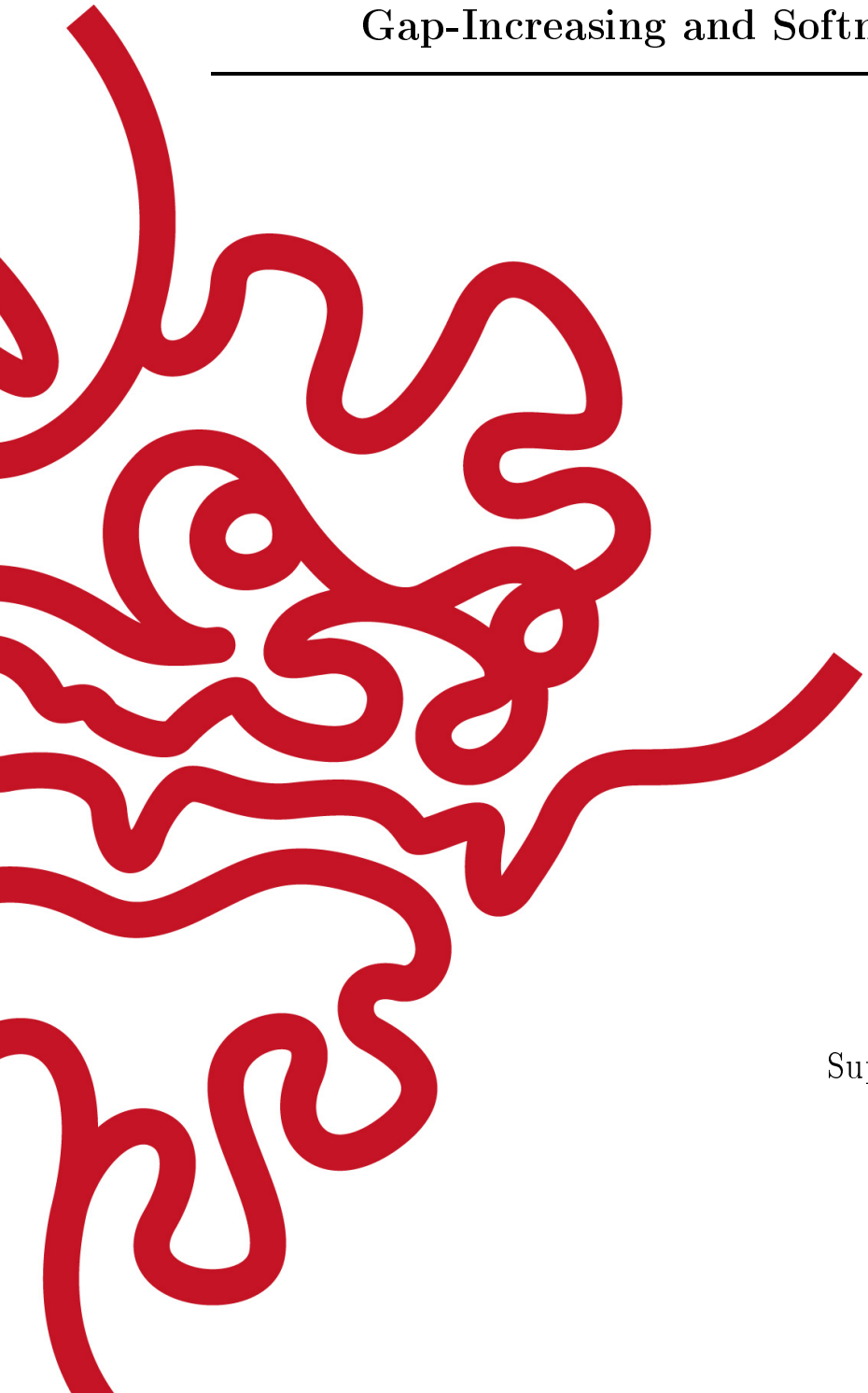
Okinawa Institute of Science and Technology
Graduate University

Thesis submitted for the degree

Doctor of Philosophy

Efficient and Noise-Tolerant Reinforcement Learning Algorithms via Theoretical Analysis of Gap-Increasing and Softmax Operators



by

Tadashi Kozuno

Supervisor: **Prof. Kenji Doya**

March, 2020

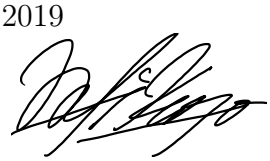
Declaration of Original and Sole Authorship

I, Tadashi Kozuno, declare that this thesis entitled *Efficient and Noise-Tolerant Reinforcement Learning Algorithms via Theoretical Analysis of Gap-Increasing and Softmax Operators* and the data presented in it are original and my own work.

I confirm that:

- No part of this work has previously been submitted for a degree at this or any other university.
- References to the work of others have been clearly acknowledged. Quotations from the work of others have been clearly indicated, and attributed to them.
- In cases where others have contributed to part of this work, such contribution has been clearly acknowledged and distinguished from my own work.
- None of this work has been previously published elsewhere, with the exception of papers on ArXiv repository and the following: "Tadashi Kozuno, Eiji Uchibe and Kenji Doya; Theoretical Analysis of Efficiency and Robustness of Softmax and Gap-Increasing Operators in Reinforcement Learning. Proceedings of Machine Learning Research, volume 89, Pages 2995-3003".

Date: June 2019

Signature: 

Abstract

Efficient and Noise-Tolerant Reinforcement Learning Algorithms via Theoretical Analysis of Gap-Increasing and Softmax Operators

Model-free deep Reinforcement Learning (RL) algorithms, a combination of deep learning and model-free RL algorithms, have attained remarkable successes in solving complex tasks such as video games. However, theoretical analyses and recent empirical results indicate its proneness to various types of value update errors including but not limited to estimation error of updates due to finite samples and function approximation error. Because real-world tasks are inherently complex and stochastic, such errors are inevitable, and thus, the development of error-tolerant RL algorithms are of great importance for applications of RL to real problems. To this end, I propose two error-tolerant algorithms for RL called Conservative Value Iteration (CVI) and Gap-increasing RetrAce for Policy Evaluation (GRAPE).

CVI unifies value-iteration-like single-stage-lookahead algorithms such as soft value iteration, advantage learning and Ψ-learning, all of which are characterized by the use of a gap-increasing operator and/or softmax operator in value updates. We provide detailed theoretical analysis of CVI that not only shows CVI's advantages but also contributes to the theory of RL in the following two points: First, it elucidates pros and cons of gap-increasing and softmax operators. Second, it provides an actual example in which performance of algorithms with max operator is worse than that of algorithms with softmax operator demonstrating the limitation of traditional greedy value updates.

GRAPE is a policy evaluation algorithm extending advantage learning (AL) and retrace, both of which have different advantages: AL is noise-tolerant as shown through our theoretical analysis of CVI, while retrace is efficient in that it is off-policy and allows the control of bias-variance trade-off. Theoretical analysis of GRAPE shows that it enjoys the merits of both algorithms. In experiments, we demonstrate the benefit of GRAPE combined with a variant of trust region policy optimization and its superiority to previous algorithms.

Through these studies, I theoretically elucidated the benefits of gap-increasing and softmax operators in both policy evaluation and control settings. While some open problems remain as explained in the final chapter, the results presented in this thesis are an important step towards a deep understanding of RL algorithms.

Acknowledgment

Firstly, I would like to thank my supervisor, Prof. Kenji Doya, who kindly accepted me to his unit. Due to an excessive number of students who wanted to join his unit, I was asked to consider joining another unit when I was selecting my thesis research unit. If he had not had accepted me, I would not be working on reinforcement learning, which is the most exciting topic for me.

Besides Prof. Doya, I am grateful to my former supervisors: Prof. Kunio Itoh at Ryukoku University and Prof. Yuki Hayashida at Osaka University. I learned how to enjoy science from them and decided to become a researcher. I also would like to mention Dr. Yuka Okazaki, a former colleague at Osaka University, who had shown me her intense passion for science and is my role model.

My sincere gratitude also goes to Prof. Evan Economo, Prof. Yoko Yazaki-Sugiyama, Prof. Jun Tani, Prof. Sile Nic Chormaic, Dr. Rémi Munos (Google DeepMind Paris), Prof. Takamitsu Matsubara (NAIST, Japan), and Dr. Bruno Scherrer (INRIA, France). Prof. Economo and Prof. Yazaki-Sugiyama are my thesis committee members and have helped me study in OIST. Prof. Tani and Dr. Munos were examiners of my thesis proposal. Dr. Munos also offered me an excellent internship opportunity at Google DeepMind Paris. Prof. Nic Chormaic chaired my thesis defense. Prof. Matsubara and Dr. Scherrer read and examined my thesis.

I am thankful to my colleagues in Neural Computation Unit. Dr. Eiji Uchibe (now at Advanced Telecommunications Research Institute International, Japan), Dr. Chris Reinke (now at INRIA, France), and Paavo Parmas gave me invaluable advice on my research. Ho Ching Chiu is a kind friend who often revised my manuscripts, and I enjoyed chatting with him. I also enjoyed talking with Masakazu Taira about neuroscience, which I used to study before. Dr. Hiroaki Hamada advised me of how to survive PhD. Besides Neural Computation Unit members, I am grateful to Dongqi Han in Cognitive Neurorobotics Research Unit for stimulating discussions, his cooperation in numerical experiments of GRAPE algorithm, and last-ditch efforts we made together before manuscript deadlines. Also I thank my close friends: Shohei Takaoka, Osamu Horiguchi, and Wataru Ohata. Wataru started to take me to OIST gym, and now workout is one of my favorite hobby.

I am thankful to other OIST members too. Dr. Steven Aird edited my manuscripts. It would be impossible to publish papers without him. People in IT section maintain Sango cluster, which I extensively used throughout my PhD research. Particularly, Dr. Jan Moren helped me a lot from the usage of Sango to optimizing codes.

Last but not the least, I would like to thank my parents Tsutomu and Shioko Kozuno for mentally helping me throughout my life. My sisters Asako Terawaki and

Aya Kozuno supported and entertained me too. My nieces, Hinano and Erino Terawaki, have supported me a lot by their cheering smiles when I was devastated by this tough PhD life.

Contents

List of Figures

List of Tables

Introduction

Many real-world decision making problems can be formulated as and solved by Reinforcement Learning (RL). To name a few, RL has been successfully applied to games (Samuel, 1959, 1967; Mnih et al., 2015; Silver et al., 2016; alp, 2019), robotics (Kober and Peters, 2014), aerobatic helicopter flight (Abbeel et al., 2007), packet routing (Boyan and Littman, 1994) and resource management (Mao et al., 2016). For detailed history of RL and its applications, see an introductory textbook (Sutton and Barto, 2018).

Unfortunately most real-world problems cannot be exactly solved for the following two reasons: (i) a decision must be made taking account of information typically expressed by a vector of real values (for example, joint angles in case of robot control), whose number of possible values is infinite; (ii) real-world problems are inherently stochastic and difficult to simulate, and thus, decisions need to be made by leveraging past experiences of real interactions with a real system. Due to the reason (i), solving real-world problems entails the use of function approximation. The reason (ii) necessitates tolerance of algorithms to stochasticity of problems.

To elaborate those two reasons, consider a simple and classical benchmark RL task called *mountain car task* (Moore, 1991; Sutton and Barto, 2018) shown in Figure 1. In this task, a learner (called an agent) observes its horizontal position x and velocity dx/dt. A pair $(x, dx/dt)$ is called a *state* of the car. Depending on a state, the agent needs to determine an *action*, i.e., to which direction and to what extent it accelerates the car.

The sets of all possible $(x, dx/dt)$ and d^2x/dt^2 values are called the *state and action spaces*, respectively. In this case, both of them are subsets of $\mathbb{R}^2$ and $\mathbb{R}$, respectively. Therefore some form of function approximation is necessary to handle such huge state and action spaces.

Although not depicted, a real-world version of this mountain car task would involve various stochasticity due to, e.g., road, wind and weather conditions. Therefore heading directly towards the goal from the bottom might be successful when the agent is lucky, whereas it might not when the agent is hapless.

As a result, it is important to theoretically understand effects of function approximation and tasks' stochasticity on performance of RL algorithms, which is the main theme of the thesis. Particularly the emphasis is on development and theoretical analysis of efficient RL algorithms with policy update regularizations/constraints (Azar et al., 2012; Rawlik, 2013; Schulman et al., 2015; Fox et al., 2016; Haarnoja et al., 2017, 2018; Abdolmaleki et al., 2018). Such algorithms have gained recent attention because of their superior empirical performance. Nonetheless they are lacking theoret-

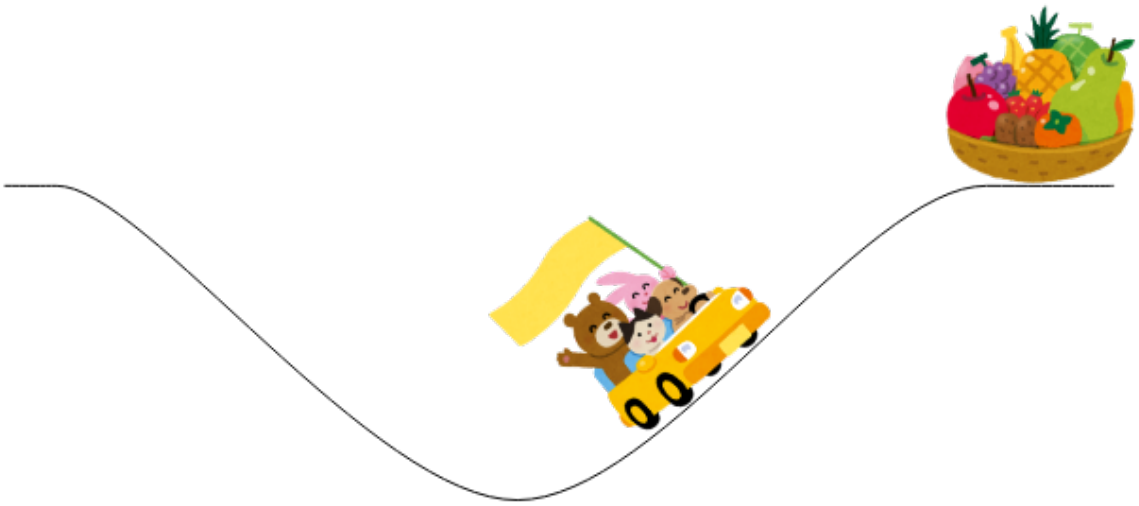


Figure 1: Mountain car task (Moore, 1991; Sutton and Barto, 2018). In this task, a learner (called an agent), shown as animals in the figure, learns to drive an under-powered car up a steep slope on the right hand side to reach the goal indicated by the fruits. As the car is under-powered, it is impossible to directly climb up the right slope. Instead the agent first needs to move away from the goal and up the opposite slope on the left hand side. Then the agent can move up to the right slope by using the gravity and applying full throttle.

ical foundation so far.

Mountain car task is a good example to highlight two major differences of RL from supervised learning, by which theoretical analysis of RL is complicated and difficult.

Firstly supervised learning is concerned with a one-shot task, while RL is with sequential decision making tasks. For example, supervised learning considers questions like "Is a dog in the given photo?". On the other, RL considers problems like learning to control a system while monitoring its state. Accordingly a long term effect of an action at each time step must be considered. In mountain car task, an agent first needs to go away from the goal so that the agent can swing up to it using the gravity.[1]

Secondly a learner in supervised setting has a direct access to correct answers, whereas an agent in RL does not. Concretely many pairs of input and desired output are given in supervised learning, while only real values called rewards are given in RL and needs to find a way to choose actions by which an agent can attain *maximum expected cumulative rewards*.

Because of these differences, typical model-free RL algorithms utilize "backup" and solve a RL problem as consecutive regression tasks wherein the backup is used to compute an output target and is updated to it. As a result, errors in a backup cause errors in subsequent backups, and thus, how "error propagates" through iterations needs to be analyzed. Such analysis is called *error propagation analysis* and frequently used (Munos, 2005, 2007; Farahmand, 2011; Scherrer et al., 2015; Scherrer, 2014). In the

[1] A similar situation frequently occurs in a real life too. For example, I have spent as long as five years of my life at OIST because I believe that a PhD degree makes my future carrier fruitful. (Let me see if it is truly fruitful using the rest of my life!)

thesis, error propagation analysis of various algorithms with policy update regularizations/constraints are carried out to understand properties of them.

Contribution

The following list summarizes contributions of the thesis.

- Proposing a new single-stage lookahead off-policy control algorithm, conservative value iteration, which unifies previous algorithms such as soft Q-learning, advantage learning and dynamic policy programming.
- Providing its error propagation analysis by which various properties of gap-increasing and softmax operators are elucidated.
- Proposing a new multi-stage lookahead off-policy policy evaluation algorithms, GRAPE and its variant called RGRAPE, based on gap-increasing operators.
- Providing their error propagation analysis that elucidates their noise-tolerance and faster convergence than the learning-rate based approach.
- Providing preliminary experimental results on GRAPE and RGRAPE that support our theoretical argument.

Chapter 1

Sequential Decision Making

This chapter lays out the mathematical foundation of Sequential Decision Making (SDM) problems necessary to understand the thesis. For accessibility, the chapter starts from basics of the set, topology and measure theories. Section 1.1 introduces mathematical notations and definitions. Section 1.2 introduces Markov Decision Processes (MDPs), which is the standard framework of SDM problems. Section 1.3 explains the policy and its value functions. In RL, actions are selected according to a so-called policy. The agent seeks for a policy that is optimal in the sense explained in Section 1.4. In Section 1.5, we describe the setting of problems we consider in the thesis. In Section 1.6, we explain a class of RL algorithms called Approximate Dynamic Programming (ADP) for this problem setting. At the end of the chapter in Section 1.7, we explain error propagation analysis, which will be used to analyze our algorithms.

1.1 Mathematical Notations and Definitions

In RL we frequently use a finite dimensional Euclidean space or its subset, both of which have some structure, such as the topology. Based on their topology, we define their σ-algebra, without which we cannot perform our rigorous analysis.

This section introduces those mathematical notations related to the set, topology and measure theories. Readers may skip this section if they are familiar with them. This section is based on two books (Bertsekas and Shreve, 1996) and (Dudley, 2002). Basic mathematical definitions are based on (Dudley, 2002), whereas some advanced ones related to SDM are based on (Bertsekas and Shreve, 1996).

Set Notations

Sets are denoted by curly upper case English letters $\mathcal{A}, \mathcal{B}, \ldots, \mathcal{Z}$. Collections of sets are denoted by Fraktur upper case letters $\mathfrak{A}, \mathfrak{B}, \ldots, \mathfrak{Z}$. Functions and scalars are denoted by lowercase English and Greek alphabet letters with some exceptions to follow conventions. (For example, value functions Q^π and V^π explained later.) Operators are denoted by bold face English and Greek alphabet letters, such as $\boldsymbol{O}$. Suppose a sequence of sets $\mathcal{X}_1, \ldots \mathcal{X}_i$. Their Cartesian product is denoted in two ways: $\prod_{i=1}^n \mathcal{X}_i$ and $\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_n$. When $\mathcal{X}_1 = \cdots = \mathcal{X}_n = \mathcal{X}$, the Cartesian product is simply denoted by $\mathcal{X}^n$. For subsets $\mathcal{Y}$ and $\mathcal{Z}$ of a set $\mathcal{X}$, their set-theoretic difference is

$\mathcal{Y} - \mathcal{Z} := \{y \in \mathcal{Y} | y \notin \mathcal{Z}\}$. The complement of $\mathcal{Y}$ is $\mathcal{Y}^c := \mathcal{X} - \mathcal{Y}$. The power set $2^{\mathcal{X}}$ is the collection of all $\mathcal{X}$'s subsets.

To follow mathematical convention, the set of real numbers, quotient numbers and integers are denoted by special letters $\mathbb{R}$, $\mathbb{Q}$ and $\mathbb{Z}$, respectively. Non-negative part of those sets are denoted by $\mathbb{R}_+$, $\mathbb{Q}_+$ and $\mathbb{Z}_+$, respectively. Positive part of those sets are denoted by $\mathbb{R}_{++}$, $\mathbb{Q}_{++}$ and $\mathbb{Z}_{++}$, respectively. A set of integers $\{a, a+1, \ldots, b-1, b\}$ is denoted by $\{a : b\}$. The empty set is denoted by $\varnothing$. Closed and open intervals are denoted by $[a, b] := \{x | a \leq x \leq b, x \in \mathbb{R}\}$ and $(a, b) := \{x | a < x < b, x \in \mathbb{R}\}$, respectively. Half-open intervals are denoted by $[a, b)$ or $(a.b]$.

Function Notations

When we define a function, we use $\mapsto$ to mean "maps to". For example, we may write a real-valued function $f(x) := x^2$ by $f : x \mapsto x^2$ or $f : x \in \mathbb{R} \mapsto x^2 \in \mathbb{R}$ when we clarify the domain and co-domain of f. When we want to clarify just a domain and co-domain of a function, we use $\to$. For example, for a function g whose domain is $\mathbb{R}$, and co-domain is $\mathbb{R}^2$, we write $g : \mathbb{R} \to \mathbb{R}^2$.

