s_h^i, a_h^i) - \delta(s_{h+1}^i)$. Conditioned on history $\mathcal{H}_h^i$ (history $\mathcal{H}_h^i$ denotes all information from the very beginning of the learning process up to and including (s_h^i, a_h^i)), we have:

$$\mathbb{E}\left[\epsilon_h^i | \mathcal{H}_h^i\right] = 0,$$

simply because s_{h+1}^i is sampled from $P_h(\cdot|s_h^i, a_h^i)$ conditioned on (s_h^i, a_h^i). Also note that $\|\epsilon_h^i\|_1 \leq 2$ for all h, i.

Since $\mu_h^\star \phi(s_h^i, a_h^i) = P_h(\cdot|s_h^i, a_h^i)$, and $\delta(s_{h+1}^i)$ is an unbiased estimate of $P_h(\cdot|s_h^i, a_h^i)$ conditioned on s_h^i, a_h^i, it is reasonable to learn $\mu^\star$ via regression from $\phi(s_h^i, a_h^i)$ to $\delta(s_{h+1}^i)$. This leads us to the following ridge linear regression:

$$\widehat{\mu}_h^n = \text{argmin}_{\mu \in \mathbb{R}^{|\mathcal{S}| \times d}} \sum_{i=0}^{n-1} \left\| \mu \phi(s_h^i, a_h^i) - \delta(s_{h+1}^i) \right\|_2^2 + \lambda \|\mu\|_F^2.$$

Ridge linear regression has the following closed-form solution:

$$\widehat{\mu}_h^n = \sum_{i=0}^{n-1} \delta(s_{h+1}^i) \phi(s_h^i, a_h^i)^\top (\Lambda_h^n)^{-1} \tag{0.3}$$

Note that $\widehat{\mu}_h^n \in \mathbb{R}^{|\mathcal{S}| \times d}$, so we never want to explicitly store it. Note that we will always use $\widehat{\mu}_h^n$ together with a specific s, a pair and a value function V (think about value iteration case), i.e., we care about $\widehat{P}_h^n(\cdot|s,a) \cdot V := (\widehat{\mu}_h^n \phi(s,a)) \cdot V$, which can be re-written as:

$$\widehat{P}_h^n(\cdot|s,a) \cdot V := (\widehat{\mu}_h^n \phi(s,a)) \cdot V = \phi(s,a)^\top \sum_{i=0}^{n-1} (\Lambda_h^n)^{-1} \phi(s_h^i, a_h^i) V(s_{h+1}^i),$$

where we use the fact that $\delta(s)^\top V = V(s)$. Thus the operator $\widehat{P}_h^n(\cdot|s,a)\cdot V$ simply requires storing all data and can be computed via simple linear algebra and the computation complexity is simply $\mathrm{poly}(d,n)$—no poly dependency on $|\mathcal{S}|$.

Let us calculate the difference between $\widehat{\mu}_h^n$ and $\mu_h^\star$.

Lemma 7.3 (Difference between $\widehat{\mu}_h$ and $\mu_h^\star$)**.** *For all n and h, we must have:*

$$\widehat{\mu}_h^n - \mu_h^\star = -\lambda\mu_h^\star\left(\Lambda_h^n\right)^{-1} + \sum_{i=1}^{n-1}\epsilon_h^i\phi(s_h^i,a_h^i)^\top\left(\Lambda_h^n\right)^{-1}.$$

Proof: We start from the closed-form solution of $\widehat{\mu}_h^n$:

$$\begin{aligned}
\widehat{\mu}_h^n &= \sum_{i=0}^{n-1}\delta(s_{h+1}^i)\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1} = \sum_{i=0}^{n-1}(P(\cdot|s_h^i,a_h^i)+\epsilon_h^n)\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1}\\
&= \sum_{i=0}^{n-1}(\mu_h^\star\phi(s_h^i,a_h^i)+\epsilon_h^i)\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1} = \sum_{i=0}^{n-1}\mu_h^*\phi(s_h^i,a_h^i)\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1} + \sum_{i=0}^{n-1}\epsilon_h^i\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1}\\
&= \sum_{i=0}^{n-1}\mu_h^\star\phi(s_h^i,a_h^i)\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1} + \sum_{i=0}^{n-1}\epsilon_h^i\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1}\\
&= \mu_h^\star(\Lambda_h^n-\lambda I)(\Lambda_h^n)^{-1} + \sum_{i=0}^{n-1}\epsilon_h^i\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1} = \mu_h^* - \lambda\mu_h^\star(\Lambda_h^n)^{-1} + \sum_{i=0}^{n-1}\epsilon_h^i\phi(s_h^i,a_h^i)^\top(\Lambda_h^n)^{-1}.
\end{aligned}$$

Rearrange terms, we conclude the proof. ■

Lemma 7.4. *Fix $V:\mathcal{S}\mapsto[0,H]$. For all n and $s,a\in\mathcal{S}\times\mathcal{A}$, and h, with probability at least $1-\delta$, we have:*

$$\left\|\sum_{i=0}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|_{(\Lambda_h^n)^{-1}} \le 3H\sqrt{\ln\frac{H\det(\Lambda_h^n)^{1/2}\det(\lambda I)^{-1/2}}{\delta}}.$$

Proof: We first check the noise terms $\{V^\top\epsilon_h^i\}_{h,i}$. Since V is independent of the data (it's a pre-fixed function), and by linear property of expectation, we have:

$$\mathbb{E}\left[V^\top\epsilon_h^i|\mathcal{H}_h^i\right]=0,\quad |V^\top\epsilon_h^i| \le \|V\|_\infty\|\epsilon_h^i\|_1 \le 2H, \forall h,i.$$

Hence, this is a Martingale difference sequence. Using the Self-Normalized vector-valued Martingale Bound (Lemma A.5), we have that for all n, with probability at least $1-\delta$:

$$\left\|\sum_{i=0}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|_{(\Lambda_h^n)^{-1}} \le 3H\sqrt{\ln\frac{\det(\Lambda_h^n)^{1/2}\det(\lambda I)^{-1/2}}{\delta}}.$$

Apply union bound over all $h\in[H]$, we get that with probability at least $1-\delta$, for all n,h:

$$\left\|\sum_{i=0}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|_{(\Lambda_h^n)^{-1}} \le 3H\sqrt{\ln\frac{H\det(\Lambda_h^n)^{1/2}\det(\lambda I)^{-1/2}}{\delta}}. \tag{0.4}$$

■

7.4 Uniform Convergence via Covering

Now we take a detour first and consider how to achieve a uniform convergence result over a function class $\mathcal{F}$ that contains infinitely many functions. Previously we know how to get uniform convergence if $\mathcal{F}$ is finite—we simply do a union bound. However, when $\mathcal{F}$ contains infinitely many functions, we cannot simply apply a union bound. We will use the covering argument here.

Consider the following ball with radius R: $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2 \leq R \in \mathbb{R}^+\}$. Fix an ϵ. An ϵ-net $\mathcal{N}_\epsilon \subset \Theta$ is a set such that for any $\theta \in \Theta$, there exists a $\theta' \in \mathcal{N}_\epsilon$, such that $\|\theta - \theta'\|_2 \leq \epsilon$. We call the smallest ϵ-net as ϵ-cover. Abuse notations a bit, we simply denote $\mathcal{N}_\epsilon$ as the ϵ-cover.

The ϵ-covering number is the size of ϵ-cover $\mathcal{N}_\epsilon$. We define the covering dimension as $\ln(|\mathcal{N}_\epsilon|)$

Lemma 7.5. *The ϵ-covering number of the ball $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2 \leq R \in \mathbb{R}^+\}$ is upper bounded by $(1 + 2R/\epsilon)^d$.*

We can extend the above definition to a function class. Specifically, we look at the following function. For a triple of (w, β, Λ) where $w \in \mathbb{R}^d$ and $\|w\|_2 \leq L$, $\beta \in [0, B]$, and Λ such that $\sigma_{\min}(\Lambda) \geq \lambda$, we define $f_{w,\beta,\Lambda} : \mathcal{S} \mapsto [0, R]$ as follows:

$$f_{w,\beta,\Lambda}(s) = \min\left\{\max_a \left(w^\top \phi(s,a) + \beta\sqrt{\phi(s,a)^\top \Lambda^{-1}\phi(s,a)}\right),\ H\right\}, \forall s \in \mathcal{S}. \tag{0.5}$$

We denote the function class $\mathcal{F}$ as:

$$\mathcal{F} = \{f_{w,\beta,\Lambda} : \|w\|_2 \leq L, \beta \in [0, B], \sigma_{\min}(\Lambda) \geq \lambda\}. \tag{0.6}$$

Note that $\mathcal{F}$ contains infinitely many functions as the parameters are continuous. However we will show that it has finite covering number that scales exponentially with respect to the number of parameters in (w, β, Λ).

Why do we look at $\mathcal{F}$? As we will see later in this chapter $\mathcal{F}$ contains all possible $\widehat{Q}_h$ functions one could encounter during the learning process.

Lemma 7.6 (ϵ-covering dimension of $\mathcal{F}$)**.** *Consider $\mathcal{F}$ defined in Eq. 0.6. Denote its ϵ-cover as $\mathcal{N}_\epsilon$ with the ℓ_∞ norm as the distance metric, i.e., $d(f_1, f_2) = \|f_1 - f_2\|_\infty$ for any $f_1, f_2 \in \mathcal{F}$. We have that:*

$$\ln(|\mathcal{N}_\epsilon|) \leq d\ln(1 + 6L/\epsilon) + \ln(1 + 6B/(\sqrt{\lambda}\epsilon)) + d^2\ln(1 + 18B^2\sqrt{d}/(\lambda\epsilon^2)).$$

Note that the ϵ-covering dimension scales quadratically with respect to d.

Proof: We start from building a net over the parameter space (w, β, Λ), and then we convert the net over parameter space to an ϵ-net over $\mathcal{F}$ under the ℓ_∞ distance metric.

We pick two functions that corresponding to parameters (w, β, Λ) and $(\hat{w}, \hat{\beta}, \widehat{\Lambda})$.

$$
\begin{aligned}
|f(s) - \hat{f}(s)| &\le \left| \max_a \left(w^\top \phi(s,a) + \beta \sqrt{\phi(s,a)^\top \Lambda^{-1} \phi(s,a)} \right) - \max_a \left(\hat{w}^\top \phi(s,a) + \hat{\beta} \sqrt{\phi(s,a)^\top \hat{\Lambda}^{-1} \phi(s,a)} \right) \right| \\
&\le \max_a \left| \left(w^\top \phi(s,a) + \beta \sqrt{\phi(s,a)^\top \Lambda^{-1} \phi(s,a)} \right) - \left(\hat{w}^\top \phi(s,a) + \hat{\beta} \sqrt{\phi(s,a)^\top \hat{\Lambda}^{-1} \phi(s,a)} \right) \right| \\
&\le \max_a \left| (w - \hat{w})^\top \phi(s,a) \right| + \max_a \left| (\beta - \hat{\beta}) \sqrt{\phi(s,a)^\top \Lambda^{-1} \phi(s,a)} \right| \\
&\qquad + \max_a \left| \hat{\beta} (\sqrt{\phi(s,a)^\top \Lambda^{-1} \phi(s,a)} - \sqrt{\phi(s,a)^\top \hat{\Lambda}^{-1} \phi(s,a)}) \right| \\
&\le \|w - \hat{w}\|_2 + |\beta - \hat{\beta}| / \sqrt{\lambda} + B \sqrt{\left| \phi(s,a)^\top (\Lambda^{-1} - \hat{\Lambda}^{-1}) \phi(s,a) \right|} \\
&\le \|w - \hat{w}\|_2 + |\beta - \hat{\beta}| / \sqrt{\lambda} + B \sqrt{\|\Lambda^{-1} - \hat{\Lambda}^{-1}\|_F}
\end{aligned}
$$

Note that Λ^{-1} is a PD matrix with $\sigma_{\max}(\Lambda^{-1}) \le 1/\lambda$.

Now we consider the $\epsilon/3$-Net $\mathcal{N}_{\epsilon/3,w}$ over $\{w : \|w\|_2 \le L\}$, $\sqrt{\lambda}\epsilon/3$-net $\mathcal{N}_{\sqrt{\lambda}\epsilon/3,\beta}$ over interval $[0, B]$ for β, and $\epsilon^2/(9B^2)$-net $\mathcal{N}_{\epsilon^2/(9B),\Lambda}$ over $\{\Lambda : \|\Lambda\|_F \le \sqrt{d}/\lambda\}$. The product of these three nets provide a ϵ-cover for $\mathcal{F}$, which means that that size of the ϵ-net $\mathcal{N}_\epsilon$ for $\mathcal{F}$ is upper bounded as:

$$
\begin{aligned}
\ln |\mathcal{N}_\epsilon| &\le \ln |\mathcal{N}_{\epsilon/3,w}| + \ln |\mathcal{N}_{\sqrt{\lambda}\epsilon/3,\beta}| + \ln |\mathcal{N}_{\epsilon^2/(9B^2),\Lambda}| \\
&\le d \ln(1 + 6L/\epsilon) + \ln(1 + 6B/(\sqrt{\lambda}\epsilon)) + d^2 \ln(1 + 18B^2\sqrt{d}/(\lambda\epsilon^2)).
\end{aligned}
$$

■

Remark Covering gives a way to represent the complexity of function class (or hypothesis class). Relating to VC, covering number is upper bound roughly by $\exp(d)$ with d being the VC-dimension. However, there are cases where VC-dimensional is infinite, but covering number if finite.

Now we can build a uniform convergence argument for all $f \in \mathcal{F}$.

Lemma 7.7 (Uniform Convergence Results)**.** *Set $\lambda = 1$. Fix $\delta \in (0,1)$. For all n, h, all s, a, and all $f \in \mathcal{F}$, with probability at least $1 - \delta$, we have:*

$$
\left| \left(\widehat{P}_h^n(\cdot|s,a) - P(\cdot|s,a) \right) \cdot f \right| \lesssim H \|\phi(s,a)\|_{(\Lambda_h^n)^{-1}} \left(\sqrt{d \ln(1 + 6L\sqrt{N})} + d\sqrt{\ln(1 + 18B^2\sqrt{d}N)} + \sqrt{\ln \frac{H}{\delta}} \right).
$$

Proof: Recall Lemma 7.4, we have with probability at least $1 - \delta$, for all n, h, for a pre-fixed V (independent of the random process):

$$
\left\| \sum_{i=1}^{n-1} \phi(s_h^i, a_h^i)(V^\top \epsilon_h^i) \right\|_{(\Lambda_h^n)^{-1}}^2 \le 9H^2 \ln \frac{H \det(\Lambda_h^n)^{1/2} \det(\lambda I)^{-1/2}}{\delta} \le 9H^2 \left(\ln \frac{H}{\delta} + d \ln(1 + N) \right)
$$

where we have used the fact that $\|\phi\|_2 \le 1$, $\lambda = 1$, and $\|\Lambda_h^n\|_2 \le N + 1$.

Denote the ϵ-cover of $\mathcal{F}$ as $\mathcal{N}_\epsilon$. With an application of a union bound over all functions in $\mathcal{N}_\epsilon$, we have that with probability at least $1 - \delta$, for all $V \in \mathcal{N}_\epsilon$, all n, h, we have:

$$
\left\| \sum_{i=1}^{n-1} \phi(s_h^i, a_h^i)(V^\top \epsilon_h^i) \right\|_{(\Lambda_h^n)^{-1}}^2 \le 9H^2 \left(\ln \frac{H}{\delta} + \ln(|\mathcal{N}_\epsilon|) + d \ln(1 + N) \right).
$$

Recall Lemma 7.6, substitute the expression of $\ln|\mathcal{N}_\epsilon|$ into the above inequality, we get:

$$\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}\leq 9H^2\left(\ln\frac{H}{\delta}+d\ln(1+6L/\epsilon)+d^2\ln(1+18B^2\sqrt{d}/\epsilon^2)+d\ln(1+N)\right).$$

Now consider an arbitrary $f\in\mathcal{F}$. By the definition of ϵ-cover, we know that for f, there exists a $V\in\mathcal{N}_\epsilon$, such that $\|f-V\|_\infty\leq\epsilon$. Thus, we have:

$$\begin{aligned}\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(f^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}&\leq 2\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}+2\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)((V-f)^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}\\&\leq 2\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(V^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}+8\epsilon^2N\\&\leq 9H^2\left(\ln\frac{H}{\delta}+d\ln(1+6L/\epsilon)+d^2\ln(1+18B^2\sqrt{d}/\epsilon^2)+d\ln(1+N)\right)+8\epsilon^2N,\end{aligned}$$

where in the second inequality we use the fact that $\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(V-f)^\top\epsilon_h^i\|^2_{(\Lambda_h^n)^{-1}}\leq 4\epsilon^2N$, which is from

$$\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(V-f)^\top\epsilon_h^i\right\|^2_{(\Lambda_h^n)^{-1}}\leq\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)2\epsilon\|^2_{(\Lambda_h^n)^{-1}}\leq\frac{4\epsilon^2}{\lambda}\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)\|_2\leq 4\epsilon^2N.$$

Set $\epsilon=1/\sqrt{N}$, we get:

$$\begin{aligned}\left\|\sum_{i=1}^{n-1}\phi(s_h^i,a_h^i)(f^\top\epsilon_h^i)\right\|^2_{(\Lambda_h^n)^{-1}}&\leq 9H^2\left(\ln\frac{H}{\delta}+d\ln(1+6L\sqrt{N})+d^2\ln(1+18B^2\sqrt{d}N)+d\ln(1+N)\right)+8\\&\lesssim H^2\left(\ln\frac{H}{\delta}+d\ln(1+6L\sqrt{N})+d^2\ln(1+18B^2\sqrt{d}N)\right),\end{aligned}$$

where we recall $\lesssim$ ignores absolute constants.

Now recall that we can express $\left(\widehat{P}_h^n(\cdot|s,a)-P(\cdot|s,a)\right)\cdot f=\phi(s,a)^\top(\widehat{\mu}_h^n-\mu_h^\star)^\top f$. Recall Lemma 7.3, we have:

$$\begin{aligned}&|(\widehat{\mu}_h^n\phi(s,a)-\mu_h^\star\phi(s,a))\cdot f|\leq\left|\lambda\phi(s,a)^\top(\Lambda_h^n)^{-1}(\mu_h^\star)^\top f\right|+\left|\sum_{i=1}^{n-1}\phi(s,a)^\top(\Lambda_h^n)^{-1}\phi(s_h^i,a_h^i)(\epsilon_h^i)^\top f\right|\\&\lesssim H\sqrt{d}\,\|\phi(s,a)\|_{(\Lambda_h^n)^{-1}}+\|\phi(s,a)\|_{(\Lambda_h^n)^{-1}}\sqrt{\left(H^2\left(\ln\frac{H}{\delta}+d\ln(1+6L\sqrt{N})+d^2\ln(1+18B^2\sqrt{d}N)\right)\right)}\\&\eqsim H\,\|\phi(s,a)\|_{(\Lambda_h^n)^{-1}}\left(\sqrt{\ln\frac{H}{\delta}}+\sqrt{d\ln(1+6L\sqrt{N})}+d\sqrt{\ln(1+18B^2\sqrt{d}N)}\right).\end{aligned}$$

■

7.5 Algorithm

Our algorithm, Upper Confidence Bound Value Iteration (UCB-VI) will use reward bonus to ensure optimism. Specifically, we will the following reward bonus, which is motivated from the reward bonus used in linear bandit:

$$b_h^n(s,a) = \beta\sqrt{\phi(s,a)^\top(\Lambda_h^n)^{-1}\phi(s,a)}, \tag{0.7}$$

where β contains poly of H and d, and other constants and log terms. Again to gain intuition, please think about what this bonus would look like when we specialize linear MDP to tabular MDP.

Algorithm 4 UCBVI for Linear MDPs

1: **Input: parameters** β, λ
2: **for** $n = 1 \dots N$ **do**
3: Compute $\widehat{P}_h^n$ for all h (Eq. 0.3)
4: Compute reward bonus b_h^n for all h (Eq. 0.7)
5: Run Value-Iteration on $\{\widehat{P}_h^n, r_h + b_h^n\}_{h=0}^{H-1}$ (Eq. 0.8)
6: Set π^n as the returned policy of VI.
7: **end for**

With the above setup, now we describe the algorithm. Every episode n, we learn the model $\widehat{\mu}_h^n$ via ridge linear regression. We then form the quadratic reward bonus as shown in Eq. 0.7. With that, we can perform the following truncated Value Iteration (always truncate the Q function at H):

$$\begin{aligned}
&\widehat{V}_H^n(s) = 0, \forall s, \\
&\widehat{Q}_h^n(s,a) = \theta^\star \cdot \phi(s,a) + \beta\sqrt{\phi(s,a)^\top(\Lambda_h^n)^{-1}\phi(s,a)} + \phi(s,a)^\top(\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n \\
&\qquad = \beta\sqrt{\phi(s,a)^\top(\Lambda_h^n)^{-1}\phi(s,a)} + (\theta^\star + (\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n)^\top \phi(s,a), \\
&\widehat{V}_h^n(s) = \min\{\max_a \widehat{Q}_h^n(s,a),\ H\}, \quad \pi_h^n(s) = \operatorname{argmax}_a \widehat{Q}_h^n(s,a).
\end{aligned} \tag{0.8}$$

Note that above $\widehat{Q}_h^n$ contains two components: a quadratic component and a linear component. And $\widehat{V}_h^n$ has the format of $f_{w,\beta,\Lambda}$ defined in Eq. 0.5.

The following lemma bounds the norm of linear weights in $\widehat{Q}_h^n$.

Lemma 7.8. *Assume $\beta \in [0, B]$. For all n, h, we have $\widehat{V}_h^n$ is in the form of Eq. 0.5, and $\widehat{V}_h^n$ falls into the following class:*

$$\mathcal{V} = \{f_{w,\beta,\Lambda} : \|w\|_2 \le W + \frac{HN}{\lambda}, \beta \in [0, B], \sigma_{\min}(\Lambda) \ge \lambda\}. \tag{0.9}$$

Proof: We just need to show that $\theta^\star + (\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n$ has its ℓ_2 norm bounded. This is easy to show as we always have $\|\widehat{V}_{h+1}^n\|_\infty \le H$ as we do truncation at Value Iteration:

$$\left\|\theta^\star + (\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n\right\|_2 \le W + \left\|(\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n\right\|_2.$$

Now we use the closed-form of $\widehat{\mu}_h^n$ from Eq. 0.3:

$$\left\|(\widehat{\mu}_h^n)^\top \widehat{V}_{h+1}^n\right\|_2 = \left\|\sum_{i=1}^{n-1} \widehat{V}_{h+1}^n(s_{h+1}^i)\phi(s_h^i, a_h^i)^\top(\Lambda_h^n)^{-1}\right\|_2 \le H\left\|(\Lambda_h^n)^{-1}\sum_{i=0}^{n-1}\phi(s_h^i, a_h^i)\right\|_2 \le \frac{Hn}{\lambda},$$

where we use the fact that $\|\widehat{V}_{h+1}^n\|_\infty \le H$, $\sigma_{\max}(\Lambda^{-1}) \le 1/\lambda$, and $\sup_{s,a}\|\phi(s,a)\|_2 \le 1$. ■

7.6 Analysis of UCBVI for Linear MDPs

In this section, we prove the following regret bound for UCBVI.

Theorem 7.9 (Regret Bound). *Set $\beta = \widetilde{O}(Hd)$, $\lambda = 1$. UCBVI (Algorithm 4) achieves the following regret bound:*

$$\mathbb{E}\left[NV^\star - \sum_{i=0}^{N} V^{\pi^n}\right] \leq \widetilde{O}\left(H^2\sqrt{d^3N}\right)$$

The main steps of the proof are similar to the main steps of UCBVI in tabular MDPs. We first prove optimism via induction, and then we use optimism to upper bound per-episode regret. Finally we use simulation lemma to decompose the per-episode regret.

In this section, to make notation simple, we set $\lambda = 1$ directly.