Topology

The topology is the most fundamental structure that can be equipped to a set. Based on the topology, important mathematical notions, such as the continuity and Borel measurability of functions, are defined. The topology is defined as follows.

Definition 1.1.1 (Topology). *For a set $\mathcal{X}$, its topology $\mathfrak{T}$ is a collection of subsets of $\mathcal{X}$ satisfying the following three conditions (Dudley, 2002, Section 2.1):*

1. *$\varnothing \in \mathfrak{T}$ and $\mathcal{X} \in \mathfrak{T}$.*
2. *For any $\mathfrak{U} \subset \mathfrak{T}$, we have $\bigcup \mathfrak{U} \in \mathfrak{T}$.*
3. *For every $\mathcal{X} \in \mathfrak{T}$ and $\mathcal{Y} \in \mathfrak{T}$, we have $\mathcal{X} \bigcap \mathcal{Y} \in \mathfrak{T}$.*

The topological space is a pair of a set and its topology. An element in a topology is called an open set, while complement of an open set is called a closed set. For brevity we denote a topological space, say $(\mathcal{X}, \mathfrak{T})$, by $\mathcal{X}$ when the topology is clear.

Note that for a set, there may be multitude of topologies. For example, both $\{\varnothing, \mathcal{X}\}$ and $2^{\mathcal{X}}$ are eligible to be a topology. The former one is called the trivial topology, while the latter the discrete topology.

Example 1.1.1 (Discrete Topology). *While the trivial topology is (likely to be) useless, the discrete topology is occasionally used. Indeed for a finite set, the discrete topology is usually used. For an example of the discrete topology, consider the set $\{1, 2, 3\}$. Its discrete topology is given by $\{\varnothing, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$.*

Example 1.1.2 (One-Dimensional Euclidean Space). *Consider the set $\mathbb{R}$ of real values and a collection $\mathfrak{O}$ consisting of sets that can be expressed as a union of open intervals. Then the collection $\mathfrak{O}$ satisfies the conditions of the topology. We usually equip $\mathbb{R}$ with this topology, and call the topological space $(\mathbb{R}, \mathfrak{O})$ as an (one-dimensional) Euclidean space. We frequently denote it simply by $\mathbb{R}$.*

Consider a function f from a topological space $(\mathcal{X}, \mathfrak{T})$ to another topological space $(\mathcal{Y}, \mathfrak{U})$. It is said to be continuous when for any open set $\mathcal{O}_Y$ in $\mathfrak{U}$, its pre-image $f^{-1}(\mathcal{O}_Y)$ is in $\mathfrak{T}$. A topological space $(\mathcal{X}, \mathfrak{T})$ is said to be homeomorphic to a topological space $(\mathcal{Y}, \mathfrak{U})$ if there exists a bijective mapping f from $\mathcal{X}$ to $\mathcal{Y}$ such that both f and its inverse mapping f^{-1} are continuous. Such a mapping is called homeomorphism.

Suppose a sequence of sets $(\mathcal{X}_i)_{i\in\mathcal{I}}$, where $\mathcal{I}$ is an index set, and each set $\mathcal{X}_i$ is equipped with a topology $\mathfrak{T}_i$. Let $\mathcal{X}$ denote its Cartesian product $\prod_i \mathcal{X}_i$. The product topology $\mathfrak{T}$ of the product set $\mathcal{X}$ is its smallest topology such that each projection function $\pi_i : (x_i)_{i\in\mathcal{I}} \in \mathcal{X} \mapsto x_i \in \mathcal{X}_i$ is continuous. If the index set $\mathcal{I}$ is finitely countable, the product topology coincides with the smallest topology containing the collection $\{\prod_i \mathcal{O}_i | \mathcal{O}_i \in \mathfrak{T}_i\}$.

Example 1.1.3 (Finite Dimensional Euclidean Space)**.** *Consider N one-dimensional Euclidean spaces $\mathbb{R}$. For their Cartesian product $\mathbb{R}^N$, we usually equip it with a collection $\mathfrak{O}_N$ that is the smallest topology containing $\{\prod_i \mathcal{O}_i | \mathcal{O}_i \in \mathfrak{O}, i = 1, \ldots, N\}$ with $\mathfrak{O}$ being the topology of $\mathbb{R}$. We call the topological space $\left(\mathbb{R}^N, \mathfrak{O}_N\right)$ as an (N-dimensional or a finite dimensional) Euclidean space. As is the case with an one-dimensional Euclidean space, we often denote $\left(\mathbb{R}^N, \mathfrak{O}_N\right)$ by $\mathbb{R}^N$.*

For a subset $\mathbb{Q}^N$ of $\mathbb{R}^N$, its closure (the smallest closed set containing it) is again $\mathbb{R}^N$. Note that $\mathbb{Q}^N$ is a countable set. Separability means this property of a topological space that there is a countable set whose closure covers the entire set. Such a countable set is said to be dense. The finite dimensional Euclidean space is separable. As we explain later, it is also complete and metrizable.

For some types of sets, there are natural topologies. A finite set is usually equipped with its discrete topology. For a subset $\mathcal{Y}$ of a set $\mathcal{X}$, the collection $\{\mathcal{Y} \bigcap \mathcal{O} | \mathcal{O} \in \mathfrak{T}\}$ with $\mathfrak{T}$ being $\mathcal{X}$'s topology can be a topology of the subset $\mathcal{Y}$. Such a topology is called the relative topology. A subset of a topological space is usually equipped with a relative topology. A Cartesian product of sets is equipped with a product topology, as in Example 1.1.3.

Metrics and Norms

The metric is a function that measures distance between two points. While the metric is not frequently used in RL, we need it for stating some theoretical results. The metric is defined as follows. (As in the case of the topological space, we denote the metric space $(\mathcal{X}, d)$ by $\mathcal{X}$ when the metric is clear.)

Definition 1.1.2 (Metric)**.** *A metric space is a pair $(\mathcal{X}, d)$ of a set $\mathcal{X}$ and a metric $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, which satisfies the following conditions (Dudley, 2002, Section 2.1):*

1. *For all $x \in \mathcal{X}$ and $y \in \mathcal{X}$, $d(x, y) = 0$ if and only if $x = y$.*
2. *For all $x \in \mathcal{X}$ and $y \in \mathcal{X}$, $d(x, y) = d(y, x)$.*
3. *For all x, y and z in $\mathcal{X}$, $d(x, z) \leq d(x, y) + d(y, z)$.*

Given a metric d over a set $\mathcal{X}$ and a positive real value $\varepsilon > 0$, an open ball is the set $\mathcal{B}_\varepsilon(x) := \{y \in \mathcal{X} | d(x, y) < \varepsilon\}$. The collection of all open balls $\mathfrak{B} := \{\mathcal{B}_\varepsilon(x) | x \in$

$\mathcal{X}, \varepsilon \in \mathbb{R}_{++}\}$ becomes a base for the topology $\{\bigcup \mathfrak{A} | \mathfrak{A} \subset \mathfrak{B}\}$. Such a topology is said to be induced (or generated) by the metric d. A metric space is usually equipped with a topology induced by its metric.

A topological space is said to be metrizable if there exists a metric such that a topology induced by the metric is consistent with the original topology. Note that a metrizable space is distinct from a metric space: a choice of a metric is open in metrizable space, whereas a metric is already determined in a metric space.

Example 1.1.4 (Finite Dimensional Euclidean Space Defined Through the Euclidean Distance)**.** *For an N-dimensional Euclidean space $\mathbb{R}^N$, the Euclidean norm*

$$\|x\|_2 := \sqrt{x_1^2 + x_2^2 + \cdots + x_{N-1}^2 + x_N^2}$$

and the Euclidean distance $d_2(x, y) := \|x - y\|_2$ are usually used. Recall that we defined a finite dimensional Euclidean space $\mathbb{R}^N$ as in Example 1.1.3. While the Euclidean space is frequently defined as the metric space $\left(\mathbb{R}^N, d_2\right)$, it can be shown that both definitions are consistent. In other words, the Euclidean space is metrizable by d_2. For the Euclidean space, we usually equip it with d_2.

A metric space is said to be complete if any Cauchy sequence converges to a point in the space. Many spaces in the thesis are complete. An example of a non-complete space is $\mathbb{Q}$ equipped with the Euclidean distance. (Consider a Cauchy sequence $(S_i | i \in \mathbb{Z}_{++})$ consisting of i leading digits of $\sqrt{2}$ converges to $\sqrt{2}$, but it is not in $\mathbb{Q}$.)

The norm is a function that measures the length of a vector. It is defined as follows. (As usual, we denote the normed vector space $(\mathcal{X}, \|\cdot\|)$ by $\mathcal{X}$ when the norm is clear.)

Definition 1.1.3 (Norm)**.** *Suppose a vector space $\mathcal{X}$ over a field $\mathbb{R}$. The norm over the vector space is a non-negative function $\|\cdot\| : \mathcal{X} \to \mathbb{R}_+$ satisfying the following conditions (Dudley, 2002, Section 5.1):*

1. *For any $x \in \mathcal{X}$ and a real value $c \in \mathbb{R}$, $\|cx\| = |c| \, \|x\|$.*
2. *For any $x \in \mathcal{X}$ and $y \in \mathcal{X}$, $\|x + y\| \leq \|x\| + \|y\|$.*
3. *For any $x \in \mathcal{X}$, $\|x\| = 0$ if and only if $x = 0$.*

If the last condition is not satisfied, $\|\cdot\|$ is called a seminorm. A normed vector space is the pair $(\mathcal{X}, \|\cdot\|)$.

We note that a norm $\|\cdot\|$ induces a metric $d(x, y) := \|x - y\|$. Therefore the norm and the metric are the almost same object.

σ-algebra, Measures and Measurable Functions

The measure is a real-valued function over a collection of sets. It measures the "area" of a set and is used as the foundation of integral and probability theory. It is defined as follows. (As is the case with other types of spaces, we denote the measurable (or measure) space by $\mathcal{X}$ when its σ-algebra and/or measure is clear.)

Definition 1.1.4 (Measurable Space and Measure)**.** *A measurable space is a pair* $(\mathcal{X}, \mathfrak{A})$ *of a set* $\mathcal{X}$ *and its* σ*-algebra* $\mathfrak{A} \subset 2^{\mathcal{X}}$*, which satisfies the following conditions (Dudley, 2002, Section 3.1):*

1. $\varnothing \in \mathfrak{A}$ *and* $\mathcal{X} \in \mathfrak{A}$*.*
2. *For any countable collection* $\mathfrak{C}$ *of subsets in* $\mathfrak{A}$*, we have* $\bigcup \mathfrak{C} \in \mathfrak{A}$*.*
3. *For every* $\mathcal{X} \in \mathfrak{A}$ *and* $\mathcal{Y} \in \mathfrak{A}$*, we have* $\mathcal{X} - \mathcal{Y} \in \mathfrak{A}$*.*

A member of $\mathfrak{A}$ *is called a measurable set. A measure space is the triplet* $(\mathcal{X}, \mathfrak{A}, \mu)$*, where* $\mu : \mathfrak{A} \to \mathbb{R}_+$ *is a measure, which satisfies the following conditions (Dudley, 2002, Section 3.1):*

1. $\mu(\varnothing) = 0$*.*
2. *For any measurable set* $\mathcal{M}$*, we have* $\mu(\mathcal{M}) \geq 0$*.*
3. *For any countable collection* $\mathfrak{C} = \{\mathcal{M}_i\}$ *of disjoint measurable sets* $\mathcal{M}_i$*, we have* $\mu(\bigcup \mathfrak{C}) = \sum_i \mu(\mathcal{M}_i)$*.*

A probability measure is a special measure satisfying $\mu(\mathcal{X}) = 1$. The sets of all measures and probability measures over the measurable space $(\mathcal{X}, \mathfrak{A})$ are denoted by $\mathcal{M}(\mathcal{X})$ and $\mathcal{P}(\mathcal{X})$, respectively.

For a topological space $(\mathcal{X}, \mathfrak{T})$, the Borel σ-algebra is a σ-algebra generated from $\mathfrak{T}$, i.e., the smallest σ-algebra containing the topology $\mathfrak{T}$. (Concretely it is a collection of subsets of $\mathcal{X}$ obtained by a countable number of union, intersection and complement operations.) A σ-algebra of a topological space is usually assumed to be a Borel σ-algebra. A Borel σ-algebra of a topological space is denoted by $\mathfrak{B}(\mathcal{X}, \mathfrak{T})$ or $\mathfrak{B}(\mathcal{X})$ when the topology is clear.

As noted before, a Cartesian product of finitely many sets is usually equipped with its product topology. Such a Cartesian product is usually equipped with a Borel σ-algebra generated from the product topology.

A measurable function $f : (\mathcal{X}, \mathfrak{A}) \to (\mathcal{Y}, \mathfrak{B})$ is a function such that for any measurable subset $\mathcal{B} \in \mathfrak{B}$, its pre-image $f^{-1}(\mathcal{B}) \in \mathfrak{A}$. To be precise, f is sometime said to be $(\mathfrak{A}, \mathfrak{B})$-measurable. When both σ-algebras are Borel σ-algebras, f is simply called a Borel-measurable function.

Example 1.1.5 (Examples of Measurable Functions)**.** *Consider a real-valued function* $f : \mathbb{R}^N \to \mathbb{R}$*. Note that its domain and co-domain are finite dimensional Euclidean spaces, and thus, their natural* σ*-algebras are the smallest* σ*-algebras containing the topologies of* $\mathbb{R}^N$ *and* $\mathbb{R}$*, i.e., Borel* σ*-algebras.*

If the function f *is continuous, it is measurable too. (Readers might be able to see that the measurability is indeed an extension of the continuity: the definition of the measurability can be obtained by replacing the topology in the definition of the continuity with the* σ*-algebra.)*

Another class of measurable functions is semi-continuous functions. In RL, lower semi-continuous functions sometimes appear.[1] *A notable property of the lower semi-continuous function is that its partial supremization, that is,* $\sup_{x_i} f(x_1, \ldots, x_N)$ *over*

[1] A function $f : \mathbb{R}^N \to \mathbb{R}$ is lower semi-continuous if and only if $\{(f(x), x) | x \in \mathbb{R}^N\}$ is closed.

one input variable x_i, is again a lower semi-continuous function. In RL, such a partial supremization plays an important role to prove the existence of an optimal policy.

Let $(\mathcal{X}, \mathfrak{A})$ be a measurable space. Consider two measures μ and ν on it. The measure ν is said to be absolutely continuous with respect to the another measure μ (written as $\nu \prec \mu$) if $\mu(\mathcal{A}) = 0$ implies $\nu(\mathcal{A}) = 0$ for any measurable set $\mathcal{A} \in \mathfrak{A}$. A σ-finite measure ρ is a measure such that there exists a sequence of subsets $(\mathcal{A}_i)_{i \in \mathbb{Z}_{++}}$ such that its union covers the original set $\mathcal{X}$ while $\rho(A_i) < \infty$ for each subset. If the measure satisfies $\rho(\mathcal{X}) < \infty$, it is said to be finite.

Regarding absolute continuity of measures, the following theorem from (Dudley, 2002) are important. (The definition of $L^1(\mathcal{X}, \mathfrak{A}, \mu)$ will be given later.) We provide it here, but for readability we repeatedly present it in later chapters when needed.

Theorem 1.1.1 (Radon-Nykodim Derivative)**.** *On the measurable space $(\mathcal{X}, \mathfrak{A})$ let μ be a σ-finite measure. Let ν be a finite measure, absolutely continuous with respect to μ. Then for some $h \in L^1(\mathcal{X}, \mathfrak{A}, \mu)$, $\nu(\mathcal{E}) = \int_{\mathcal{E}} h(x)\mu(dx)$ for all $\mathcal{E}$ in $\mathfrak{A}$.*

The function h in Theorem 1.1.1 is called the Radon-Nykodim derivative of ν with respect to μ and denoted by ν/μ. Note that probability measures are always finite, and thus, the absolute continuity only matters for the existence of the Radon-Nykodim derivative.

Example 1.1.6 (Example of Radon-Nykodim Derivative)**.** *Although the definition of the Radon-Nykodim derivative looks intimidating, it can be understood as a measure-theoretic version of the importance sampling ratio. Consider a finite set $\{1 : N\}$ and two measures μ, ν over it. The absolute continuity, $\nu \prec \mu$, means that $\mu(i) = 0 \Rightarrow \nu(i) = 0$ for any i. Thus for any set $\mathcal{S}$ of integers between 1 to N, $\nu(\mathcal{S}) = \sum_{j \in \mathcal{S}, \mu(j) \neq 0} \frac{\nu(j)}{\mu(j)} \mu(j)$, and hence, $\frac{\nu(\cdot)}{\mu(\cdot)}$ is the Radon-Nykodim derivative of ν with respect to μ.*

Function Spaces and Norms of Functions

Suppose two real-valued functions $f, g : \mathcal{X} \to \mathbb{R}$. Addition $f + g$ of the two functions is defined such that $(f + g)(x) = f(x) + g(x)$ for any point $x \in \mathcal{X}$. Negation $-g$ of the function g is defined such that $(-g)(x) := -g(x)$. Addition of f and $-g$ is simply denoted by $f - g$. Multiplication of a scalar $c \in \mathbb{R}$ and the function f is denoted by cf and defined by $(cf)(x) = f(x)$.

Suppose a topological space $\mathcal{X}$. The set of all continuous functions from $\mathcal{X}$ to $\mathbb{R}$ is denoted by $\mathcal{C}(\mathcal{X})$. The set of all bounded Borel-measurable real-valued functions is denoted by $\mathcal{B}(\mathcal{X})$. For a positive real value $L \in \mathbb{R}_{++}$, its subset consisting of all functions bounded by L is denoted by $\mathcal{B}(\mathcal{X}, L)$.

For a bounded measurable function $f : \mathcal{X} \to \mathbb{R}$, L^∞-norm is defined by $\|f\|_\infty := \sup_x |f(x)|$. Suppose a measure space $(\mathcal{X}, \mathfrak{A}, \mu)$ and a real value $p \in [1, \infty)$. A set of functions $L^p(\mathcal{X}, \mathfrak{A}, \mu)$ is the set of all measurable functions, say f, such that its integral $I := \int_{\mathcal{X}} |f(x)|^p \mu(dx)$ is finite. The space is usually equipped with $L^p(\mathcal{X}, \mathfrak{A}, \mu)$-norm defined by $\|f\|_{p,\mu} := I^{1/p}$, and it is called $L^p(\mathcal{X}, \mathfrak{A}, \mu)$ space. When the measurable space is clear, $L^p(\mathcal{X}, \mathfrak{A}, \mu)$-norm is simply denoted as $L^p(\mu)$-norm.[2]

[2]Precisely speaking, $L^p(\mu)$-norm is a seminorm because $\|f\|_{p,\mu} = 0$ may not imply $f(x) = 0$. For

Operators and Arithmetic Operations with Operators

Suppose two normed vector spaces of functions over a field $\mathbb{R}$. An operator is a mapping from one of the vector space to the other. An operator $\boldsymbol{o}$ is said to be linear if $\boldsymbol{o}(f+g) = \boldsymbol{o}f + \boldsymbol{o}g$ for any functions f and g. A linear operator $\boldsymbol{o}$ from a normed function space $(\mathcal{X}, |\cdot|)$ to another normed function space $(\mathcal{Y}, \|\cdot\|)$ is continuous if and only if its operator norm $\|\boldsymbol{o}\|_{op} := \sup_{f\in\mathcal{X}} \|\boldsymbol{o}f\| / |f|$ is finite.

For an operator $\boldsymbol{o}$ from a normed vector space $(\mathcal{X}, |\cdot|)$ to another normed vector space $(\mathcal{Y}, \|\cdot\|)$ and a scalar $c \in \mathbb{R}$, multiplication of c and $\boldsymbol{o}$ is defined as an operator $c\boldsymbol{o}$ such that $c\boldsymbol{o} : f \mapsto c(\boldsymbol{o}f)$. For an operator $\boldsymbol{o}$ from a normed vector space $(\mathcal{X}, |\cdot|)$ to itself and a positive integer $n \in \mathbb{Z}_{++}$, an operator $\boldsymbol{o}^n$ is recursively defined as $\boldsymbol{o}^n : f \mapsto \boldsymbol{o}(\boldsymbol{o}^{n-1}f)$, i.e., applications of the same operator for n times. Suppose two operators $\boldsymbol{o}_1$ and $\boldsymbol{o}_2$. Their addition $\boldsymbol{o}_1 + \boldsymbol{o}_2$ and multiplication $\boldsymbol{o}_1\boldsymbol{o}_2$ are defined as $\boldsymbol{o}_1 + \boldsymbol{o}_2 : f \mapsto \boldsymbol{o}_1 f + \boldsymbol{o}_2 f$ and $\boldsymbol{o}_1\boldsymbol{o}_2 : f \mapsto \boldsymbol{o}_1(\boldsymbol{o}_2 f)$, respectively. (Their domains and co-domains must be appropriately defined accordingly.)