7.6.1 Proving Optimism

Proving optimism requires us to first bound model error which we have built in the uniform convergence result shown in Lemma 7.7, namely, the bound we get for $(\widehat{P}_h^n(\cdot|s,a) - P(\cdot|s,a)) \cdot f$ for all $f \in \mathcal{V}$. Recall Lemma 7.7 but this time replacing $\mathcal{F}$ by $\mathcal{V}$ defined in Eq. 0.9. With probability at least $1-\delta$, for all n, h, s, a and for all $f \in \mathcal{V}$,

$$\begin{aligned}
\left|(\widehat{P}_h^n(\cdot|s,a) - P(\cdot|s,a)) \cdot f\right| &\leq H\left\|\phi(s,a)\right\|_{(\Lambda_h^n)^{-1}} \left(\sqrt{\ln\frac{H}{\delta}} + \sqrt{d\ln(1+6(W+HN)\sqrt{N})} + d\sqrt{\ln(1+18B^2\sqrt{d}N)}\right) \\
&\lesssim Hd\left\|\phi(s,a)\right\|_{(\Lambda_h^n)^{-1}} \left(\sqrt{\ln\frac{H}{\delta}} + \sqrt{\ln(WN+HN^2)} + \sqrt{\ln(B^2dN)}\right) \\
&\lesssim \left\|\phi(s,a)\right\|_{(\Lambda_h^n)^{-1}} \underbrace{Hd\left(\sqrt{\ln\frac{H}{\delta}} + \sqrt{\ln(W+H)} + \sqrt{\ln B} + \sqrt{\ln d} + \sqrt{\ln N}\right)}_{:=\beta}.
\end{aligned}$$

Denote the above inequality as event $\mathcal{E}_{model}$. Below we are going to condition on $\mathcal{E}_{model}$ being hold. Note that here for notation simplicity, we denote

$$\beta = Hd\left(\sqrt{\ln\frac{H}{\delta}} + \sqrt{\ln(W+H)} + \sqrt{\ln B} + \sqrt{\ln d} + \sqrt{\ln N}\right) = \widetilde{O}(Hd).$$

remark Note that in the definition of $\mathcal{V}$ (Eq. 0.9), we have $\beta \in [0, B]$. And in the above formulation of β, note that B appears inside a log term. So we need to set B such that $\beta \leq B$ and we can get the correct B by solving the inequality $\beta \leq B$ for B.

Lemma 7.10 (Optimism). *Assume event $\mathcal{E}_{model}$ is true. for all n and h,*

$$\widehat{V}_h^n(s) \geq V_h^\star(s), \forall s.$$

Proof: We consider a fixed episode n. We prove via induction. Assume that $\widehat{V}^n_{h+1}(s) \geq V^\star_{h+1}(s)$ for all s. For time step h, we have:

$$\begin{aligned}
&\hat{Q}^n_h(s,a) - Q^\star_h(s,a) \\
&= \theta^\star \cdot \phi(s,a) + \beta\sqrt{\phi(s,a)^\top(\Lambda^n_h)^{-1}\phi(s,a)} + \phi(s,a)^\top(\widehat{\mu}^n_h)^\top \widehat{V}^n_{h+1} - \theta^\star \cdot \phi(s,a) - \phi(s,a)^\top(\mu^\star_h)^\top V^\star_{h+1} \\
&\geq \beta\sqrt{\phi(s,a)^\top(\Lambda^n_h)^{-1}\phi(s,a)} + \phi(s,a)^\top \left(\widehat{\mu}^n_h - \mu^\star_h\right)^\top \widehat{V}^n_{h+1},
\end{aligned}$$

where in the last inequality we use the inductive hypothesis that $\widehat{V}^n_{h+1}(s) \geq V^\star_{h+1}(s)$, and $\mu^\star_h\phi(s,a)$ is a valid distribution (note that $\widehat{\mu}^n_h\phi(s,a)$ is not necessarily a valid distribution). We need to show that the bonus is big enough to offset the model error $\phi(s,a)^\top \left(\widehat{\mu}^n_h - \mu^\star_h\right)^\top \widehat{V}^n_{h+1}$. Since we have event $\mathcal{E}_{model}$ being true, we have that:

$$\left|(\widehat{P}^n_h(\cdot|s,a) - P(\cdot|s,a)) \cdot \widehat{V}^n_{h+1}\right| \lesssim \beta\|\phi(s,a)\|_{(\Lambda^n_h)^{-1}},$$

as by the construction of $\mathcal{V}$, we know that $\widehat{V}^n_{h+1} \in \mathcal{V}$.

This concludes the proof. ■

7.6.2 Regret Decomposition

Now we can upper bound the per-episode regret as follows:

$$V^\star - V^{\pi_n} \leq \widehat{V}^n_0(s_0) - V^{\pi_n}_0(s_0).$$

We can further bound the RHS of the above inequality using simulation lemma. Recall Eq. 0.4 that we derived in the note for tabular MDP (Chapter 6:

$$\widehat{V}^n_0(s_0) - V^{\pi_n}_0(s_0) \leq \sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d^{\pi_n}_h}\left[b^n_h(s,a) + \left(\widehat{P}^n_h(\cdot|s,a) - P(\cdot|s,a)\right)\cdot \widehat{V}^n_{h+1}\right].$$

(recall that the simulation lemma holds for any MDPs—it's not specialized to tabular).

In the event $\mathcal{E}_{model}$, we already know that for any s,a,h,n, we have $\left(\widehat{P}^n_h(\cdot|s,a) - P(\cdot|s,a)\right)\cdot\widehat{V}^n_{h+1} \lesssim \beta\|\phi(s,a)\|_{(\Lambda^n_h)^{-1}} = b^n_h(s,a)$. Hence, under $\mathcal{E}_{model}$, we have:

$$\widehat{V}^n_0(s_0) - V^{\pi_n}_0(s_0) \leq \sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d^{\pi_n}_h}\left[2b^n_h(s,a)\right] \lesssim \sum_{h=0}^{H-1} \mathbb{E}_{s,a\sim d^{\pi_n}_h}\left[b^n_h(s,a)\right].$$

Sum over all episodes, we have the following statement.

Lemma 7.11 (Regret Bound)**.** *Assume the event $\mathcal{E}_{model}$ holds. We have:*

$$\sum_{n=0}^{N-1}\left(V^\star_0(s_0) - V^{\pi_n}_0(s_0)\right) \leq \sum_{n=0}^{N-1}\sum_{h=0}^{H-1} \mathbb{E}_{s^n_h,a^n_h\sim d^{\pi_n}_h}\left[b^n_h(s^n_h,a^n_h)\right]$$

7.6.3 Concluding the Final Regret Bound

We first consider the following elliptical potential argument, which is similar to what we have seen in the linear bandit lecture.

Lemma 7.12 (Elliptical Potential)**.** *Consider an arbitrary sequence of state action pairs s_h^i, a_h^i. Assume* $\sup_{s,a} \|\phi(s,a)\|_2 \leq 1$. *Denote* $\Lambda_h^n = I + \sum_{i=0}^{n-1} \phi(s_h^i, a_h^i)\phi(s_h^i, a_h^i)^\top$. *We have:*

$$\sum_{i=0}^{N-1} \phi(s_h^i, a_h^i)(\Lambda_h^i)^{-1}\phi(s_h^i, a_h^i) \leq 2\ln\left(\frac{\det(\Lambda_h^{N+1})}{\det(I)}\right) \lesssim 2d\ln(N).$$

Proof: By the Lemma 3.7 and 3.8 in the linear bandit lecture note,

$$\begin{aligned}\sum_{i=1}^{N} \phi(s_h^i, a_h^i)(\Lambda_h^i)^{-1}\phi(s_h^i, a_h^i) &\leq 2\sum_{i=1}^{N} \ln(1 + \phi(s_h^i, a_h^i)(\Lambda_h^i)^{-1}\phi(s_h^i, a_h^i)) \\ &\leq 2\ln\left(\frac{\det(\Lambda_h^{N+1})}{\det(I)}\right) \\ &\leq 2d\ln(1 + \frac{N+1}{d\lambda}) \lesssim 2d\ln(N).\end{aligned}$$

where the first inequality uses that for $0 \leq y \leq 1$, $\ln(1+y) \geq y/2$. ■

Now we use Lemma 7.11 together with the above inequality to conclude the proof.

Proof:[Proof of main Theorem 7.9]

We split the expected regret based on the event $\mathcal{E}_{model}$.

$$\begin{aligned}\mathbb{E}\left[NV^\star - \sum_{n=1}^{N} V^{\pi_n}\right] &= \mathbb{E}\left[\mathbf{1}\{\mathcal{E}_{model} \text{ holds}\}\left(NV^\star - \sum_{n=1}^{N} V^{\pi_n}\right)\right] \\ &\quad + \mathbb{E}\left[\mathbf{1}\{\mathcal{E}_{model} \text{ doesn't hold}\}\left(NV^\star - \sum_{n=1}^{N} V^{\pi_n}\right)\right] \\ &\leq \mathbb{E}\left[\mathbf{1}\{\mathcal{E}_{model} \text{ holds}\}\left(NV^\star - \sum_{n=1}^{N} V^{\pi_n}\right)\right] + \delta NH \\ &\lesssim \mathbb{E}\left[\sum_{n=1}^{N}\sum_{h=0}^{H-1} b_h^n(s_h^n, a_h^n)\right] + \delta NH.\end{aligned}$$

Note that:

$$\begin{aligned}\sum_{n=1}^{N}\sum_{h=0}^{H-1} b_h^n(s_h^n, a_h^n) &= \beta\sum_{n=1}^{N}\sum_{h=0}^{H-1}\sqrt{\phi(s_h^n, a_h^n)^\top(\Lambda_h^n)^{-1}\phi(s_h^n, a_h^n)} \\ &= \beta\sum_{h=0}^{H-1}\sum_{n=1}^{N}\sqrt{\phi(s_h^n, a_h^n)^\top(\Lambda_h^n)^{-1}\phi(s_h^n, a_h^n)} \\ &\leq \beta\sum_{h=0}^{H-1}\sqrt{N\sum_{n=1}^{N}\phi(s_h^n, a_h^n)(\Lambda_h^n)^{-1}\phi(s_h^n, a_h^n)} \lesssim \beta H\sqrt{Nd\ln(N)}.\end{aligned}$$

Recall that $\beta = \widetilde{O}(Hd)$. This concludes the proof. ■

7.7 Bibliographic Remarks and Further Readings

There are number of ways to linearly parameterize an MDP such that it permits for efficient reinforcement learning (both statistically and computationally). The first observation that such assumptions lead to statistically efficient algorithms was due to [Jiang et al., 2017] due to that these models have low Bellman rank (as we shall see in Chapter 8). The first statistically and computationally efficient algorithm for a linearly parameterized MDP model was due to [Yang and Wang, 2019a,b]. Subsequently, [Jin et al., 2020] provided a computationally and statistically efficient algorithm for simplified version of this model, which is the model we consider here. The model of [Jia et al., 2020, Ayoub et al., 2020, Zhou et al., 2020] provides another linearly parameterized model, which can viewed as parameterizing $P(s'|s,a)$ as a linear combination of feature functions $\phi(s,a,s')$. One notable aspect of the model we choose to present here, where $P_h(\cdot|s,a) = \mu_h^\star \phi(s,a)$, is that this model has a number of free parameters that is $|\mathcal{S}| \cdot d$ (note that μ is unknown and is of size $|\mathcal{S}| \cdot d$), and yet the statistical complexity does not depend on $|\mathcal{S}|$. Notably, this implies that accurate model estimation request $O(|\mathcal{S}|)$ samples, while the regret for reinforcement learning is only polynomial in d. The linearly parameterized models of [Jia et al., 2020, Ayoub et al., 2020, Zhou et al., 2020] are parameterized by $O(d)$ parameters, and, while $O(d)$ free parameters suggests lower model capacity (where accurate model based estimation requires only polynomial in d samples), these models are incomparable to the linearly parameterized models presented in this chapter;

It is worth observing that all of these models permit statistically efficient estimation due to that they have bounded Belman rank [Jiang et al., 2017] (and bounded Witness rank [Sun et al., 2019]), a point which we return to in the next Chapter.

The specific linear model we consider here was originally introduced by [Jin et al., 2020]. The non-parametric model-based algorithm we study here was first introduced by [Lykouris et al., 2019] (but under the context of adversarial attacks).

The analysis we present here does not easily extend to infinite dimensional feature ϕ (e.g., RBF kernel); here, [Agarwal et al., 2020a] provide an algorithm and an analysis that extends to infinite dimensional ϕ, i.e. where we have a Reproducing Kernel Hilbert Space (RKHS) and the regret is based on the concept of Information Gain.

Chapter 8

Parametric Models with Bounded Bellman Rank

Our previous lectures on exploration in RL focused on the UCBVI algorithm designed for the tabular MDPs and Linear MDPs. While linear MDPs extends tabular MDPs to the function approximation regime, it is still limited in linear function approximation, and indeed the assumption that a Bellman backup of any function is still a linear function is a strong assumption. In this chapter, we consider the setting beyond tabular and linear representation. We aim to design algorithm with general function approximation that works for a large family of MDPs that subsumes not only tabular MDPs and linear MDPs, but also other models such as Linear function approximation with Bellman Completion (this generalizes linear MDPs), reactive predictive state representation (PSRs), and reactive Partially Observable Markov Decision Process (POMDPs).

8.1 Problem setting

We consider finite-horizon episodic time-dependent Markov Decision Process (MDP) $\mathcal{M} = (\{\mathcal{S}\}_h, \{\mathcal{A}\}_h, P_h, s_0, r, H)$ where $\mathcal{S}_h$ is the state space for time step h and we assume $\mathcal{S}_0, \mathcal{S}_2, \ldots, \mathcal{S}_{H-1}$ are disjoint ($\mathcal{A}_0, \mathcal{A}_2, \ldots, \mathcal{A}_{H-1}$ are disjoint as well). We assume $\mathcal{S}_0 = \{s_0\}$ though we can generalize to an arbitrary $\mathcal{S}_0$ with an initial state distribution.

Remark This setting indeed generalizes our previous finite horizon setting where we have a fixed $\mathcal{S}$ and $\mathcal{A}$, but time-dependent P_h and r_h, as we just need to add time step h into state space $\mathcal{S}$ to create $\mathcal{S}_h$, i.e., every state $s \in \mathcal{S}_h$ only contains h. The benefit of using the above slightly more general notation is that this allows us to ignore h in P, r, and in π, V and Q as now state action contains time step h.

The goal of an agent is to maximize the cumulative expected reward it obtains over H steps:

$$\max_{\pi} V^{\pi}(s_0).$$

In this chapter, we focus on a PAC (Probably Approximately Correct) guarantee. Namely, our goal is to find a policy $\hat{\pi}$ such that $V^{\pi}(s_0) \geq V^{\star}(s_0)$.

We make the following boundedness assumption on the rewards.

Assumption 8.1. Almost surely, for any trajectory τ and step h, $0 \leq r_h \leq 1$. Additionally, $0 \leq \sum_{h=0}^{H-1} r_h \leq 1$ almost surely for any trajectory τ.

While the first part of the assumption is the standard boundedness assumption we have made throughout, the second assumes that the trajectory level rewards are also bounded by 1, instead of H, which is helpful for capturing sparse-reward goal-directed problems with rewards only at one point in a successful trajectory. While normalization of the trajectory level reward also keeps the net reward bounded, this makes the total reward only scale as $1/H$ if the rewards are sparse along the trajectory.

8.2 Value-function approximation

We consider a model-free, value function based approach here. More specifically, denote $\mathcal{S} = \cup_h \mathcal{S}_h$ and $\mathcal{A} := \cup_h \mathcal{A}_h$, we assume that we are given the following function class:

$$\mathcal{F} = \{f : \mathcal{S} \times \mathcal{A} \to [0, 1]\}.$$

Since we want to learn a near-optimal behavior, we seek to approximate the Q-value function of the optimal policy, namely $Q^\star$ using $f \in \mathcal{F}$. To this end, we start with a simplifying assumption that $Q^\star$ lies in $\mathcal{F}$. In practice, this can be weakened to having a good approximation for $Q^\star$ in $\mathcal{F}$, but we focus on exact containment for the cleanest setting. Formally, we make the following *realizability* assumption.

Assumption 8.2 (Value-function realizability)**.** The function class $\mathcal{F}$ satisfies $Q^\star \in \mathcal{F}$.

Armed with this assumption, we may ask whether we can find $Q^\star$ using a number of samples which does not scale as $|\mathcal{X}|$, trading it off for a statistical complexity measure for $\mathcal{F}$ such as $\ln |\mathcal{F}|$. The next result, adapted from Krishnamurthy et al. [2016] shows that this is not possible.

Theorem 8.3. *Fix $H, K \in \mathbb{N}$ with $K \geq 2$ and $\epsilon \in (0, \sqrt{1/8}]$. For any algorithm, there exists an MDP with a horizon of H and K actions, a class of predictors $\mathcal{F}$ with $|\mathcal{F}| = K^H$ and $Q^\star \in \mathcal{F}$ and a constant $c > 0$ such that the probability that the algorithm outputs a policy $\widehat{\pi}$ with $V(\widehat{\pi}) \geq V^\star - \epsilon$ after collecting T trajectories from the MDP is at most $2/3$ for all $T \leq cK^H/\epsilon^2$.*

In words, the theorem says that for any algorithm, there exists an MDP where it cannot find a good policy in fewer than an exponential number of samples in the planning horizon, even when $Q^\star \in \mathcal{F}$ Furthermore, the size of the class $\mathcal{F}$ required for this result is K^H, so that a logarithmic dependence on $|\mathcal{F}|$ will not explain the lower bound. The lower bound construction basically uses the binary tree example that we have seen in the Generalization chapter.

The lower bound indicates that in order to learn in polynomial sample complexity, we need additional assumptions. While we have seen that polynomially sample complexity is possible in linear MDPs and tabular MDPs, our goal in this chapter is to significantly weaken the structural assumption on the MDPs.

8.3 Bellman Rank

Having concluded that we cannot find a near optimal policy using a reasonable number of samples with just the realizability assumption, it is clear that additional structural assumptions on the problem are required in order to make progress. We now give one example of such a structure, named Bellman rank, which was introduced by Jiang et al. [2017]. In order to motivate and define this quantity, we need some additional notation. For a function $f \in \mathcal{F}$, let us define $\pi_f(x) = \operatorname{argmax}_{a \in \mathcal{A}} f(x, a)$. Namely π_f is the greedy action selector with respect to f. For a policy π, function $f \in \mathcal{F}$ and $h \in [H]$, let us also define the *average Bellman error* of the function approximator f:

$$\mathcal{E}(f; \pi, h) = \mathbb{E}_{s_h, a_h \sim d_h^\pi} \left[f(s_h, a_h) - r_h - \mathbb{E}_{s_{h+1} \sim P(\cdot|s_h, a_h)} \max_{a \in \mathcal{A}_{h+1}} f(s_{h+1}, a) \right]. \tag{0.1}$$

This is called the average Bellman error as it is not the error on an individual state action s, a, but an expected error under the state-action distribution induced by the policy π. To see why the definition might be natural, we note that the following property of $Q^\star$ from the Bellman optimality equations.

Fact 8.4. $\mathcal{E}(Q^\star, \pi, h) = 0$, for all policies π and levels h.

The fact holds due to $Q^\star$ satisfying the Bellman optimality condition. We also have seen that for any f such that $f(s,a) = r(s,a) + \mathbb{E}_{s' \sim P(\cdot|s,a)} \max_{a'} f(s', a')$ for all $s, a \in \mathcal{S} \times \mathcal{A}$, then $f = Q^\star$.

Thus, if we ever discover a function f such that $\mathcal{E}(f; \pi, h) \neq 0$ for an arbitrary policy π and time step h, then we know that $f \neq Q^\star$. The average Bellman error allows us to detect wrong function approximators, i.e., f such that $f \neq Q^\star$.

We now make a structural assumption on average Bellman errors, which allows us to reason about the Bellman errors induced by all policies π_f in a sample-efficient manner. For any $h \in [H-1]$, let is define the ***Bellman error matrix*** $\mathcal{E}_h \in \mathbb{R}^{|\mathcal{F}| \times |\mathcal{F}|}$ as

$$[\mathcal{E}_h]_{g,f} = \mathcal{E}(f; \pi_g, h). \tag{0.2}$$

That is, each entry in the matrix captures the Bellman error of the function indexed by the column f under the greedy policy π_g induced by the row g at step h. With this notation, we define the Bellman rank of the MDP and a function class $\mathcal{F}$ below.

Definition 8.5 (Bellman Rank)**.** The Bellman rank of an MDP and a function class $\mathcal{F}$ is the smallest integer M such that $\text{rank}(\mathcal{E}_h) \leq M$ for all $h \in [H-1]$.

Intuitively, if the Bellman rank is small, then for any level h, the number of linearly independent rows is small. That is, the average Bellman error for any function under most policies can be expressed as linear combination of the Bellman errors of that function on a small set of policies corresponding to the linearly independent rows. Note that the definition presented here is a modification of the original definition from Jiang et al. [2017] in order to precisely capture the linear MDP example we covered before. [Jiang et al., 2017] showed that Bellman rank can be further upper bounded in terms of latent quantities such as the rank of the transition matrix, or the number of latent states if the MDP has an equivalent formulation as an MDP with a small number of latent states. We refer the reader to Jiang et al. [2017] for detailed examples, as well as connections of Bellman rank with other rank type notions in the RL literature to measure problem complexity.

8.3.1 Examples

Before moving on the the algorithm, we first present three examples that admit small Bellman Rank: tabular MDPs, linear MDPs, and Bellman Completion with Linear Function class. We note that the setting of Bellman Completion with Linear Function class subsumes linear MDPs, and linear MDPs subsumes tabular MDPs.

Tabular MDPs

For tabular MDPs, we can directly rewrite the Bellman error as follows. Focusing on an arbitrary h:

$$\begin{aligned}
\mathcal{E}(f; \pi_g, h) &= \sum_{s_h, a_h \in \mathcal{S}_h \times \mathcal{A}_h} d_h^{\pi_g}(s_h, a_h) \left(f(s_h, a_h) - r(s_h, a_h) - \mathbb{E}_{s_{h+1} \sim P(\cdot|s_h, a_h)} \max_a f(s_{h+1}, a) \right) \\
&= \left\langle d_h^{\pi_g}, f(\cdot, \cdot) - r(\cdot, \cdot) - \mathbb{E}_{s_{h+1} \sim P(\cdot|\cdot,\cdot)} \max_a f(s_{h+1}, a) \right\rangle.
\end{aligned}$$

Namely, $\mathcal{E}(f; \pi_g, h)$ can be written as an inner product of two vectors whose dimension is $|\mathcal{S}_h||\mathcal{A}_h|$. Note that in tabular MDPs, number of states and number of actions are the natural complexity quantity in sample complexity and regret bounds.

Linear MDPs

Recall the linear MDP definition. In linear MDPs, we know that $Q^\star$ is a linear function with respect to feature vector ϕ, i.e., $Q^\star(s_h, a_h) = (w^\star)^\top \phi(s_h, a_h)$. We will parameterize $\mathcal{F}$ to be the following linear function class:

$$\mathcal{F} = \{w^\top \phi(s, a) : w \in \mathbb{R}^d, \|w\|_2 \leq W\},$$

where we assume $\|w^\star\|_2 \leq W$. Recall that the transition P and reward are also linear, i.e., $P(\cdot|s_h, a_h) = \mu^\star \phi(s_h, a_h), r(s_h, a_h) = \phi(s_h, a_h)^\top \theta^\star$. In this case, for the average Bellman error, we have:

$$\begin{aligned}
\mathcal{E}(f; \pi_g, h) &= \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \left[w^\top \phi(s_h, a_h) - (\theta^\star)^\top \phi(s_h, a_h) - \mathbb{E}_{s_{h+1} \sim P(\cdot|s_h, a_h)} \max_{a \in \mathcal{A}_{h+1}} w^\top \phi(s_{h+1}, a) \right] \\
&= \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \left[w^\top \phi(s_h, a_h) - (\theta^\star)^\top \phi(s_h, a_h) - \phi(s_h, a_h)^\top (\mu^\star)^\top \left(\max_{a \in \mathcal{A}_{h+1}} w^\top \phi(\cdot, a) \right) \right] \\
&= \left\langle w - \theta^\star - (\mu^\star)^\top \left(\max_{a \in \mathcal{A}_{h+1}} w^\top \phi(\cdot, a) \right), \quad \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \phi(s_h, a_h) \right\rangle.
\end{aligned}$$

Namely, Bellman error is written as the inner product of two vectors whose dimension is d. Thus, Bellman rank is at most d for linear MDPs.

Bellman Completion in Linear Function Approximation

The last example we study here further generalizes linear MDPs. The setting *Bellman Completion in Linear Function Approximation* is defined as follows.

Definition 8.6 (Bellman Completion under Linear Function Class)**.** We assume $r(s_h, a_h) = \theta^\star \cdot \phi(s_h, a_h)$ and we consider the following linear function class $\mathcal{F} = \{w^\top \phi(s, a) : w \in \mathbb{R}^d, \|w\|_2 \leq W\}$. Recall the Bellman operator $\mathcal{T}$. We have Bellman completion under $\mathcal{F}$ if and only if for any $f \in \mathcal{F}$, $\mathcal{T} f \in \mathcal{F}$.

We can verify that linear MDP is a special instance here.

Consider $f(s, a) := w^\top \phi(s, a)$, and denote $\mathcal{T} f(s, a) := \widetilde{w}^\top \phi(s_h, a_h)$. We can rewrite the Bellman error as follows:

$$\begin{aligned}
\mathcal{E}(f; \pi_g, h) &= \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \left[w^\top \phi(s_h, a_h) - (\theta^\star)^\top \phi(s_h, a_h) - \mathbb{E}_{s_{h+1} \sim P(\cdot|s_h, a_h)} \max_{a \in \mathcal{A}_{h+1}} w^\top \phi(s_{h+1}, a) \right] \\
&= \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \left[w^\top \phi(s_h, a_h) - \widetilde{w}^\top \phi(s_h, a_h) \right] \\
&= \left\langle w - \widetilde{w}, \quad \mathbb{E}_{s_h, a_h \sim d_h^{\pi_g}} \left[\phi(s_h, a_h) \right] \right\rangle.
\end{aligned}$$

Namely, we can write the bellman error as an inner product of two vectors with dimension d. This indicates that the Bellman Rank is at most d.

8.3.2 Examples that do not have low Bellman Rank

[Jiang et al., 2017] also demonstrates a few more examples (with a slightly modification of the definition of average Bellman error) also admits low-Bellman rank including reactive POMDPs and reactive PSRs. So far in the literature, the only example that doesn't admit low Bellman rank but has polynomial sample complexity algorithms is the factored MDPs [Kearns and Koller, 1999]. Sun et al. [2019] showed that Bellman rank can be exponential in Factored MDPs.

8.4 Algorithm

Having defined our main structural assumption, we now describe an algorithm whose sample complexity depends on the Bellman rank, with no explicit dependence on $|\mathcal{S}|$ and $|\mathcal{A}|$ and only logarithmic scaling with $|\mathcal{F}|$. For ease of presentation, we will assume that all the expectations can be measured exactly with no errors, which serves to illustrate the key idea of *Explore-or-Terminate*. For a more careful analysis with finite samples, we refer the reader to Jiang et al. [2017]. The algorithm, named OLIVE for Optimism Led Iterative Value-function Elimination is an iterative algorithm which successively prunes value functions that have non-zero average Bellman error (recall $Q^\star$ has zero Bellman error at any state-action pair). It then uses the principle of optimism in the face of uncertainty to select its next policy which allows us to detect if a new optimal policy has been found. The algorithm is described in Algorithm 6.

Algorithm 5 The OLIVE algorithm for MDPs with low Bellman rank

Input: Function class $\mathcal{F}$.

1: Initialize $\mathcal{F}_0 = \mathcal{F}$.
2: **for** $t = 1, 2, \ldots,$ **do**
3: Define $f_t = \operatorname{argmax}_{f \in \mathcal{F}_{t-1}} \max_a f(s_0, a)$ and $\pi_t = \pi_{f_t}$.
4: **if** $\max_a f_t(s_0, a) = V^{\pi_t}$ **then return** π_t.
5: **else**
6: Update $\mathcal{F}_t = \{f \in \mathcal{F}_{t-1} \ : \ \mathcal{E}(f; \pi_t, h) = 0, \text{ for all } h \in [H-1]\}$.
7: **end if**
8: **end for**

The key step here is that during elimination procedure (Line 6), we enumerate all remaining $f \in \mathcal{F}_{t-1}$ under a fixed roll-in policy π_t, i.e., we can estimate $\mathcal{E}(f; \pi_t, h)$ for all f via a dataset that is collected from π_t. Assume we get the following dataset $\{s_h^i, a_h^i, r_h^i, s_{h+1}^i\}_{i=1}^N$ where $s_h^i, a_h^i \sim d_h^{\pi_t}$, and $s_{h+1}^i \sim P(\cdot | s_h^i, a_h^i)$. For any $f \in \mathcal{F}_{h-1}$, we can form the following empirical estimate of $\mathcal{E}(f; \pi_t, h)$:

$$\widetilde{\mathcal{E}}(f; \pi_t, h) = \frac{1}{N} \sum_{i=1}^{N} \left(f(s_h^i, a_h^i) - r_h^i - \max_a f(s_{h+1}^i, a) \right).$$

We can apply Hoeffding's inequality (Lemma A.1) and a union bound over $\mathcal{F}_{t-1}$ together here to get a uniform convergence argument, i.e., for any $f \in \mathcal{F}_{t-1}$, we will have $\left| \widetilde{\mathcal{E}}(f; \pi_t, h) - \mathcal{E}(f; \pi_t, h) \right| = \widetilde{O}(\sqrt{\ln(|\mathcal{F}|/\delta)/N})$.

Since we assume that all the expectations are available exactly, the main complexity analysis in OLIVE concerns the number of iterations before it terminates. When we estimate expectations using samples, this iteration complexity is critical as it also scales the sample complexity of the algorithm. We will state and prove the following theorem regarding the iteration complexity of OLIVE.

We need the following lemma to show that performance difference between $\max_a f(s_0, a)$ and the value of its induced greedy policy V^{π_t}.

Lemma 8.7 (Performance Difference)**.** *For any f, we have:*

$$\max_a f(x_0, a) - V^{\pi_f}(x_0) = \sum_{h=0}^{H-1} \mathbb{E}_{x_h, a_h \sim d_h^{\pi_f}} \left[f(x_h, a_h) - r(x_h, a_h) - \mathbb{E}_{x_{h+1} \sim P(\cdot | x_h, a_h)} \left[\max_a f(x_{h+1}, a) \right] \right].$$

We leave the proof of the above lemma in HW2.

Theorem 8.8. *For any MDP and $\mathcal{F}$ with Bellman rank M, OLIVE terminates in at most MH iterations and outputs $\pi^\star$.*

Proof: Consider an iteration t of OLIVE. Due to Assumption 8.2 and Fact 8.4, we know that $Q^\star \in \mathcal{F}_{t-1}$. Suppose OLIVE terminates at this iteration and returns π_t. Then we have

$$V^{\pi_t} = V_{f_t} = \max_{f \in \mathcal{F}_{t-1}} V_f \geq V_{Q^\star} = V^\star,$$

since $Q^\star \in \mathcal{F}_{t-1}$. So the algorithm correctly outputs an optimal policy when it terminates.

On the other hand, if it does not terminate then $V^{\pi_t} \neq V_{f_t}$ and Lemma 8.7 implies that $\mathcal{E}(f_t, \pi_t, h) > 0$ for some step $h \in [H-1]$. This certainly ensures that $f_t \notin \mathcal{F}_t$, but has significantly stronger implications. Note that $f_t \in \mathcal{F}_{t-1}$ implies that $\mathcal{E}(f_t, \pi_s, h) = 0$ for all $s < t$ and $h \in [H-1]$. Since we just concluded that $\mathcal{E}(f_t, \pi_t, h) > 0$ for some h, it must be the case that the row corresponding to π_t is linearly independent of those corresponding to $\pi_1, \ldots, \pi_{t-1}$ in the matrix $\mathcal{E}_h$. Consequently, at each non-final iteration, OLIVE finds a row that is linearly independent of all previous identified rows (i.e., indexed by $f_1, f_2, \ldots, f_{t-1}$). Since $\mathcal{E}_h$ has rank M, the total number of linearly independent rows one could find is at most M! Recall that there are at most H many Bellman error matrices, then the algorithm must terminate in number of rounds HM, which gives the statement of the theorem. ■

The proof of the theorem makes it precise that the factorization underlying Bellman rank really plays the role of an efficient basis for exploration in complex MDPs. Extending these ideas to noisy estimates of expectations requires some care since algebraic notions like rank are not robust to noise. Instead Jiang et al. [2017] use a more general volumetric argument to analyze the noisy case, as well as describe robustness to requirements of exact low-rank factorization and realizability.

Unfortunately, the OLIVE algorithm is not computationally efficient, and a computational hardness result was discovered by Dann et al. [2018]. Developing both statistically and computationally efficient exploration algorithms for RL with rich observations is an area of active research.

However, whether there exists a computationally efficient algorithm for low Bellman rank setting is still an open problem.

8.5 Extension to Model-based Setting

We briefly discuss the extension to model-based setting. Different from model-free value-based setting, in model-based learning, we start with a model class that contains possible transitions. To ease presentation, we denote the ground truth model as $P^\star$.

$$\mathcal{P} = \{P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})\}.$$

Again we assume realizability:

Assumption 8.9. We assume $P^\star \in \mathcal{P}$.

Given any $P \in \mathcal{P}$, we can define the optimal value function and optimal Q function under P. Specifically, we denote $V_P^\star$ and $Q_P^\star$ as the optimal value and Q function under model P, and π_P as the optimal policy under P.

We introduce a class of witness functions (a.k.a discriminators),

$$\mathcal{G} = \{g : \mathcal{S} \times \mathcal{A} \times \mathcal{A} \mapsto \mathbb{R}\}.$$

We denote the *witness model-misfit* for a model P as follows:

$$\mathcal{W}(P; \bar{P}, h, \mathcal{G}) := \sup_{g \in \mathcal{G}} \mathbb{E}_{s_h, a_h \sim d_h^{\pi_{\bar{P}}}} \left[\mathbb{E}_{s_{h+1} \sim P(\cdot|s_h, a_h)} g(s_h, a_h, s_{h+1}) - \mathbb{E}_{s_{h+1} \sim P^\star(\cdot|s_h, a_h)} g(s_h, a_h, s_{h+1}) \right].$$

Namely, we are using a discriminator class $\mathcal{G}$ to distinguish two distributions $d_h^{\pi_{\bar{P}}} \circ P$ and $d_h^{\pi_{\bar{P}}} \circ P^\star$. Note that for the ground truth model $P^\star$, we always have witness misfit being zero, i.e., $\mathcal{W}(P^\star; \bar{P}, h, \mathcal{G}) = 0$ for any $\bar{P}, h, \mathcal{G}$.

To further compare to Bellman rank, we assume the following assumption on the discriminators:

Assumption 8.10. We assume $\mathcal{G}$ contains functions $r + V_P^\star$ for all $P \in \mathcal{P}$, i.e.,

$$\{r(s,a) + V_P^\star(s') : P \in \mathcal{P}\} \subseteq \mathcal{G}.$$

With this assumption, we can show that *Bellman Domination*, i.e., $\mathcal{W}(P; \bar{P}, h, \mathcal{G}) \geq \mathcal{E}(Q_P^\star; \pi_{\bar{P}}, h)$, i.e., the average Bellman error of $Q_P^\star$ under the state-action distribution induced by the optimal policy of $\bar{P}$.

We define Witness rank as follows.

Definition 8.11 (Witness Rank)**.** Consider the Witness misfit matrix $\mathcal{W}_h \in \mathbb{R}^{|\mathcal{P}| \times |\mathcal{P}|}$, where $[\mathcal{W}_h]_{\bar{P},P} := \mathcal{W}(P; \bar{P}, h, \mathcal{G})$. Witness rank is the smallest number M such that $\text{rank}(\mathcal{W}_h) \leq M$ for all h.

Sun et al. [2019] provides a more refined definition of Witness rank which is at least as small as the Bellman rank under the value functions $\mathcal{Q} = \{Q_P^\star : P \in \mathcal{P}\}$, and shows that for factored MDPs, Bellman rank with $\mathcal{Q}$ could be exponentially large with respect to horizon H while witness rank remains small and captures the right complexity measure in factored MDPs.

Indeed one can show that there exists a factored MDP, such that no model-free algorithms using $\mathcal{Q}$ as a function class input is able to achieve polynomial sample complexity, while the algorithm OLIME which we present later, achieves polynomially sample complexity [Sun et al., 2019]. This demonstrates an exponential separation in sample complexity between model-based approaches and model-free approaches.

We close here by providing the algorithm *Optimism-Led Iterative Model Elimination* (OLIME) below.

Algorithm 6 The OLIME algorithm for MDPs with low Witness rank

Input: model class $\mathcal{P}$.
1: Initialize $\mathcal{P}_0 = \mathcal{P}$.
2: **for** $t = 1, 2, \ldots,$ **do**
3: Define $P_t = \text{argmax}_{P \in \mathcal{P}_{t-1}} V_P^\star(s_0)$ and $\pi_t = \pi_{P_t}$.
4: **if** $V_{P_t}^{\pi_t} = V^{\pi_t}$ **then return** π_t.
5: **else**
6: Update $\mathcal{P}_t = \{f \in \mathcal{P}_{t-1} \ : \ \mathcal{W}(P; \pi_t, h, \mathcal{G}) = 0, \text{ for all } h \in [H-1]\}$.
7: **end if**
8: **end for**

The algorithm use optimism and maintains $\mathcal{P}_t$ in each iteration. Every iteration, it picks the most optimistic model P_t from the current model class $\mathcal{P}_t$. Again optimism allows us to identify if the algorithm has find the optimal policy. If the algorithm does not terminate, then we are guaranteed to find a model P_t that admits non-zero witness misfit. We then update $\mathcal{P}_t$ by eliminating all models that have non-zero model misfit under the distribution of π_t. Again, following the similar argument we had for OLIVE, one can show that OLIME terminates in at most MH iterations, with M being the witness rank.

8.6 Bibliographic Remarks and Further Readings

Bellman rank was original proposed by Jiang et al. [2017]. Note that the definition of our average Bellman error is a simple modification of the original average Bellman error definition in [Jiang et al., 2017]. Our definition allows it

to capture linear MDPs and the Bellman Completion in Linear Function Approximation, without assuming discrete action space and paying a polynomial dependency on the number of actions.

Witness rank was proposed by Sun et al. [2019] for model-based setting. Sun et al. [2019] showed that witness rank captures the correct complexity in Factored MDPs [Kearns and Koller, 1999] while Bellman rank could be exponentially large in H when applied to Factored MDPs. Sun et al. [2019] also demonstrates an exponential sample complexity gap between model-based algorithms and model-free algorithms with general function approximation.

Part 3

Policy Optimization

Chapter 9

Policy Gradient Methods and Non-Convex Optimization

For a distribution ρ over states, define:

$$V^\pi(\rho) := \mathbb{E}_{s_0 \sim \rho}[V^\pi(s_0)]\,,$$

where we slightly overload notation. Consider a class of parametric policies $\{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$. The optimization problem of interest is:

$$\max_{\theta \in \Theta} V^{\pi_\theta}(\rho)\,.$$

We drop the MDP subscript in this chapter.

One immediate issue is that if the policy class $\{\pi_\theta\}$ consists of deterministic policies then π_θ will, in general, not be differentiable. This motivates us to consider policy classes that are stochastic, which permit differentiability.

Example 9.1 (Softmax policies)**.** It is instructive to explicitly consider a "tabular" policy representation, given by the *softmax policy*:

$$\pi_\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}\,, \tag{0.1}$$

where the parameter space is $\Theta = \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$. Note that (the closure of) the set of softmax policies contains all stationary and deterministic policies.

Example 9.2 (Log-linear policies)**.** For any state, action pair s, a, suppose we have a feature mapping $\phi_{s,a} \in \mathbb{R}^d$. Let us consider the policy class

$$\pi_\theta(a|s) = \frac{\exp(\theta \cdot \phi_{s,a})}{\sum_{a' \in \mathcal{A}} \exp(\theta \cdot \phi_{s,a'})}$$

with $\theta \in \mathbb{R}^d$.

Example 9.3 (Neural softmax policies)**.** Here we may be interested in working with the policy class

$$\pi_\theta(a|s) = \frac{\exp\left(f_\theta(s,a)\right)}{\sum_{a' \in \mathcal{A}} \exp\left(f_\theta(s,a')\right)}$$

where the scalar function $f_\theta(s,a)$ may be parameterized by a neural network, with $\theta \in \mathbb{R}^d$.

9.1 Policy Gradient Expressions and the Likelihood Ratio Method

Let τ denote a trajectory, whose unconditional distribution $\Pr_\mu^\pi(\tau)$ under π with starting distribution μ, is

$$\Pr_\mu^\pi(\tau) = \mu(s_0)\pi(a_0|s_0)P(s_1|s_0,a_0)\pi(a_1|s_1)\cdots . \tag{0.2}$$

We drop the μ subscript when it is clear from context.

It is convenient to define the discounted total reward of a trajectory as:

$$R(\tau) := \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$$

where s_t, a_t are the state-action pairs in τ. Observe that:

$$V^{\pi_\theta}(\mu) = \mathbb{E}_{\tau\sim \Pr_\mu^{\pi_\theta}}[R(\tau)] .$$

Theorem 9.4. *(Policy gradients) The following are expressions for $\nabla_\theta V^{\pi_\theta}(\mu)$:*

- *REINFORCE:*

$$\nabla V^{\pi_\theta}(\mu) = \mathbb{E}_{\tau\sim \Pr_\mu^{\pi_\theta}}\left[R(\tau)\sum_{t=0}^{\infty}\nabla \log \pi_\theta(a_t|s_t)\right]$$

- *Action value expression:*

$$\begin{aligned}\nabla V^{\pi_\theta}(\mu) &= \mathbb{E}_{\tau\sim \Pr_\mu^{\pi_\theta}}\left[\sum_{t=0}^{\infty}\gamma^t Q^{\pi_\theta}(s_t,a_t)\nabla \log \pi_\theta(a_t|s_t)\right] \\ &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\Big[Q^{\pi_\theta}(s,a)\nabla \log \pi_\theta(a|s)\Big]\end{aligned}$$

- *Advantage expression:*

$$\nabla V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\Big[A^{\pi_\theta}(s,a)\nabla \log \pi_\theta(a|s)\Big]$$

The alternative expressions are more helpful to use when we turn to Monte Carlo estimation.

Proof: We have:

$$\begin{aligned}\nabla V^{\pi_\theta}(\mu) &= \nabla \sum_\tau R(\tau)\Pr_\mu^{\pi_\theta}(\tau) \\ &= \sum_\tau R(\tau)\nabla \Pr_\mu^{\pi_\theta}(\tau) \\ &= \sum_\tau R(\tau)\Pr_\mu^{\pi_\theta}(\tau)\nabla \log \Pr_\mu^{\pi_\theta}(\tau) \\ &= \sum_\tau R(\tau)\Pr_\mu^{\pi_\theta}(\tau)\nabla \log\left(\mu(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0,a_0)\pi_\theta(a_1|s_1)\cdots\right) \\ &= \sum_\tau R(\tau)\Pr_\mu^{\pi_\theta}(\tau)\left(\sum_{t=0}^{\infty}\nabla \log \pi_\theta(a_t|s_t)\right)\end{aligned}$$

which completes the proof of the first claim.

For the second claim, for any state s_0

$$\begin{aligned}
& \nabla V^{\pi_\theta}(s_0) \\
= \; & \nabla \sum_{a_0} \pi_\theta(a_0|s_0) Q^{\pi_\theta}(s_0, a_0) \\
= \; & \sum_{a_0} \Big(\nabla \pi_\theta(a_0|s_0)\Big) Q^{\pi_\theta}(s_0, a_0) + \sum_{a_0} \pi_\theta(a_0|s_0) \nabla Q^{\pi_\theta}(s_0, a_0) \\
= \; & \sum_{a_0} \pi_\theta(a_0|s_0) \Big(\nabla \log \pi_\theta(a_0|s_0)\Big) Q^{\pi_\theta}(s_0, a_0) \\
& + \sum_{a_0} \pi_\theta(a_0|s_0) \nabla \Big(r(s_0, a_0) + \gamma \sum_{s_1} P(s_1|s_0, a_0) V^{\pi_\theta}(s_1)\Big) \\
= \; & \sum_{a_0} \pi_\theta(a_0|s_0) \Big(\nabla \log \pi_\theta(a_0|s_0)\Big) Q^{\pi_\theta}(s_0, a_0) + \gamma \sum_{a_0, s_1} \pi_\theta(a_0|s_0) P(s_1|s_0, a_0) \nabla V^{\pi_\theta}(s_1) \\
= \; & \mathbb{E}_{\tau \sim \Pr_{s_0}^{\pi_\theta}} \left[Q^{\pi_\theta}(s_0, a_0) \nabla \log \pi_\theta(a_0|s_0)\right] + \gamma \mathbb{E}_{\tau \sim \Pr_{s_0}^{\pi_\theta}} \left[\nabla V^{\pi_\theta}(s_1)\right].
\end{aligned}$$

By linearity of expectation,

$$\begin{aligned}
& \nabla V^{\pi_\theta}(\mu) \\
= \; & \mathbb{E}_{\tau \sim \Pr_{\mu}^{\pi_\theta}} \left[Q^{\pi_\theta}(s_0, a_0) \nabla \log \pi_\theta(a_0|s_0)\right] + \gamma \mathbb{E}_{\tau \sim \Pr_{\mu}^{\pi_\theta}} \left[\nabla V^{\pi_\theta}(s_1)\right] \\
= \; & \mathbb{E}_{\tau \sim \Pr_{\mu}^{\pi_\theta}} \left[Q^{\pi_\theta}(s_0, a_0) \nabla \log \pi_\theta(a_0|s_0)\right] + \gamma \mathbb{E}_{\tau \sim \Pr_{\mu}^{\pi_\theta}} \left[Q^{\pi_\theta}(s_1, a_1) \nabla \log \pi_\theta(a_1|s_1)\right] + \dots .
\end{aligned}$$

where the last step follows from recursion. This completes the proof of the second claim.

The proof of the final claim is left as an exercise to the reader. ■

9.2 (Non-convex) Optimization

It is worth explicitly noting that $V^{\pi_\theta}(s)$ is non-concave in θ for the softmax parameterizations, so the standard tools of convex optimization are not applicable.

Lemma 9.5. *(Non-convexity) There is an MDP M (described in Figure 0.1) such that the optimization problem $V^{\pi_\theta}(s)$ is not concave for both the direct and softmax parameterizations.*

Proof: Recall the MDP in Figure 0.1. Note that since actions in terminal states s_3, s_4 and s_5 do not change the expected reward, we only consider actions in states s_1 and s_2. Let the "up/above" action as a_1 and "right" action as a_2. Note that

$$V^{\pi}(s_1) = \pi(a_2|s_1)\pi(a_1|s_2) \cdot r$$

Now consider

$$\theta^{(1)} = (\log 1, \log 3, \log 3, \log 1), \quad \theta^{(2)} = (-\log 1, -\log 3, -\log 3, -\log 1)$$

where θ is written as a tuple $(\theta_{a_1,s_1}, \theta_{a_2,s_1}, \theta_{a_1,s_2}, \theta_{a_2,s_2})$. Then, for the softmax parameterization, we have:

$$\pi^{(1)}(a_2|s_1) = \frac{3}{4}; \quad \pi^{(1)}(a_1|s_2) = \frac{3}{4}; \quad V^{(1)}(s_1) = \frac{9}{16} r$$

and

$$\pi^{(2)}(a_2|s_1) = \frac{1}{4}; \quad \pi^{(2)}(a_1|s_2) = \frac{1}{4}; \quad V^{(2)}(s_1) = \frac{1}{16} r.$$

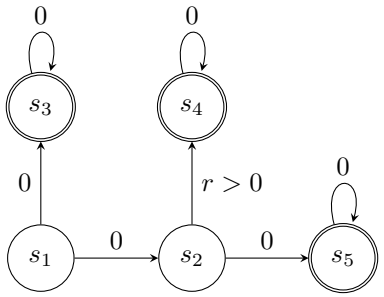

Figure 0.1: (Non-concavity example) A deterministic MDP corresponding to Lemma 9.5 where $V^{\pi_\theta}(s)$ is not concave. Numbers on arrows represent the rewards for each action.

Also, for $\theta^{(\text{mid})} = \frac{\theta^{(1)}+\theta^{(2)}}{2}$,

$$\pi^{(\text{mid})}(a_2|s_1) = \frac{1}{2}; \quad \pi^{(\text{mid})}(a_1|s_2) = \frac{1}{2}; \quad V^{(\text{mid})}(s_1) = \frac{1}{4}r.$$

This gives

$$V^{(1)}(s_1) + V^{(2)}(s_1) > 2V^{(\text{mid})}(s_1),$$

which shows that V^π is non-concave. ■

9.2.1 Gradient ascent and convergence to stationary points

Let us say a function $f : \mathbb{R}^d \to \mathbb{R}$ is β *-smooth* if

$$\|\nabla f(w) - \nabla f(w')\| \leq \beta \|w - w'\|, \tag{0.3}$$

where the norm $\|\cdot\|$ is the Euclidean norm. In other words, the derivatives of f do not change too quickly.