Example 1.1.7 (Differentiation as an Operator)**.** *When $\mathcal{X}$ is the set of all infinitely differentiable functions from $\mathbb{R}$ to $\mathbb{R}$, $\frac{\boldsymbol{d}}{\boldsymbol{dx}}$ is an operator from $(\mathcal{X}, |\cdot|)$ to itself defined by*

$$\frac{\boldsymbol{d}}{\boldsymbol{dx}} : f \in \mathcal{X} \mapsto \frac{\boldsymbol{d}}{\boldsymbol{dx}} f \in \mathcal{X} \text{ such that } \forall y \in \mathbb{R}, \left(\frac{\boldsymbol{d}}{\boldsymbol{dx}} f\right)(y) = \frac{df}{dx}(y).$$

It can be applied infinitely many times, and its n-th power is given by

$$\left(\frac{\boldsymbol{d}}{\boldsymbol{dx}}\right)^n : f \mapsto \left(\frac{\boldsymbol{d}}{\boldsymbol{dx}}\right)^n f \text{ such that } \forall y \in \mathbb{R}, \left[\left(\frac{\boldsymbol{d}}{\boldsymbol{dx}}\right)^n f\right](y) = \frac{d^n f}{dx^n}(y).$$

This operator is also linear.

Suppose an operator $\boldsymbol{o}$ from a normed vector space $(\mathcal{X}, |\cdot|)$ to itself. The operator $\boldsymbol{o}$ is said to be a contraction mapping with modulus $L \in [0, 1)$ if for any functions f and g in the set $\mathcal{X}$, $|\boldsymbol{o}f - \boldsymbol{o}g| \leq L|f - g|$ holds. Banach's fixed point theorem states that there is a unique fixed point for any operator that is a contraction mapping. If the operator $\boldsymbol{o}$ is linear and a contraction mapping, a Neumann series $(\boldsymbol{I} - \boldsymbol{o})^{-1}$ of $\boldsymbol{o}$ is well-defined and given by $(\boldsymbol{I} - \boldsymbol{o})^{-1} := \sum_{i=0}^{\infty} \boldsymbol{o}^i$. As the notation implies, it satisfies that $(\boldsymbol{I} - \boldsymbol{o})^{-1}(\boldsymbol{I} - \boldsymbol{o}) = (\boldsymbol{I} - \boldsymbol{o})(\boldsymbol{I} - \boldsymbol{o})^{-1} = \boldsymbol{I}$.

Borel Spaces and Borel-Stochastic Kernels

A topological space $\mathcal{X}$ is said to be a Borel space if it is homeomorphic to a Borel subset $\mathcal{B} \in \mathfrak{B}(\mathcal{Y})$ of a complete separable metric space $\mathcal{Y}$ (Bertsekas and Shreve, 1996). Many spaces in the thesis are Borel spaces. For example, $\mathbb{R}^n$ is a complete separable metric space homeomorphic to itself. Because compact Borel-measurable subsets of a complete

example, suppose that μ is a discrete probability measure whose support is a set $\mathcal{Y}$. Then $\|f\|_{p,\mu} = 0$ holds as long as $f(t) = 0$ for all points t in the support $\mathcal{Y}$. To solve this issue, the quotient set of $L^p(\mu)$ by equivalence $f \sim g \iff \|f - g\|_{p,\mu} = 0$ with $L^p(\mu)$-norm is considered instead of $L^p(\mathcal{X}, \mathfrak{A}, \mu)$ space. However this subtlety is not important to the discussion of the thesis, and thus, it is ignored.

separable metric space are complete and separable by Corollary 7.6.2 of Bertsekas and Shreve (1996), they are Borel spaces. Thus the state and action spaces $\mathcal{X}, \mathcal{A}$ (explained later), both of which are compact subsets of finite dimensional Euclidean spaces, are Borel spaces. Furthermore, a Cartesian product of two Borel spaces is again a Borel space with its product topology by Proposition 7.13 of Bertsekas and Shreve (1996). Therefore, $\mathcal{X} \times \mathcal{A}$ is a Borel space too.

Suppose two Borel spaces $\mathcal{X}$ and $\mathcal{Y}$ equipped with Borel σ-algebras. A stochastic kernel p on $\mathcal{Y}$ given $\mathcal{X}$ is a function from $\mathcal{X}$ to $\mathcal{P}(\mathcal{Y})$. A probability measure obtained by mapping $x \in \mathcal{X}$ with p is denoted by $p(\cdot|x)$.

Suppose a stochastic kernel p on $\mathcal{Y}$ given $\mathcal{X}$. It is said to be Borel-measurable if and only if for any Borel-measurable function $f : \mathcal{X} \to \mathcal{Y}$, a function $x \in \mathcal{X} \mapsto \int_{\mathcal{Z}} f(y)p(dy|x)$ is Borel-measurable for every $\mathcal{Z} \in \mathfrak{B}(\mathcal{Y})$. (This can be relatively easily proven based on Proposition 7.26 of (Bertsekas and Shreve, 1996) and the definition of Lebesgue integration.) In other words, it preserve Borel-measurability of functions. This property is important to define value functions in Section 1.3.

1.2 Markov Decision Processes

In this section, we introduce Markov Decesion Process (MDP), a mathematical framework for SDM problems. Definition of MDPs varies depending on a problem setting at hand (Puterman, 1994; Bertsekas and Shreve, 1996; Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 2018). Since only the infinite-time horizon discounted MDP is considered in the thesis, we introduce it here. We shall simply call it an MDP. We impose several assumptions on MDPs to mainly ensure the existence of an optimal policy. They are summarized at the end of this section.

Definition 1.2.1 (Markov Decision Process)**.** *An MDP is a tuple* $(\mathcal{X}, \mathcal{A}, T, \rho_0, \gamma)$. $\mathcal{X}$ *is the state space assumed to be either a countable or compact subset of a finite dimensional Euclidean space.* $\mathcal{A}$ *is the action space assumed to be a finite set* $\{1, 2, \ldots, |\mathcal{A}|\}$. *The transition kernel* $T : \mathcal{X} \times \mathcal{A} \to \mathcal{P}(\mathcal{X} \times \mathbb{R})$ *is a Borel-measurable stochastic kernel on* $\mathcal{X} \times \mathbb{R}$ *given* $\mathcal{X} \times \mathcal{A}$. ρ_0 *is a probability measure over* $\mathcal{X}$ *defining an initial state distribution. The discount factor* $\gamma \in [0, 1)$ *is a positive real value used to discount future rewards.*

The components of the MDP are understood as follows: first, an agent is placed to a state $X_0 \in \mathcal{X}$ sampled from ρ_0. The agent reacts to it by executing an action $A_0 \in \mathcal{A}$ sampled from a policy $\pi(\cdot|X_0)$ (explained later). Then state transition to a subsequent state X_1 with an immediate reward R_1 occurs such that the pair (X_1, R_0) is sampled from $T(\cdot|X_0, A_0)$. The agent again reacts to the new state X_1 by executing an action $A_1 \sim \pi(\cdot|X_1)$ followed by state transition to a new state X_2 with an immediate reward R_1. This process is continued forever.[3]

The transition kernel T is inconvenient for theoretical analysis. Instead its marginal called the state transition probability kernel and the expected reward function are frequently used.

[3] In practice the state is reset to $X_0' \sim \rho_0$ after some time steps. More practical setting is explained later in Section 1.5.

Definition 1.2.2 (State Transition Probability Kernel and Expected Reward Function)**.** *From Corollary 7.27.1 of Bertsekas and Shreve (1996), T can be decomposed to two Borel-measurable stochastic kernels $R : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathcal{P}(\mathbb{R})$ and $P : \mathcal{X} \times \mathcal{A} \to \mathcal{P}(\mathcal{X})$ such that*

$$T(\mathcal{Y} \times \mathcal{B}|x, a) = \int_{\mathcal{Y}} R(\mathcal{B}|x, a, y)P(dy|x, a)$$

holds for any $\mathcal{Y} \in \mathfrak{B}(\mathcal{X})$ and $\mathcal{B} \in \mathfrak{B}(\mathbb{R})$. We call $R(\cdot|x, a, y)$ and $P(\cdot|x, a)$ reward and state transition probability kernels, respectively. The (expected) reward function is defined by

$$r(x, a) := \int_{\mathcal{Y}} \rho R(d\rho|x, a, y)P(dy|x, a),$$

which is guaranteed to be Borel-measurable by Proposition 7.29 of Bertsekas and Shreve (1996). We assume r to be bounded by a positive real value $r_{max} \in \mathbb{R}_{++}$.

Here is a list of assumptions imposed on MDPs.

Assumption 1.2.1 (Assumptions on MDPs)**.** *The following assumptions are imposed on MDPs.*

- *The state space $\mathcal{X}$ is either a countable or compact subset of a finite dimensional Euclidean space.*
- *The action space $\mathcal{A}$ is a finite set $\{1, 2, \ldots, |\mathcal{A}|\}$.*
- *The transition kernel $T : \mathcal{X} \times \mathcal{A} \to \mathcal{P}(\mathcal{X} \times \mathbb{R})$ is Borel-measurable.*
- *The reward function $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is a Borel-measurable function bounded by a positive real value $r_{max} \in \mathbb{R}_{++}$*

The assumption of the action space can be relaxed: if the reward function is a lower semi-continuous, the action space can be a compact subset of a finite dimensional Euclidean space. For details, see Puterman (1994).

1.3 Policies and Value Functions

An agent chooses actions according to a policy, which is a stochastic kernel on $\mathcal{A}$ given $\mathcal{X}$. When actions are selected according to a policy, we say that the policy is followed. Value functions of a policy are expected cumulative discounted rewards when it is followed, and they play important roles in defining the optimality of a policy as well as learning an optimal policy. We explain them more concretely in this section.

The policy is defined as follows.

Definition 1.3.1 (Policy)**.** *The policy is a Borel-measurable stochastic kernel on $\mathcal{A}$ given $\mathcal{X}$. Given a policy π and a state $x \in \mathcal{X}$, an agent chooses action $a \sim \pi(\cdot|x)$. When an agent is selecting actions according to a policy, an agent is said to be following the policy.*

Let $\boldsymbol{P}$ be a linear operator such that $(\boldsymbol{P}V)(x,a) = \int_{\mathcal{X}} V(y)P(dy|x,a)$ for any function V in $\mathcal{B}(\mathcal{X})$. (Recall that P is the state transition probability kernel.) For a policy π, let $\boldsymbol{\pi}$ be a linear operator such that $(\boldsymbol{\pi}Q)(x) = \int_{\mathcal{A}} Q(x,a)\pi(da|x)$ for any function Q in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$. Let $\boldsymbol{P}_\pi$ denote an operator $\boldsymbol{P}\boldsymbol{\pi}$. The Bellman operator $\boldsymbol{B}_\pi$ for a policy π is an operator $Q \mapsto r + \gamma \boldsymbol{P}_\pi Q$. Note that $\boldsymbol{P}V$, $\boldsymbol{P}_\pi Q$ and $\boldsymbol{B}_\pi Q$ are Borel-measurable.

Consider two functions Q_1 and Q_2 in $\mathcal{B}(\mathcal{X}\times\mathcal{A}, L)$ bounded by a positive real value $L \in \mathbb{R}_{++}$. It follows that $\boldsymbol{B}_\pi Q_1 - \boldsymbol{B}_\pi Q_2 = \gamma(\boldsymbol{P}_{\boldsymbol{\pi}} Q_1 - \boldsymbol{P}_{\boldsymbol{\pi}} Q_2)$. Therefore Jensen's inequality and definition of sup imply that

$$\begin{aligned}
\|\boldsymbol{B}_\pi Q_1 - \boldsymbol{B}_\pi Q_2\|_\infty &= \gamma \sup_{(x,a)\in\mathcal{X}\times\mathcal{A}} \left| \int_{\mathcal{X}\times\mathcal{A}} Q_1(y,b) - Q_2(y,b)\pi(db|y)P(dy|x,a) \right| \\
&\leq \gamma \int_{\mathcal{X}\times\mathcal{A}} \sup_{(y,b)\in\mathcal{X}\times\mathcal{A}} |Q_1(y,b) - Q_2(y,b)|\, \pi(db|y)P(dy|x,a) \\
&= \gamma \|Q_1 - Q_2\|_\infty .
\end{aligned}$$

In other words, $\boldsymbol{B}_\pi$ is a contraction mapping. Accordingly from Banach's fixed point theorem, there exists a unique fixed point q_π obtained as a limit of $Q_n := \boldsymbol{B}_\pi Q_{n-1}$, where an initial function Q_0 is an arbitrary function in $\mathcal{B}(\mathcal{X}\times\mathcal{A}, L)$. By induction, one can deduce that $Q_n = \sum_{m=0}^{n} \gamma^m (\boldsymbol{P}_\pi)^m r + \gamma^{n+1} (\boldsymbol{P}_\pi)^{n+1} Q$. As the fixed point plays an important role in RL, we formally define it.

Definition 1.3.2 (Value and Advantage Function)**.** *Suppose a policy π. A function $Q_\pi := \sum_{t=0}^{\infty} \gamma^t (\boldsymbol{P}_\pi)^t r$ is called the Q-value function for the policy π. A similar function $V_\pi := \boldsymbol{\pi}\left(\sum_{t=0}^{\infty} \gamma^t (\boldsymbol{P}_\pi)^t r\right)$ is called the state-value function. They are collectively called as value functions. The advantage function A_π is defined by $A_\pi := Q_\pi - V_\pi$.*

As the expected reward function is bounded by r_{max}, value functions are bounded by $V_{max} := r_{max}/(1-\gamma)$. Note that value and advantage functions are Borel-measurable. Indeed the Q-value function Q_π is a limit of a sequence of Borel-measurable functions, and thus, it belongs to $\mathcal{B}(\mathcal{X}\times\mathcal{A})$. The state value function V_π and advantage function A_π also clearly belong to $\mathcal{B}(\mathcal{X}\times\mathcal{A})$ too.

The definitions of value functions seem to be different from the standard ones

$$Q_\pi(x,a) := \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) \middle| X_0 = x, A_0 = a \right] \text{ and } V_\pi(x) := \int_{\mathcal{A}} Q_\pi(x,a)\pi(da|x),$$

where the superscript π of $\mathbb{E}^\pi$ indicates that π is followed, and X_t and A_t are a state and action at time t, respectively. Those definitions turn out to be equivalent to ours.

We first see the meaning of $\mathbb{E}^\pi$ above. Suppose a policy π and a probability measure ρ over $\mathcal{X}\times\mathcal{A}$. Proposition 7.28 of Bertsekas and Shreve (1996) states that there is a unique probability measure ρ_T over $(\mathcal{X}\times\mathcal{A})^{T+1}$ such that

$$\begin{aligned}
&\rho_T(\mathcal{Y}_0, \mathcal{B}_0, \ldots, \mathcal{Y}_T, \mathcal{B}_T) \\
&= \int_{\mathcal{B}_0}\int_{\mathcal{Y}_0} \cdots \int_{\mathcal{B}_T}\int_{\mathcal{Y}_T} f(x_0, a_0, \ldots, x_T, a_T) \prod_{t=1}^{T} \pi(da_t|x_t)P(dx_t|x_{t-1}, a_{t-1})\rho(d(x_0 a_0))
\end{aligned}$$

for any Borel-measurable function f and Borel-measurable subsets $\mathcal{Y}_t \in \mathfrak{B}(\mathcal{X}), \mathcal{B}_t \in \mathfrak{B}(\mathcal{A})$. ρ_T describes the probability that a short trajectory $(X_0, A_0, \ldots, X_T, A_T)$ is in the set $\mathcal{Y}_0 \times \mathcal{B}_0 \times \cdots \times \mathcal{Y}_T \times \mathcal{B}_T$. Proposition 7.28 of Bertsekas and Shreve (1996) indeed states a stronger result that T can be taken to ∞ such that $\rho_T(\mathcal{Y}_0, \mathcal{B}_0, \ldots, \mathcal{Y}_T, \mathcal{B}_T) = \rho_\infty(\mathcal{Y}_0, \mathcal{B}_0, \ldots, \mathcal{Y}_T, \mathcal{B}_T, \mathcal{X}, \mathcal{A}, \ldots)$.

Now assume that ρ is concentrated to a state-action pair (x, a). Let S_T denote expected cumulative rewards $\mathbb{E}^\pi[\sum_{t=0}^T \gamma^t r(X_t, A_t) | X_0 = x, A_0 = a]$ by time T. Then we have that

$$\begin{aligned} S_T &= \int_{\mathcal{A}\times\mathcal{X}\times\cdots\mathcal{A}\times\mathcal{X}} \sum_{t=0}^T \gamma^t r(X_t, A_t) \rho_T\left(d(x_0 a_0 \cdots x_T a_T)\right) \\ &= \int_{\mathcal{A}} \int_{\mathcal{X}} \cdots \int_{\mathcal{A}} \int_{\mathcal{X}} \sum_{t=0}^T \gamma^t r(X_t, A_t) \prod_{t=1}^T \pi(da_t|x_t) P(dx_t|x_{t-1}, a_{t-1}) \\ &= \sum_{t=0}^T \gamma^t \left[(\boldsymbol{P}_\pi)^t r\right](x, a). \end{aligned}$$

Note that $\mathbb{E}^\pi[\lim_{T\to\infty} \sum_{t=0}^T \gamma^t r(X_t, A_t) | X_0 = x, A_0 = a]$ is uniformly bounded from below and above by $S_T - \gamma^{T+1} V_{max}$ and $S_T + \gamma^{T+1} V_{max}$ for any T, respectively. Therefore by Lebesgue's dominated convergence theorem,

$$\mathbb{E}^\pi\left[\lim_{T\to\infty} \sum_{t=0}^T \gamma^t r(X_t, A_t) | X_0 = x, A_0 = a\right] = \lim_{T\to\infty} \sum_{t=0}^T \gamma^t \left[(\boldsymbol{P}_\pi)^t r\right](x, a).$$

1.4 Optimal Policy and Its Existence

An agent aims at finding a policy that leads to the maximum expected cumulative discounted rewards. Such a policy is called an optimal policy and is explained here in more details.

The max operator $\boldsymbol{m}$ is defined as an operator such that $(\boldsymbol{m}Q)(x) := \max_{a\in\mathcal{A}} Q(x, a)$ for any function Q in $\mathcal{B}(\mathcal{X} \times \mathcal{A})$. The function $\boldsymbol{m}Q$ is Borel-measurable.[4] The Bellman optimality operator $\boldsymbol{B}$ is defined as an operator $Q \mapsto r + \gamma \boldsymbol{P}\boldsymbol{m}Q$. The Bellman operator is also a contraction mapping, and thus, there exists a unique fixed point Q_*.

We now confirm that $Q_* \geq Q_\pi$ for any policy π. To this end, note that $\boldsymbol{B}f \leq \boldsymbol{B}g$ holds for any two functions f and g in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$ satisfying $f \leq g$. This property is called monotonicity. By combining the monotonicity with a fact that $\boldsymbol{B}Q_\pi \geq \boldsymbol{B}_\pi Q_\pi = Q_\pi$, we deduce that $Q_* = \lim_{i\to\infty} (\boldsymbol{B})^i Q_\pi \geq Q_\pi$.

Let π_* be a greedy policy with respect to Q_*, that is,

$$\pi_*(a|x) = \begin{cases} 1 & \text{if } a = \arg\max_{b\in\mathcal{A}} Q_*(x, a) \\ 0 & \text{otherwise} \end{cases}.$$

[4] This is because the action space is finite. However $\boldsymbol{m}Q$ may not be Borel-measurable when the action space is not finite. When the action space is continuous, we need to restrict the space of functions we consider.

(If there are several maximizers, choose one of them by a prescribed manner.) Clearly, a Bellman operator $\boldsymbol{B}_{\pi_*} : Q \mapsto r + \gamma \boldsymbol{P}\boldsymbol{\pi}_* Q$ for a policy π_* has a fixed point Q_*, which is the Q-value function of the policy π_*. Therefore the policy π_* is optimal in the sense that it leads to the maximum amount of expected cumulative rewards. We formally state the definition.

Definition 1.4.1 (Optimal Policy and Optimal Value Functions)**.** *The optimal policy π_* is defined as a policy for which $Q_{\pi_*} \geq Q_\pi$ holds for any policy π. The Q-value function of the optimal policy is called the optimal Q-value function, which is simply denoted as Q_*. The optimal state-value function V_* and advantage function A_* are similarly defined as the state value function and advantage function of the optimal policy. The aim of the agent is to find the optimal policy π_*.*

One may conjecture that it would be possible to obtain more expected cumulative rewards than those obtained by the optimal policy by using a non-stationary policy. However, it is not the case (Puterman, 1994; Bertsekas and Shreve, 1996). That is the reason why almost all RL algorithms are concerned only with stationary policies, which only depend on a current state.

1.5 Problem Description

In this section, we explain a problem setting we consider in the thesis.

In the explanation of MDPs, we stated that an agent interacts with an MDP forever. However in practice, the interaction is divided into several blocks called episodes; at the beginning of an episode, an agent is placed to $X_0 \sim \rho_0$; after several time steps H or state transition to a special state called a terminal state, the episode ends, and a new one starts.

In recent algorithms, an agent is often equipped with a buffer to which a tuple (x, a, r, y, π, d) - a state, action, reward, subsequent state, probability of the action (optional), and binary value with 1 indicating that y is a terminal state - is constantly appended, as an agent gets new data (experience). When the buffer is full, the oldest tuple at the beginning is removed, and a new one is appended to the end. It returns samples when queried. Values and/or policy updates are carried out using samples from the buffer.

Taking those situations into account, we consider a problem setting wherein an agent interacts with an environment while storing data, updating value estimate, and improving a policy as in Algorithm 1. In Section 1.6, we explain algorithms for solving this kind of problems.

1.6 Approximate Dynamic Programming

The problem, whose setting is given in Section 1.5, can be approximately solved by Approximate Dynamic Programming (ADP), an approximate version of DP. In this section, we explain three classical and basic DP as well as their ADP versions because they are highly related to our algorithms.

Algorithm 1 Problem Setting

Require: A buffer $\mathcal{D}$, horizon H, total time steps T, value update frequency $f_v \in \mathbb{Z}_{++}$, policy update frequency $f_p \in \mathbb{Z}_{++}$, and a sequence of behavior policies $(\mu_i)_{i\in\mathbb{Z}_+}$, which may be dependent on policies π_i learned through the interaction with the environment.

Initialize an initial function $Q_0 \in \mathcal{B}(\mathcal{X}\times\mathcal{A})$, policy π_0, policy update counter $i \leftarrow 0$ and episode step counter $h \leftarrow 0$.
Sample an initial state $x_0 \sim \rho_0$.
for t from 0 to T **do**
  Execute an action $a_t \sim \mu_i(\cdot|x_t)$.
  Observe a next state and reward $(y_t, r_t) \sim T(\cdot|x_t, a_t)$.
  Set d_t to 0 if y_t is not a terminal state otherwise 1.
  Discard the first data in $\mathcal{D}$ if it is full.
  Append $(x_t, a_t, y_t, r_t, \mu_j(a_t|x_t), d_t)$ to the end of the buffer $\mathcal{D}$.
  if $d_t = 1$ or $h + 1 \equiv 0 \pmod{H}$ **then**
    Reset $x_{t+1} \sim \rho_0$ and $h \leftarrow 0$.
  else
    Update $x_{t+1} \leftarrow y_t$ and $h \leftarrow h + 1$.
  end if
  if $t + 1 \equiv 0 \pmod{f_v}$ **then**
    Update $Q_i \leftarrow \mathrm{QUpdate}(Q_i, \pi_i, \mathcal{D}')$ using samples $\mathcal{D}'$ from $\mathcal{D}$ if needed.
  end if
  if $t + 1 \equiv 0 \pmod{f_p}$ **then**
    Update $\pi_{i+1} \leftarrow \mathrm{PolicyUpdate}(Q_i, \pi_i, \mathcal{D}')$ using samples $\mathcal{D}'$ from $\mathcal{D}$ if needed.
    $i \leftarrow i + 1$.
  end if
end for
return π_i.

1.6.1 Temporal Difference Learning

A crucial component of Policy Iteration (PI) and Actor-Critic (AC) is policy evaluation, i.e., computation of Q-value or advantage functions (QUpdate in Algorithm 1).

Arguably the most famous policy evaluation algorithm is online Temporal Difference Learning (λ) (TD(λ)) (Sutton and Barto, 2018), which is an online stochastic approximation of an algorithm having the following abstract DP-style update rule

$$Q_{i+1} := Q_i + (\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_\pi)^{-1} (\boldsymbol{B}_\pi Q_i - Q_i),$$

where λ is a fixed positive real value in a closed interval $[0, 1]$. It can be shown that the sequence $(Q_i)_{i\in\mathbb{Z}_+}$ uniformly converges to Q_π. While this algorithm is a DP-version of online TD(λ), we also call this algorithm TD(λ). When we refer to the online one, we call it online TD(λ).

TD(λ) is feasible only when (i) the state and action spaces are finite and sufficiently small so that functions Q_i can be expressed by a finite dimensional vector (often called a table), and (ii) $\boldsymbol{B}_{\pi_i} Q_i$ can be exactly computed.

Algorithm 2 Approximate TD(0)

Require: An initial function $Q_0 \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$, a target policy π, the number of iterations $i \in \mathbb{Z}_{++}$, series of function classes $(\mathcal{F}_j)_{j\in[1:i]}, \mathcal{F}_j \subset \mathcal{B}(\mathcal{X} \times \mathcal{A})$, sample distributions $(\mu_j)_{j\in[0:i-1]}, \mu_j \in \mathcal{P}(\mathcal{X} \times \mathcal{A})$ and numbers of samples $(N_j)_{j\in[0:i-1]}, N_j \in \mathbb{Z}_{++}$.
for j from 0 to $i-1$ **do**
Sample $(x_n, a_n, r_n, y_n)_{n\in[1:N_j]}$ such that $(x_n, a_n) \sim \mu_j, (r_n, y_n) \sim T(\cdot, \cdot|x_n, a_n)$.
$$Q_{j+1} \leftarrow \operatorname*{arg\,min}_{Q\in\mathcal{F}_{j+1}} \sum_{n=1}^{N_j} \left(r_n + \gamma \sum_{b_n\in\mathcal{A}} \pi(b_n|y_n) Q_j(y_n, b_n) - Q(x_n, a_n) \right)^2$$
end for
return Q_i.

The conditions (i) and (ii) are too demanding. Approximate TD(λ) is literally an approximation of TD(λ) and has the following abstract update rule

$$Q_{i+1} := Q_i + (\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_\pi)^{-1} (\boldsymbol{B}_\pi Q_i - Q_i) + \varepsilon_i, \tag{1.1}$$

where an error function $\varepsilon_i \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$ abstractly expresses value update errors due to function approximation and finite sample estimation. Although this algorithm is called Approximate TD(λ) (ATD(λ)), we frequently omit the qualifier "approximate" keeping in mind that we are mainly interested in approximate versions of DP algorithm.

In Algorithm 2, an example implementation of ATD(λ) with $\lambda = 0$ is shown. The algorithm is more abstract than Algorithm 1 because some objects, such as μ_j, are not specified. One can regard μ_j as approximate distribution of the buffer $\mathcal{D}$ in Algorithm 1 at j-th value update.

1.6.2 Value Iteration

The description of an optimal policy π_* in Section 1.4 suggests an algorithm with the following abstract update rule:

$$\pi_i \in \mathcal{G}(Q_i) \text{ and } Q_{i+1} := \boldsymbol{B}_{\pi_i} Q_i$$

where i starts from 0 with an initial function Q_0 in $\mathcal{B}(\mathcal{X} \times \mathcal{A})$, and $\pi_i \in \mathcal{G}(Q_i)$ means that π_i is chosen from the set $\mathcal{G}(Q_i)$ of all greedy policies with respect to Q_i. This algorithm is called Value Iteration (VI).

Approximate Value Iteration (AVI) (Bertsekas and Tsitsiklis, 1996; Munos, 2005, 2007), occasionally referred to as different names such as fitted Q-iteration, is an approximated version of VI. Its implementation is shown in Algorithm 3. Its update can be abstractly described as

$$\pi_i \in \mathcal{G}(Q_i) \text{ and } Q_{i+1} := \boldsymbol{B}_{\pi_i} Q_i + \varepsilon_i,$$

where $\varepsilon_i \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$ is an error function.

There are some theoretical results on AVI such as error propagation analysis (Bertsekas and Tsitsiklis, 1996; Munos, 2005, 2007; Farahmand, 2011; Scherrer et al., 2015)

Algorithm 3 Approximate Value Iteration (AVI)

Require: Number of iterations $i \in \mathbb{Z}_{++}$, series of function classes $(\mathcal{F}_j)_{j\in[1:i]}, \mathcal{F}_j \subset \mathcal{B}(\mathcal{X}\times\mathcal{A})$, sample distributions $(\mu_j)_{j\in[0:i-1]}, \mu_j \in \mathcal{P}(\mathcal{X}\times\mathcal{A})$ and numbers of samples $(N_j)_{j\in[0:j-1]}, N_j \in \mathbb{Z}_{++}$.
Initialize an initial function $Q_0 \in \mathcal{B}(\mathcal{X}\times\mathcal{A})$.
for j from 0 to $i-1$ **do**
  Update π_j to a greedy policy with respect to Q_j.
  Sample $(x_n, a_n, r_n, y_n)_{n\in[1:N_j]}$ such that $(x_n, a_n) \sim \mu_j, (r_n, y_n) \sim T(\cdot,\cdot|x_n, a_n)$.
  $$Q_{j+1} \leftarrow \arg\min_{Q\in\mathcal{F}_{j+1}} \sum_{n=1}^{N_j} \left(r_n + \gamma \max_{b_n\in\mathcal{A}} Q_j(y_n, b_n) - Q(x_n, a_n) \right)^2$$
end for
Update π_i to a greedy policy with respect to Q_i.
return Q_i, π_i.

and PAC-bound like performance guarantee (Farahmand, 2011; Scherrer et al., 2015), the latter of which is based on the former. The error propagation analysis, however, shows that AVI is not robust to value update errors. In the following chapters, error propagation analysis of more general algorithms (CVI) is carried out. AVI's proneness to errors are discussed in detail by comparing AVI to the general algorithms.

1.6.3 Policy Iteration

An alternative to VI is PI (Bertsekas and Tsitsiklis, 1996). Its abstract update rule is the following:

$$\pi_i \in \mathcal{G}(Q_i) \text{ and } Q_{i+1} := Q_{\pi_i},$$

where i starts from 0 with an initial function Q_0 in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$. Note that the Q-value function Q_{π_i} can be estimated by ATD(λ), but any policy evaluation algorithm, such as Gap-increasing RetrAce Policy Evaluation (GRAPE) in Chapter 3, is available.

An approximated version of PI is Approximate Policy Iteration (API) whose abstract update is the following:

$$\pi_i \in \mathcal{G}(Q_i) \text{ and } Q_{i+1} := Q_{\pi_i} + \varepsilon_i.$$

There are several variants of PI (Kakade and Langford, 2002; Scherrer, 2014; Schulman et al., 2015; Abdolmaleki et al., 2018). In the following chapters, they are discussed in detail by contrasting difference between our algorithms and them. We briefly mention that PI has the almost same error bound as that of VI, and thus, PI is prone to errors.

1.7 Error Propagation Analysis of ATD(0)

In later chapters, we carry out theoretical analysis of various algorithms using a technique called *error propagation analysis*. For example, due to ε_i in Equation (1.1), $\|Q_\pi - Q_i\|_{p,\rho}$ may not converge to zero, and it is important to analyze how $\|Q_\pi - Q_i\|_{p,\rho}$ is related to $\|\varepsilon_i\|_{q,\mu_i}$. To this end, error propagation analysis is frequently used. As it is easy for ATD(λ) with $\lambda = 0$ while capturing its some important steps, we perform error propagation analysis of ATD(λ) with $\lambda = 0$ in this section.

For error propagation analysis, we try to establish an upper bound of, for example, $L^p(\rho)$-norm of $Q_\pi - Q_i$ using $L^q(\mu_j)$-norm of ε_j. Note that different probability measures are used in those norms. Allowing different probability measures is important because data distribution μ_j and evaluation distribution ρ are in many cases different. For instance, ρ is frequently an initial state-action pair distribution, whereas μ_j is a distribution of data obtained by following a some exploratory policy. In order to correct this discrepancy, concentrability coefficients are necessary (Munos, 2005, 2007; Farahmand, 2011; Scherrer et al., 2015).

For a policy π and a probability measure ρ over $\mathcal{X} \times \mathcal{A}$, let ρP_π be a probability measure over $\mathcal{X} \times \mathcal{A}$ defined by

$$\rho P_\pi (\mathcal{Y}, \mathcal{B}) = \int_{\mathcal{X}\times\mathcal{A}} \int_{\mathcal{Y}} \pi (\mathcal{B}|x_1) P (dx_1|x_0, a_0) \rho (d(x_0 a_0)) ,$$

where $\mathcal{B}$ and $\mathcal{Y}$ are Borel-measurable subsets in $\mathfrak{B}(\mathcal{A})$ and $\mathfrak{B}(\mathcal{X})$, respectively. In words, ρP_π is an expected state-action probability measure at time $T = 1$ taking an action A_0 at an initial state X_0 sampled as $(X_0, A_0) \sim \rho$. For a sequence of policies $(\pi_t)_{t\in[0:T]}$, a probability measure $\rho P_{\pi_0} \cdots P_{\pi_T}$ over $\mathcal{X} \times \mathcal{A}$ can be recursively defined as

$$\rho P_{\pi_0} \cdots P_{\pi_T} := \left(\rho P_{\pi_0} \cdots P_{\pi_{T-1}}\right) P_{\pi_T}, \tag{1.2}$$

Concentrability coefficients are defined as follows.

Definition 1.7.1 (Concentrability Coefficients)**.** *For a sequence of policies $(\pi_t)_{t\in[0:T]}$ and probability measures ρ and μ over $\mathcal{X}\times\mathcal{A}$, let $\rho P_{\pi_0} \cdots P_{\pi_T} \in \mathcal{P}(\mathcal{X}\times\mathcal{A})$ be an expected state-action probability measure at time T defined in Equation (1.2). A concentrability coefficient $c(p, \rho, \mu; \pi_0, \ldots, \pi_T)$ is defined as*

$$c(p, \rho, \mu; \pi_0, \ldots, \pi_T) := \begin{cases} \left\| \frac{\rho P_{\pi_0} \cdots P_{\pi_T}}{\mu} \right\|_{p,\mu} & \text{if } \rho P_{\pi_0} \cdots P_{\pi_T} \prec \mu \\ \infty & \text{otherwise} \end{cases}, \tag{1.3}$$

where $p \in [1, \infty)$.

As we explained in Example 1.1.6, the Radon-Nykodim derivative is the (function of) importance sampling ratio. Accordingly the concentrability coefficient is the norm of the importance sampling ratio (viewed as a function over $\mathcal{X} \times \mathcal{A}$).

For later use, we define an expectation functional of a probability measure.

Definition 1.7.2 (Expectation Functional)**.** *For a probability measure ρ over $\mathcal{X} \times \mathcal{A}$,*

a functional $\boldsymbol{\rho} : \mathcal{B}(\mathcal{X} \times \mathcal{A}) \to \mathbb{R}$ *is defined by*

$$\boldsymbol{\rho} Q = \int_{\mathcal{X}\times\mathcal{A}} Q(x,a)\rho\left(d(xa)\right). \tag{1.4}$$

For a sequence of policies $(\pi_t)_{t\in[0:T]}$*, a functional* $\boldsymbol{\rho}\boldsymbol{P}_{\pi_T}\cdots\boldsymbol{P}_{\pi_0} : \mathcal{B}(\mathcal{X}\times\mathcal{A}) \to \mathbb{R}$ *is defined accordingly as*

$$\boldsymbol{\rho}\boldsymbol{P}_{\pi_0}\cdots\boldsymbol{P}_{\pi_T} Q = \int_{\mathcal{X}\times\mathcal{A}} Q(x_T,a_T)\rho P_{\pi_0}\cdots P_{\pi_T}\left(d(x_T a_T)\right). \tag{1.5}$$

This functional returns $\mathbb{E}[f(X_T,A_T)]$*, which is expected value of* $f(X_T,A_T)$ *when an action* A_t *at each time step* $0 \le t \le T$ *is selected according to* $\pi_t(\cdot|X_t)$*, and an initial state-action pair* (X_0,A_0) *is sampled from* ρ*.*

Note that these notations are consistent with other operator notations. For example, we have $\boldsymbol{\rho}\left(\boldsymbol{P}_{\pi_0}\cdots\boldsymbol{P}_{\pi_T}Q\right) = \left(\boldsymbol{\rho}\boldsymbol{P}_{\pi_0}\cdots\boldsymbol{P}_{\pi_T}Q\right)$.

We are now ready to start error propagation analysis of ATD(λ) with $\lambda = 0$. The first step is relating final difference $Q_\pi - Q_i$ with initial one $Q_\pi - Q_0$ as in the following lemma.

Lemma 1.7.1 (A Point-Wise Error Bound for ATD(λ))**.** *For ATD(λ) in Equation* (1.1)*, the following holds:*

$$Q_\pi - Q_i = \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^i \boldsymbol{M}^i\left(Q_\pi - Q_0\right) - \sum_{j=0}^{i-1}\left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^j \boldsymbol{M}^j\varepsilon_{i-j-1},$$

where $\boldsymbol{M}$ *is the operator* $(1-\gamma\lambda)\left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\boldsymbol{P}_\pi$*.*

Proof of Lemma 1.7.1.

$$\begin{aligned} Q_\pi - Q_i &= Q_\pi - Q_{i-1} - \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(\boldsymbol{B}_\pi Q_{i-1} - Q_{i-1}\right) - \varepsilon_{i-1} \\ &\overset{\text{(a)}}{=} Q_\pi - \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(r + \gamma(1-\lambda)\boldsymbol{P}_\pi Q_{i-1}\right) - \varepsilon_{i-1} \\ &\overset{\text{(b)}}{=} \gamma(1-\lambda)\left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\boldsymbol{P}_\pi\left(Q_\pi - Q_{i-1}\right) - \varepsilon_{i-1}, \end{aligned}$$

where the deduction accords with the following:

(a) $Q_{i-1} = \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)Q_{i-1} = \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(Q_{i-1} - \gamma\lambda\boldsymbol{P}_\pi Q_{i-1}\right)$

(b) $Q_\pi = \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)Q_\pi = \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_\pi\right)^{-1}\left(r + \gamma\boldsymbol{P}_\pi Q_\pi - \gamma\lambda\boldsymbol{P}_\pi Q_\pi\right).$

By induction, it is clear that the claim holds. □

As a corollary, an L^∞-norm error bound can be immediately obtained.

Corollary 1.7.2 (An L^∞-Norm Error Bound for ATD(λ))**.** *For ATD(λ) in Equation (1.1), the following holds:*

$$\|Q_\pi - Q_i\|_\infty \leq \sum_{j=0}^{i-1} \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^j \|\varepsilon_{i-j-1}\|_\infty + \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^i \|Q_\pi - Q_0\|_\infty . \tag{1.6}$$

Proof of Corollary 1.7.2. The operator $\boldsymbol{M}$ is $(1-\gamma\lambda)\sum_{t=0}^\infty (\gamma\lambda)^t (\boldsymbol{P}_\pi)^{t+1}$. Therefore, it is clearly monotone. In other words, $\boldsymbol{M}f \leq \boldsymbol{M}g$ holds for any functions f and g in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$ satisfying $f \leq g$. Furthermore, $\boldsymbol{M}f = f$ holds for any constant function f. Accordingly $-\|\varepsilon_{i-j-1}\|_\infty \leq \boldsymbol{M}^j\varepsilon_{i-j-1} \leq \|\varepsilon_{i-j-1}\|_\infty$, i.e., $\left\|\boldsymbol{M}^j\varepsilon_{i-j-1}\right\|_\infty \leq \|\varepsilon_{i-j-1}\|_\infty$ holds. Combination of this fact with triangle inequality leads to

$$\begin{aligned}
\|Q_\pi - Q_i\|_\infty &\leq \sum_{j=0}^{i-1} \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^j \left\|\boldsymbol{M}^j\varepsilon_{i-j-1}\right\|_\infty + \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^i \left\|\boldsymbol{M}^i\left(Q_\pi - Q_0\right)\right\|_\infty \\
&\leq \sum_{j=0}^{i-1} \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^j \|\varepsilon_{i-j-1}\|_\infty + \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^i \|Q_\pi - Q_0\|_\infty .
\end{aligned}$$

This concludes the proof. □

This corollary shows two things: the convergence rate and how the effect of past errors decays. When there are no errors, the right hand side of the inequality Equation (1.6) reduces to $[\gamma(1-\lambda)/(1-\gamma\lambda)]^i \|Q_\pi - Q_0\|_\infty$. Therefore the decay rate of the initial error $\|Q_\pi - Q_0\|_\infty$ is given by $[\gamma(1-\lambda)/(1-\gamma\lambda)]^i$. When there are errors, we can rewrite the first term of the right hand side as follows:

$$\begin{aligned}
&\sum_{j=0}^{i-1} \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^j \|\varepsilon_{i-j-1}\|_\infty \\
&= \|\varepsilon_{i-1}\|_\infty + \frac{\gamma(1-\lambda)}{1-\gamma\lambda}\|\varepsilon_{i-2}\|_\infty + \left(\frac{\gamma(1-\lambda)}{1-\gamma\lambda}\right)^2 \|\varepsilon_{i-3}\|_\infty + \cdots
\end{aligned}$$

This shows that the effect of past errors decays with the rate $[\gamma(1-\lambda)/(1-\gamma\lambda)]^j$.