Gradient ascent, with a fixed stepsize η, follows the update rule:

$$\theta_{t+1} = \theta_t + \eta \nabla V^{\pi_{\theta_t}}(\mu).$$

It is convenient to use the shorthand notation:

$$\pi^{(t)} := \pi_{\theta_t}, \qquad V^{(t)} := V^{\pi_{\theta_t}}$$

The next lemma is standard in non-convex optimization.

Lemma 9.6. *(Convergence to Stationary Points) Assume that for all $\theta \in \Theta$, V^{π_θ} is β-smooth and bounded below by V_*. Suppose we use the constant stepsize $\eta = 1/\beta$. For all T, we have that*

$$\min_{t \leq T} \|\nabla V^{(t)}(\mu)\|^2 \leq \frac{2\beta(V^*(\mu) - V^{(0)}(\mu))}{T}.$$

9.2.2 Monte Carlo estimation and stochastic gradient ascent

One difficulty is that even if we know the MDP M, computing the gradient may be computationally intensive. It turns out that we can obtain unbiased estimates of π with only simulation based access to our model, i.e. assuming we can obtain sampled trajectories $\tau \sim \Pr_\mu^{\pi_\theta}$.

With respect to a trajectory τ, define:

$$\begin{aligned}\widehat{Q^{\pi_\theta}}(s_t, a_t) &:= \sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \\ \widehat{\nabla V^{\pi_\theta}}(\mu) &:= \sum_{t=0}^{\infty} \gamma^t \widehat{Q^{\pi_\theta}}(s_t, a_t) \nabla \log \pi_\theta(a_t|s_t)\end{aligned}$$

We now show this provides an unbiased estimated of the gradient:

Lemma 9.7. *(Unbiased gradient estimate) We have :*

$$\mathbb{E}_{\tau \sim \Pr_\mu^{\pi_\theta}}\left[\widehat{\nabla V^{\pi_\theta}}(\mu)\right] = \nabla V^{\pi_\theta}(\mu)$$

Proof: Observe:

$$\begin{aligned}\mathbb{E}[\widehat{\nabla V^{\pi_\theta}}(\mu)] &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \widehat{Q^{\pi_\theta}}(s_t, a_t) \nabla \log \pi_\theta(a_t|s_t)\right] \\ &\overset{(a)}{=} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\widehat{Q^{\pi_\theta}}(s_t, a_t)|s_t, a_t] \nabla \log \pi_\theta(a_t|s_t)\right] \\ &\overset{(b)}{=} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t Q^{\pi_\theta}(s_t, a_t) \nabla \log \pi_\theta(a_t|s_t)\right]\end{aligned}$$

where (a) follows from the tower property of the conditional expectations and (b) follows from that the Markov property implies $\mathbb{E}[\widehat{Q^{\pi_\theta}}(s_t, a_t)|s_t, a_t] = Q^{\pi_\theta}(s_t, a_t)$. ■

Hence, the following procedure is a stochastic gradient ascent algorithm:

1. initialize θ_0.
2. For $t = 0, 1, \ldots$
   (a) Sample $\tau \sim \Pr_\mu^{\pi_\theta}$.
   (b) Update:
   $$\theta_{t+1} = \theta_t + \eta_t \widehat{\nabla V^{\pi_\theta}}(\mu)$$
   where η_t is the stepsize and $\widehat{\nabla V^{\pi_\theta}}(\mu)$ estimated with τ.

Note here that we are ignoring that τ is an infinte length sequence. It can be truncated appropriately so as to control the bias.

The following is standard result with regards to non-convex optimization. Again, with reasonably bounded variance, we will obtain a point θ_t with small gradient norm.

Lemma 9.8. *(Stochastic Convergence to Stationary Points) Assume that for all $\theta \in \Theta$, V^{π_θ} is β-smooth and bounded below by V_*. Suppose the variance is bounded as follows:*

$$\mathbb{E}[\|\widehat{\nabla V^{\pi_\theta}}(\mu) - \nabla V^{\pi_\theta}(\mu)\|^2] \leq \sigma^2$$

For $t \leq \beta(V^*(\mu) - V^{(0)}(\mu))/\sigma^2$, suppose we use a constant stepsize of $\eta_t = 1/\beta$, and thereafter, we use $\eta_t = \sqrt{2/(\beta T)}$. For all T, we have:

$$\min_{t \leq T} \mathbb{E}[\|\nabla V^{(t)}(\mu)\|^2] \leq \frac{2\beta(V^*(\mu) - V^{(0)}(\mu))}{T} + \sqrt{\frac{2\sigma^2}{T}}.$$

Baselines and stochastic gradient ascent

A significant practical issue is that the variance σ^2 is often large in practice. Here, a form of variance reduction is often critical in practice. A common method is as follows.

Let $f : \mathcal{S} \to \mathbb{R}$.

1. Construct f as an estimate of $V^{\pi_\theta}(\mu)$. This can be done using any previous data.
2. Sample a new trajectory τ, and define:

$$\begin{aligned}\widehat{Q^{\pi_\theta}}(s_t, a_t) &:= \sum_{t'=t}^{\infty} \gamma^{t'-t} r(s_{t'}, a_{t'}) \\ \widehat{\nabla V^{\pi_\theta}}(\mu) &:= \sum_{t=0}^{\infty} \gamma^t \Big(\widehat{Q^{\pi_\theta}}(s_t, a_t) - f(s_t)\Big) \nabla \log \pi_\theta(a_t|s_t)\end{aligned}$$

We often refer to $f(s)$ as a *baseline* at state s.

Lemma 9.9. *(Unbiased gradient estimate with Variance Reduction) For any procedure used to construct to the baseline function $f : \mathcal{S} \to \mathbb{R}$, if the samples used to construct f are independent of the trajectory τ, where $\widehat{Q^{\pi_\theta}}(s_t, a_t)$ is constructed using τ, then:*

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \Big(\widehat{Q^{\pi_\theta}}(s_t, a_t) - f(s_t)\Big) \nabla \log \pi_\theta(a_t|s_t)\right] = \nabla V^{\pi_\theta}(\mu)$$

where the expectation is with respect to both the random trajectory τ and the random function $f(\cdot)$.

Proof: For any function $g(s)$,

$$\mathbb{E}\left[\nabla \log \pi(a|s) g(s)\right] = \sum_a \nabla \pi(a|s) g(s) = g(s) \sum_a \nabla \pi(a|s) = g(s) \nabla \sum_a \pi(a|s) = g(s) \nabla 1 = 0$$

Using that $f(\cdot)$ is independent of τ, we have that for all t

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t f(s_t) \nabla \log \pi_\theta(a_t|s_t)\right] = 0$$

The result now follow froms Lemma 9.7. ■

9.3 Bibliographic Remarks and Further Readings

The REINFOCE algorithm is due to [Williams, 1992], which is an example of the likelihood ratio method for gradient estimation [Glynn, 1990].

For standard optimization results in non-convex optimization (e.g. Lemma 9.6 and 9.8), we refer the reader to [Beck, 2017]. Our results for convergence rates for SGD to approximate stationary points follow from [Ghadimi and Lan, 2013].

Chapter 10

Optimality

We now seek to understand the global convergence properties of policy gradient methods, when given exact gradients. Here, we will largely limit ourselves to the (tabular) softmax policy class in Example 9.1.

Given our a starting distribution ρ over states, recall our objective is:

$$\max_{\theta \in \Theta} V^{\pi_\theta}(\rho)\,.$$

where $\{\pi_\theta | \theta \in \Theta \subset \mathbb{R}^d\}$ is some class of parametric policies.

While we are interested in good performance under ρ, we will see how it is helpful to optimize under a different measure μ. Specifically, we consider optimizing $V^{\pi_\theta}(\mu)$, i.e.

$$\max_{\theta \in \Theta} V^{\pi_\theta}(\mu)\,,$$

even though our ultimate goal is performance under $V^{\pi_\theta}(\rho)$.

We now consider the softmax policy parameterization (0.1). Here, we still have a non-concave optimization problem in general, as shown in Lemma 9.5, though we do show that global optimality can be reached under certain regularity conditions. From a practical perspective, the softmax parameterization of policies is preferable to the direct parameterization, since the parameters θ are unconstrained and standard unconstrained optimization algorithms can be employed. However, optimization over this policy class creates other challenges as we study in this section, as the optimal policy (which is deterministic) is attained by sending the parameters to infinity.

This chapter will study three algorithms for this problem, for the softmax policy class. The first performs direct policy gradient ascent on the objective without modification, while the second adds a log barrier regularizer to keep the parameters from becoming too large, as a means to ensure adequate exploration. Finally, we study the natural policy gradient algorithm and establish a global optimality convergence rate, with no dependence on the dimension-dependent factors.

The presentation in this chapter largely follows the results in [Agarwal et al., 2020c].

10.1 Vanishing Gradients and Saddle Points

To understand the necessity of optimizing under a distribution μ that is different from ρ, let us first give an informal argument that some condition on the state distribution of π, or equivalently μ, is necessary for stationarity to imply

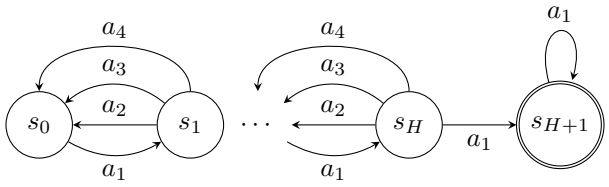

Figure 0.1: (Vanishing gradient example) A deterministic, chain MDP of length $H + 2$. We consider a policy where $\pi(a|s_i) = \theta_{s_i,a}$ for $i = 1, 2, \ldots, H$. Rewards are 0 everywhere other than $r(s_{H+1}, a_1) = 1$. See Proposition 10.1.

optimality. For example, in a sparse-reward MDP (where the agent is only rewarded upon visiting some small set of states), a policy that does not visit *any* rewarding states will have zero gradient, even though it is arbitrarily suboptimal in terms of values. Below, we give a more quantitative version of this intuition, which demonstrates that even if π chooses all actions with reasonable probabilities (and hence the agent will visit all states if the MDP is connected), then there is an MDP where a large fraction of the policies π have vanishingly small gradients, and yet these policies are highly suboptimal in terms of their value.

Concretely, consider the chain MDP of length $H + 2$ shown in Figure 0.1. The starting state of interest is state s_0 and the discount factor $\gamma = H/(H + 1)$. Suppose we work with the direct parameterization, where $\pi_\theta(a|s) = \theta_{s,a}$ for $a = a_1, a_2, a_3$ and $\pi_\theta(a_4|s) = 1 - \theta_{s,a_1} - \theta_{s,a_2} - \theta_{s,a_3}$. Note we do not over-parameterize the policy. For this MDP and policy structure, if we were to initialize the probabilities over actions, say deterministically, then there is an MDP (obtained by permuting the actions) where all the probabilities for a_1 will be less than $1/4$.

The following result not only shows that the gradient is exponentially small in H, it also shows that many higher order derivatives, up to $O(H/\log H)$, are also exponentially small in H.

Proposition 10.1 (Vanishing gradients at suboptimal parameters)**.** Consider the chain MDP of Figure 0.1, with $H + 2$ states, $\gamma = H/(H + 1)$, and with the direct policy parameterization (with $3|\mathcal{S}|$ parameters, as described in the text above). Suppose θ is such that $0 < \theta < 1$ (componentwise) and $\theta_{s,a_1} < 1/4$ (for all states s). For all $k \leq \frac{H}{40 \log(2H)} - 1$, we have $\left\|\nabla^k_\theta V^{\pi_\theta}(s_0)\right\| \leq (1/3)^{H/4}$, where $\nabla^k_\theta V^{\pi_\theta}(s_0)$ is a tensor of the k_{th} order derivatives of $V^{\pi_\theta}(s_0)$ and the norm is the operator norm of the tensor.[1] Furthermore, $V^\star(s_0) - V^{\pi_\theta}(s_0) \geq (H + 1)/8 - (H + 1)^2/3^H$.

We do not prove this lemma here (see Section 10.5). The lemma illustrates that lack of good exploration can indeed be detrimental in policy gradient algorithms, since the gradient can be small either due to π being near-optimal, or, simply because π does not visit advantageous states often enough. Furthermore, this lemma also suggests that varied results in the non-convex optimization literature, on escaping from saddle points, do not directly imply global convergence due to that the higher order derivatives are small.

While the chain MDP of Figure 0.1, is a common example where *sample* based estimates of gradients will be 0 under random exploration strategies; there is an exponentially small in H chance of hitting the goal state under a random exploration strategy. Note that this lemma is with regards to *exact* gradients. This suggests that even with exact computations (along with using exact higher order derivatives) we might expect numerical instabilities.

10.2 Policy Gradient Ascent

Let us now return to the softmax policy class, from Equation 0.1, where:

$$\pi_\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})},$$

[1] The operator norm of a k_{th}-order tensor $J \in \mathbb{R}^{d^{\otimes k}}$ is defined as $\sup_{u_1,\ldots,u_k \in \mathbb{R}^d \,:\, \|u_i\|_2 = 1} \langle J, u_1 \otimes \ldots \otimes u_d \rangle$.

where the number of parameters in this policy class is $|\mathcal{S}||\mathcal{A}|$.

Observe that:

$$\frac{\partial \log \pi_\theta(a|s)}{\partial \theta_{s',a'}} = \mathbb{1}\Big[s = s'\Big]\Big(\mathbb{1}\Big[a = a'\Big] - \pi_\theta(a'|s)\Big)$$

where $\mathbb{1}[\mathcal{E}]$ is the indicator of $\mathcal{E}$ being true.

Lemma 10.2. *For the softmax policy class, we have:*

$$\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s)\pi_\theta(a|s)A^{\pi_\theta}(s,a) \tag{0.1}$$

Proof: Using the advantage expression for the policy gradient (see Theorem 9.4),

$$\begin{aligned}
\frac{\partial V^{\pi_\theta}(\mu)}{\partial \theta_{s',a'}} &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi_\theta}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\Big[A^{\pi_\theta}(s,a)\frac{\partial \log \pi_\theta(a|s)}{\partial \theta_{s',a'}}\Big] \\
&= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi_\theta}}\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\Big[A^{\pi_\theta}(s,a)\mathbb{1}\Big[s = s'\Big]\Big(\mathbb{1}\Big[a = a'\Big] - \pi_\theta(a'|s)\Big)\Big] \\
&= \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s')\mathbb{E}_{a\sim\pi_\theta(\cdot|s')}\Big[A^{\pi_\theta}(s',a)\Big(\mathbb{1}\Big[a = a'\Big] - \pi_\theta(a'|s')\Big)\Big] \\
&= \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s')\left(\mathbb{E}_{a\sim\pi_\theta(\cdot|s')}\Big[A^{\pi_\theta}(s',a)\mathbb{1}\big[a = a'\big]\Big] - \pi_\theta(a'|s')\mathbb{E}_{a\sim\pi_\theta(\cdot|s')}\Big[A^{\pi_\theta}(s',a)\Big]\right) \\
&= \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s')\pi_\theta(a'|s')A^{\pi_\theta}(s',a') - 0\,,
\end{aligned}$$

where the last step uses that for any policy $\sum_a \pi(a|s)A^\pi(s,a) = 0$. ■

The update rule for gradient ascent is:

$$\theta^{(t+1)} = \theta^{(t)} + \eta\nabla_\theta V^{(t)}(\mu). \tag{0.2}$$

Recall from Lemma 9.5 that, even for the case of the softmax policy class (which contains all stationary policies), our optimization problem is non-convex. Furthermore, due to the exponential scaling with the parameters θ in the softmax parameterization, *any* policy that is nearly deterministic will have gradients close to 0. Specifically, for any sequence of policies π^{θ_t} that becomes deterministic, $\|\nabla V^{\pi^{\theta_t}}\| \to 0$.

In spite of these difficulties, it turns out we have a positive result that gradient ascent asymptotically converges to the global optimum for the softmax parameterization.

Theorem 10.3 (Global convergence for softmax parameterization)**.** *Assume we follow the gradient ascent update rule as specified in Equation* (0.2) *and that the distribution μ is strictly positive i.e. $\mu(s) > 0$ for all states s. Suppose $\eta \leq \frac{(1-\gamma)^3}{8}$, then we have that for all states s, $V^{(t)}(s) \to V^\star(s)$ as $t \to \infty$.*

The proof is somewhat technical, and we do not provide a proof here (see Section 10.5).

A few remarks are in order. Theorem 10.3 assumed that optimization distribution μ was *strictly* positive, i.e. $\mu(s) > 0$ for all states s. We conjecture that any gradient ascent may not globally converge if this condition is not met. The concern is that if this condition is not met, then gradient ascent may not globally converge due to that $d_\mu^{\pi_\theta}(s)$ effectively scales down the learning rate for the parameters associated with state s (see Equation 0.1).

Furthermore, there is strong reason to believe that the convergence rate for this is algorithm (in the worst case) is exponentially slow in some of the relevant quantities, such as in terms of the size of state space. We now turn to a regularization based approach to ensure convergence at a polynomial rate in all relevant quantities.

10.3 Log Barrier Regularization

Due to the exponential scaling with the parameters θ, policies can rapidly become near deterministic, when optimizing under the softmax parameterization, which can result in slow convergence. Indeed a key challenge in the asymptotic analysis in the previous section was to handle the growth of the absolute values of parameters as they tend to infinity. Recall that the relative-entropy for distributions p and q is defined as:

$$\mathrm{KL}(p, q) := \mathbb{E}_{x\sim p}[-\log q(x)/p(x)].$$

Denote the uniform distribution over a set $\mathcal{X}$ by $\mathrm{Unif}_{\mathcal{X}}$, and define the following log barrier regularized objective as:

$$\begin{aligned} L_\lambda(\theta) &:= V^{\pi_\theta}(\mu) - \lambda\, \mathbb{E}_{s\sim \mathrm{Unif}_{\mathcal{S}}}\left[\mathrm{KL}(\mathrm{Unif}_{\mathcal{A}}, \pi_\theta(\cdot|s))\right] \\ &= V^{\pi_\theta}(\mu) + \frac{\lambda}{|\mathcal{S}|\,|\mathcal{A}|}\sum_{s,a}\log \pi_\theta(a|s) + \lambda \log|\mathcal{A}|\,, \end{aligned} \tag{0.3}$$

where λ is a regularization parameter. The constant (i.e. the last term) is not relevant with regards to optimization. This regularizer is different from the more commonly utilized entropy regularizer, a point which we return to later.

The policy gradient ascent updates for $L_\lambda(\theta)$ are given by:

$$\theta^{(t+1)} = \theta^{(t)} + \eta \nabla_\theta L_\lambda(\theta^{(t)}). \tag{0.4}$$

We now see that any approximate first-order stationary points of the entropy-regularized objective is approximately globally optimal, provided the regularization is sufficiently small.

Theorem 10.4. *(Log barrier regularization) Suppose θ is such that:*

$$\|\nabla_\theta L_\lambda(\theta)\|_2 \leq \epsilon_{opt}$$

and $\epsilon_{opt} \leq \lambda/(2|\mathcal{S}|\,|\mathcal{A}|)$. Then we have that for all starting state distributions ρ:

$$V^{\pi_\theta}(\rho) \geq V^\star(\rho) - \frac{2\lambda}{1-\gamma}\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty.$$

We refer to $\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty$ as the *distribution mismatch coefficient*. The above theorem shows the importance of having an appropriate measure $\mu(s)$ in order for the approximate first-order stationary points to be near optimal.

Proof: The proof consists of showing that $\max_a A^{\pi_\theta}(s,a) \leq 2\lambda/(\mu(s)|\mathcal{S}|)$ for all states. To see that this is sufficient, observe that by the performance difference lemma (Lemma 1.16),

$$\begin{aligned} V^\star(\rho) - V^{\pi_\theta}(\rho) &= \frac{1}{1-\gamma}\sum_{s,a} d_\rho^{\pi^\star}(s)\pi^\star(a|s)A^{\pi_\theta}(s,a) \\ &\leq \frac{1}{1-\gamma}\sum_{s} d_\rho^{\pi^\star}(s)\max_{a\in\mathcal{A}} A^{\pi_\theta}(s,a) \\ &\leq \frac{1}{1-\gamma}\sum_{s} 2d_\rho^{\pi^\star}(s)\lambda/(\mu(s)|\mathcal{S}|) \\ &\leq \frac{2\lambda}{1-\gamma}\max_s\left(\frac{d_\rho^{\pi^\star}(s)}{\mu(s)}\right). \end{aligned}$$

which would then complete the proof.

We now proceed to show that $\max_a A^{\pi_\theta}(s,a) \leq 2\lambda/(\mu(s)|\mathcal{S}|)$. For this, it suffices to bound $A^{\pi_\theta}(s,a)$ for any state-action pair s, a where $A^{\pi_\theta}(s,a) \geq 0$ else the claim is trivially true. Consider an (s,a) pair such that $A^{\pi_\theta}(s,a) > 0$. Using the policy gradient expression for the softmax parameterization (see Equation 0.1),

$$\frac{\partial L_\lambda(\theta)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s)\pi_\theta(a|s)A^{\pi_\theta}(s,a) + \frac{\lambda}{|\mathcal{S}|}\left(\frac{1}{|\mathcal{A}|} - \pi_\theta(a|s)\right). \tag{0.5}$$

The gradient norm assumption $\|\nabla_\theta L_\lambda(\theta)\|_2 \leq \epsilon_{\text{opt}}$ implies that:

$$\begin{aligned}\epsilon_{\text{opt}} \geq \frac{\partial L_\lambda(\theta)}{\partial \theta_{s,a}} &= \frac{1}{1-\gamma} d_\mu^{\pi_\theta}(s)\pi_\theta(a|s)A^{\pi_\theta}(s,a) + \frac{\lambda}{|\mathcal{S}|}\left(\frac{1}{|\mathcal{A}|} - \pi_\theta(a|s)\right) \\ &\geq \frac{\lambda}{|\mathcal{S}|}\left(\frac{1}{|\mathcal{A}|} - \pi_\theta(a|s)\right),\end{aligned}$$

where we have used $A^{\pi_\theta}(s,a) \geq 0$. Rearranging and using our assumption $\epsilon_{\text{opt}} \leq \lambda/(2|\mathcal{S}|\,|\mathcal{A}|)$,

$$\pi_\theta(a|s) \geq \frac{1}{|\mathcal{A}|} - \frac{\epsilon_{\text{opt}}|\mathcal{S}|}{\lambda} \geq \frac{1}{2|\mathcal{A}|}.$$

Solving for $A^{\pi_\theta}(s,a)$ in (0.5), we have:

$$\begin{aligned}A^{\pi_\theta}(s,a) &= \frac{1-\gamma}{d_\mu^{\pi_\theta}(s)}\left(\frac{1}{\pi_\theta(a|s)}\frac{\partial L_\lambda(\theta)}{\partial \theta_{s,a}} + \frac{\lambda}{|\mathcal{S}|}\left(1 - \frac{1}{\pi_\theta(a|s)|\mathcal{A}|}\right)\right) \\ &\leq \frac{1-\gamma}{d_\mu^{\pi_\theta}(s)}\left(2|\mathcal{A}|\epsilon_{\text{opt}} + \frac{\lambda}{|\mathcal{S}|}\right) \\ &\leq 2\frac{1-\gamma}{d_\mu^{\pi_\theta}(s)}\frac{\lambda}{|\mathcal{S}|} \\ &\leq 2\lambda/(\mu(s)|\mathcal{S}|),\end{aligned}$$

where the penultimate step uses $\epsilon_{\text{opt}} \leq \lambda/(2|\mathcal{S}|\,|\mathcal{A}|)$ and the final step uses $d_\mu^{\pi_\theta}(s) \geq (1-\gamma)\mu(s)$. This completes the proof. ■

The policy gradient ascent updates for $L_\lambda(\theta)$ are given by:

$$\theta^{(t+1)} = \theta^{(t)} + \eta \nabla_\theta L_\lambda(\theta^{(t)}). \tag{0.6}$$

By combining the above theorem with the convergence of gradient ascent to first order stationary points (Lemma 9.6), we obtain the following corollary.

Corollary 10.5. *(Iteration complexity with log barrier regularization) Let $\beta_\lambda := \frac{8\gamma}{(1-\gamma)^3} + \frac{2\lambda}{|\mathcal{S}|}$. Starting from any initial $\theta^{(0)}$, consider the updates* (0.6) *with $\lambda = \frac{\epsilon(1-\gamma)}{2\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty}$ and $\eta = 1/\beta_\lambda$. Then for all starting state distributions ρ, we have*

$$\min_{t<T}\left\{V^\star(\rho) - V^{(t)}(\rho)\right\} \leq \epsilon \quad \textit{whenever} \quad T \geq \frac{320|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)^6\,\epsilon^2}\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty^2.$$

The corollary shows the importance of balancing how the regularization parameter λ is set relative to the desired accuracy ϵ, as well as the importance of the initial distribution μ to obtain global optimality.