Sometimes error propagation analysis may result in a loose bound. As shown in the following proposition, the error bound (1.6) is not improvable, and thus, it is tight.

Proposition 1.7.3 (Tightness of the L^∞-Norm Error Bound (1.6) for ATD(λ))**.** *The L^∞-Norm Error Bound (1.6) for ATD(λ) is tight meaning that there exists an MDP and a sequence $(\varepsilon_i)_{i\in\mathbb{Z}_+}$ of error functions such that the error bound holds with equality.*

Proof. Consider an MDP in which a reward function r takes a constant value 1. Assume that an initial function $Q_0(x,a)$ takes a constant value $-1/(1-\gamma)$, and ε_j is $-r$ for

any $j \in \mathbb{Z}_+$. Then we have that $Q_i(x,a) = -\kappa^i/(1-\gamma)$ where $\kappa := \frac{\gamma(1-\lambda)}{1-\gamma\lambda}$. Thus

$$\|Q_\pi - Q_i\|_\infty = \frac{1+\kappa^i}{1-\gamma},\ \|Q_\pi - Q_0\|_\infty = \frac{2}{1-\gamma} \text{ and } \sum_{j=0}^{i-1} \kappa^j \|\varepsilon_{i-j-1}\|_\infty = \frac{1-\kappa^i}{1-\gamma}$$
$$\implies \kappa^i \|Q_\pi - Q_0\|_\infty + \sum_{j=0}^{i-1} \kappa^j \|\varepsilon_{i-j-1}\|_\infty = \frac{1+\kappa^i}{1-\gamma} = \|Q_\pi - Q_0\|_\infty .$$

This concludes the proof. □

Remark 1.7.1. *The keys of Proposition 1.7.3's proof are that (i) $r(x,a)$ takes a constant value 1, and that (ii) ε_j is $-r$. In such a case, an agent virtually perceives no reward. Therefore a sequence of functions $(Q_i)_{i\in\mathbb{Z}_+}$, which consists of an estimate of Q_π, must converge to a constant function taking 0. On the other, the true Q-value function Q_π takes a constant value $1/(1-\gamma)$. Any decent policy evaluation algorithm must show the almost same behavior that $\lim_{i\to\infty} \|Q_\pi - Q_i\|_\infty = 1/(1-\gamma)$, without any assumption on ε_j.*

Once a point-wise error bound as in Lemma 1.7.1 is obtained, it is tedious but straightforward to obtain $L^p(\rho)$-norm error bounds, as we will do in the proof of the following theorem.

Theorem 1.7.4 ($L^p(\rho)$-Norm Error Bounds for ATD(λ))**.** *Suppose probability measures ρ and μ_j over $\mathcal{X}\times\mathcal{A}$, where $j \in [0\,{:}\,i]$. For ATD(λ) in Equation (1.1) with $\lambda = 0$, the following holds:*

$$\|Q_\pi - Q_i\|_{p,\rho} \le \sum_{j=0}^{i-1} \gamma^j c_j^{\frac{1}{p}} \|\varepsilon_{i-j-1}\|_{q,\mu_{i-j-1}} + \gamma^i c_i^{\frac{1}{p}} \|Q_\pi - Q_0\|_{q,\mu_i} ,$$

where p and q are positive real values in $[0,\infty)$ and $q \in (p,\infty)$, respectively, and

$$c_j := c\left(\frac{q}{q-p}, \rho, \mu_{i-j-1}; \overbrace{\pi,\dots,\pi}^{j}\right), \text{ and } c_i := c\left(\frac{q}{q-p}, \rho, \mu_i; \overbrace{\pi,\dots,\pi}^{i}\right)$$

are concentrability coefficients.

Proof of Theorem 1.7.4. For a function Q in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$, let $|Q|$ denote a function $|Q|(x,a) := |Q(x,a)|$. In addition, let Q^p denote a function $Q^p(x,a) = Q(x,a)^p$.

By triangle inequality, $|Q_\pi - Q_i| \le \sum_{j=0}^{i-1} \gamma^j \left|\boldsymbol{P}_\pi^j \varepsilon_{i-j-1}\right| + \gamma^i \left|\boldsymbol{P}_\pi^i (Q_\pi - Q_0)\right|$. Furthermore, for a function Q in $\mathcal{B}(\mathcal{X}\times\mathcal{A})$, we have that $\left|\boldsymbol{P}_\pi^j Q\right| \le \boldsymbol{P}_\pi^j |Q|$ from Jensen's inequality. Thus

$$|Q_\pi - Q_i| \le \sum_{j=0}^{i-1} \gamma^j \boldsymbol{P}_\pi^j |\varepsilon_{i-j-1}| + \gamma^i \boldsymbol{P}_\pi^i |(Q_\pi - Q_0)| .$$

Taking p-th power of both sides,

$$\begin{aligned}
|Q_\pi - Q_i|^p &\leq \left(\sum_{j=0}^{i-1} \gamma^j \boldsymbol{P}_\pi^j \left|\varepsilon_{i-j-1}\right| + \gamma^i \boldsymbol{P}_\pi^i \left|Q_\pi - Q_0\right| \right)^p \\
&= \Lambda^p \left(\sum_{j=0}^{i-1} \frac{\gamma^j \lambda_j}{\Lambda} \boldsymbol{P}_\pi^j \left| \frac{1}{\lambda_j} \varepsilon_{i-j-1} \right| + \frac{\lambda_i \gamma^i}{\Lambda} \boldsymbol{P}_\pi^i \left| \frac{1}{\lambda_i} \left(Q_\pi - Q_0\right) \right| \right)^p \\
&\leq \Lambda^{p-1} \left(\sum_{j=0}^{i-1} \gamma^j \lambda_j^{1-p} \boldsymbol{P}_\pi^j \left|\varepsilon_{i-j-1}\right|^p + \lambda_i^{1-p} \gamma^i \boldsymbol{P}_\pi^i \left|Q_\pi - Q_0\right|^p \right),
\end{aligned}$$

where the last line follows from Jensen's inequality[5], and $(\lambda_j)_{j\in[0:i]}$ is a sequence of currently unspecified positive real values such that $\sum_{j=0}^{i} \gamma^j \lambda_j = \Lambda$.

By using the expectation functional (1.4), we deduce that

$$\boldsymbol{\rho} \left|Q_\pi - Q_i\right|^p \leq \Lambda^{p-1} \left(\sum_{j=0}^{i-1} \gamma^j \lambda_j^{1-p} \boldsymbol{\rho} \boldsymbol{P}_\pi^j \left|\varepsilon_{i-j-1}\right|^p + \lambda_i^{1-p} \gamma^i \boldsymbol{\rho} \boldsymbol{P}_\pi^i \left|Q_\pi - Q_0\right|^p \right).$$

When a probability measure $\rho P_\pi^j := \rho \overbrace{P_\pi \cdots P_\pi}^{j}$ is absolutely continuous with respect to μ_{i-j-1}, its Radon-Nykodim derivative $\rho P_\pi^j / \mu_{i-j-1}$ exists, and we have

$$\begin{aligned}
\boldsymbol{\rho} \boldsymbol{P}_\pi^j \left|\varepsilon_{i-j-1}\right|^p &= \int_{\mathcal{X}\times\mathcal{A}} \frac{\rho P_\pi^j}{\mu_{i-j-1}}(x,a) \left|\varepsilon_{i-j-1}(x,a)\right|^p \mu_{i-j-1}(d(xa)) \\
&\leq \left\| \frac{\rho P_\pi^j}{\mu_{i-j-1}} \right\|_{\frac{q}{q-p},\mu_{i-j-1}} \left\|\varepsilon_{i-j-1}\right\|_{q,\mu_{i-j-1}}^p,
\end{aligned}$$

where Hölder's inequality is used. Even if ρP_π^j is not absolutely continuous with respect to μ_{i-j-1},

$$\boldsymbol{\rho} \boldsymbol{P}_\pi^j \left|\varepsilon_{i-j-1}\right|^p \leq c^\rho_{\frac{q}{q-p},\mu_{i-j-1}}(\pi; j) \left\|\varepsilon_{i-j-1}\right\|_{q,\mu_{i-j-1}}^p,$$

holds by Definition 1.7.1 of concentrability coefficients. As a result,

$$\begin{aligned}
\boldsymbol{\rho} \left|Q_\pi - Q_i\right|^p \leq \Lambda^{p-1} \sum_{j=0}^{i-1} \gamma^j \lambda_j^{1-p} c^\rho_{\frac{q}{q-p},\mu_{i-j-1}}(\pi; j) \left\|\varepsilon_{i-j-1}\right\|_{q,\mu_{i-j-1}}^p \\
+ \Lambda^{p-1} \lambda_i^{1-p} \gamma^i c^\rho_{\frac{q}{q-p},\mu_i}(\pi; i) \left\|Q_\pi - Q_0\right\|_{q,\mu_i}^p.
\end{aligned}$$

Finally, by setting λ_j to $c^\rho_{\frac{q}{q-p},\mu_{i-j-1}}(\pi; j)^{\frac{1}{p}} \left\|\varepsilon_{i-j-1}\right\|_{q,\mu_{i-j-1}}$ for $0 \leq j \leq i-1$, and λ_i to $c^\rho_{\frac{q}{q-p},\mu_i}(\pi; i)^{\frac{1}{p}} \left\|Q_\pi - Q_0\right\|_{q,\mu_i}$ for $j = i$, we deduce that $\boldsymbol{\rho} \left|Q_\pi - Q_i\right|^p \leq \Lambda^{p-1} \sum_{j=0}^{i} \gamma^j \lambda_j$.

[5] used firstly to exchange the order of the summation and p-th power, and secondly to exchange the order of expectation by $\boldsymbol{P}_\pi$ and p-th power

In other words,

$$\begin{aligned}&\|Q_\pi - Q_i\|_{p,\rho} \\ &\leq \sum_{j=0}^{i-1} \gamma^j c^\rho_{\frac{q}{q-p},\mu_{i-j-1}}(\pi; j)^{\frac{1}{p}} \left\|\varepsilon_{i-j-1}\right\|_{q,\mu_{i-j-1}} + \gamma^i c^\rho_{\frac{q}{q-p},\mu_i}(\pi; i)^{\frac{1}{p}} \left\|Q_\pi - Q_0\right\|_{q,\mu_i}.\end{aligned}$$

This concludes the proof. □

We again emphasize that the proof of Theorem 1.7.4 shows that obtaining $L^p(\rho)$-norm error bounds are relatively straightforward when one has a point-wise error bound as in Lemma 1.7.1. Indeed techniques used in error propagation analysis of other algorithms are very similar to those used in the proof of Theorem 1.7.4.

Chapter 2

Error-Tolerant Control via Entropy Regularized Value Iteration

In this chapter, we introduce and analyze a class of RL algorithms wherein policy updates are regularized by the entropy and Kullback–Leibler (KL) divergence (also known as the relative entropy, and thus, we call regularization using both of them as entropy regularization). This chapter is based on our work (Kozuno et al., 2019) but slightly extends it.

An idea to regularize policy updates of VI and PI with either KL divergence (Azar et al., 2012; Rawlik, 2013; Abdolmaleki et al., 2018) or entropy (Fox et al., 2016; Haarnoja et al., 2017) can be found in many research articles. In this chapter, we consider using both regularizes as in our paper (Kozuno et al., 2019). Interestingly the use of both entropy regularizers yield different algorithms depending on which function to store. For example, storing action preference leads to a softened version of Advantage Learning (AL) by Baird III (1999) and Bellemare et al. (2016).

Unfortunately, however, the current theoretical understanding on effects of the entropy regularizers is limited. Very recently, Geist et al. (2019) have provided error propagation analysis of algorithms with policy update regularization that includes KL divergence as a special case. However their analysis captures no benefit of policy update regularization on the contrary to works by Azar et al. (2012) and Kozuno et al. (2019). We show that the entropy regularizers do have benefits.

Our analysis in this chapter is an important step towards understanding algorithms with policy update regularization using the entropy regularizers. The following lists our contribution in this chapter:

1. (Theorem 2.2.1) Novel performance bounds for the previous algorithms (soft Q-learning and AL).

2. (Theorem 2.2.1) Algorithms with a gap-increasing operator are noise-tolerant. α controls the trade-off between noise-tolerance and convergence rate.

3. (Theorem 2.2.3) Algorithms with a hard gap-increasing operator have almost the same error dependency as does AVI.

4. (Theorem 2.2.6) Algorithms with a softmax operator are error-tolerant, but

asymptotic performance may be poor. β controls the quality of asymptotic performance.

5. (Theorem 2.2.6) Algorithms with a soft gap-increasing operator enjoy both noise-tolerance and error-tolerance, while avoiding poor asymptotic performance.

Here, error-tolerance refers to the tolerance of algorithms to errors such as function approximation error, whereas noise-tolerance refers to the tolerance of algorithms to stochastic errors that may cancel when averaged as in Equation (3.4).

2.1 Conservative Value Iteration

In this section, we derive an algorithm that we call Conservative Value Iteration (CVI). It is characterized by the use of two types of regularization, KL divergence and entropy. Interestingly it is possible to implement CVI in two ways: one directly stores Q-value augmented with KL divergence penalty and entropy bonus, while the other stores action-preferences. The latter one has a close connection to gap-increasing algorithms such as Dynamic Policy Programming (DPP) (Azar et al., 2012) and AL (Baird III, 1999; Bellemare et al., 2016).

Let u denote a uniform distribution $u(a) := 1/|\mathcal{A}|$ over $\mathcal{A}$. Consider an algorithm with the following update:

$$\pi_i(\cdot|x) := \arg\max_{\pi} V^{\pi}_{\pi_{i-1}}(x) \tag{2.1}$$

$$Q_{i+1}(x,a) := r(x,a) + \gamma \mathbb{E}\left[V^{\pi_i}_{\pi_{i-1}}(X_1) \middle| X_0 = x, A_0 = a \right], \tag{2.2}$$

where an initial function Q_0 is a constant function taking 0, an initial baseline policy $\pi_{i-1}(\cdot|x)$ is assumed to be a uniform distribution u for each state x, and $V^{\pi}_{\pi_{i-1}}$ is a function over states and defined as

$$V^{\pi}_{\pi_{i-1}}(x) := \sum_{a\in\mathcal{A}} \pi(a) Q_i(x,a) - \sigma D_{KL}\left(\pi \| u\right) - \tau D_{KL}\left(\pi \| \pi_{i-1}(\cdot|x)\right)$$

with $D_{KL}\left(p\|q\right)$ denoting the KL divergence $\sum_{a\in\mathcal{A}} p(a) \ln\left(p(a)/q(a)\right)$. The algorithm can be understood as VI with a policy update regularizer $-\sigma D_{KL}\left(\pi\|u\right)$ (KL divergence regularizer) and $-\tau D_{KL}\left(\pi\|\pi_{i-1}(\cdot|x)\right)$ (entropy regularizer), both of which augment a state value $\sum_{a\in\mathcal{A}} \pi(a) Q_i(X_1,a)$ at the next state.

It turns out that the update rules (2.1) and (2.2) can be rewritten as more explicit update rules. For notational simplicity, let us rewrite $\pi_i(\cdot|x)$, $\pi_{i-1}(\cdot|x)$ and $Q_i(x,a)$ as p, q and $Q(a)$, respectively. Then to get explicit expression of π_i and $V^{\pi_i}_{\pi_{i-1}}$ in updates (2.1) and (2.2), we need to solve the following optimization problem:

$$\max_p \sum_{a\in\mathcal{A}} p(a)Q(a) - \sigma D_{KL}\left(p\|u\right) - \tau D_{KL}\left(p\|q\right) \text{ subject to } \sum_{a\in\mathcal{A}} p(a) = 1.$$

(Other inequality constraints will turn out to be unnecessary.) The standard argument

of convex optimization tells us that the following must be satisfied at the solution p°:

$$0 = Q(a) - \sigma\left(\ln\frac{p^\circ(a)}{u(a)} + 1\right) - \tau \ln p^\circ(a) + \lambda.$$

Therefore we deduce that

$$p^\circ(a) = \frac{q(a)^\alpha \exp\left(\beta Q(a)\right)}{\sum_{b\in\mathcal{A}} q(b)^\alpha \exp\left(\beta Q(b)\right)},$$

where we defined two positive real values $\alpha := \tau/(\sigma+\tau)$ and $\beta := 1/(\sigma+\tau)$.

Plugging p° into the original objective, we deduce that

$$\sum_{a\in\mathcal{A}} p^\circ(a)Q(a) - \sigma D_{KL}\left(p^\circ \| u\right) - \tau D_{KL}\left(p^\circ\|q\right) = \frac{1}{\beta}\ln\sum_{a\in\mathcal{A}}\frac{q(a)^\alpha\exp\left(\beta Q(a)\right)}{|\mathcal{A}|^{1-\alpha}}.$$

Thus the algorithm (2.1) and (2.2) has the following simpler update:

$$\pi_i(a|x) := \frac{\pi_{i-1}(a|x)^\alpha \exp\left(\beta Q_i(x,a)\right)}{\sum_{b\in\mathcal{A}}\pi_{i-1}(b|x)^\alpha\exp\left(\beta Q_i(x,b)\right)} \tag{2.3}$$

$$Q_{i+1} := \boldsymbol{B}^{\alpha,\beta}_{\pi_{i-1}} Q_i := r + \gamma \boldsymbol{P}\boldsymbol{m}^{\alpha,\beta}_{\pi_{i-1}} Q_i, \tag{2.4}$$

where we defined an operator $\boldsymbol{m}^{\alpha,\beta}_\pi : \mathcal{B}(\mathcal{X}\times\mathcal{A}) \to \mathcal{B}(\mathcal{X})$ such that

$$\boldsymbol{m}^{\alpha,\beta}_\pi Q(x) := \frac{1}{\beta}\ln\sum_{a\in\mathcal{A}}\frac{\pi(a|x)^\alpha\exp\left(\beta Q(x,a)\right)}{|\mathcal{A}|^{1-\alpha}}$$

We recall that the initial function Q_0 is assumed to be a constant function taking 0, and an initial baseline policy $\pi_{i-1}(\cdot|x)$ is assumed to be a uniform distribution u for each state x. We call this algorithm CVI-Q.

From this derivation, only finite β is allowed. However we allow infinite β. In this case, policy updates are greedy policy updates.

Interestingly another implementation of updates (2.1) and (2.2) is possible. Let $\Psi_i(x,a)$ be $Q_i(x,a) + \alpha\beta^{-1}\ln\pi_{i-1}(a|x) + \alpha\beta^{-1}\ln|\mathcal{A}|$. Then the policy π_i at the i-th iteration satisfies $\pi_i(a|x) \propto \exp\left(\beta\Psi_i(x,a)\right)$. Furthermore we have that

$$\boldsymbol{m}^{\alpha,\beta}_{\pi_{i-1}} Q_i(x) = \frac{1}{\beta}\ln\sum_{a\in\mathcal{A}}\frac{\exp\left(\beta\left[Q_i(x,a) + \frac{\alpha}{\beta}\ln\pi_{i-1}(a|x)\right]\right)}{|\mathcal{A}|^{1-\alpha}} := \boldsymbol{m}^\beta\Psi_i(x),$$

where $\boldsymbol{m}^\beta : \mathcal{B}(\mathcal{X}\times\mathcal{A}) \to \mathcal{B}(\mathcal{X})$ is an operator defined as $\boldsymbol{m}^\beta := \boldsymbol{m}^{0,\beta}_u$, which is known as the mellowmax operator (Asadi and Littman, 2017). Accordingly we deduce that

$$\begin{aligned}\Psi_{i+1}(x,a) &= Q_{i+1}(x,a) + \frac{\alpha}{\beta}\ln\pi_i(a|x) + const.\\ &= r(x,a) + \gamma\boldsymbol{P}\boldsymbol{m}^\beta\Psi_i(x,a) + \alpha\left(\Psi_i(x,a) - \boldsymbol{m}^\beta\Psi_i(x)\right) + const.\end{aligned}$$

As the constant has no effect on the policy π_i at the i-th iteration, CVI-Q can be equivalently implemented by

$$\pi_i(a|x) := \frac{\exp\left(\beta\Psi_i(x,a)\right)}{\sum_{b\in\mathcal{A}}\exp\left(\beta\Psi_i(x,b)\right)} \tag{2.5}$$

$$\Psi_{i+1} := r + \gamma \boldsymbol{P}\boldsymbol{m}^{\beta}\Psi_i + \alpha\left(\Psi_i - \boldsymbol{m}^{\beta}\Psi_i\right), \tag{2.6}$$

We call this algorithm CVI-Ψ.

2.1.1 Approximate Versions of CVI

We are interested in performance of CVI under value update errors. To theoretically analyze it, we introduce approximate versions of CVI.