Proof:[of Corollary 10.5] Let β_λ be the smoothness of $L_\lambda(\theta)$. A valid upper bound on β_λ is:

$$\beta_\lambda = \frac{8\gamma}{(1-\gamma)^3} + \frac{2\lambda}{|\mathcal{S}|},$$

where we leave the proof as an exercise to the reader.

Using Theorem 10.4, the desired optimality gap ϵ will follow if we set $\lambda = \frac{\epsilon(1-\gamma)}{2\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty}$ and if $\|\nabla_\theta L_\lambda(\theta)\|_2 \leq \lambda/(2|\mathcal{S}|\,|\mathcal{A}|)$. In order to complete the proof, we need to bound the iteration complexity of making the gradient sufficiently small.

By Lemma 9.6, after T iterations of gradient ascent with stepsize of $1/\beta_\lambda$, we have

$$\min_{t \leq T} \left\|\nabla_\theta L_\lambda(\theta^{(t)})\right\|_2^2 \leq \frac{2\beta_\lambda(L_\lambda(\theta^\star) - L_\lambda(\theta^{(0)}))}{T} \leq \frac{2\beta_\lambda}{(1-\gamma)\,T}, \tag{0.7}$$

where β_λ is an upper bound on the smoothness of $L_\lambda(\theta)$. We seek to ensure

$$\epsilon_{\text{opt}} \leq \sqrt{\frac{2\beta_\lambda}{(1-\gamma)\,T}} \leq \frac{\lambda}{2|\mathcal{S}|\,|\mathcal{A}|}$$

Choosing $T \geq \frac{8\beta_\lambda\,|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)\,\lambda^2}$ satisfies the above inequality. Hence,

$$\begin{aligned}
\frac{8\beta_\lambda\,|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)\,\lambda^2} &\leq \frac{64\,|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)^4\,\lambda^2} + \frac{16\,|\mathcal{S}||\mathcal{A}|^2}{(1-\gamma)\,\lambda} \\
&\leq \frac{80\,|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)^4\,\lambda^2} \\
&= \frac{320\,|\mathcal{S}|^2|\mathcal{A}|^2}{(1-\gamma)^6\,\epsilon^2}\left\|\frac{d_\rho^{\pi^\star}}{\mu}\right\|_\infty^2
\end{aligned}$$

where we have used that $\lambda < 1$. This completes the proof. ■

Entropy vs. log barrier regularization. A commonly considered regularizer is the entropy, where the regularizer would be:

$$\frac{1}{|\mathcal{S}|}\sum_s H(\pi_\theta(\cdot|s)) = \frac{1}{|\mathcal{S}|}\sum_s \sum_a -\pi_\theta(a|s)\log\pi_\theta(a|s).$$

Note the entropy is far less aggressive in penalizing small probabilities, in comparison to the log barrier, which is equivalent to the relative entropy. In particular, the entropy regularizer is always bounded between 0 and $\log|\mathcal{A}|$, while the relative entropy (against the uniform distribution over actions), is bounded between 0 and infinity, where it tends to infinity as probabilities tend to 0. Here, it can be shown that he convergence rate is asymptotically $O(1\epsilon)$ (see Section 10.5) though it is unlikely that the convergence rate for this method is polynomial in other relevant quantities, including $|\mathcal{S}|$, $|\mathcal{A}|$, $1/(1-\gamma)$, and the distribution mismatch coefficient. The polynomial convergence rate using the log barrier (KL) regularizer crucially relies on the aggressive nature in which the relative entropy prevents small probabilities.

10.4 The Natural Policy Gradient

Observe that a policy constitutes a family of probability distributions $\{\pi_\theta(\cdot|s)|s \in \mathcal{S}\}$. We now consider a pre-conditioned gradient descent method based on this family of distributions. Recall that the Fisher information matrix

of a parameterized density $p_\theta(x)$ is defined as $\mathbb{E}_{x\sim p_\theta}\left[\nabla \log p_\theta(x) \nabla \log p_\theta(x)^\top\right]$. Now we let us define $\mathcal{F}_\rho^\theta$ as an (average) Fisher information matrix on the family of distributions $\{\pi_\theta(\cdot|s)|s\in\mathcal{S}\}$ as follows:

$$\mathcal{F}_\rho^\theta := \mathbb{E}_{s\sim d_\rho^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[(\nabla \log \pi_\theta(a|s)) \nabla \log \pi_\theta(a|s)^\top\right] .$$

Note that the average is under the state-action visitation frequencies. The NPG algorithm performs gradient updates in the geometry induced by this matrix as follows:

$$\theta^{(t+1)} = \theta^{(t)} + \eta F_\rho(\theta^{(t)})^\dagger \nabla_\theta V^{(t)}(\rho), \tag{0.8}$$

where $M^\dagger$ denotes the Moore-Penrose pseudoinverse of the matrix M.

Throughout this section, we restrict to using the initial state distribution $\rho \in \Delta(\mathcal{S})$ in our update rule in (0.8) (so our optimization measure μ and the performance measure ρ are identical). Also, we restrict attention to states $s \in \mathcal{S}$ reachable from ρ, since, without loss of generality, we can exclude states that are not reachable under this start state distribution[2].

For the softmax parameterization, this method takes a particularly convenient form; it can be viewed as a soft policy iteration update.

Lemma 10.6. *(Softmax NPG as soft policy iteration) For the softmax parameterization* (0.1)*, the NPG updates* (0.8) *take the form:*

$$\theta^{(t+1)} = \theta^{(t)} + \frac{\eta}{1-\gamma} A^{(t)} + \eta v \quad \textit{and} \quad \pi^{(t+1)}(a|s) = \pi^{(t)}(a|s) \frac{\exp\left(\eta A^{(t)}(s,a)/(1-\gamma)\right)}{Z_t(s)},$$

where $Z_t(s) = \sum_{a\in\mathcal{A}} \pi^{(t)}(a|s) \exp\left(\eta A^{(t)}(s,a)/(1-\gamma)\right)$. Here, v is only a state dependent offset (i.e. $v_{s,a} = c_s$ for some $c_s \in \mathbb{R}$ for each state s), and, owing to the normalization $Z_t(s)$, v has no effect on the update rule.

It is important to note that while the ascent direction was derived using the gradient $\nabla_\theta V^{(t)}(\rho)$, which depends on ρ, the NPG update rule actually has no dependence on the measure ρ. Furthermore, there is no dependence on the state distribution $d_\rho^{(t)}$, which is due to the pseudoinverse of the Fisher information cancelling out the effect of the state distribution in NPG.

Proof: By definition of the Moore-Penrose pseudoinverse, we have that $(\mathcal{F}_\rho^\theta)^\dagger \nabla V^{\pi_\theta}(\rho) = w_\star$ if an only if $w_\star$ is the minimum norm solution of:

$$\min_w \|\nabla V^{\pi_\theta}(\rho) - \mathcal{F}_\rho^\theta w\|^2 .$$

Let us first evaluate $\mathcal{F}_\rho^\theta w$. For the softmax policy parameterization, Lemma 0.1 implies:

$$w^\top \nabla_\theta \log \pi_\theta(a|s) = w_{s,a} - \sum_{a'\in\mathcal{A}} w_{s,a'} \pi_\theta(a'|s) := w_{s,a} - \overline{w}_s$$

where $\overline{w}_s$ is not a function of a. This implies that:

$$\begin{aligned}\mathcal{F}_\rho^\theta w &= \mathbb{E}_{s\sim d_\rho^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla \log \pi_\theta(a|s)\left(w^\top \nabla_\theta \log \pi_\theta(a|s)\right)\right] \\ &= \mathbb{E}_{s\sim d_\rho^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\nabla \log \pi_\theta(a|s)\left(w_{s,a} - \overline{w}_s\right)\right] = \mathbb{E}_{s\sim d_\rho^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[w_{s,a} \nabla \log \pi_\theta(a|s)\right],\end{aligned}$$

where the last equality uses that $\overline{w}_s$ is not a function of s. Again using the functional form of derivative of the softmax policy parameterization, we have:

$$\left[\mathcal{F}_\rho^\theta w\right]_{s',a'} = d^{\pi_\theta}(s')\pi_\theta(a'|s')\left(w_{s',a'} - \overline{w}_{s'}\right).$$

[2]Specifically, we restrict the MDP to the set of states $\{s\in\mathcal{S} : \exists \pi \text{ such that } d_\rho^\pi(s) > 0\}$.

This implies:

$$\|\nabla V^{\pi_\theta}(\rho) - \mathcal{F}_\rho^\theta w\|^2 = \sum_{s,a}\left(d^{\pi_\theta}(s)\pi_\theta(a|s)\left(\frac{1}{1-\gamma}A^{\pi_\theta}(s,a) - \left[\mathcal{F}_\rho^\theta w\right]_{s,a}\right)\right)^2$$
$$= \sum_{s,a}\left(d^{\pi_\theta}(s)\pi_\theta(a|s)\left(\frac{1}{1-\gamma}A^{\pi_\theta}(s,a) - w_{s,a} - \sum_{a'\in\mathcal{A}} w_{s,a'}\pi_\theta(a'|s)\right)\right)^2.$$

Due to that $w = \frac{1}{1-\gamma}A^{\pi_\theta}$ leads to 0 error, the above implies that all 0 error solutions are of the form $w = \frac{1}{1-\gamma}A^{\pi_\theta} + v$, where v is only a state dependent offset (i.e. $v_{s,a} = c_s$ for some $c_s \in \mathbb{R}$ for each state s). The first claim follows due to that the minimum norm solution is one of these solutions. The proof of the second claim now follows by the definition of the NPG update rule, along with that v has no effect on the update rule due to the normalization constant $Z_t(s)$. ■

We now see that this algorithm enjoys a dimension free convergence rate.

Theorem 10.7 (Global convergence for NPG)**.** *Suppose we run the NPG updates* (0.8) *using $\rho \in \Delta(\mathcal{S})$ and with $\theta^{(0)} = 0$. Fix $\eta > 0$. For all $T > 0$, we have:*

$$V^{(T)}(\rho) \geq V^*(\rho) - \frac{\log|\mathcal{A}|}{\eta T} - \frac{1}{(1-\gamma)^2 T}.$$

Note in the above the theorem that the NPG algorithm is directly applied to the performance measure $V^\pi(\rho)$, and the guarantees are also with respect to ρ. In particular, there is no distribution mismatch coefficient in the rate of convergence.

Now setting $\eta \geq (1-\gamma)^2 \log|\mathcal{A}|$, we see that NPG finds an ϵ-optimal policy in a number of iterations that is at most:

$$T \leq \frac{2}{(1-\gamma)^2\epsilon},$$

which has no dependence on the number of states or actions, despite the non-concavity of the underlying optimization problem.

The proof strategy we take borrows ideas from the classical multiplicative weights algorithm (see Section!10.5).

First, the following improvement lemma is helpful:

Lemma 10.8 (Improvement lower bound for NPG)**.** *For the iterates $\pi^{(t)}$ generated by the NPG updates* (0.8)*, we have for all starting state distributions μ*

$$V^{(t+1)}(\mu) - V^{(t)}(\mu) \geq \frac{(1-\gamma)}{\eta}\mathbb{E}_{s\sim\mu}\log Z_t(s) \geq 0.$$

Proof: First, let us show that $\log Z_t(s) \geq 0$. To see this, observe:

$$\log Z_t(s) = \log\sum_a \pi^{(t)}(a|s)\exp(\eta A^{(t)}(s,a)/(1-\gamma))$$
$$\geq \sum_a \pi^{(t)}(a|s)\log\exp(\eta A^{(t)}(s,a)/(1-\gamma)) = \frac{\eta}{1-\gamma}\sum_a \pi^{(t)}(a|s)A^{(t)}(s,a) = 0.$$

where we have used Jensen's inequality on the concave function $\log x$ and that $\sum_a \pi^{(t)}(a|s)A^{(t)}(s,a) = 0$. By the

performance difference lemma,

$$\begin{aligned}
V^{(t+1)}(\mu) - V^{(t)}(\mu) &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{(t+1)}} \sum_a \pi^{(t+1)}(a|s) A^{(t)}(s,a) \\
&= \frac{1}{\eta}\mathbb{E}_{s\sim d_\mu^{(t+1)}} \sum_a \pi^{(t+1)}(a|s) \log \frac{\pi^{(t+1)}(a|s) Z_t(s)}{\pi^{(t)}(a|s)} \\
&= \frac{1}{\eta}\mathbb{E}_{s\sim d_\mu^{(t+1)}} \mathrm{KL}(\pi_s^{(t+1)} || \pi_s^{(t)}) + \frac{1}{\eta}\mathbb{E}_{s\sim d_\mu^{(t+1)}} \log Z_t(s) \\
&\geq \frac{1}{\eta}\mathbb{E}_{s\sim d_\mu^{(t+1)}} \log Z_t(s) \geq \frac{1-\gamma}{\eta}\mathbb{E}_{s\sim\mu} \log Z_t(s),
\end{aligned}$$

where the last step uses that $d_\mu^{(t+1)} \geq (1-\gamma)\mu$ (by (0.5)) and that $\log Z_t(s) \geq 0$. ■

With this lemma, we now prove Theorem 10.7.

Proof:[of Theorem 10.7] Since ρ is fixed, we use $d^\star$ as shorthand for $d_\rho^{\pi^\star}$; we also use π_s as shorthand for the vector of $\pi(\cdot|s)$. By the performance difference lemma (Lemma 1.16),

$$\begin{aligned}
V^{\pi^\star}(\rho) - V^{(t)}(\rho) &= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^\star} \sum_a \pi^\star(a|s) A^{(t)}(s,a) \\
&= \frac{1}{\eta}\mathbb{E}_{s\sim d^\star} \sum_a \pi^\star(a|s) \log \frac{\pi^{(t+1)}(a|s) Z_t(s)}{\pi^{(t)}(a|s)} \\
&= \frac{1}{\eta}\mathbb{E}_{s\sim d^\star} \left(\mathrm{KL}(\pi_s^\star || \pi_s^{(t)}) - \mathrm{KL}(\pi_s^\star || \pi_s^{(t+1)}) + \sum_a \pi^*(a|s) \log Z_t(s) \right) \\
&= \frac{1}{\eta}\mathbb{E}_{s\sim d^\star} \left(\mathrm{KL}(\pi_s^\star || \pi_s^{(t)}) - \mathrm{KL}(\pi_s^\star || \pi_s^{(t+1)}) + \log Z_t(s) \right),
\end{aligned}$$

where we have used the closed form of our updates from Lemma 10.6 in the second step.

By applying Lemma 10.8 with $d^\star$ as the starting state distribution, we have:

$$\frac{1}{\eta}\mathbb{E}_{s\sim d^\star} \log Z_t(s) \leq \frac{1}{1-\gamma}\left(V^{(t+1)}(d^\star) - V^{(t)}(d^\star) \right)$$

which gives us a bound on $\mathbb{E}_{s\sim d^\star} \log Z_t(s)$.

Using the above equation and that $V^{(t+1)}(\rho) \geq V^{(t)}(\rho)$ (as $V^{(t+1)}(s) \geq V^{(t)}(s)$ for all states s by Lemma 10.8), we have:

$$\begin{aligned}
V^{\pi^\star}(\rho) - V^{(T-1)}(\rho) &\leq \frac{1}{T}\sum_{t=0}^{T-1} (V^{\pi^\star}(\rho) - V^{(t)}(\rho)) \\
&= \frac{1}{\eta T}\sum_{t=0}^{T-1} \mathbb{E}_{s\sim d^\star}(\mathrm{KL}(\pi_s^\star || \pi_s^{(t)}) - \mathrm{KL}(\pi_s^\star || \pi_s^{(t+1)})) + \frac{1}{\eta T}\sum_{t=0}^{T-1} \mathbb{E}_{s\sim d^\star} \log Z_t(s) \\
&\leq \frac{\mathbb{E}_{s\sim d^\star}\mathrm{KL}(\pi_s^\star || \pi^{(0)})}{\eta T} + \frac{1}{(1-\gamma)T}\sum_{t=0}^{T-1} \left(V^{(t+1)}(d^\star) - V^{(t)}(d^\star) \right) \\
&= \frac{\mathbb{E}_{s\sim d^\star}\mathrm{KL}(\pi_s^\star || \pi^{(0)})}{\eta T} + \frac{V^{(T)}(d^\star) - V^{(0)}(d^\star)}{(1-\gamma)T} \\
&\leq \frac{\log|\mathcal{A}|}{\eta T} + \frac{1}{(1-\gamma)^2 T}.
\end{aligned}$$

The proof is completed using that $V^{(T)}(\rho) \geq V^{(T-1)}(\rho)$. ■

10.5 Bibliographic Remarks and Further Readings

The natural policy gradient method was originally presented in [Kakade, 2001]; a number of arguments for this method have been provided based on information geometry [Kakade, 2001, Bagnell and Schneider, 2003, Peters and Schaal, 2008].

The convergence rates in this chapter are largely derived from [Agarwal et al., 2020c]. The proof strategy for the NPG analysis has origins in the online regret framework in changing MDPs [Even-Dar et al., 2009], which would result in a worst rate in comparison to [Agarwal et al., 2020c]. This observation that the proof strategy from [Even-Dar et al., 2009] provided a convergence rate for the NPG was made in [Neu et al., 2017]. The faster NPG rate we present here is due to [Agarwal et al., 2020c]. The analysis of the MD-MPI algorithm [Geist et al., 2019] also implies a $O(1/T)$ rate for the NPG, though with worse dependencies on other parameters.

Building on ideas in [Agarwal et al., 2020c], [Mei et al., 2020] showed that, for the softmax policy class, both the gradient ascent and entropy regularized gradient ascent asymptotically converge at a $O(1/t)$; it is unlikely these methods are have finite rate which are polynomial in other quantities (such as the $|\mathcal{S}|$, $|\mathcal{A}|$, $1/(1-\gamma)$, and the distribution mismatch coefficient).

[Mnih et al., 2016] introduces the entropy regularizer (also see [Ahmed et al., 2019] for a more detailed empirical investigation).

Chapter 11

Function Approximation and the NPG

We now analyze the case of using parametric policy classes:

$$\Pi = \{\pi_\theta \mid \theta \in \mathbb{R}^d\},$$

where Π may not contain all stochastic policies (and it may not even contain an optimal policy). In contrast with the tabular results in the previous sections, the policy classes that we are often interested in are not fully expressive, e.g. $d \ll |\mathcal{S}||\mathcal{A}|$ (indeed $|\mathcal{S}|$ or $|\mathcal{A}|$ need not even be finite for the results in this section); in this sense, we are in the regime of function approximation.

We focus on obtaining *agnostic* results, where we seek to do as well as the best policy in this class (or as well as some other comparator policy). While we are interested in a solution to the (unconstrained) policy optimization problem

$$\max_{\theta \in \mathbb{R}^d} V^{\pi_\theta}(\rho),$$

(for a given initial distribution ρ), we will see that optimization with respect to a different distribution will be helpful, just as in the tabular case,

We will consider variants of the NPG update rule (0.8):

$$\theta \leftarrow \theta + \eta F_\rho(\theta)^\dagger \nabla_\theta V^\theta(\rho)\,. \tag{0.1}$$

Our analysis will leverage a close connection between the NPG update rule (0.8) with the notion of *compatible function approximation*. We start by formalizing this connection. The compatible function approximation error also provides a measure of the expressivity of our parameterization, allowing us to quantify the relevant notion of approximation error for the NPG algorithm.

The main results in this chapter establish the effectiveness of NPG updates where there is error both due to statistical estimation (where we may not use exact gradients) and approximation (due to using a parameterized function class). We will see an precise estimation/approximation decomposition based on the compatible function approximation error.

The presentation in this chapter largely follows the results in [Agarwal et al., 2020c].

11.1 Compatible function approximation and the NPG

We now introduce the notion of *compatible function approximation*, which both provides some intuition with regards to policy gradient methods and it will help us later on with regards to characterizing function approximation.

Lemma 11.1 (Gradients and compatible function approximation)**.** *Let $w^\star$ denote the following minimizer:*

$$w^\star \in \mathbb{E}_{s\sim d_\mu^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[\left(A^{\pi_\theta}(s,a) - w \cdot \nabla_\theta \log \pi_\theta(a|s) \right)^2 \right],$$

where the squared error above is referred to as the compatible function approximation*. Denote the best linear predictor of $A^{\pi_\theta}(s,a)$ using $\nabla_\theta \log \pi_\theta(a|s)$ by $\widehat{A}^{\pi_\theta}(s,a)$, i.e.*

$$\widehat{A}^{\pi_\theta}(s,a) := w^\star \cdot \nabla_\theta \log \pi_\theta(a|s).$$

We have that:

$$\nabla_\theta V^{\pi_\theta}(\mu) = \frac{1}{1-\gamma} \mathbb{E}_{s\sim d_\mu^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[\nabla_\theta \log \pi_\theta(a|s) \widehat{A}^{\pi_\theta}(s,a) \right].$$

Proof: The first order optimality conditions for $w^\star$ imply

$$\mathbb{E}_{s\sim d_\mu^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[\left(A^{\pi_\theta}(s,a) - w^\star \cdot \nabla_\theta \log \pi_\theta(a|s) \right) \nabla_\theta \log \pi_\theta(a|s) \right] = 0 \tag{0.2}$$

Rearranging and using the definition of $\widehat{A}^{\pi_\theta}(s,a)$,

$$\begin{aligned} \nabla_\theta V^{\pi_\theta}(\mu) &= \frac{1}{1-\gamma} \mathbb{E}_{s\sim d_\mu^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[\nabla_\theta \log \pi_\theta(a|s) A^{\pi_\theta}(s,a) \right] \\ &= \frac{1}{1-\gamma} \mathbb{E}_{s\sim d_\mu^{\pi_\theta}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} \left[\nabla_\theta \log \pi_\theta(a|s) \widehat{A}^{\pi_\theta}(s,a) \right], \end{aligned}$$

which completes the proof. ■

The next lemma shows that the weight vector above is precisely the NPG ascent direction. Precisely,

Lemma 11.2. *We have that:*

$$F_\rho(\theta)^\dagger \nabla_\theta V^\theta(\rho) = \frac{1}{1-\gamma} w^\star, \tag{0.3}$$

where $w^\star$ is a minimizer of the following regression problem:

$$w^\star \in \mathrm{argmin}_w \, \mathbb{E}_{s\sim d_\rho^{\pi_\theta}, a\sim\pi_\theta(\cdot|s)} \left[(w^\top \nabla_\theta \log \pi_\theta(\cdot|s) - A^{\pi_\theta}(s,a))^2 \right] .$$

Proof: The above is a straightforward consequence of the first order optimality conditions (see Equation 0.2). Specifically, Equation 0.2, along with the advantage expression for the policy gradient (see Theorem 9.4), imply that $w^\star$ must satisfy:

$$\nabla_\theta V^\theta(\rho) = \frac{1}{1-\gamma} F_\rho(\theta) w^\star$$

which completes the proof. ■

This lemma implies that we might write the NPG update rule as:

$$\theta \leftarrow \theta + \frac{\eta}{1-\gamma} w^\star .$$

where $w^\star$ is minimizer of the compatible function approximation error (which depends on θ.

The above regression problem can be viewed as "compatible" function approximation: we are approximating $A^{\pi_\theta}(s,a)$ using the $\nabla_\theta \log \pi_\theta(\cdot|s)$ as features. We also consider a variant of the above update rule, Q-NPG, where instead of using advantages in the above regression we use the Q-values. This viewpoint provides a methodology for approximate updates, where we can solve the relevant regression problems with samples.

11.2 Examples: NPG and Q-NPG

In practice, the most common policy classes are of the form:

$$\Pi = \left\{ \pi_\theta(a|s) = \frac{\exp\left(f_\theta(s,a)\right)}{\sum_{a' \in \mathcal{A}} \exp\left(f_\theta(s,a')\right)} \;\middle|\; \theta \in \mathbb{R}^d \right\}, \tag{0.4}$$

where f_θ is a differentiable function. For example, the tabular softmax policy class is one where $f_\theta(s,a) = \theta_{s,a}$. Typically, f_θ is either a linear function or a neural network. Let us consider the NPG algorithm, and a variant Q-NPG, in each of these two cases.

11.2.1 Log-linear Policy Classes and Soft Policy Iteration

For any state-action pair (s,a), suppose we have a feature mapping $\phi_{s,a} \in \mathbb{R}^d$. Each policy in the log-linear policy class is of the form:

$$\pi_\theta(a|s) = \frac{\exp(\theta \cdot \phi_{s,a})}{\sum_{a' \in \mathcal{A}} \exp(\theta \cdot \phi_{s,a'})},$$

with $\theta \in \mathbb{R}^d$. Here, we can take $f_\theta(s,a) = \theta \cdot \phi_{s,a}$.

With regards to compatible function approximation for the log-linear policy class, we have:

$$\nabla_\theta \log \pi_\theta(a|s) = \overline{\phi}^{\,\theta}_{s,a}, \text{ where } \quad \overline{\phi}^{\,\theta}_{s,a} = \phi_{s,a} - \mathbb{E}_{a' \sim \pi_\theta(\cdot|s)}[\phi_{s,a'}],$$

that is, $\overline{\phi}^{\,\theta}_{s,a}$ is the centered version of $\phi_{s,a}$. With some abuse of notation, we accordingly also define $\bar{\phi}^\pi$ for any policy π. Here, using (0.3), the NPG update rule (0.1) is equivalent to:

$$\text{NPG:} \quad \theta \leftarrow \theta + \eta w_\star, \qquad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} \left[\left(A^{\pi_\theta}(s,a) - w \cdot \overline{\phi}^{\,\theta}_{s,a} \right)^2 \right].$$

(We have rescaled the learning rate η in comparison to (0.1)). Note that we recompute $w_\star$ for every update of θ. Here, the compatible function approximation error measures the expressivity of our parameterization in how well linear functions of the parameterization can capture the policy's advantage function.

We also consider a variant of the NPG update rule (0.1), termed *Q-NPG*, where:

$$Q\text{-NPG:} \quad \theta \leftarrow \theta + \eta w_\star, \qquad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} \left[\left(Q^{\pi_\theta}(s,a) - w \cdot \phi_{s,a} \right)^2 \right].$$

Note we do not center the features for Q-NPG; observe that $Q^\pi(s,a)$ is also not 0 in expectation under $\pi(\cdot|s)$, unlike the advantage function.

(NPG/Q-NPG and Soft-Policy Iteration) We now see how we can view both NPG and Q-NPG as an incremental (soft) version of policy iteration, just as in Lemma 10.6 for the tabular case. Rather than writing the update rule in terms of the parameter θ, we can write an equivalent update rule directly in terms of the (log-linear) policy π:

$$\text{NPG:} \;\; \pi(a|s) \leftarrow \pi(a|s) \exp(w_\star \cdot \phi_{s,a}) / Z_s, \quad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^\pi, a \sim \pi(\cdot|s)} \left[\left(A^\pi(s,a) - w \cdot \overline{\phi}^{\,\pi}_{s,a} \right)^2 \right],$$

where Z_s is normalization constant. While the policy update uses the original features ϕ instead of $\overline{\phi}^{\,\pi}$, whereas the quadratic error minimization is terms of the centered features $\overline{\phi}^{\,\pi}$, this distinction is not relevant due to that we may also instead use $\overline{\phi}^{\,\pi}$ (in the policy update) which would result in an equivalent update; the normalization makes the update invariant to (constant) translations of the features. Similarly, an equivalent update for Q-NPG, where we update π directly rather than θ, is:

$$Q\text{-NPG:} \;\; \pi(a|s) \leftarrow \pi(a|s) \exp(w_\star \cdot \phi_{s,a}) / Z_s, \quad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^\pi, a \sim \pi(\cdot|s)} \left[\left(Q^\pi(s,a) - w \cdot \phi_{s,a} \right)^2 \right].$$

(On the equivalence of NPG and Q-NPG) If it is the case that the compatible function approximation error is 0, then it straightforward to verify that the NPG and Q-NPG are equivalent algorithms, in that their corresponding policy updates will be equivalent to each other.

11.2.2 Neural Policy Classes

Now suppose $f_\theta(s,a)$ is a neural network parameterized by $\theta \in \mathbb{R}^d$, where the policy class Π is of form in (0.4). Observe:

$$\nabla_\theta \log \pi_\theta(a|s) = g_\theta(s,a), \text{where} \quad g_\theta(s,a) = \nabla_\theta f_\theta(s,a) - \mathbb{E}_{a' \sim \pi_\theta(\cdot|s)}[\nabla_\theta f_\theta(s,a')],$$

and, using (0.3), the NPG update rule (0.1) is equivalent to:

$$\text{NPG:} \quad \theta \leftarrow \theta + \eta w_\star, \qquad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} \Big[\big(A^{\pi_\theta}(s,a) - w \cdot g_\theta(s,a) \big)^2 \Big]$$

(Again, we have rescaled the learning rate η in comparison to (0.1)).

The Q-NPG variant of this update rule is:

$$Q\text{-NPG:} \quad \theta \leftarrow \theta + \eta w_\star, \qquad w_\star \in \text{argmin}_w \, \mathbb{E}_{s \sim d_\rho^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} \Big[\big(Q^{\pi_\theta}(s,a) - w \cdot \nabla_\theta f_\theta(s,a) \big)^2 \Big].$$

11.3 The NPG "Regret Lemma"

It is helpful for us to consider NPG more abstractly, as an update rule of the form

$$\theta^{(t+1)} = \theta^{(t)} + \eta w^{(t)}. \tag{0.5}$$

We will now provide a lemma where $w^{(t)}$ is an *arbitrary* (bounded) sequence, which will be helpful when specialized.

Recall a function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be β-smooth if for all $x, x' \in \mathbb{R}^d$:

$$\|\nabla f(x) - \nabla f(x')\|_2 \leq \beta \|x - x'\|_2 \,,$$

and, due to Taylor's theorem, recall that this implies:

$$\left| f(x') - f(x) - \nabla f(x) \cdot (x' - x) \right| \leq \frac{\beta}{2} \|x' - x\|_2^2 \,. \tag{0.6}$$

The following analysis of NPG is draws close connections to the mirror-descent approach used in online learning (see Section 11.6), which motivates us to refer to it as a "regret lemma".

Lemma 11.3. *(NPG Regret Lemma) Fix a comparison policy $\widetilde{\pi}$ and a state distribution ρ. Assume for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$ that $\log \pi_\theta(a|s)$ is a β-smooth function of θ. Consider the update rule* (0.5)*, where $\pi^{(0)}$ is the uniform distribution (for all states) and where the sequence of weights $w^{(0)}, \ldots, w^{(T)}$, satisfies $\|w^{(t)}\|_2 \leq W$ (but is otherwise arbitrary). Define:*

$$\text{err}_t = \mathbb{E}_{s \sim \widetilde{d}} \mathbb{E}_{a \sim \widetilde{\pi}(\cdot|s)} \Big[A^{(t)}(s,a) - w^{(t)} \cdot \nabla_\theta \log \pi^{(t)}(a|s) \Big].$$

We have that:

$$\min_{t<T} \Big\{ V^{\widetilde{\pi}}(\rho) - V^{(t)}(\rho) \Big\} \leq \frac{1}{1-\gamma} \left(\frac{\log |\mathcal{A}|}{\eta T} + \frac{\eta \beta W^2}{2} + \frac{1}{T} \sum_{t=0}^{T-1} \text{err}_t \right).$$

This lemma is the key tool in understanding the role of function approximation of various algorithms. We will consider one example in detail with regards to the log-linear policy class (from Example 9.2).

Note that when $\mathrm{err}_t = 0$, as will be the case with the (tabular) softmax policy class with exact gradients, we obtain a convergence rate of $O(\sqrt{1/T})$ using a learning rate of $\eta = O(\sqrt{1/T})$. Note that this is slower than the faster rate of $O(1/T)$, provided in Theorem 10.7. Obtaining a bound that leads to a faster rate in the setting with errors requires more complicated dependencies on err_t than those stated above.

Proof: By smoothness (see (0.6)),

$$\begin{aligned}\log \frac{\pi^{(t+1)}(a|s)}{\pi^{(t)}(a|s)} &\geq \nabla_\theta \log \pi^{(t)}(a|s) \cdot \left(\theta^{(t+1)} - \theta^{(t)}\right) - \frac{\beta}{2}\|\theta^{(t+1)} - \theta^{(t)}\|_2^2 \\ &= \eta \nabla_\theta \log \pi^{(t)}(a|s) \cdot w^{(t)} - \eta^2 \frac{\beta}{2}\|w^{(t)}\|_2^2.\end{aligned}$$

We use $\widetilde{d}$ as shorthand for $d_\rho^{\widetilde{\pi}}$ (note ρ and $\widetilde{\pi}$ are fixed); for any policy π, we also use π_s as shorthand for the vector $\pi(\cdot|s)$. Using the performance difference lemma (Lemma 1.16),

$$\begin{aligned}&\mathbb{E}_{s\sim\widetilde{d}}\Big(\mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t)}) - \mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t+1)})\Big) \\ &= \mathbb{E}_{s\sim\widetilde{d}}\mathbb{E}_{a\sim\widetilde{\pi}(\cdot|s)}\left[\log \frac{\pi^{(t+1)}(a|s)}{\pi^{(t)}(a|s)}\right] \\ &\geq \eta \mathbb{E}_{s\sim\widetilde{d}}\mathbb{E}_{a\sim\widetilde{\pi}(\cdot|s)}\left[\nabla_\theta \log \pi^{(t)}(a|s) \cdot w^{(t)}\right] - \eta^2\frac{\beta}{2}\|w^{(t)}\|_2^2 \qquad \text{(using previous display)} \\ &= \eta \mathbb{E}_{s\sim\widetilde{d}}\mathbb{E}_{a\sim\widetilde{\pi}(\cdot|s)}\left[A^{(t)}(s,a)\right] - \eta^2\frac{\beta}{2}\|w^{(t)}\|_2^2 \\ &\quad + \eta \mathbb{E}_{s\sim\widetilde{d}}\mathbb{E}_{a\sim\widetilde{\pi}(\cdot|s)}\left[\nabla_\theta \log \pi^{(t)}(a|s) \cdot w^{(t)} - A^{(t)}(s,a)\right] \\ &= (1-\gamma)\eta\Big(V^{\widetilde{\pi}}(\rho) - V^{(t)}(\rho)\Big) - \eta^2\frac{\beta}{2}\|w^{(t)}\|_2^2 - \eta\,\mathrm{err}_t\end{aligned}$$

Rearranging, we have:

$$V^{\widetilde{\pi}}(\rho) - V^{(t)}(\rho) \leq \frac{1}{1-\gamma}\left(\frac{1}{\eta}\mathbb{E}_{s\sim\widetilde{d}}\Big(\mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t)}) - \mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t+1)})\Big) + \frac{\eta\beta}{2}W^2 + \mathrm{err}_t\right)$$

Proceeding,

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}(V^{\widetilde{\pi}}(\rho) - V^{(t)}(\rho)) &\leq \frac{1}{\eta T(1-\gamma)}\sum_{t=0}^{T-1}\mathbb{E}_{s\sim\widetilde{d}}(\mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t)}) - \mathrm{KL}(\widetilde{\pi}_s||\pi_s^{(t+1)})) \\ &\quad + \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\left(\frac{\eta\beta W^2}{2} + \mathrm{err}_t\right) \\ &\leq \frac{\mathbb{E}_{s\sim\widetilde{d}}\mathrm{KL}(\widetilde{\pi}_s||\pi^{(0)})}{\eta T(1-\gamma)} + \frac{\eta\beta W^2}{2(1-\gamma)} + \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\mathrm{err}_t \\ &\leq \frac{\log|\mathcal{A}|}{\eta T(1-\gamma)} + \frac{\eta\beta W^2}{2(1-\gamma)} + \frac{1}{T(1-\gamma)}\sum_{t=0}^{T-1}\mathrm{err}_t,\end{aligned}$$

which completes the proof. ■

11.4 Q-NPG: Performance Bounds for Log-Linear Policies

For a state-action distribution υ, define:

$$L(w; \theta, \upsilon) := \mathbb{E}_{s,a\sim\upsilon}\left[\left(Q^{\pi_\theta}(s,a) - w \cdot \phi_{s,a}\right)^2\right].$$

The iterates of the Q-NPG algorithm can be viewed as minimizing this loss under some (changing) distribution υ.

We now specify an approximate version of Q-NPG. It is helpful to consider a slightly more general version of the algorithm in the previous section, where instead of optimizing under a starting state distribution ρ, we have a different starting *state-action* distribution ν. The motivation for this is similar in spirit to our log barrier regularization: we seek to maintain exploration (and estimation) over the action space even if the current policy does not have coverage over the action space.

Analogous to the definition of the state visitation measure, d^π_μ, we can define a visitation measure over states *and* actions induced by following π after $s_0, a_0 \sim \nu$. We overload notation using d^π_ν to also refer to the state-action visitation measure; precisely,

$$d^\pi_\nu(s,a) := (1-\gamma)\mathbb{E}_{s_0,a_0\sim\nu}\sum_{t=0}^{\infty}\gamma^t \Pr^\pi(s_t = s, a_t = a | s_0, a_0) \tag{0.7}$$

where $\Pr^\pi(s_t = s, a_t = a|s_0, a_0)$ is the probability that $s_t = s$ and $a_t = a$, after starting at state s_0, taking action a_0, and following π thereafter. While we overload notation for visitation distributions ($d^\pi_\mu(s)$ and $d^\pi_\nu(s,a)$) for notational convenience, note that the state-action measure d^π_ν uses the subscript ν, which is a state-action measure.

Q-NPG will be defined with respect to the *on-policy* state action measure starting with $s_0, a_0 \sim \nu$. As per our convention, we define

$$d^{(t)} := d^{\pi^{(t)}}_\nu.$$

The approximate version of this algorithm is:

$$\text{Approx. Q-NPG:}\quad \theta^{(t+1)} = \theta^{(t)} + \eta w^{(t)}, \qquad w^{(t)} \approx \operatorname{argmin}_{\|w\|_2\le W} L(w; \theta^{(t)}, d^{(t)}), \tag{0.8}$$

where the above update rule also permits us to constrain the norm of the update direction $w^{(t)}$ (alternatively, we could use ℓ_2 regularization as is also common in practice). The exact minimizer is denoted as:

$$w^{(t)}_\star \in \operatorname{argmin}_{\|w\|_2\le W} L(w; \theta^{(t)}, d^{(t)}).$$

Note that $w^{(t)}_\star$ depends on the current parameter $\theta^{(t)}$.

Our analysis will take into account both the *excess risk* (often also referred to as estimation error) and the *approximation error*. The standard approximation-estimation error decomposition is as follows:

$$L(w^{(t)}; \theta^{(t)}, d^{(t)}) \;=\; \underbrace{L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w^{(t)}_\star; \theta^{(t)}, d^{(t)})}_{\text{Excess risk}} + \underbrace{L(w^{(t)}_\star; \theta^{(t)}, d^{(t)})}_{\text{Approximation error}}$$

Using a sample based approach, we would expect $\epsilon_{\text{stat}} = O(1/\sqrt{N})$ or better, where N is the number of samples used to estimate. $w^{(t)}_\star$ In constrast, the approximation error is due to modeling error, and does not tend to 0 with more samples. We will see how these two errors have strikingly different impact on our final performance bound.

Note that we have already considered two cases where $\epsilon_{\text{approx}} = 0$. For the tabular softmax policy class, it is immediate that $\epsilon_{\text{approx}} = 0$. A more interesting example (where the state and action space could be infinite) is provided by the linear parameterized MDP model from Chapter 7. Here, provided that we use the log-linear policy class (see

Section 11.2.1) with features corresponding to the linear MDP features, it is straightforward to see that $\epsilon_{\text{approx}} = 0$ for this log-linear policy class. More generally, we will see the effect of model misspecification in our performance bounds.

We make the following assumption on these errors:

Assumption 11.4 (Approximation/estimation error bounds)**.** Let $w^{(0)}, w^{(1)}, \ldots w^{(T-1)}$ be the sequence of iterates used by the Q-NPG algorithm Suppose the following holds for all $t < T$:

1. (*Excess risk*) Suppose the estimation error is bounded as follows:

$$L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{stat}}$$

2. (*Approximation error*) Suppose the approximation error is bounded as follows:

$$L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{approx}}.$$

We will also see how, with regards to our estimation error, we will need a far more mild notion of coverage. Here, with respect to any state-action distribution υ, define:

$$\Sigma_\upsilon = \mathbb{E}_{s,a\sim\upsilon}\left[\phi_{s,a}\phi_{s,a}^\top\right].$$

We make a the following conditioning assumption:

Assumption 11.5 (Relative condition number)**.** Fix a state distribution ρ (this will be what ultimately be the performance measure that we seek to optimize). Consider an arbitrary comparator policy $\pi^\star$ (not necessarily an optimal policy). With respect to $\pi^\star$, define the state-action measure $d^\star$ as

$$d^\star(s,a) = d_\rho^{\pi^\star}(s) \cdot \text{Unif}_{\mathcal{A}}(a)$$

i.e. $d^\star$ samples states from the comparators state visitation measure, $d_\rho^{\pi^\star}$ and actions from the uniform distribution. Define

$$\sup_{w\in\mathbb{R}^d} \frac{w^\top \Sigma_{d^\star} w}{w^\top \Sigma_\nu w} = \kappa,$$

and assume that κ is finite.

We later discuss why it is reasonable to expect that κ is not a quantity related to the size of the state space.

The main result of this chapter shows how the approximation error, the excess risk, and the conditioning, determine the final performance.

Theorem 11.6. *Fix a state distribution ρ; a state-action distribution ν; an arbitrary comparator policy $\pi^\star$ (not necessarily an optimal policy). Suppose Assumption 11.5 holds with respect to these choices and that $\|\phi_{s,a}\|_2 \leq B$ for all s, a. Suppose the Q-NPG update rule (in* (0.8)*) starts with $\theta^{(0)} = 0$, $\eta = \sqrt{2\log|\mathcal{A}|/(B^2W^2T)}$, and the (random) sequence of iterates satisfies Assumption 11.4. We have that:*

$$\mathbb{E}\left[\min_{t<T}\left\{V^{\pi^\star}(\rho) - V^{(t)}(\rho)\right\}\right] \leq \frac{BW}{1-\gamma}\sqrt{\frac{2\log|\mathcal{A}|}{T}} + \sqrt{\frac{4|\mathcal{A}|}{(1-\gamma)^3}\left(\kappa\cdot\epsilon_{\text{stat}} + \left\|\frac{d^\star}{\nu}\right\|_\infty \cdot \epsilon_{\text{approx}}\right)}.$$

Note when $\epsilon_{\text{approx}} = 0$, our convergence rate is $O(\sqrt{1/T})$ plus a term that depends on the excess risk; hence, provided we obtain enough samples, then ϵ_{stat} will also tend to 0, and we will be competitive with the comparison policy $\pi^\star$.

The above also shows the striking difference between the effects of estimation error and approximation error. A few remarks are now in order.

Transfer learning, distribution shift, and the approximation error. In large scale problems, the worst case distribution mismatch factor $\left\| \frac{d^\star}{\nu} \right\|_\infty$ is unlikely to be small. However, this factor is ultimately due to transfer learning. Our approximation error is with respect to the fitting distribution $d^{(t)}$, where we assume that $L(w_\star^{(t)}; \theta^{(t)}, d^{(t)}) \leq \epsilon_{\text{approx}}$. As the proof will show, the relevant notion of approximation error will be $L(w_\star^{(t)}; \theta^{(t)}, d^\star)$, where $d^\star$ is the *fixed* comparators measure. In others words, to get a good performance bound we need to successfully have low transfer learning error to the fixed measure $d^\star$. Furthermore, in many modern machine learning applications, this error is often is favorable, in that it is substantially better than worst case theory might suggest.

See Section 11.6 for further remarks on this point.

Dimension dependence in κ and the importance of ν. It is reasonable to think about κ as being dimension dependent (or worse), but it is not necessarily related to the size of the state space. For example, if $\|\phi_{s,a}\|_2 \leq B$, then $\kappa \leq \frac{B^2}{\sigma_{\min}(\mathbb{E}_{s,a\sim\nu}[\phi_{s,a}\phi_{s,a}^\top])}$ though this bound may be pessimistic. Here, we also see the importance of choice of ν in having a small (relative) condition number; in particular, this is the motivation for considering the generalization which allows for a starting state-action distribution ν vs. just a starting state distribution μ (as we did in the tabular case). Roughly speaking, we desire a ν which provides good coverage over the features. As the following lemma shows, there always exists a universal distribution ν, which can be constructed only with knowledge of the feature set (without knowledge of $d^\star$), such that $\kappa \leq d$.

Lemma 11.7. *($\kappa \leq d$ is always possible) Let $\Phi = \{\phi(s,a) | (s,a) \in \mathcal{S} \times \mathcal{A}\} \subset \mathbb{R}^d$ and suppose Φ is a compact set. There always exists a state-action distribution ν, which is supported on at most d^2 state-action pairs and which can be constructed only with knowledge of Φ (without knowledge of the MDP or $d^\star$), such that:*

$$\kappa \leq d.$$

Proof: The distribution can be found through constructing the minimal volume ellipsoid containing Φ, i.e. the Lowner-John ellipsoid. **To be added...** ■

Direct policy optimization vs. approximate value function programming methods Part of the reason for the success of the direct policy optimization approaches is to due their more mild dependence on the approximation error. Here, our theoretical analysis has a dependence on a distribution mismatch coefficient, $\left\| \frac{d^\star}{\nu} \right\|_\infty$, while approximate value function methods have even worse dependencies. See Chapter 3. As discussed earlier and as can be seen in the regret lemma (Lemma 11.3), the distribution mismatch coefficient is due to that the relevant error for NPG is a transfer error notion to a *fixed* comparator distribution, while approximate value function methods have more stringent conditions where the error has to be small under, essentially, the distribution of *any* other policy.

11.4.1 Analysis

Proof: (of Theorem 11.6) For the log-linear policy class, due to that the feature mapping ϕ satisfies $\|\phi_{s,a}\|_2 \leq B$, then it is not difficult to verify that $\log \pi_\theta(a|s)$ is a B^2-smooth function. Using this and the NPG regret lemma (Lemma 11.3), we have:

$$\min_{t<T} \left\{ V^{\pi^\star}(\rho) - V^{(t)}(\rho) \right\} \leq \frac{BW}{1-\gamma} \sqrt{\frac{2\log|\mathcal{A}|}{T}} + \frac{1}{(1-\gamma)T} \sum_{t=0}^{T-1} \text{err}_t \,.$$

where we have used our setting of η.