An approximate version of CVI-Q is given by

$$\pi_i(a|x) := \frac{\pi_{i-1}(a|x)^{\alpha}\exp\left(\beta Q_i(x,a)\right)}{\sum_{b\in\mathcal{A}}\pi_{i-1}(b|x)^{\alpha}\exp\left(\beta Q_i(x,b)\right)} \tag{2.7}$$

$$Q_{i+1} := \boldsymbol{B}^{\alpha,\beta}_{\pi_{i-1}}Q_i + \varepsilon_i := r + \gamma\boldsymbol{P}\boldsymbol{m}^{\alpha,\beta}_{\pi_{i-1}}Q_i + \varepsilon_i, \tag{2.8}$$

where $\varepsilon_i \in \mathcal{B}(\mathcal{X}\times\mathcal{A})$ is a value update error function at i-th iteration.

An approximate version of CVI-Ψ is given by

$$\pi_i(a|x) := \frac{\exp\left(\beta\Psi_i(x,a)\right)}{\sum_{b\in\mathcal{A}}\exp\left(\beta\Psi_i(x,b)\right)} \tag{2.9}$$

$$\Psi_{i+1} := \boldsymbol{B}^{\beta}\Psi_i + \alpha\left(\Psi_i - \boldsymbol{m}^{\beta}\Psi_i\right) + \varepsilon_i := r + \gamma\boldsymbol{P}\boldsymbol{m}^{\beta}\Psi_i + \alpha\left(\Psi_i - \boldsymbol{m}^{\beta}\Psi_i\right) + \varepsilon_i. \tag{2.10}$$

Recall that ε_i appearing in value updates (2.8) and (2.10) may be completely different; ε_i is used in both updates just for notational simplicity.

2.1.2 Equivalence of ACVI-Q and Ψ

Although Approximate Conservative Value Iteration (ACVI)-Q and Ψ are seemingly different, they are equivalent as we now explain. For clarity, let ε_i^Q and ε_i^{Ψ} denote ε_i in (2.8) and (2.10), respectively. Furthermore let π_i^Q and π_i^{Ψ} denote π_i in (2.7) and (2.9), respectively. The following lemma tells us the equivalence of ACVI-Q and Ψ when $\varepsilon_i^Q = \varepsilon_i^{\Psi}$.

Lemma 2.1.1 (The Equivalence of ACVI-Q and Ψ)**.** *Suppose that $\varepsilon_i^Q = \varepsilon_i^{\Psi} := \varepsilon_i$ holds for all i. Then $\pi_i^Q = \pi_i^{\Psi} := \pi_i$ holds for all i.*

Proof. Recall that Q_0 and Ψ_0 are assumed to be constant functions taking 0. Furthermore $\pi_0^Q(\cdot|x)$, $\pi_0^{\Psi}(\cdot|x)$ and $\pi_{-1}^Q(\cdot|x)$ are assumed to be a uniform distribution over $|\mathcal{A}|$ at each state x.

We are going to prove that $Q_i(x,a) + \alpha\beta^{-1}\ln\pi_{i-1}(a|x) = \Psi_i(x,a) + const.$ and $\pi_i^Q = \pi_i^{\Psi}$ for all i. It holds for $i = 0$ by definition. Suppose that it holds for all

$j \in [0:i]$. Then

$$Q_{i+1} = \boldsymbol{B}^{\alpha,\beta}_{\pi_{i-1}} Q_i + \varepsilon_i = r + \gamma \boldsymbol{P}\boldsymbol{m}^{\alpha,\beta}_{\pi_{i-1}} Q_i + \varepsilon_i.$$

From

$$\boldsymbol{m}^{\alpha,\beta}_{\pi_{i-1}} Q_i(x) = \frac{1}{\beta} \ln \sum_{a \in \mathcal{A}} \frac{\exp\left(\beta \left[Q_i(x,a) + \frac{\alpha}{\beta} \ln \pi_{i-1}(a|x)\right]\right)}{|\mathcal{A}|^{1-\alpha}} = \boldsymbol{m}^{\beta} \Psi_i(x) + const.,$$

we deduce that $Q_{i+1} = r + \gamma \boldsymbol{P}\boldsymbol{m}^{\beta}\Psi_i + \varepsilon_i + const.$ As $\beta^{-1} \ln \pi_i(a|x) = \Psi_i(x,a) - \boldsymbol{m}^{\beta}\Psi_i(x) + const.$, we deduce that $Q_{i+1}(x,a) + \alpha\beta^{-1} \ln \pi_i(a|x) = \Psi_{i+1}(x,a) + const.$ This concludes the proof. □

Thanks to this lemma, error propagation analysis of ACVI-Q can be done by just replacing ε_i^{Ψ} appearing in performance bound of ACVI-Ψ with ε_i^{Q}.

2.2 Error Propagation Analysis of ACVI

In this section, we perform error propagation analysis of ACVI-Q and Ψ. For readability, we defer all proofs to Section 2.5.

We begin with some definitions. We define a sequence of policies $(\boldsymbol{\mu}_i)_{i \in \mathbb{Z}_+}$ such that

$$\boldsymbol{\mu}_i \Psi_i := \boldsymbol{m}^{\beta} \Psi_i. \tag{2.11}$$

Such policies always exist (Asadi and Littman, 2017). We also define

$$E_i := \sum_{j=0}^{i} \alpha^j \varepsilon_{i-j}$$

for all i. Furthermore we use a shorthand notation

$$\alpha_{j:i} := \begin{cases} \sum_{k=j}^{i} \alpha^k & \text{if } i \geq j \\ 0 & otherwise \end{cases}$$

for two integers $i, j \in \mathbb{Z}$.

2.2.1 Regularization Agnostic Performance Bound

In this Section 2.2.1, we derive performance bounds for ACVI-Q and Ψ that are not fully capturing effects of policy update regularization. Indeed the performance bounds here imply that ACVI with $\beta = \infty$ would work best. (In other words, they are agnostic of effects of policy update regularization.) However Azar et al. (2012), Fox et al. (2016) and Haarnoja et al. (2017) have noted that finite β actually works best. Nonetheless they provide rates of convergence and allow us to see noise-tolerance of ACVI.

We have the following theorem that provides L^{∞}-norm performance bound.

Theorem 2.2.1 (Regularization Agnostic L^∞-Norm Performance Bound for ACVI-Q and Ψ)**.** *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. The following point-wise upper bound for $Q_* - Q_{\pi_i}$ holds for any non-negative integer i:*

$$\begin{aligned}&\|Q_* - Q_{\pi_i}\|_\infty \\ &\leq \frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\|E_{i-j-1}\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}|. \end{aligned} \tag{2.12}$$

where $\sum_{j=1}^{0} Q_j$ is a constant function taking 0 for any sequence of functions Q_j. Furthermore the same bound holds for $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.7).

Instead of L^∞-norm, $L^p(\rho)$-norm performance bound is possible. For readability, we provide definition of concentrability coefficients (Definition 1.7.1) here again. Suppose a sequence of policies $(\pi_t)_{t\in[0:T]}$ and probability measures $\rho, \mu \in \mathcal{P}(\mathcal{X}\times\mathcal{A})$. Let $\rho P_{\pi_0}\cdots P_{\pi_T} \in \mathcal{P}(\mathcal{X}\times\mathcal{A})$ be an expected state-action probability measure at time T (cf. Equation (1.2)). A concentrability coefficient $c(p,\rho,\nu;\pi_0,\ldots,\pi_T)$ is defined as

$$c(p,\rho,\nu;\pi_0,\ldots,\pi_T) := \begin{cases} \left\|\frac{\rho P_{\pi_0}\cdots P_{\pi_T}}{\nu}\right\|_{p,\nu} & \text{if } \rho P_{\pi_0}\cdots P_{\pi_T} \prec \nu \\ \infty & \text{otherwise} \end{cases}, \tag{2.13}$$

where $p \in [1,\infty)$. We define short-hand notations for the following concentrability coefficients:

$$\begin{aligned} c^*_j &:= c(2,\rho,\nu;\overbrace{\pi_*,\ldots,\pi_*}^{j}), \\ c_{j,k} &:= c(2,\rho,\nu;\overbrace{\rho_i,\ldots,\rho_i}^{j},\rho_i,\rho_{i-1},\ldots,\rho_k). \end{aligned}$$

Theorem 2.2.2 (Regularization Agnostic $L^p(\rho)$-Norm Performance Bound for ACVI-Q and Ψ)**.** *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. The following $L^p(\rho)$-norm performance bound for ACVI-Q and Ψ hold for any non-negative integer i:*

$$\|Q_* - Q_{\pi_i}\|_{\rho,p} \leq \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}| + \frac{2\gamma}{1-\gamma}\mathcal{E}_{\nu,2p,i}, \tag{2.14}$$

where

$$\begin{aligned} C_j &:= \frac{1-\gamma}{2}\left(\left(c^*_{j+1}\right)^{1/p} + \sum_{k=0}^{\infty}\gamma^k\left(c^{1/p}_{k,j+1} + \gamma c^{1/p}_{k,j}\right)\right), \\ \mathcal{E}_{\nu,2p,i} &:= \sup_{\pi_i,\ldots,\pi_0}\sum_{j=0}^{i-1}\gamma^j C_j\left\|\frac{E_{i-j-1}}{\alpha_{0:i}}\right\|_{\nu,2p}. \end{aligned}$$

Remark 2.2.1. *Setting $\alpha = 1$ yields a performance bound for DPP. We note two differences from a known performance bound for DPP by Azar et al. (2012).*

First, the convergence rate

$$\frac{2\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} = \frac{2\gamma V_{max}(1-\gamma^{i+1})}{(1-\gamma)(i+1)}$$

is $1-\gamma$ times smaller than the corresponding term in Azar et al. (2012)'s performance bound (Theorem 5), thanks to a new proof technique.[1] *As γ is typically close to 1, this improvement is not negligible.*

Second, our bound is an $L^p(\rho)$-norm performance bound. In general, $L^p(\rho)$-norm performance bound is considered to be tighter (Farahmand, 2011). Indeed, with a slight complication of the argument using Lebesgue's decomposition theorem (Dudley, 2002), the $L^p(\rho)$-norm performance bound (2.14) can be tighten such that it never exceeds L^∞-norm.

The L^p-norm performance bound (2.14) allows us to understand several properties of ACVI-Q and Ψ controlled by α.

Firstly the convergence rates of ACVI-Q and Ψ

$$\frac{2\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} = O\left(\frac{1}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j}\right) \tag{2.15}$$

are controlled by α. Note that it becomes $O(\gamma^i)$ when $\alpha = 0$, which is the convergence rate of AVI (Munos, 2005, 2007; Farahmand, 2011; Scherrer et al., 2015). Figure 2.2 visualizes the convergence rates of ACVI-Q and Ψ. As is seen, relatively high α such as $\gamma \approx 0.95$ does not slow the convergence. However α almost equal to 1 noticeably slows it. Figure 2.2 visualizes the number of iterations i at which the convergence rates becomes smaller than 0.1. It again shows that the convergence rates of ACVI-Q and Ψ are the almost same as that of AVI when α is less than 0.95. However they drastically slows down when α is higher than 0.95.

Importantly the L^p-norm performance bound (2.14) of ACVI-Q and Ψ shows that *a higher α leads to a greater noise-tolerance.* It states that

$$\left\| \frac{1}{\alpha_{0:i}} E_{i-j-1} \right\|_{\nu,2p} = \left\| \frac{1}{\alpha_{0:i}} \sum_{k=0}^{i-j-1} \alpha^k \varepsilon_{i-j-k-1} \right\|_{\nu,2p}$$

essentially determines the loss $\|Q_* - Q_{\pi_i}\|_{\rho,p}$. Now suppose for simplicity that $\varepsilon_j(x,a)$ is sampled independently from a distribution with a mean of 0 and a standard deviation of 1 for any j, state x and action a. Then a standard deviation of $\alpha_{0:i}^{-1} E_{i-j-1}$ is given by $\alpha_{0:i}^{-1}\sqrt{1+\alpha^2+\cdots+\alpha^{2(i-j)}}$. When $\alpha = 0.9$, it converges to approximately 0.23, which is four times smaller than 1. Although $\varepsilon_j(s,a)$ is unlikely to satisfy the assumptions in reality, a similar result is expected in a model-free setting where errors contain noise

[1] Their bound contains a mistake: $\|E_j\|_\infty$ in their bound must be multiplied by two.

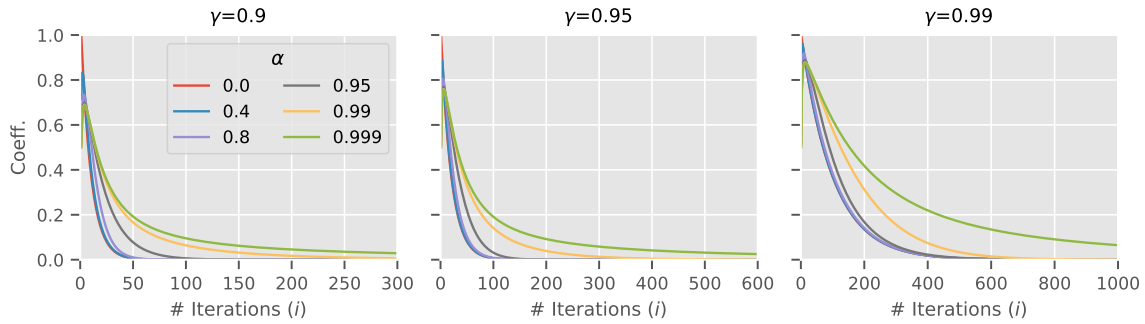


Figure 2.1: Convergence rates comparison of ACVI-Q and Ψ. The discount factor γ is indicated on top of each panel. α is indicated by the line colors as shown in the legend in the left panel. In each panel, the vertical axis is value of the convergence rate (2.15) at different iteration i. The horizontal axes show values of α. As shown, the convergence becomes extremely slower when α is higher than 0.95.

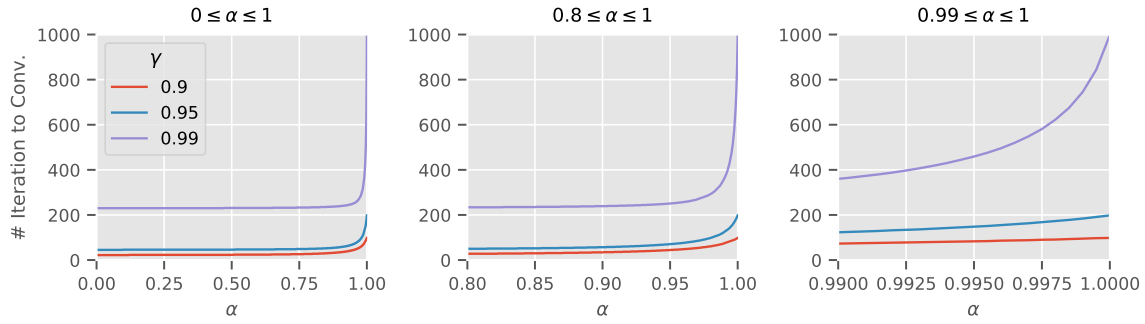


Figure 2.2: The number of iterations to convergence of CVI-Q and Ψ. The discount factor γ is indicated by the line colors as shown in the legend in the left panel. In each panel, the vertical axis is the number of iterations i at which ACVI's convergence rate (2.15) becomes less than 0.1. Note that the vertical axes are in log scale. The horizontal axes show values of α, and different panels show different ranges of α for visibility. As shown, the convergence is extremely slower when α is higher than 0.95.

stemming from the stochasticity of MDPs. The greatest robustness can be attained when $\alpha = 1$, with which, however, the convergence is much slower.

Next let us consider effects of errors when they do not cancel out by averaging. For simplicity, we consider L^∞-norm performance bound (2.12).

We first consider the net effect of errors. We have

$$\begin{aligned}\frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j-1}\right\|_\infty &\le \frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\sum_{k=0}^{i-j-1}\alpha^k\left\|\varepsilon_{i-j-k-1}\right\|_\infty \\ &= \frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}c_{i-j-1}\left\|\varepsilon_{i-j-1}\right\|_\infty,\end{aligned}$$

where we defined $c_{i-j-1} := \sum_{k=0}^{j}\alpha^k\gamma^{j-k}/\alpha_{0:i}$. Thus the net effects of errors can be

understood by

$$\lim_{i\to\infty}\sum_{j=0}^{i-1} c_{i-j-1} = \lim_{i\to\infty}\frac{1}{\alpha_{0:i}}\sum_{j=0}^{i-1}\frac{\alpha^{j+1}-\gamma^{j+1}}{\alpha-\gamma} = \frac{1}{\alpha-\gamma}\left(1-\frac{1-\alpha}{1-\gamma}\right) = \frac{1}{1-\gamma},$$

in which we assumed $\alpha \neq \gamma$. Thus the net effect of errors are the same regardless of α.

We next consider how effects of errors decay. Error decay of ε_{i-j-1} is expressed by c_{i-j-1}. Figure 2.3 visualizes it illustrating enlarged and lessened effect of the past ($j \approx i-1$) and recent errors ($j \approx 0$) for a large α, respectively. (Note that it is completely same as Figure 3.4.)

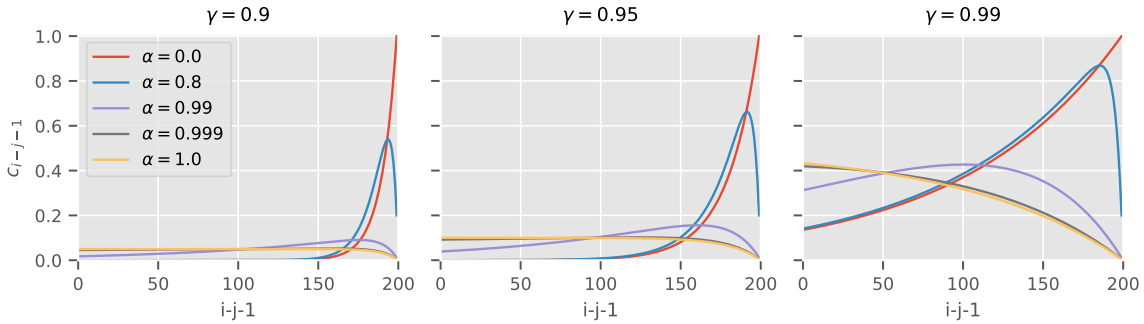


Figure 2.3: Error decay of ACVI-Q and Ψ. Lines show the coefficient $c_{i-j-1} := \sum_{k=0}^{j}\alpha^{j-k}\gamma^k/\alpha_{0:i-1}$ with various α as in the legend. γ is shown on top of each panel. The horizontal axis is $i-j-1$. As clearly shown, effects of errors at early iterations lingers if α is high. However if each error function has the same L^∞-norm, the net effect of errors is the same across different α as argued in the main text.

We finally touch on the term

$$\frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}|,$$

which is an inevitable loss due to the use of softmax. It converges to $\gamma(1-\alpha)(1-\gamma)^{-2}\beta^{-1}\ln|\mathcal{A}|$. Thus, unless $\alpha = 1$ or $\beta = \infty$, it is not 0. However, in Section 2.2.3: ref, we show that a small β may be preferable despite this inevitable loss.

2.2.2 Tightness of Regularization Agnostic Performance Bounds

The performance bounds in Section 2.2.1 show that error-tolerance of ACVI is the same as that of AVI. Furthermore they imply that using infinite β is best. The following theorem states that the performance bounds (2.12) is essentially not improvable when $\beta = \infty$.

Theorem 2.2.3. *When $\beta = \infty$, there exists an MDP and a sequence of ε_k satisfying*

the following: for any real value $\delta \in (0, \infty)$, there is a positive integer I such that

$$\frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j-1}\right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} \leq \left\|Q_* - Q_{\pi_i}\right\|_\infty + \delta \tag{2.16}$$

holds for any $i \geq I$.

Suppose that there is a performance bound b_i that is smaller than the right hand side of the performance bounds (2.12). Then the inequality (2.16) states that

$$b_i < \frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j-1}\right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} \leq b_i + \delta$$

for a large enough i. In other words, the difference between b_i and our performance bound (2.12) is within $(0, \delta]$, and our bound is arbitrarily close to b_i.

2.2.3 Regularization Aware Performance Bounds

Theorem 2.2.1 states that $\beta = \infty$, i.e., algorithms with a hard gap-increasing operator are the optimal choice. However, there is experimental evidence that a finite β leads to better results (Azar et al. (2012); Fox et al. (2016); Haarnoja et al. (2017)). In this subsection, we provide novel form of performance bounds for ACVI that show benefits of setting β to a finite value especially when errors are huge.

The following proposition provides a bound of KL divergence between π_i and π_{i-1}. It is utilized in the novel form of performance bounds.