We make the following decomposition of err_t:

$$\begin{aligned}\mathrm{err}_t = \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[A^{(t)}(s,a) - w^{(t)}_\star \cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big] \\ + \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[\big(w^{(t)}_\star - w^{(t)}\big)\cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big].\end{aligned}$$

For the first term, using that $\nabla_\theta \log \pi_\theta(a|s) = \phi_{s,a} - \mathbb{E}_{a'\sim\pi_\theta(\cdot|s)}[\phi_{s,a'}]$ (see Section 11.2.1), we have:

$$\begin{aligned}& \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[A^{(t)}(s,a) - w^{(t)}_\star \cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big] \\ = \;& \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[Q^{(t)}(s,a) - w^{(t)}_\star \cdot \phi_{s,a}\Big] - \mathbb{E}_{s\sim d^\star_\rho, a'\sim\pi^{(t)}(\cdot|s)}\Big[Q^{(t)}(s,a') - w^{(t)}_\star \cdot \phi_{s,a'}\Big] \\ \leq \;& \sqrt{\mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\left(Q^{(t)}(s,a) - w^{(t)}_\star \cdot \phi_{s,a}\right)^2} + \sqrt{\mathbb{E}_{s\sim d^\star_\rho, a'\sim\pi^{(t)}(\cdot|s)}\left(Q^{(t)}(s,a') - w^{(t)}_\star \cdot \phi_{s,a'}\right)^2} \\ \leq \;& 2\sqrt{|\mathcal{A}|\mathbb{E}_{s\sim d^\star_\rho, a\sim \mathrm{Unif}_{\mathcal{A}}}\left[\left(Q^{(t)}(s,a) - w^{(t)}_\star \cdot \phi_{s,a}\right)^2\right]} = 2\sqrt{|\mathcal{A}| L(w^{(t)}_\star; \theta^{(t)}, d^\star)}.\end{aligned}$$

where we have used the definition of $d^\star$ and $L(w^{(t)}_\star; \theta^{(t)}, d^\star)$ in the last step. Using following crude upper bound,

$$L(w^{(t)}_\star; \theta^{(t)}, d^\star) \leq \left\|\frac{d^\star}{d^{(t)}}\right\|_\infty L(w^{(t)}_\star; \theta^{(t)}, d^{(t)}) \leq \frac{1}{1-\gamma}\left\|\frac{d^\star}{\nu}\right\|_\infty L(w^{(t)}_\star; \theta^{(t)}, d^{(t)}),$$

(where the last step uses the defintion of $d^{(t)}$, see Equation 0.7), we have that:

$$\mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[A^{(t)}(s,a) - w^{(t)}_\star \cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big] \leq 2\sqrt{\frac{|\mathcal{A}|}{1-\gamma}\left\|\frac{d^\star}{\nu}\right\|_\infty L(w^{(t)}_\star; \theta^{(t)}, d^{(t)})}. \tag{0.9}$$

For the second term, let us now show that:

$$\begin{aligned}& \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[\big(w^{(t)}_\star - w^{(t)}\big)\cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big] \\ & \leq \; 2\sqrt{\frac{|\mathcal{A}|\kappa}{1-\gamma}\Big(L(w^{(t)}; \theta^{(t)}, d^{(t)}) - L(w^{(t)}_\star; \theta^{(t)}, d^{(t)})\Big)}\end{aligned} \tag{0.10}$$

To see this, first observe that a similar argument to the above leads to:

$$\begin{aligned}& \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[\big(w^{(t)}_\star - w^{(t)}\big)\cdot \nabla_\theta \log \pi^{(t)}(a|s)\Big] \\ & = \mathbb{E}_{s\sim d^\star_\rho, a\sim\pi^\star(\cdot|s)}\Big[\big(w^{(t)}_\star - w^{(t)}\big)\cdot \phi_{s,a}\Big] - \mathbb{E}_{s\sim d^\star_\rho, a'\sim\pi^{(t)}(\cdot|s)}\Big[\big(w^{(t)}_\star - w^{(t)}\big)\cdot \phi_{s,a'}\Big] \\ & \leq 2\sqrt{|\mathcal{A}|\mathbb{E}_{s,a\sim d^\star}\left[\left(\big(w^{(t)}_\star - w^{(t)}\big)\cdot \phi_{s,a}\right)^2\right]} = 2\sqrt{|\mathcal{A}| \cdot \|w^{(t)}_\star - w^{(t)}\|^2_{\Sigma_{d^\star}}},\end{aligned}$$

where we use the notation $\|x\|^2_M := x^\top M x$ for a matrix M and a vector x. From the definition of κ,

$$\|w^{(t)}_\star - w^{(t)}\|^2_{\Sigma_{d^\star}} \leq \kappa \|w^{(t)}_\star - w^{(t)}\|^2_{\Sigma_\nu} \leq \frac{\kappa}{1-\gamma}\|w^{(t)}_\star - w^{(t)}\|^2_{\Sigma_{d^{(t)}}}$$

using that $(1-\gamma)\nu \leq d^{\pi^{(t)}}_\nu$ (see (0.7)). Due to that $w^{(t)}_\star$ minimizes $L(w; \theta^{(t)}, d^{(t)})$ over the set $\mathcal{W} := \{w : \|w\|_2 \leq W\}$, for any $w \in \mathcal{W}$ the first-order optimality conditions for $w^{(t)}_\star$ imply that:

$$(w - w^{(t)}_\star) \cdot \nabla L(w^{(t)}_\star; \theta^{(t)}, d^{(t)}) \geq 0.$$

Therefore, for any $w \in \mathcal{W}$,

$$\begin{aligned}
&L(w;\theta^{(t)},d^{(t)}) - L(w_\star^{(t)};\theta^{(t)},d^{(t)}) \\
&= \mathbb{E}_{s,a\sim d^{(t)}}\left[\left(w\cdot\phi(s,a) - w_\star\cdot\phi(s,a) + w_\star\cdot\phi(s,a) - Q^{(t)}(s,a)\right)^2\right] - L(w_\star^{(t)};\theta^{(t)},d^{(t)}) \\
&= \mathbb{E}_{s,a\sim d^{(t)}}\left[\left(w\cdot\phi(s,a) - w_\star\cdot\phi(s,a)\right)^2\right] + 2(w-w_\star)\mathbb{E}_{s,a\sim d^{(t)}}\left[\phi(s,a)\left(w_\star\cdot\phi(s,a) - Q^{(t)}(s,a)\right)\right] \\
&= \|w - w_\star^{(t)}\|^2_{\Sigma_{d^{(t)}}} + (w-w_\star^{(t)})\cdot\nabla L(w_\star^{(t)};\theta^{(t)},d^{(t)}) \\
&\geq \|w - w_\star^{(t)}\|^2_{\Sigma_{d^{(t)}}}.
\end{aligned}$$

Noting that $w^{(t)} \in \mathcal{W}$ by construction in Algorithm 0.8 yields the claimed bound on the second term in (0.10).

Using the bounds on the first and second terms in (0.9) and (0.10), along with concavity of the square root function, we have that:

$$\mathrm{err}_t \leq 2\sqrt{\frac{|\mathcal{A}|}{1-\gamma}\left\|\frac{d^\star}{\nu}\right\|_\infty L(w_\star^{(t)};\theta^{(t)},d^{(t)})} + 2\sqrt{\frac{|\mathcal{A}|\kappa}{1-\gamma}\left(L(w^{(t)};\theta^{(t)},d^{(t)}) - L(w_\star^{(t)};\theta^{(t)},d^{(t)})\right)}.$$

The proof is completed by substitution and using our assumptions on ϵ_{stat} and ϵ_{bias}. ■

11.5 Q-NPG Sample Complexity

To be added...

11.6 Bibliographic Remarks and Further Readings

The notion of *compatible function approximation* was due to [Sutton et al., 1999], which also proved the claim in Lemma 11.1. The close connection of the NPG update rule to compatible function approximation (Lemma 0.3) was noted in [Kakade, 2001].

The regret lemma (Lemma 11.3) for the NPG analysis has origins in the online regret framework in changing MDPs [Even-Dar et al., 2009]. The convergence rates in this chapter are largely derived from [Agarwal et al., 2020c]. The Q-NPG algorithm for the log-linear policy classes is essentially the same algorithm as POLITEX [Abbasi-Yadkori et al., 2019], with a distinction that it is important to use a state action measure over the initial distribution. The analysis and error decomposition of Q-NPG is from [Agarwal et al., 2020c], which has a more general analysis of NPG with function approximation under the regret lemma. This more general approach also permits the analysis of neural policy classes, as shown in [Agarwal et al., 2020c]. Also, [Liu et al., 2019] provide an analysis of the TRPO algorithm [Schulman et al., 2015] (essentially the same as NPG) for neural network parameterizations in the somewhat restrictive linearized "neural tangent kernel" regime.

Chapter 12

CPI, TRPO, and More

In this chapter, we consider conservative policy iteration (CPI) and trust-region constrained policy optimization (TRPO). Both CPI and TRPO can be understood as making small incremental update to the policy by forcing that the new policy's state action distribution is not too far away from the current policy's. We will see that CPI achieves that by forming a new policy that is a mixture of the current policy and a local greedy policy, while TRPO forcing that by explicitly adding a KL constraint (over polices' induced trajectory distributions space) in the optimization procedure. We will show that TRPO gives an equivalent update procedure as Natural Policy Gradient.

Along the way, we discuss the benefit of incremental policy update, by contrasting it to another family of policy update procedure called *Approximate Policy Iteration* (API), which performs local greedy policy search and could potentially lead to abrupt policy change. We show that API in general fails to converge or make local improvement, unless under a much stronger concentrability ratio assumption.

The algorithm and analysis of CPI is adapted from the original one in [Kakade and Langford, 2002], and the we follow the presentation of TRPO from [Schulman et al., 2015], while making a connection to the NGP algorithm.

12.1 Conservative Policy Iteration

As the name suggests, we will now describe a more conservative version of the policy iteration algorithm, which shifts the next policy away from the current policy with a small step size to prevent drastic shifts in successive state distributions.

We consider the discounted MDP here $\{\mathcal{S}, \mathcal{A}, P, r, \gamma, \rho\}$ where ρ is the initial state distribution. Similar to Policy Gradient Methods, we assume that we have a restart distribution μ (i.e., the μ-restart setting). Throughout this section, for any policy π, we denote d^π_μ as the state-action visitation starting from $s_0 \sim \mu$ instead of ρ, and d^π the state-action visitation starting from the true initial state distribution ρ, i.e., $s_0 \sim \rho$. Similarly, we denote V^π_μ as expected discounted total reward of policy π starting at μ, while V^π as the expected discounted total reward of π with ρ as the initial state distribution. We assume $\mathcal{A}$ is discrete but $\mathcal{S}$ could be continuous.

CPI is based on the concept of *Reduction to Supervised Learning*. Specifically we will use the Approximate Greedy Policy Selector defined in Chapter 3 (Definition 3.1). We recall the definition of the ε-approximate Greedy Policy Selector $\mathcal{G}_\varepsilon(\pi, \Pi, \mu)$ below. Given a policy π, policy class Π, and a restart distribution μ, denote $\widehat{\pi} = \mathcal{G}_\varepsilon(\pi, \Pi, \mu)$, we have that:

$$\mathbb{E}_{s\sim d^\pi_\mu} A^\pi(s, \widehat{\pi}(s)) \geq \max_{\widetilde{\pi}\in\Pi} \mathbb{E}_{s\sim d^\pi_\mu} A^\pi(s, \widetilde{\pi}(s)) - \varepsilon.$$

Recall that in Chapter 3 we explained two approach to implement such approximate oracle: one with a reduction to classification oracle, and the other one with a reduction to regression oracle.

12.1.1 The CPI Algorithm

CPI, summarized in Alg. 7, will iteratively generate a sequence of policies π^i. Note we use $\pi_\alpha = (1-\alpha)\pi + \alpha\pi'$ to refer to a randomized policy which at any state s, chooses an action according to π with probability $1-\alpha$ and according to π' with probability α. The greedy policy π' is computed using the ε-approximate greedy policy selector $\mathcal{G}_\varepsilon(\pi^t, \Pi, \mu)$. The algorithm is terminate when there is no significant one-step improvement over π^t, i.e., $\mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s))) \leq \varepsilon$.

Algorithm 7 Conservative Policy Iteration (CPI)

Input: Initial policy $\pi^0 \in \Pi$, accuracy parameter ε.

for $t = 0, 1, 2 \ldots$ **do**
  $\pi' = \mathcal{G}_\varepsilon(\pi^t, \Pi, \mu)$
  if $\mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s)) \leq \varepsilon$ **then**
    Return π^t
  end if
  Update $\pi^{t+1} = (1-\alpha)\pi^t + \alpha\pi'$
end for

The main intuition behind the algorithm is that the stepsize α controls the difference between state distributions of π^t and π^{t+1}. Let us look into the performance difference lemma to get some intuition on this conservative update. From PDL, we have:

$$V_\mu^{\pi^{t+1}} - V_\mu^{\pi^t} = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} A^{\pi^t}(s, \pi^{t+1}(s)) = \frac{\alpha}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} A^{\pi^t}(s, \pi'(s)),$$

where the last equality we use the fact that $\pi^{t+1} = (1-\alpha)\pi^t + \alpha\pi'$ and $A^{\pi^t}(s, \pi^t(s)) = 0$ for all s. Thus, if we can search for a policy $\pi' \in \Pi$ that maximizes $\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} A^{\pi^t}(s, \pi'(s))$ and makes $\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} A^{\pi^t}(s, \pi'(s)) > 0$, then we can guarantee policy improvement. However, at episode t, we do not know the state distribution of π^{t+1} and all we have access to is $d_\mu^{\pi^t}$. Thus, we explicitly make the policy update procedure to be conservative such that $d_\mu^{\pi^t}$ and the new policy's distribution $d_\mu^{\pi^{t+1}}$ is guaranteed to be not that different. Thus we can hope that $\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} A^{\pi^t}(s, \pi'(s))$ is close to $\mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s))$, and the latter is something that we can manipulate using the greedy policy selector.

Below we formalize the above intuition and show that with small enough α, we indeed can ensure monotonic policy improvement.

We start from the following lemma which shows that π^{t+1} and π^t are close to each other in terms of total variation distance at any state, and $d_\mu^{\pi^{t+1}}$ and $d_\mu^{\pi^t}$ are close as well.

Lemma 12.1 (Similar Policies imply similar state visitations)**.** *Consider any t, we have that:*

$$\left\|\pi^{t+1}(\cdot|s) - \pi^t(\cdot|s)\right\|_1 \leq 2\alpha, \forall s;$$

Further, we have:

$$\left\|d_\mu^{\pi^{t+1}} - d_\mu^{\pi^t}\right\|_1 \leq \frac{2\alpha\gamma}{1-\gamma}.$$

Proof: The first claim in the above lemma comes from the definition of policy update:

$$\left\|\pi^{t+1}(\cdot|s) - \pi^t(\cdot|s)\right\|_1 = \alpha \left\|\pi^t(\cdot|s) - \pi'(\cdot|s)\right\|_1 \leq 2\alpha.$$

Denote $\mathbb{P}_h^\pi$ as the state distribution resulting from π at time step h with μ as the initial state distribution. We consider bounding $\|\mathbb{P}_h^{\pi^{t+1}} - \mathbb{P}_h^{\pi^{t+1}}\|_1$ with $h \geq 1$.

$$\begin{aligned}
\mathbb{P}_h^{\pi^{t+1}}(s') - \mathbb{P}_h^{\pi^{t+1}}(s') &= \sum_{s,a} \left(\mathbb{P}_{h-1}^{\pi^{t+1}}(s)\pi^{t+1}(a|s) - \mathbb{P}_{h-1}^{\pi^t}(s)\pi^t(a|s)\right) P(s'|s,a) \\
&= \sum_{s,a} \left(\mathbb{P}_{h-1}^{\pi^{t+1}}(s)\pi^{t+1}(a|s) - \mathbb{P}_{h-1}^{\pi^{t+1}}(s)\pi^t(a|s) + \mathbb{P}_{h-1}^{\pi^{t+1}}(s)\pi^t(a|s) - \mathbb{P}_{h-1}^{\pi^t}(s)\pi^t(a|s)\right) P(s'|s,a) \\
&= \sum_s \mathbb{P}_{h-1}^{\pi^{t+1}}(s) \sum_a \left(\pi^{t+1}(a|s) - \pi^t(a|s)\right) P(s'|s,a) \\
&\quad + \sum_s \left(\mathbb{P}_{h-1}^{\pi^{t+1}}(s) - \mathbb{P}_{h-1}^{\pi^t}(s)\right) \sum_a \pi^t(a|s) P(s'|s,a).
\end{aligned}$$

Take absolute value on both sides, we get:

$$\begin{aligned}
\sum_{s'} \left|\mathbb{P}_h^{\pi^{t+1}}(s') - \mathbb{P}_h^{\pi^{t+1}}(s')\right| &\leq \sum_s \mathbb{P}_{h-1}^{\pi^{t+1}}(s) \sum_a \left|\pi^{t+1}(a|s) - \pi^t(a|s)\right| \sum_{s'} P(s'|s,a) \\
&\quad + \sum_s \left|\mathbb{P}_{h-1}^{\pi^{t+1}}(s) - \mathbb{P}_{h-1}^{\pi^t}(s)\right| \sum_{s'} \sum_a \pi^t(a|s) P(s'|s,a) \\
&\leq 2\alpha + \|\mathbb{P}_{h-1}^{\pi^{t+1}} - \mathbb{P}_{h-1}^{\pi^t}\|_1 \leq 4\alpha + \|\mathbb{P}_{h-2}^{\pi^{t+1}} - \mathbb{P}_{h-2}^{\pi^t}\|_1 = 2h\alpha.
\end{aligned}$$

Now use the definition of d_μ^π, we have:

$$d_\mu^{\pi^{t+1}} - d_\mu^{\pi^t} = (1-\gamma) \sum_{h=0}^\infty \gamma^h \left(\mathbb{P}_h^{\pi^{t+1}} - \mathbb{P}_h^{\pi^t}\right).$$

Add ℓ_1 norm on both sides, we get:

$$\left\|d_\mu^{\pi^{t+1}} - d_\mu^{\pi^t}\right\|_1 \leq (1-\gamma) \sum_{h=0}^\infty \gamma^h 2h\alpha$$

It is not hard to verify that $\sum_{h=0}^\infty \gamma^h h = \frac{\gamma}{(1-\gamma)^2}$. Thus, we can conclude that:

$$\left\|d_\mu^{\pi^{t+1}} - d_\mu^{\pi^t}\right\|_1 \leq \frac{2\alpha\gamma}{1-\gamma}.$$

■

The above lemma states that if π^{t+1} and π^t are close in terms of total variation distance for every state, then the total variation distance between the resulting state visitations from π^{t+1} and π^t will be small up to a effective horizon $1/(1-\gamma)$ amplification.

The above lemma captures the key of the conservative policy update. Via the conservative policy update, we make sure that $d_\mu^{\pi^{t+1}}$ and $d_\mu^{\pi^t}$ are close to each other in terms of total variation distance. Now we use the above lemma to show a monotonic policy improvement.

Theorem 12.2 (Monotonic Improvement in CPI)**.** *Consider any episode t. Denote $\mathbb{A} = \mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s))$. We have:*

$$V_\mu^{\pi^{t+1}} - V_\mu^{\pi^t} \geq \frac{\alpha}{1-\gamma}\left(\mathbb{A} - \frac{2\alpha\gamma}{(1-\gamma)^2}\right)$$

Set $\alpha = \frac{\mathbb{A}(1-\gamma)^2}{4\gamma}$*, we get:*

$$V_\mu^{\pi^{t+1}} - V_\mu^{\pi^t} \geq \frac{\mathbb{A}^2(1-\gamma)}{8\gamma}.$$

The above lemma shows that as long as we still have positive one-step improvement, i.e., $\mathbb{A} > 0$, then we guarantee that π^{t+1} is strictly better than π^t.

Proof: Via PDL, we have:

$$V_\mu^{\pi^{t+1}} - V_\mu^{\pi^t} = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} \alpha A^{\pi^t}(s, \pi'(s)).$$

Recall Lemma 12.1, we have:

$$\begin{aligned}
(1-\gamma)\left(V_\mu^{\pi^{t+1}} - V_\mu^{\pi^t}\right) &= \mathbb{E}_{s\sim d_\mu^{\pi^t}} \alpha A^{\pi^t}(s, \pi'(s)) + \mathbb{E}_{s\sim d_\mu^{\pi^{t+1}}} \alpha A^{\pi^t}(s, \pi'(s)) - \mathbb{E}_{s\sim d_\mu^{\pi^t}} \alpha A^{\pi^t}(s, \pi'(s)) \\
&\geq \mathbb{E}_{s\sim d_\mu^{\pi^t}} \alpha A^{\pi^t}(s, \pi'(s)) - \alpha \sup_{s,a,\pi} |A^\pi(s,a)| \|d_\mu^{\pi^t} - d_\mu^{\pi^{t+1}}\|_1 \\
&\geq \mathbb{E}_{s\sim d_\mu^{\pi^t}} \alpha A^{\pi^t}(s, \pi'(s)) - \frac{\alpha}{1-\gamma}\|d_\mu^{\pi^t} - d_\mu^{\pi^{t+1}}\|_1 \\
&\geq \mathbb{E}_{s\sim d_\mu^{\pi^t}} \alpha A^{\pi^t}(s, \pi'(s)) - \frac{2\alpha^2\gamma}{(1-\gamma)^2} = \alpha\left(\mathbb{A} - \frac{2\alpha\gamma}{(1-\gamma)^2}\right)
\end{aligned}$$

where the first inequality we use the fact that for any two distributions P_1 and P_2 and any function f, $|\mathbb{E}_{x\sim P_1} f(x) - \mathbb{E}_{x\sim P_2} f(x)| \leq \sup_x |f(x)| \|P_1 - P_2\|_1$, for the second inequality, we use the fact that $|A^\pi(s,a)| \leq 1/(1-\gamma)$ for any π, s, a, and the last inequality uses Lemma 12.1.

For the second part of the above theorem, note that we want to maximum the policy improvement as much as possible by choosing α. So we can pick α which maximizes $\alpha(\mathbb{A} - 2\alpha\gamma/(1-\gamma)^2)$. This gives us the α we claimed in the lemma. Plug in α back into $\alpha(\mathbb{A} - 2\alpha\gamma/(1-\gamma)^2)$, we conclude the second part of the theorem.

■

The above theorem indicates that with the right choice of α, we guarantee that the policy is making improvement as long as $\mathbb{A} > 0$. Recall the termination criteria in CPI where we terminate CPI when $\mathbb{A} \leq \varepsilon$. Putting these results together, we obtain the following overall convergence guarantee for the CPI algorithm.

Theorem 12.3 (Local optimality of CPI)**.** *Algorithm 7 terminates in at most* $8\gamma/\epsilon^2$ *steps and outputs a policy* π^t *satisfying* $\max_{\pi\in\Pi} \mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi(s)) \leq 2\varepsilon$.

Proof: Note that our reward is bounded in $[0,1]$ which means that $V_\mu^\pi \in [0, 1/(1-\gamma)]$. Note that we have shown in Theorem 12.2, every iteration t, we have policy improvement at least $\frac{\mathbb{A}^2(1-\gamma)}{8\gamma}$, where recall $\mathbb{A}$ at episode t is defined as $\mathbb{A} = \mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s))$. If the algorithm does not terminate at episode t, then we guarantee that:

$$V_\mu^{\pi^{t+1}} \geq V_\mu^{\pi^t} + \frac{\varepsilon^2(1-\gamma)}{8\gamma}$$

Since V_μ^π is upper bounded by $1/(1-\gamma)$, so we can at most make improvement $8\gamma/\epsilon^2$ many iterations.

Finally, recall that ε-approximate greedy policy selector $\pi' = \mathcal{G}_\varepsilon(\pi^t, \Pi, \mu)$, we have:

$$\max_{\pi\in\Pi} \mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi(s)) \leq \mathbb{E}_{s\sim d_\mu^{\pi^t}} A^{\pi^t}(s, \pi'(s)) + \varepsilon \leq 2\varepsilon$$

This concludes the proof. ■

Theorem 12.3 can be viewed as a local optimality guarantee in a sense. It shows that when CPI terminates, we cannot find a policy $\pi \in \Pi$ that achieves local improvement over the returned policy more than ε. However, this does not necessarily imply that the value of π is close to $V^\star$. However, similar to the policy gradient analysis, we can turn this local guarantee into a global one when the restart distribution μ covers $d^{\pi^\star}$. We formalize this intuition next.

Theorem 12.4 (Global optimality of CPI)**.** *Upon termination, we have a policy π such that:*

$$V^\star - V^\pi \leq \frac{2\varepsilon + \epsilon_\Pi}{(1-\gamma)^2} \left\| \frac{d^{\pi^\star}}{\mu} \right\|_\infty,$$

where $\epsilon_\Pi := \mathbb{E}_{s\sim d^\pi_\mu} \max_{a\in\mathcal{A}} A^\pi(s,a) - \max_{\pi\in\Pi} \mathbb{E}_{s\sim d^\pi_\mu} \max_{a\in\mathcal{A}} A^\pi(s,\pi(s))$.

In other words, if our policy class is rich enough to approximate the policy $\max_{a\in\mathcal{A}} A^\pi(s,a)$ under d^π_μ, i.e., ϵ_Π is small, and μ covers $d^{\pi^\star}$ in a sense that $\left\| \frac{d^{\pi^\star}}{\mu} \right\|_\infty \leq \infty$, CPI guarantees to find a near optimal policy.

Proof: By the performance difference lemma,

$$\begin{aligned}
V^\star - V^\pi &= \frac{1}{1-\gamma} \mathbb{E}_{s\sim d^{\pi^\star}} A^\pi(s, \pi^\star(s)) \\
&\leq \frac{1}{1-\gamma} \mathbb{E}_{s\sim d^{\pi^\star}} \max_{a\in\mathcal{A}} A^\pi(s,a) \\
&\leq \frac{1}{1-\gamma} \left\| \frac{d^{\pi^*}}{d^\pi_\mu} \right\|_\infty \mathbb{E}_{s\sim d^\pi_\mu} \max_{a\in\mathcal{A}} A^\pi(s,a) \\
&\leq \frac{1}{(1-\gamma)^2} \left\| \frac{d^{\pi^*}}{\mu} \right\|_\infty \mathbb{E}_{s\sim d^\pi_\mu} \max_{a\in\mathcal{A}} A^\pi(s,a) \\
&=\leq \frac{1}{(1-\gamma)^2} \left\| \frac{d^{\pi^*}}{\mu} \right\|_\infty \left[\max_{\hat\pi\in\Pi} \mathbb{E}_{s\sim d^\pi_\mu} A^\pi(s,\hat\pi(s)) - \max_{\hat\pi\in\Pi} \mathbb{E}_{s\sim d^\pi_\mu} A^\pi(s,\hat\pi(s)) + \mathbb{E}_{s\sim d^\pi_\mu} \max_{a\in\mathcal{A}} A^\pi(s,a) \right] \\
&\leq \frac{1}{(1-\gamma)^2} \left\| \frac{d^{\pi^*}}{\mu} \right\|_\infty (2\varepsilon + \epsilon_\Pi),
\end{aligned}$$

where the second inequality holds due to the fact that $\max_a A^\pi(s,a) \geq 0$, the third inequality uses the fact that $d^\pi_\mu(s) \geq (1-\gamma)\mu(s)$ for any s and π, and the last inequality uses the definition ϵ_Π and Theorem 12.3. ■

It is informative to contrast CPI and policy gradient algorithms due to the similarity of their guarantees. Both provide local optimality guarantees. For CPI, the local optimality always holds, while for policy gradients it requires a smooth value function as a function of the policy parameters. If the distribution mismatch between an optimal policy and the output of the algorithm is not too large, then both algorithms further yield a near optimal policy. The similarities are not so surprising. Both algorithms operate by making local improvements to the current policy at each iteration, by inspecting its advantage function. The changes made to the policy are controlled using a stepsize parameter in both the approaches. It is the actual mechanism of the improvement which differs in the two cases. Policy gradients assume that the policy's reward is a differentiable function of the parameters, and hence make local improvements through gradient ascent. The differentiability is certainly an assumption and does not necessarily hold for all policy classes. An easy example is when the policy itself is not an easily differentiable function of its parameters. For instance, if the policy is parametrized by regression trees, then performing gradient updates can be challenging.

In CPI, on the other hand, the basic computational primitive required on the policy class is the ability to maximize the advantage function relative to the current policy. Notice that Algorithm 7 does not necessarily restrict to a policy class, such as a set of parametrized policies as in policy gradients. Indeed, due to the reduction to supervised learning approach (e.g., using the weighted classification oracle CO), we can parameterize policy class via decision tree, for

instance. This property makes CPI extremely attractive. Any policy class over which efficient supervised learning algorithms exist can be adapted to reinforcement learning with performance guarantees.

A second important difference between CPI and policy gradients is in the notion of locality. Policy gradient updates are local in the parameter space, and we hope that this makes small enough changes to the state distribution that the new policy is indeed an improvement on the older one (for instance, when we invoke the performance difference lemma between successive iterates). While this is always true in expectation for correctly chosen stepsizes based on properties of stochastic gradient ascent on smooth functions, the variance of the algorithm and lack of robustness to suboptimal stepsizes can make the algorithm somewhat finicky. Indeed, there are a host of techniques in the literature to both lower the variance (through control variates) and explicitly control the state distribution mismatch between successive iterates of policy gradients (through trust region techniques). On the other hand, CPI explicitly controls the amount of perturbation to the state distribution by carefully mixing policies in a manner which does not drastically alter the trajectories with high probability. Indeed, this insight is central to the proof of CPI, and has been instrumental in several follow-ups, both in the direct policy improvement as well as policy gradient literature.

12.2 Trust Region Methods and Covariant Policy Search

So far we have seen policy gradient methods and CPI which all uses a small step-size to ensure incremental update in policies. Another popular approach for incremental policy update is to explicitly forcing small change in policies' distribution via a trust region constraint. More specifically, let us go back to the general policy parameterization π_θ. At iteration t with the current policy π_{θ_t}, we are interested in the following local trust-region constrained optimization:

$$\max_\theta \mathbb{E}_{s\sim d_\mu^{\pi_{\theta_t}}} \mathbb{E}_{a\sim\pi_\theta(\cdot|s)} A^{\pi_{\theta_t}}(s,a)$$
$$\text{s.t., } KL\left(\Pr_\mu^{\pi_{\theta_t}} || \Pr_\mu^{\pi_\theta}\right) \le \delta,$$

where recall $\Pr_\mu^\pi(\tau)$ is the trajectory distribution induced by π starting at $s_0 \sim \mu$, and $KL(P_1||P_2)$ are KL-divergence between two distribution P_1 and P_2. Namely we explicitly perform local policy search with a constraint forcing the new policy not being too far away from $\Pr_\mu^{\pi_{\theta_t}}$ in terms of KL divergence.

As we are interested in small local update in parameters, we can perform sequential quadratic programming here, i.e., we can further linearize the objective function at θ_t and quadratize the KL constraint at θ_t to form a local quadratic programming:

$$\max_\theta \left\langle \mathbb{E}_{s\sim d_\mu^{\pi_{\theta_t}}} \mathbb{E}_{a\sim\pi_{\theta_t}(\cdot|s)} \nabla_\theta \ln \pi_{\theta_t}(a|s) A^{\pi_{\theta_t}}(s,a), \theta \right\rangle \tag{0.1}$$
$$\text{s.t., } \langle \nabla_\theta KL\left(\Pr_\mu^{\pi_{\theta_t}} || \Pr_\mu^{\pi_\theta}\right)|_{\theta=\theta_t}, \theta-\theta_t\rangle + \frac{1}{2}(\theta-\theta_t)^\top \left(\nabla_\theta^2 KL\left(\Pr_\mu^{\pi_{\theta_t}} || \Pr_\mu^{\pi_\theta}\right)|_{\theta=\theta_t}\right)(\theta-\theta_t) \le \delta, \tag{0.2}$$

where we denote $\nabla^2 KL|_{\theta=\theta_t}$ as the Hessian of the KL constraint measured at θ_t. Note that KL divergence is not a valid metric as it is not symmetric. However, its local quadratic approximation can serve as a valid local distance metric, as we prove below that the Hessian $\nabla^2 KL|_{\theta=\theta_t}$ is a positive semi-definite matrix. Indeed, we will show that the Hessian of the KL constraint is exactly equal to the fisher information matrix, and the above quadratic programming exactly reveals a Natural Policy Gradient update. Hence Natural policy gradient can also be interpreted as performing sequential quadratic programming with KL constraint over policy's trajectory distributions.

To match to the practical algorithms in the literature (e.g., TRPO), below we focus on episode finite horizon setting again (i.e., an MDP with horizon H).

Claim 12.5. *Consider a finite horizon MDP with horizon H. Consider any fixed θ_t. We have:*

$$\nabla_\theta KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right)|_{\theta=\theta_t} = 0,$$
$$\nabla^2_\theta KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right)|_{\theta=\theta_t} = H\mathbb{E}_{s,a\sim d^{\pi_{\theta_t}}} \nabla \ln \pi_{\theta_t}(a|s) \left(\nabla \ln \pi_{\theta_t}(a|s)\right)^\top.$$

Proof: We first recall the trajectory distribution in finite horizon setting.

$$\mathrm{Pr}_\mu^\pi(\tau) = \mu(s_0) \prod_{h=0}^{H-1} \pi(a_h|s_h) P(s_{h+1}|s_h, a_h).$$

We first prove that the gradient of KL is zero. First note that:

$$\begin{aligned} KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right) &= \sum_\tau \mathrm{Pr}_\mu^{\pi_{\theta_t}}(\tau) \ln \frac{\mathrm{Pr}_\mu^{\pi_{\theta_t}}(\tau)}{\mathrm{Pr}_\mu^{\pi_\theta}(\tau)} \\ &= \sum_\tau \mathrm{Pr}_\mu^{\pi_{\theta_t}}(\tau) \left(\sum_{h=0}^{H-1} \ln \frac{\pi_{\theta_t}(a_h|s_h)}{\pi_\theta(a_h|s_h)}\right) = \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \ln \frac{\pi_{\theta_t}(a_h|s_h)}{\pi_\theta(a_h|s_h)}. \end{aligned}$$

Thus, for $\nabla_\theta KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right)$, we have:

$$\begin{aligned} \nabla KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right)|_{\theta=\theta_t} &= \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} - \nabla \ln \pi_{\theta_t}(a_h|s_h) \\ &= -\sum_{h=0}^{H-1} \mathbb{E}_{s_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \mathbb{E}_{a_h\sim\pi_{\theta_t}(\cdot|s_h)} \nabla \ln \pi_{\theta_t}(a_h|s_h) = 0, \end{aligned}$$

where we have seen the last step when we argue the unbiased nature of policy gradient with an action independent baseline.

Now we move to the Hessian.

$$\begin{aligned} \nabla^2 KL\left(\mathrm{Pr}_\mu^{\pi_{\theta_t}} || \mathrm{Pr}_\mu^{\pi_\theta}\right)|_{\theta=\theta_t} &= -\sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \nabla^2 \ln \pi_\theta(a_h|s_h)|_{\theta=\theta_t} \\ &= -\sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \left(\nabla\left(\frac{\nabla\pi_\theta(a|s)}{\pi_\theta(a|s)}\right)\right) \\ &= -\sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \left(\frac{\nabla^2\pi_\theta(a_h|s_h)}{\pi_\theta(a_h|s_h)} - \frac{\nabla\pi_\theta(a_h|s_h)\nabla\pi_\theta(a_h|s_h)^\top}{\pi_\theta^2(a_h|s_h)}\right) \\ &= \sum_{h=0}^{H-1} \mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \nabla_\theta \ln \pi_\theta(a_h|s_h) \nabla_\theta \ln \pi_\theta(a_h|s_h)^\top, \end{aligned}$$

where in the last equation we use the fact that $\mathbb{E}_{s_h,a_h\sim\mathbb{P}_h^{\pi_{\theta_t}}} \frac{\nabla^2\pi_\theta(a_h|s_h)}{\pi_\theta(a_h|s_h)} = 0$. ■

The above claim shows that a second order taylor expansion of the KL constraint over trajectory distribution gives a local distance metric at θ_t:

$$(\theta - \theta_t) F_{\theta_t} (\theta - \theta_t),$$

where again $F_{\theta_t} := H\mathbb{E}_{s,a\sim d^{\pi_{\theta_t}}} \nabla \ln \pi_{\theta_t}(a|s) \left(\nabla \ln \pi_{\theta_t}(a|s)\right)^\top$ is proportional to the fisher information matrix. Note that F_{θ_t} is a PSD matrix and thus $d(\theta, \theta_t) := (\theta - \theta_t) F_{\theta_t} (\theta - \theta_t)$ is a valid distance metric. By sequential quadratic programming, we are using local geometry information of the trajectory distribution manifold induced by the parameterization θ, rather the naive Euclidean distance in the parameter space. Such a method is sometimes referred to as *Covariant Policy Search*, as the policy update procedure will be invariant to linear transformation of parameterization (See Section 12.3 for further discussion).

Now using the results from Claim 12.5, we can verify that the local policy optimization procedure in Eq. 0.2 exactly recovers the NPG update, where the step size is based on the trust region parameter δ. Denote $\Delta = \theta - \theta_t$, we have:

$$\max_{\Delta} \langle \Delta, \nabla_\theta V^{\pi_{\theta_t}} \rangle,$$
$$s.t., \Delta^\top F_{\theta_t} \Delta^\top \leq \delta,$$

which gives the following update procedure:

$$\theta_{t+1} = \theta_t + \Delta = \theta_t + \sqrt{\frac{\delta}{(\nabla V^{\pi_{\theta_t}})^\top F_{\theta_t}^{-1} \nabla V^{\pi_{\theta_t}}}} \cdot F_{\theta_t}^{-1} \nabla V^{\pi_{\theta_t}},$$

where note that we use the self-normalized learning rate computed using the trust region parameter δ.

12.2.1 Proximal Policy Optimization

To be added...

12.3 Bibliographic Remarks and Further Readings

The analysis of CPI is adapted from the original one in [Kakade and Langford, 2002]. There have been a few further interpretations of CPI. One interesting perspective is that CPI can be treated as a boosting algorithm [Scherrer and Geist, 2014].

More generally, CPI and NPG are part of family of *incremental* algorithms, including Policy Search by Dynamic Programming (PSDP) [Bagnell et al., 2004] and MD-MPI [Geist et al., 2019]. PSDP operates in a finite horizon setting and optimizes a sequence of time-dependent policies; from the last time step to the first time step, every iteration of, PSDP only updates the policy at current time step while holding the future policies fixed — thus making incremental update on the policy. See [Scherrer, 2014] for more a detailed discussion and comparison of some of these approaches. Mirror Descent-Modified Policy Iteration (MD-MPI) algorithm [Geist et al., 2019] is a family of actor-critic style algorithms which is based on regularization and is incremental in natural; with negative entropy as the Bregman divergence (for the tabular case), it recovers the NPG the tabular case (for the softmax parameterization).

Broadly speaking, these incremental algorithms can improve upon the stringent concentrability conditions for approximate value iteration methods, presented in Chapter 3. Scherrer [2014] provide a more detailed discussion of bounds which depend on these density ratios. As discussed in the last chapter, the density ratio for NPG can be interpreted as a factor due to transfer learning to a single, *fixed* distribution.

The interpretation of NPG as Covariant Policy Search is due to [Bagnell and Schneider, 2003], as the policy update procedure will be invariant to linear transformation of parameterization; see [Bagnell and Schneider, 2003] for a more detailed discussion on this.

The TRPO algorithm is due to [Schulman et al., 2015]. The original TRPO provide performance guarantees, largely relying on making a reduction to CPI. Here, we make the connection of TRPO to NPG, which was subsequently

observed by many; this provides a sharper analysis (e.g. via the results presented in Chapter 11). In practice, a popular variant is the Proximal Policy Optimization (PPO) algorithm [Schulman et al., 2017].

Part 4

Further Topics

Chapter 13

Linear Quadratic Regulators

To be added...

Chapter 14

Imitation Learning and Behavioral Cloning

To be added...

Chapter 15

Offline Reinforcement Learning

To be added...

Bibliography

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, 2011.

Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvari, and Gellért Weisz. POLITEX: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, pages 3692–3702, 2019.

Naoki Abe and Philip M. Long. Associative reinforcement learning using linear probabilistic concepts. In *Proc. 16th International Conf. on Machine Learning*, pages 3–11. Morgan Kaufmann, San Francisco, CA, 1999.

Alekh Agarwal, Mikael Henaff, Sham Kakade, and Wen Sun. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *NeurIPS*, 2020a.

Alekh Agarwal, Sham Kakade, and Lin F. Yang. Model-based reinforcement learning with a generative model is minimax optimal. In *COLT*, volume 125, pages 67–83, 2020b.

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift, 2020c.

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans, editors. *Understanding the impact of entropy on policy optimization*, 2019. URL `https://arxiv.org/abs/1811.11214`.

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71(1):89–129, 2008.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47 (2-3):235–256, 2002. ISSN 0885-6125.

Alex Ayoub, Zeyu Jia, Csaba Szepesvári, Mengdi Wang, and Lin F. Yang. Model-based reinforcement learning with value-targeted regression. *arXiv preprint arXiv:2006.01107*, 2020. URL `https://arxiv.org/abs/2006.01107`.

Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J Kappen. Minimax pac bounds on the sample complexity of reinforcement lear ning with a generative model. *Machine learning*, 91(3):325–349, 2013.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of Machine Learning Research*, volume 70, pages 263–272, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

J. Andrew Bagnell and Jeff Schneider. Covariant policy search. *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1019–1024, 2003. URL `http://dl.acm.org/citation.cfm?id=1630659.1630805`.

J Andrew Bagnell, Sham M Kakade, Jeff G Schneider, and Andrew Y Ng. Policy search by dynamic programming. In *Advances in neural information processing systems*, pages 831–838, 2004.

A. Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017. doi: 10.1137/1.9781611974997.

Richard Bellman. Dynamic programming and Lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, pages 1042–1051, 2019.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *COLT*, pages 355–366, 2008.

Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 5713–5723, 2017.

Christoph Dann, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E. Schapire. On oracle-efficient PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems 31*, 2018.

Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2019.

Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Online Markov decision processes. *Mathematics of Operations Research*, 34(3):726–736, 2009.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. *arXiv preprint arXiv:1901.11275*, 2019.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Peter W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM*, 33(10):75–84, 1990. ISSN 0001-0782.

Daniel Hsu, Sham Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78, 11 2008.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Zeyu Jia, Lin Yang, Csaba Szepesvari, and Mengdi Wang. Model-based reinforcement learning with value-targeted regression. *Proceedings of Machine Learning Research*, 120:666–686, 2020.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E. Schapire. Contextual decision processes with low Bellman rank are PAC-learnable. In *International Conference on Machine Learning*, 2017.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143, 2020.

S. Kakade. A natural policy gradient. In *NIPS*, 2001.

Sham Kakade and John Langford. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the 19th International Conference on Machine Learning*, volume 2, pages 267–274, 2002.

Sham Machandranath Kakade. *On the sample complexity of reinforcement learning*. PhD thesis, University of College London, 2003.

Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pages 740–747, 1999.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49 (2-3):209–232, 2002.

Michael J Kearns and Satinder P Singh. Finite-sample convergence rates for q-learning and indirect algorithms. In *Advances in neural information processing systems*, pages 996–1002, 1999.

Michael J. Kearns, Yishay Mansour, and Andrew Y. Ng. Approximate planning in large pomdps via reusable trajectories. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 1001–1007. MIT Press, 2000.

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pages 1840–1848, 2016.

Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Analysis of classification-based policy iteration algorithms. *The Journal of Machine Learning Research*, 17(1):583–612, 2016.

Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. Breaking the sample size barrier in model-based reinforcement learning with a generative model. *CoRR*, abs/2005.12900, 2020.

Boyi Liu, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural proximal/trust region policy optimization attains globally optimal policy. *CoRR*, abs/1906.10306, 2019. URL `http://arxiv.org/abs/1906.10306`.

Thodoris Lykouris, Max Simchowitz, Aleksandrs Slivkins, and Wen Sun. Corruption robust exploration in episodic reinforcement learning. *arXiv preprint arXiv:1911.08689*, 2019.

Yishay Mansour and Satinder Singh. On the complexity of policy iteration. *UAI*, 01 1999.

C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods, 2020.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

Rémi Munos. Error bounds for approximate policy iteration. In *ICML*, volume 3, pages 560–567, 2003.

Rémi Munos. Error bounds for approximate value iteration. In *AAAI*, 2005.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *CoRR*, abs/1705.07798, 2017.

Ian Osband and Benjamin Van Roy. On lower bounds for regret in reinforcement learning. *ArXiv*, abs/1608.02732, 2016.

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomput.*, 71(7-9):1180–1190, 2008. ISSN 0925-2312.

Martin Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994.

H. Robbins. Some aspects of the sequential design of experiments. In *Bulletin of the American Mathematical Society*, volume 55, 1952.

Bruno Scherrer. Approximate policy iteration schemes: a comparison. In *International Conference on Machine Learning*, pages 1314–1322, 2014.

Bruno Scherrer and Matthieu Geist. Local policy search in a convex space and conservative policy iteration as boosted policy search. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2014.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

John Schulman, F. Wolski, Prafulla Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Aaron Sidford, Mengdi Wang, Xian Wu, Lin F. Yang, and Yinyu Ye. Near-optimal time and sample complexities for for solving discounted markov decision process with a generative model. In *Advances in Neural Information Processing Systems 31*, 2018.

Satinder Singh and Richard Yee. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16(3):227–233, 1994.

Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based rl in contextual decision processes: Pac bounds and exponential improvements over model-free approaches. In *Conference on Learning Theory*, pages 2898–2933, 2019.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 99, pages 1057–1063, 1999.

Csaba Szepesvári and Rémi Munos. Finite time bounds for sampling based fitted value iteration. In *Proceedings of the 22nd international conference on Machine learning*, pages 880–887. ACM, 2005.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Lin F Yang and Mengdi Wang. Reinforcement leaning in feature space: Matrix bandit, kernels, and regret bound. *arXiv preprint arXiv:1905.10389*, 2019a.

Lin F. Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pages 6995–7004, 2019b.

Yinyu Ye. A new complexity result on solving the markov decision problem. *Math. Oper. Res.*, 30:733–749, 08 2005.

Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the markov decision problem with a fixed discount rate. *Math. Oper. Res.*, 36(4):593–603, 2011.

Dongruo Zhou, Jiafan He, and Quanquan Gu. Provably efficient reinforcement learning for discounted mdps with feature mapping. *arXiv preprint arXiv:2006.13165*, 2020.

Appendix A

Concentration

Lemma A.1. *(Hoeffding's inequality) Suppose $X_1, X_2, \ldots X_n$ are a sequence of independent, identically distributed (i.i.d.) random variables with mean μ. Let $\bar{X}_n = n^{-1}\sum_{i=1}^n X_i$. Suppose that $X_i \in [b_-, b_+]$ with probability 1, then*

$$P(\bar{X}_n \geq \mu + \epsilon) \leq e^{-2n\epsilon^2/(b_+ - b_-)^2}.$$

Similarly,

$$P(\bar{X}_n \leq \mu - \epsilon) \leq e^{-2n\epsilon^2/(b_+ - b_-)^2}.$$

The Chernoff bound implies that with probability $1 - \delta$:

$$\bar{X}_n - EX \leq (b_+ - b_-)\sqrt{\ln(1/\delta)/(2n)}\,.$$

Theorem A.2 (Hoeffding-Azuma Inequality)**.** *Suppose $X_1, \ldots, X_T$ is a martingale difference sequence where each X_t is a σ sub-Gaussian, Then, for every $1 > \delta > 0$,*

$$\Pr(\sum X_i \geq \epsilon) \leq \exp\left(\frac{-\epsilon^2}{2N\sigma^2}\right).$$

Lemma A.3. *(Bernstein's inequality) Suppose $X_1, \ldots, X_n$ are independent random variables. Let $\bar{X}_n = n^{-1}\sum_{i=1}^n X_i$, $\mu = \mathbb{E}\bar{X}_n$, and Var$(X_i)$ denote the variance of X_i. If $X_i - EX_i \leq b$ for all i, then*

$$P(\bar{X}_n \geq \mu + \epsilon) \leq \exp\left[-\frac{n^2\epsilon^2}{2\sum_{i=1}^n \mathit{Var}(X_i) + 2nb\epsilon/3}\right].$$

If all the variances are equal, the Bernstein inequality implies that, with probability at least $1 - \delta$,

$$\bar{X}_n - EX \leq \sqrt{2\mathrm{Var}(X)\ln(1/\delta)/n} + \frac{2b\ln(1/\delta)}{3n}.$$

The following concentration bound is a simple application of the McDiarmid's inequality [McDiarmid, 1989] (e.g. see [Hsu et al., 2008] for proof).

Proposition A.4. (Concentration for Discrete Distributions) Let z be a discrete random variable that takes values in $\{1, \ldots, d\}$, distributed according to q. We write q as a vector where $\vec{q} = [\Pr(z = j)]_{j=1}^d$. Assume we have N iid samples, and that our empirical estimate of $\vec{q}$ is $[\widehat{q}]_j = \sum_{i=1}^N \mathbf{1}[z_i = j]/N$.

We have that $\forall \epsilon > 0$:

$$\Pr\left(\|\widehat{q} - \vec{q}\|_2 \geq 1/\sqrt{N} + \epsilon\right) \leq e^{-N\epsilon^2}.$$

which implies that:

$$\Pr\left(\|\widehat{q} - \vec{q}\|_1 \geq \sqrt{d}(1/\sqrt{N} + \epsilon)\right) \leq e^{-N\epsilon^2}.$$

Lemma A.5 (Self-Normalized Bound for Vector-Valued Martingales; [Abbasi-Yadkori et al., 2011])**.** *Let $\{\varepsilon_i\}_{i=1}^{\infty}$ be a real-valued stochastic process with corresponding filtration $\{\mathcal{F}_i\}_{i=1}^{\infty}$ such that ε_i is $\mathcal{F}_i$ measurable, $\mathbb{E}[\varepsilon_i|\mathcal{F}_{i-1}] = 0$, and ε_i is conditionally σ-sub-Gaussian with $\sigma \in \mathbb{R}^+$. Let $\{X_i\}_{i=1}^{\infty}$ be a stochastic process with $X_i \in \mathcal{H}$ (some Hilbert space) and X_i being $\mathcal{F}_t$ measurable. Assume that a linear operator $\Sigma : \mathcal{H} \to \mathcal{H}$ is positive definite, i.e., $x^\top \Sigma x > 0$ for any $x \in \mathcal{H}$. For any t, define the linear operator $\Sigma_t = \Sigma_0 + \sum_{i=1}^{t} X_i X_i^\top$ (here $xx^\top$ denotes outer-product in $\mathcal{H}$). With probability at least $1 - \delta$, we have for all $t \geq 1$:*

$$\left\|\sum_{i=1}^{t} X_i \varepsilon_i\right\|^2_{\Sigma_t^{-1}} \leq \sigma^2 \log\left(\frac{\det(\Sigma_t)\det(\Sigma)^{-1}}{\delta^2}\right).$$