Proposition 2.2.4. *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9). If $\|\varepsilon_k\| \leq \varepsilon$ holds for any integer $j \in \{1, 2, \ldots\}$, policies in the sequence satisfies for any i, $\max_s D_{KL}(\pi_i(\cdot|s)|\pi_{i-1}(\cdot|s)) \leq \delta_i$, where δ_i is*

$$\delta_i := 4\beta\left(\frac{1-\gamma^i}{1-\gamma}\varepsilon + r_{max}\sum_{j=0}^{i-1}\alpha^j\gamma^{i-j-1}\right).$$

Using this bound of KL divergence, we obtain the L^∞-norm performance bounds for ACVI-Q and Ψ.

Theorem 2.2.5. *Suppose a sequence of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9). If $\|\varepsilon_k\| \leq \varepsilon$ holds for any integer $j \in \{1, 2, \ldots\}$, the following L^∞-norm performance bound holds:*

$$\begin{aligned}\left\|Q_* - Q_{\pi_i}\right\|_\infty \leq 2\gamma\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j-1}\right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} \\ + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)}\ln|\mathcal{A}| + \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma}\sum_{j=0}^{i-1}\gamma^j\delta_{i-j}^{1/2}.\end{aligned} \tag{2.17}$$

We next provide $L^p(\rho)$-norm performance bounds. To succinctly state them, we need the following short-hand notation for concentrability coefficients:

$$d_{k,j} := c(2, \rho, \nu; \overbrace{\pi_i, \ldots, \pi_i}^{k}, \pi_i, \pi_{i-1}, \ldots, \pi_j)$$

With this notation, we have the following theorem.

Theorem 2.2.6. *Suppose a sequence of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9). If $\|\varepsilon_k\| \le \varepsilon$ holds for any integer $j \in \{1, 2, \ldots\}$, the following $L^p(\rho)$-norm ($p \in [1, \infty)$) performance bounds hold:*

$$\|Q_* - Q_{\pi_i}\|_{\rho,p} \le 2\gamma\mathcal{E}'_{\nu,2p,i} + \frac{2\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln|\mathcal{A}| + \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma} \sum_{j=0}^{i-1} \gamma^j \delta_{i-j}^{1/2}, \tag{2.18}$$

where

$$D_j := \frac{\left(c^*_{j+1}\right)^{1/p} + d^{1/p}_{0,j}}{2},$$

$$\mathcal{E}'_{\nu,2p,i} := \sup_{\pi_i,\ldots,\pi_0} \sum_{j=0}^{i-1} \gamma^j D_j \left\| \frac{E_{i-j-1}}{\alpha_{0:i}} \right\|_{\nu,2p}.$$

Remark 2.2.2. *By taking the minimum of the bounds (2.14) and (2.18), we obtain a bound that is clearly no worse than both bounds.*

To understand differences, let us compare (2.12) with (2.17). Their major differences are the following: (i) $\mathcal{E}'_{\nu,2p,i}$ is multiplied by 2γ in (2.17), whereas it is multiplied by $2\gamma/(1-\gamma)$ in (2.12). (ii) There is an additional term *const.* $\sum_{j=0}^{i-1} \gamma^j \delta_{i-j}^{1/2}$ in (2.17). (iii) in (2.17), the loss of using softmax

$$\frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln|\mathcal{A}|$$

is smaller than the corresponding term in (2.12) by a factor of $1-\gamma$.

The first difference indicates that *algorithms using the softmax operator are error-tolerant*. As we explained, gap-increasing operators make algorithms noise-tolerant. However, if errors are not noise, the argument is nullified. In contrast, algorithms using the softmax operator have great tolerance to any type of error. The price to pay for this tolerance is the second difference, which decreases monotonically in β. Thus, a small β leads to better performance. Note that a small β results in the increase

$$\frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln|\mathcal{A}|$$

To compensate for it, α must be large enough. Therefore, the use of the softmax operator alone is not sufficient.

In addition, Theorem 2.2.6 shows another benefit of a finite β: concentrability coefficients D_j is better than C_j. To see this, note that C_j contains $\sum_{k=0}^{\infty} \gamma^k c_{k,j}^{1/p} = \sum_{k=0}^{\infty} \gamma^k d_{k,j}^{1/p}$, which clearly satisfies

$$\sum_{k=0}^{\infty} \gamma^k d_{k,j}^{1/p} \geq d_{0,j}^{\frac{1}{p}}$$

As a consequence, $D_j = \infty$ implies $C_j = \infty$. Furthermore, it is possible to construct an example in which D_j is finite, but C_j is infinite. In this sense, D_j in (2.18) is better than C_j.

Finally, we note that $\alpha \in [0, 1)$ together with a finite β forces a policy π_i to be stochastic. As a result, concentrability coefficients of ACVI with such α and β before taking $\sup_{\pi_i,\ldots,\pi_0}$ are expected to be smaller compared to algorithms with either $\alpha = 1$ or $\beta = \infty$. However, our analysis fails in capturing it.

2.3 Related Research

Before concluding this chapter, we provide a quick review of related results to clarify our contributions compared to existing works.

There are many algorithms that regularize policy and/or value updates by either the entropy (G-Learning (GL) (Fox et al., 2016), Soft Q-Learning (SQL) (Haarnoja et al., 2017), Softmax Deep Q-Network (SDQN) (Song et al., 2019)) or KL divergence (DPP (Azar et al., 2012), Ψ-learning (Rawlik, 2013), Trust Region Policy Optimization (TRPO) (Schulman et al., 2015), Maximum a Posteriori policy Optimization (MPO) (Abdolmaleki et al., 2018)). These algorithms vary mainly depending on the following three factors:

1. An algorithm is based on PI or VI scheme.
2. A constraint is used instead of a regularization.
3. In addition to policy updates, value updates are regularized or not.

Table 2.1 summarizes those algorithms, based on these factors. Figure 2.4 summarizes algorithms generalized by CVI.

Despite the proliferation of algorithms with a regularization, most of the works did not provide theoretical explanation on why the regularization helps the learning: most of them just provide convergence results or results that only hold when no errors are involved. Because algorithms without the regularization works perfectly when no errors are involved, those results are not sufficient to understand benefits of the regularization.

A notable exception is a work by Azar et al. (2012), which provided an error propagation analysis of DPP, a special case of CVI with $\alpha = 1$. Theorem 2.2.1 extends their result to a case with $\alpha \leq 1$. Theorem 2.2.2 extends their result to $L^p(\rho)$-norm performance bound. This result is obtained by leveraging proof techniques used in

Table 2.1: A summary of algorithms using the entropy or KL divergence regularization (or constraint). The symbol ✓means "Yes", and "reg." is an abbreviation of "regularization". As for DPP, it is difficult to say that it uses value update regularization because it does not exactly equivalent to VI with the KL divergence regularization.

	GL	SQL	SDQN	DPP	Ψ-learning	TRPO	MPO
Using VI scheme	✓	✓	✓	✓	✓		
Using KL reg.				✓	✓	✓	✓
Using constraint						✓	✓
Value update reg.	✓	✓	✓	?	✓		

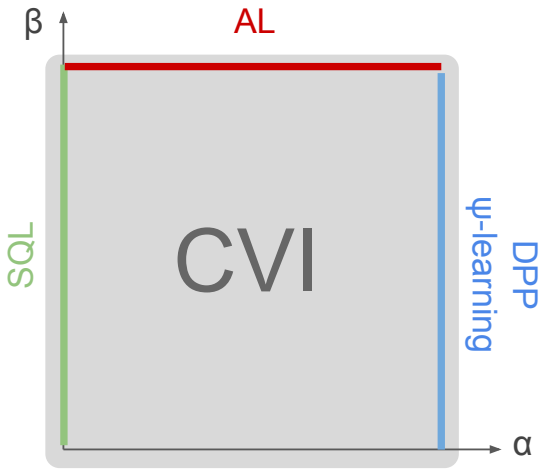


Figure 2.4: A summary of DP algorithms generalized by CVI. In the figure, VI, SQL, AL and DPP (Ψ-learning) correspond to the top left corner, left edge, top edge, and right edge, respectively. CVI unifies these algorithms.

(Scherrer et al., 2015), which lead to some improvements of coefficients in our bound compared to one obtained in (Azar et al., 2012).

While the analysis of Azar et al. (2012) shows the benefit of the KL regularization, it fails to explain a benefit of using a finite KL regularization coefficient (β), which is observed and noted in (Azar et al., 2012; Fox et al., 2016; Haarnoja et al., 2017). On the other hand, Theorems 2.2.5 and 2.2.6 sheds some light on its benefit.

Recently Geist et al. (2019) have provided error propagation analysis of a general algorithm. Their results and ours differ in three points:

1. First we considered an algorithm with both entropy and KL regularizations, and showed its equivalence to a general algorithm including a variety of previous algorithms, namely VI, DPP, AL and SQL. On the other hand, Geist et al. (2019) considered an algorithm with either a strongly convex regularizer or a Bregman divergence regularization, the former of which includes the entropy regularization, and the latter of which includes the KL divergence regularization. Therefore our results and theirs are generalization of existing works to different directions.

2. Second they bounded the regret $\sum_{j=1}^{i} \left\| Q_* - Q_{\pi_j} \right\|_\infty / i$, which is an average of

the losses $\left\|Q_* - Q_{\pi_j}\right\|_\infty$ at all iterations. We note that it is really easy to obtain a regret bound of CVI from our bounds of the loss.

3. Third they could not show a benefit of using a regularizer. For a strongly convex regularization, they showed a performance bound (Corollary 1), regarding which they noted "As this is the same bound (up to the fact that it deals with regularized MDPs) as the one of AMPI...". (AMPI stands for approximate modified policy iteration, which is a classical ADP algorithm we did not explain in this thesis.) For a Bregman divergence regularization, they provided a bound of the regret (Corollary 4), which consist of sum of $\|\varepsilon_j\|_\infty$ terms, and they noted "Yet, we highlight again the fact that we bound a regret, and bounding the regret of AMPI would provide a similar result."

From these differences, we believe that our results are, although somewhat limited in sense that we only considered the entropy and KL regularizations, an important step towards understanding the effect of regularizations.

2.4 Conclusion

Soft Q-learning, AL, and DPP, all of which employ value-iteration-like, single-stage lookahead updates using the softmax operator and/or gap-increasing operator, demonstrated their superiority to VI (Baird III (1999); Azar et al. (2012); Rawlik (2013); Bellemare et al. (2016); Fox et al. (2016); Haarnoja et al. (2017)). However, they are not theoretically well understood. In this chapter, we proposed and analyzed CVI that unifies them to explain their theoretical properties, such as performance guarantees under non-exact update settings and roles of their hyper-parameters.

The performance bounds without KL divergence improve the existing performance bound for DPP and comprise the first performance bound for soft Q-learning and AL. They also clarify the role of a hyper-parameter α in gap-increasing operators: α controls the trade-off between tolerance to stochastic error and convergence rate.

We also found that performance bounds without KL divergence are essentially tight as long as greedy value updates and a greedy policy are used. Furthermore, they imply that as long as greedy value updates and a greedy policy are used, tolerance of algorithms to non-stochastic errors are almost the same as that of VI.

Performance bounds with KL divergence show that the limitation by greedy value updates and a greedy policy can be overcome when the softmax operator is used. However, the softmax operator alone may lead to poor asymptotic performance, which is controlled by β. Algorithms with a soft gap-increasing operator enjoy both noise-tolerance and error-tolerance, while avoiding poor asymptotic performance.

However, there are following open questions:

- *Is the sample complexity of gap-increasing algorithms minimax optimal?* In this thesis, I carried out error propagation analysis, wherein I did not consider how using the gap-increasing operator changes the error functions. Sample complexity analysis takes such changes into account and provides a deeper understanding of algorithms.

- *Do other regularizations, such as a Bregman divergence regularization, have similar property or any other benefits?* While the KL divergence is a type of the Bregman divergence, it is not clear if benefits of more general regularization with Bregman divergence (or any other divergences and probability metrics) exist and can be proven.

- *Does a policy regularization have any benefit in terms of exploration?* The exploration is surely a vital part of RL algorithm. However, error propagation analysis cannot capture the aspect of exploration.

- *Is the regularization agnostic L^∞-norm performance bound for ACVI-Q and Ψ (bound (2.12)) tight for arbitrary β?* While I could show that it is tight when $\beta = \infty$, it is unclear if the bound is tight. A key to its proof is that in the worst case, a set of greedy policies may contain the best and worst policy, the latter of which is intentionally chosen to prove the tightness. When β is finite, this fact cannot be used.

Addressing those questions give more insights into algorithms using the softmax operator and/or a gap-increasing operator as well as algorithms with various types of regularizations.

While some open problems remain, the present chapter is an important step toward understanding algorithms using the softmax operator and/or a gap-increasing operator.

2.5 Proofs

In this section, we provide proofs in this chapter. We begin with recalling some definitions. We define a sequence of policies $(\boldsymbol{\mu}_i)_{i\in\mathbb{Z}_+}$ such that

$$\boldsymbol{\mu}_i \Psi_i := \boldsymbol{m}^\beta \Psi_i.$$

Such policies always exist (Asadi and Littman, 2017). We also define

$$E_i := \sum_{j=0}^{i} \alpha^j \varepsilon_{i-j}$$

for all i. Furthermore we use a shorthand notation

$$\alpha_{j:i} := \begin{cases} \sum_{k=j}^{i} \alpha^k & \text{if } i \geq j \\ 0 & \textit{otherwise} \end{cases}$$

for two integers $i, j \in \mathbb{Z}$. We call the following policy a Boltzmann policy (given a function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$):

$$b^\beta(a|x; Q) := \frac{\exp\left(\beta Q(x,a)\right)}{\sum_{b\in\mathcal{A}} \exp\left(\beta Q(x,b)\right)},$$

where $\beta \in (0, \infty)$ is the inverse temperature. A Boltzmann-softmax operator $\boldsymbol{b}^\beta$ is defined such that

$$\left(\boldsymbol{b}^\beta Q\right)(x) := \sum_{a \in \mathcal{A}} b^\beta(a|x; Q) Q(x, a)$$

for any state $x \in \mathcal{X}$ and function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$. Note that the Boltzmann-softmax operator is not linear, as it depends on input function Q. We define $\boldsymbol{m}^\infty$ and $\boldsymbol{b}^\infty$ to be $\boldsymbol{m}$. As we show later, $\boldsymbol{m}^\beta Q \leq \boldsymbol{b}^\beta Q$ and $\lim_{\beta \to \infty} \boldsymbol{b}^\beta Q = \lim_{\beta \to \infty} \boldsymbol{m}^\beta f = \boldsymbol{m} Q$ hold.

2.5.1 Auxiliary Lemmas

For error propagation analysis of ACVI, we need several lemmas. The following lemma shows a relationship between the mellowmax and Boltzmann-softmax operators.

Lemma 2.5.1. *For any inverse temperature $\beta \in (0, \infty)$ and function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$,*

$$\frac{1}{\beta} \ln |\mathcal{A}| \geq \boldsymbol{b}^\beta Q - \boldsymbol{m}^\beta Q \geq 0. \tag{2.19}$$

Proof. The entropy of $b^\beta(\cdot|x; Q)$ is

$$H := -\sum_{a \in \mathcal{A}} b^\beta(a|x; Q) \ln b^\beta(a|x; Q).$$

It can be rewritten as

$$\begin{aligned} H &= -\sum_{a \in \mathcal{A}} \frac{\exp\left(\beta Q(x, a)\right)}{Z} \left(\beta Q(x, a) - \ln Z\right) \\ &= \ln Z - \beta \left(\boldsymbol{b}^\beta Q\right)(x) \\ &= \beta \left(\boldsymbol{m}^\beta Q\right)(x) - \beta \left(\boldsymbol{b}^\beta Q\right)(x) + \ln |\mathcal{A}|, \end{aligned}$$

where $Z := \sum_{a \in \mathcal{A}} \exp\left(\beta Q(s, a)\right)$, and the last line is obtained by using

$$\frac{1}{\beta} \ln \frac{Z}{|\mathcal{A}|} = \left(\boldsymbol{m}^\beta Q\right)(x)$$

Because $0 \leq H \leq \ln |\mathcal{A}|$, the claim holds. □

Lemma 2.5.3, which is proven by using the following lemma, states that the mellowmax and Boltzmann-softmax operators are close to the max operator.

Lemma 2.5.2. *For any inverse temperature $\beta \in (0, \infty)$, state $x \in \mathcal{X}$ and function $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$, $\left(\boldsymbol{m}^\beta Q\right)(x)$ is non-decreasing in β while $\left(\boldsymbol{m}^\beta Q\right)(x) + \left(\ln |\mathcal{A}|\right)/\beta$ is non-increasing in β.*

Proof. The former claim holds since

$$\frac{\partial}{\partial \beta} \left(\boldsymbol{m}^\beta Q\right)(x) = \frac{1}{\beta} \left(\left(\boldsymbol{b}^\beta Q\right)(x) - \left(\boldsymbol{m}^\beta Q\right)(x)\right) \geq 0,$$

where the inequality is due to Lemma 2.5.1.

On the other hand,

$$\frac{\partial}{\partial\beta}\left(\left(\boldsymbol{m}^{\beta}Q\right)(x)+\frac{1}{\beta}\ln|\mathcal{A}|\right)=\frac{1}{\beta}\left(\left(\boldsymbol{b}^{\beta}Q\right)(x)-\left(\boldsymbol{m}^{\beta}Q\right)(x)-\frac{1}{\beta}\ln|\mathcal{A}|\right)\leq 0,$$

where the inequality is again due to Lemma 2.5.1. □

Lemma 2.5.3. *For any inverse temperature $\beta\in(0,\infty)$ and function $Q\in\mathcal{B}(\mathcal{X}\times\mathcal{A})$,*

$$\boldsymbol{m}Q-\boldsymbol{b}^{\beta}Q\leq\boldsymbol{m}Q-\boldsymbol{m}^{\beta}Q\leq\frac{1}{\beta}\ln|\mathcal{A}|.$$

Proof. From Lemma 2.5.2, $\left(\boldsymbol{m}^{\beta}Q\right)(x)+\left(\ln|\mathcal{A}|\right)/\beta$ is non-increasing in β. Therefore, for any $x\in\mathcal{X}$,

$$\left(\boldsymbol{m}^{\beta}Q\right)(x)+\frac{1}{\beta}\ln|\mathcal{A}|\geq\lim_{\beta\to\infty}\left(\boldsymbol{m}^{\beta}Q\right)(x)=\left(\boldsymbol{m}Q\right)(x),$$

where the last equality is proven in (Asadi and Littman, 2017). From Lemma 2.5.1, $\boldsymbol{m}^{\beta}f\leq\boldsymbol{b}^{\beta}f$, and thus, the claim holds. □

The following lemma not only makes our theoretical analysis simpler, but also shows that the behavior of CVI-Ψ is determined by a series of functions whose update rule is simpler.

Lemma 2.5.4. *Suppose sequences of policies $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(\Psi_i)_{i\in\mathbb{Z}_+}$ obtained by the update (2.10). For any positive integer $i\geq 1$,*

$$\Psi_i=\alpha_{0:i-1}q_i-\alpha_{1:i-1}\boldsymbol{\mu}_{i-1}q_{i-1}\tag{2.20}$$

holds, where q_i is recursively defined by $q_0:=\Phi_0$ and

$$\alpha_{0:i}q_{i+1}:=\alpha_{0:i}r+\alpha_{0:i-1}\gamma\boldsymbol{P}_{\mu_i}q_i+E_i.\tag{2.21}$$

Proof. We prove the claim by induction. For $i=1$, $\Psi_1=\boldsymbol{B}_{\mu_0}\Psi_0+\varepsilon_0=\alpha_{0:0}q_1+E_0$. Therefore the claim holds for $i=1$.

Suppose that up to i ($i>1$), the claim holds. Then, we deduce that

$$\begin{aligned}\boldsymbol{B}_{\mu_i}\Psi_i&=\boldsymbol{B}_{\mu_i}\left(\alpha_{0:i-1}q_i-\alpha_{1:i-1}\boldsymbol{\mu}_{i-1}q_{i-1}\right)\\&=\left(\alpha_{0:i}-\alpha_{1:i}\right)r+\alpha_{0:i-1}\gamma\boldsymbol{P}_{\mu_i}q_i-\alpha_{1:i-1}\gamma\boldsymbol{P}_{\mu_{i-1}}q_{i-1}\\&=\alpha_{0:i}r+\alpha_{0:i-1}\gamma\boldsymbol{P}_{\mu_i}q_i-\alpha\left(\alpha_{0:i-1}r+\alpha_{0:i-2}\gamma\boldsymbol{P}_{\mu_{i-1}}q_{i-1}\right)\\&=\alpha_{0:i}q_{i+1}-\alpha_{1:i}q_i-E_i+\alpha E_{i-1}\\&=\alpha_{0:i}q_{i+1}-\alpha_{1:i}q_i-\varepsilon_i.\end{aligned}$$

Furthermore we deduce that $\Psi_i - \boldsymbol{\mu}_i \Psi_i = \alpha_{0:i-1} q_i - \alpha_{0:i-1} \boldsymbol{\mu}_i q_i$. Combining these results

$$\begin{aligned}\Psi_{i+1} &= \boldsymbol{B}_{\mu_i} \Psi_i + \alpha \left(\Psi_i - \boldsymbol{\mu}_i \Psi_i\right) + \varepsilon_i \\ &= \alpha_{0:i} q_{i+1} - \alpha_{1:i} q_i + \alpha_{1:i} q_i - \alpha_{1:i} \boldsymbol{\mu}_i q_i \\ &= \alpha_{0:i} q_{i+1} - \alpha_{1:i} \boldsymbol{\mu}_i q_i.\end{aligned}$$

This concludes the proof. □

The following corollary shows that μ_i is almost greedy.

Corollary 2.5.5. *Suppose sequences of policies $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. Then for any non-negative integer $i \geq 0$*

$$\boldsymbol{\mu}_i \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) \geq \boldsymbol{m} \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) - \frac{\ln |\mathcal{A}|}{\alpha_{0:i}\beta}$$

holds.

Proof. From Lemma 2.5.4, we have $\boldsymbol{\mu}_i \left(\alpha_{0:i-1} q_i\right) = \boldsymbol{m}^{\beta} \left(\alpha_{0:i-1} q_i\right)$. As a result,

$$\boldsymbol{\mu}_i \left(\alpha_{0:i-1} q_i\right) = \boldsymbol{m}^{\beta} \left(\alpha_{0:i-1} q_i\right) = \boldsymbol{m}^{\alpha_{0:i}\beta} \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) \geq \boldsymbol{m} \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) - \frac{\ln |\mathcal{A}|}{\alpha_{0:i}\beta},$$

where the inequality follows from Lemma 2.5.3. □

2.5.2 Proof of Theorems 2.2.1 and 2.2.2

We decompose $Q_* - Q^{\pi_i}$ to $Q_* - Q$ and $-\left(Q_{\pi_i} - Q\right)$, where $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$ is some function. Then we prove an upper bound of $Q_* - Q$ and a lower bound of $Q_{\pi_i} - Q$ to establish a point-wise upper bound of $Q_* - Q_{\pi_i}$.

Lemma 2.5.6. *Suppose sequences of policies $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. The following upper bound for $Q_* - q_{i+1}$ holds for any non-negative integer $i \geq 0$:*

$$Q_* - q_{i+1} \leq -\frac{1}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \left(\boldsymbol{P}^*\right)^j E_{i-j} + \frac{\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln |\mathcal{A}|. \quad (2.22)$$

Proof of Lemma 2.5.6. We prove the claim by induction. From (2.21) and Lemma 2.5.3,

$$q_{i+1} = r + \gamma \boldsymbol{P}_{\mu_i} \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) + \frac{E_i}{\alpha_{0:i}} \geq r + \gamma \boldsymbol{P}_* \left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}} q_i\right) + \frac{E_i}{\alpha_{0:i}} - \frac{\gamma \ln |\mathcal{A}|}{\alpha_{0:i}\beta}.$$

Accordingly, for $i = 0$,

$$Q_* - q_1 = \gamma \boldsymbol{P}_* Q_* - \gamma \boldsymbol{P}_{\mu_0} q_0 - \frac{E_0}{\alpha_{0:0}} = \gamma \boldsymbol{P}_* Q_* - \frac{E_0}{\alpha_{0:0}} \leq \frac{\gamma V_{max}}{\alpha_{0:0}} - \frac{E_0}{\alpha_{0:0}},$$

where the first equality follows because q_0 is a constant function taking 0, and the last inequality is due to $Q_* \leq V_{max}$. Therefore, the claim holds for $i = 0$.

Suppose that the claim holds up to i. Then we deduce that

$$\begin{aligned} Q_* - q_{i+1} &\leq \frac{\gamma\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_*\left(Q_* - q_i\right) + \frac{\gamma\alpha^i}{\alpha_{0:i}}\boldsymbol{P}_*Q_* - \frac{E_i}{\alpha_{0:i}} + \frac{\gamma\ln|\mathcal{A}|}{\alpha_{0:i}\beta} \\ &\leq \frac{\gamma\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_*\left(Q_* - q_i\right) + \frac{\gamma\alpha^i}{\alpha_{0:i}}V_{max} - \frac{E_i}{\alpha_{0:i}} + \frac{\gamma\ln|\mathcal{A}|}{\alpha_{0:i}\beta}, \end{aligned}$$

where the inequalities are obtained similarly to the case in which $i = 0$. By the assumption of the induction,

$$\begin{aligned} Q_* - q_{i+1} &\leq \frac{\gamma\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_*\left(Q_* - q_i\right) + \frac{\gamma\alpha^i}{\alpha_{0:i}}\boldsymbol{P}_*Q_* - \frac{E_i}{\alpha_{0:i}} + \frac{\gamma\ln|\mathcal{A}|}{\alpha_{0:i}\beta} \\ &\leq -\frac{\gamma}{\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\left(\boldsymbol{P}^*\right)^j E_{i-j-1} + \frac{\gamma^2 V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\alpha^{i-j-1} + \frac{\gamma^2(1-\gamma^{i-1})}{\alpha_{0:i}\beta(1-\gamma)}\ln|\mathcal{A}| \\ &\qquad + \frac{\gamma\alpha^i}{\alpha_{0:i}}V_{max} - \frac{E_i}{\alpha_{0:i}} + \frac{\gamma\ln|\mathcal{A}|}{\alpha_{0:i}\beta} \\ &= -\frac{1}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\left(\boldsymbol{P}^*\right)^j E_{i-j} + \frac{\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)}\ln|\mathcal{A}|. \end{aligned}$$

Therefore, the claim holds. □

Lemma 2.5.7. *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. The following upper bound for $Q_{\pi_i} - q_{i+1}$ holds for any non-negative integer $i \geq 0$:*

$$Q_{\pi_i} - q_{i+1} \geq -\frac{1}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\boldsymbol{Q}_{i,i-j}E_{i-j} - \frac{\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} - \frac{\gamma^2(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}|, \tag{2.23}$$

where

$$\boldsymbol{Q}_{i,i-j} := \begin{cases} \boldsymbol{I} & \text{for } j = 0 \\ (\boldsymbol{I} - \gamma\boldsymbol{P}_{\pi_i})^{-1}\boldsymbol{P}_{\pi_i}\cdots\boldsymbol{P}_{\pi_{i-j+1}}(\boldsymbol{I} - \gamma\boldsymbol{P}_{\pi_{i-j}}) & \text{for } 1 \leq j \leq i \end{cases}.$$

Proof of Lemma 2.5.7. We first note that for any non-negative integer i,

$$\boldsymbol{\mu}_i\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right) \leq \boldsymbol{\pi}_i\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right) \tag{2.24}$$

holds. This is clear from Lemma 2.5.4.

For any non-negative integer i, we deduce that

$$\begin{aligned}
&(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i})\,(Q_{\pi_i}-q_{i+1})\\
&=\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{B}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)+\frac{E_i}{\alpha_{0:i}}\right)-\gamma\boldsymbol{P}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{E_i}{\alpha_{0:i}}\\
&\geq\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{B}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)+\frac{E_i}{\alpha_{0:i}}\right)-\gamma\boldsymbol{P}_{\pi_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{E_i}{\alpha_{0:i}}\\
&=\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{B}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)\right)-\gamma\boldsymbol{P}_{\pi_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i,
\end{aligned}$$

where the inequality (2.24) is used. Accordingly from Lemma 2.5.3,

$$\begin{aligned}
&(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i})\,(Q_{\pi_i}-q_{i+1})\\
&\geq\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{B}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i\\
&\geq\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{B}_{\pi_{i-1}}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i-\frac{\gamma^2\ln|\mathcal{A}|}{\alpha_{0:i}\beta}\\
&=\gamma\boldsymbol{P}_{\pi_i}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_{i-1}}\right)\left(Q_{\pi_{i-1}}-\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i\right)-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i-\frac{\gamma^2\ln|\mathcal{A}|}{\alpha_{0:i}\beta}\\
&=\frac{\gamma\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\left[\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_{i-1}}\right)\left(Q_{\pi_{i-1}}-q_i\right)+\alpha^i r\right]-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i-\frac{\gamma^2\ln|\mathcal{A}|}{\alpha_{0:i}\beta}\\
&\geq\frac{\gamma\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_{i-1}}\right)\left(Q_{\pi_{i-1}}-q_i\right)-\gamma\alpha^i r_{max}-\frac{1}{\alpha_{0:i}}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)E_i-\frac{\gamma^2\ln|\mathcal{A}|}{\alpha_{0:i}\beta}.
\end{aligned}$$

By continuing the same argument, we deduce that

$$\begin{aligned}
&(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i})\,(Q_{\pi_i}-q_{i+1})\\
&\geq\frac{\gamma^i}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\cdots\boldsymbol{P}_{\pi_1}\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_0}\right)\left(Q_{\pi_0}-q_1\right)-\frac{\gamma r_{max}}{\alpha_{0:i}}\sum_{j=0}^{i-1}\alpha^{i-j}\gamma^j\\
&\qquad-\frac{1}{\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_i}\right)\boldsymbol{Q}_{i,i-j}E_{i-j}-\frac{\gamma^2(1-\gamma^i)\ln|\mathcal{A}|}{\alpha_{0:i}\beta(1-\gamma)}.
\end{aligned}$$

Since

$$\begin{aligned}
(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_0})\,(Q_{\pi_0}-q_1)&=\boldsymbol{B}_{\pi_0}q_1-q_1\\
&=\gamma\boldsymbol{P}_{\pi_0}\left(r+E_0\right)-E_0\\
&\geq-\gamma r_{max}-\left(\boldsymbol{I}-\gamma\boldsymbol{P}_{\pi_0}\right)E_0
\end{aligned}$$

we finally obtain

$$
\begin{aligned}
&\left(\boldsymbol{I}-\gamma \boldsymbol{P}_{\pi_i}\right)\left(Q_{\pi_i}-q_{i+1}\right) \\
&\geq -\frac{\gamma^i}{\alpha_{0:i}} \boldsymbol{P}_{\pi_i} \cdots \boldsymbol{P}_{\pi_1}\left(\boldsymbol{I}-\gamma \boldsymbol{P}_{\pi_0}\right) E_0-\frac{1}{\alpha_{0:i}} \sum_{j=0}^{i-1} \gamma^j\left(\boldsymbol{I}-\gamma \boldsymbol{P}_{\pi_i}\right) \boldsymbol{Q}_{i, i-j} E_{i-j} \\
&\qquad\qquad -\frac{\gamma^{i+1} r_{max}}{\alpha_{0:i}} V_{max}-\frac{\gamma r_{max}}{\alpha_{0:i}} \sum_{j=0}^{i-1} \alpha^{i-j} \gamma^j-\frac{\gamma^2\left(1-\gamma^i\right) \ln |\mathcal{A}|}{\alpha_{0:i} \beta(1-\gamma)} \\
&=-\frac{1}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j\left(\boldsymbol{I}-\gamma \boldsymbol{P}_{\pi_i}\right) \boldsymbol{Q}_{i, i-j} E_{i-j}-\frac{\gamma r_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \alpha^{i-j} \gamma^j-\frac{\gamma^2\left(1-\gamma^i\right) \ln |\mathcal{A}|}{\alpha_{0:i} \beta(1-\gamma)}.
\end{aligned}
$$

Recall that $\left(\boldsymbol{I}-\gamma \boldsymbol{P}_{\pi_i}\right)^{-1}$ is monotone and linear. Therefore by applying it to both sides of the inequality, it is confirmed that the claim holds. □

By combining Lemmas 2.5.6 and 2.5.7, the following proposition is obtained. (Note that the first summation in the inequality (2.25) is from $j=1$ to i because $\boldsymbol{Q}_{j,j} = \left(\boldsymbol{P}_*\right)^0 = \boldsymbol{I}$ for $j=0$.)

Proposition 2.5.8 (Point-wise Performance Bound for ACVI-Q and Ψ)**.** *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. The following point-wise upper bound for $Q_* - Q_{\pi_i}$ holds for any non-negative integer i:*

$$
\begin{aligned}
&Q_*-Q_{\pi_i} \qquad\qquad (2.25)\\
&\leq \sum_{j=1}^{i} \frac{\gamma^j}{\alpha_{0:i}}\left(\boldsymbol{Q}_{i, i-j} E_{i-j}-\left(\boldsymbol{P}^*\right)^j E_{i-j}\right)+\frac{2 \gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j}+\frac{\gamma\left(1-\gamma^i\right)}{\alpha_{0:i} \beta(1-\gamma)^2} \ln |\mathcal{A}|,
\end{aligned}
$$

where $\sum_{j=1}^{0} Q_j$ is a constant function taking 0 for any sequence of functions Q_j. Furthermore the same bound holds for $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.7).

As a corollary of Proposition 2.5.8, l_∞-norm performance bound for ACVI-Q and Ψ can be obtained.

Proof of Theorem 2.2.1. From Proposition 2.5.8 and $|Q_*(s,a)-Q_{\pi_i}(s,a)| = Q_*(s,a)-Q_{\pi_i}(s,a)$,

$$
\begin{aligned}
&\left\|Q_*-Q_{\pi_i}\right\|_\infty \\
&=\max_{s,a}\left(Q_*-Q_{\pi_i}\right)(s,a) \\
&=\max_{s,a}\left(Q_*-q_{i+1}-\left(Q_{\pi_i}-q_{i+1}\right)\right)(s,a) \\
&\leq \sum_{j=1}^{i} \frac{\gamma^j}{\alpha_{0:i}}\left(\left\|\boldsymbol{Q}_{i, i-j} E_{i-j}\right\|_\infty+\left\|E_{i-j}\right\|_\infty\right)+\frac{2 \gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j}+\frac{\gamma\left(1-\gamma^i\right)}{\alpha_{0:i} \beta(1-\gamma)^2} \ln |\mathcal{A}|.
\end{aligned}
$$

Because $\left\|\boldsymbol{Q}_{i,i-j}Q\right\|_\infty \leq (1+\gamma)\left\|Q\right\|_\infty/(1-\gamma)$ for any $Q \in \mathcal{B}(\mathcal{X}\times\mathcal{A})$,

$$\begin{aligned}
&\left\|Q_* - Q_{\pi_i}\right\|_\infty \\
&\leq \frac{2}{1-\gamma}\sum_{j=1}^{i}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j}\right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}| \\
&= \frac{2\gamma}{1-\gamma}\sum_{j=0}^{i-1}\frac{\gamma^j}{\alpha_{0:i}}\left\|E_{i-j-1}\right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)^2}\ln|\mathcal{A}|.
\end{aligned}$$

This concludes the proof. □

A proof of L^p-norm performance bound for ACVI-Q and Ψ is similar to that for ATD(λ) (Theorem 1.7.4). However we omit it because it is notationally very cluttered.

2.5.3 Proof of Theorem 2.2.3

Figure 2.5: A deterministic environment used to prove the asymptotic tightness of the performance bounds (2.12) in Theorem 2.2.1. This environment is taken from Bertsekas and Tsitsiklis (1996) and Scherrer and Lesner (2012) in which existing performance bounds for AVI and API are proven to be tight. There are two actions: s (stay) and m (move). Except for state 0, staying costs an agent $-r(k,s) = 2\sum_{l=0}^{k-1}\gamma^l\varepsilon$, where $\varepsilon \in (0,\infty)$ is a fixed positive real value, and l is an index of a state. At state 0, no cost is incurred. Therefore, an optimal action is m (move) at all states.

We are going to prove Theorem 2.2.3. Since the proof is lengthy, we first provide a sketch of the proof.

Proof Sketch

Consider a deterministic environment depicted in Figure 2.5. Expected immediate reward of staying at state k is given as $r(k,s) = -2\sum_{l=0}^{k-1}\gamma^l\varepsilon$, where $\varepsilon \in (0,\infty)$ is a prescribed positive real value. We assume that

- For any state k and action a, $\Psi_0(k,a) = 0$.
- For any state k and action a, $E_j(k,a) = 0$ except state $k = j+1$ and $k = j+2$ where

$$\begin{aligned}
E_j(j+1,s) &= \alpha_{0:j}\varepsilon, & E_k(k+1,m) &= -\alpha_{0:j}\varepsilon - \alpha^k\gamma\frac{1-\gamma^k}{1-\gamma}\varepsilon, \\
E_k(k+2,s) &= 0, & E_k(k+2,m) &= \alpha_{0:j}\varepsilon + \alpha^{k+1}\frac{1-\gamma^{k+1}}{1-\gamma}\varepsilon.
\end{aligned}$$

Under these assumptions, we prove that for any positive integer $i \geq 1$, (i) $q_i(i,s) = q_i(i,m)$ and (ii) $q_i(i+l,s) < q_i(i+l,m)$ for any positive integer $l \in \mathbb{Z}_{++}$. Thus, from Lemma 2.5.4, one of greedy policies with respect to q_i chooses action s (stay) at state i resulting in cumulative rewards of $-2\sum_{t=0}^{\infty}\gamma^t\sum_{j=0}^{i-1}\gamma^j\varepsilon$. We set π_i to that greedy policy. As a result,

$$\|Q_* - Q_{\pi_i}\|_\infty = Q_*(i,s) - Q_{\pi_i}(i,s) = \frac{2\gamma(1-\gamma^i)}{(1-\gamma)^2}\varepsilon$$

since $Q_*(i,s) = -2\sum_{j=0}^{i-1}\gamma^j\varepsilon$ is cumulative rewards when s is taken once at state i and m is repeatedly taken afterwards.

On the other hand, it is obvious that either

$$\|E_j\|_\infty = |E_j(j+1,m)| \ \text{ or } \ \|E_j\|_\infty = |E_j(j+2,m)|$$

holds. In any case, we have

$$\|E_j\|_\infty = \alpha_{0:j}\varepsilon + O(\alpha^j).$$

Thus, the right hand side of the performance bounds (2.12) become

$$\begin{aligned}
\text{r.h.s.} &= \frac{2\gamma\varepsilon}{1-\gamma}\sum_{j=0}^{i-1}\gamma^{i-j-1}\frac{\alpha_{0:j}}{\alpha_{0:i}} + o(1) \\
&= \frac{2\gamma\varepsilon}{1-\gamma}\sum_{j=0}^{i-1}\gamma^{i-j-1}\left(1 - \alpha^j\frac{\alpha_{0:i-j+1}}{\alpha_{0:i}}\right) + o(1) \\
&= \frac{2\gamma(1-\gamma^i)}{(1-\gamma)^2}\varepsilon - \frac{2\varepsilon}{(1-\gamma)\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\alpha^{i-j}\alpha_{0:j} + o(1).
\end{aligned}$$

The second term converges to 0. Indeed, when $0 \leq \alpha < 1$ and $\alpha \neq \gamma$,

$$0 \leq \frac{1}{\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\alpha^{i-j}\alpha_{0:j} \leq \alpha^i\sum_{j=0}^{i-1}\left(\frac{\gamma}{\alpha}\right)^j = \frac{\alpha^i - \gamma^i}{\alpha - \gamma}.$$

(When $\alpha = \gamma$, the right hand side is $i\alpha^i$ and converges to 0.) When $\alpha = 1$,

$$0 \leq \frac{1}{\alpha_{0:i}}\sum_{j=0}^{i-1}\gamma^j\alpha^{i-j}\alpha_{0:j} = \frac{1}{i+1}\sum_{j=0}^{i-1}\gamma^j(j+1) = \frac{1-\gamma^i}{(1-\gamma)^2(i+1)} - \frac{\gamma^i i}{(1-\gamma)(i+1)},$$

where the second equality is obtained as follows: let S_i denote $\sum_{j=0}^{i-1}\gamma^j(j+1)$. Because

$$\begin{aligned} S_i - \gamma S_i &= \sum_{j=0}^{i-1}\gamma^j(j+1) - \sum_{j=0}^{i-1}\gamma^{j+1}(j+1) \\ &= \sum_{j=0}^{i-1}\gamma^j(j+1) - \sum_{j=1}^{i}\gamma^j j \\ &= \sum_{j=0}^{i-1}\gamma^j - \gamma^i i, \end{aligned}$$

it follows that $S_i = \dfrac{1-\gamma^i}{(1-\gamma)^2} - \dfrac{\gamma^i i}{1-\gamma}$. As a result,

$$\lim_{i\to\infty} \text{r.h.s.} = \frac{2\gamma\varepsilon}{(1-\gamma)^2} = \lim_{i\to\infty} \left\|Q_* - Q_{\pi_i}\right\|_\infty .$$

Full Proof

By induction, we prove that for any positive integer $i \geq 1$

$$q_i(i,s) = q_i(i,m) = -\frac{1-\gamma^i}{1-\gamma}\varepsilon, \tag{2.26}$$

$$q_i(i+1,m) = \frac{\alpha_{0:i}}{\alpha_{0:i-1}}\frac{1-\gamma^i}{1-\gamma}\varepsilon, \tag{2.27}$$

$$q_i(i+l,s) < q_i(i+l,m), \tag{2.28}$$

where $l \in \{1,2,\ldots\}$.

Recall that the update rule of q_j is

$$q_j = r + \gamma\frac{\alpha_{0:j-2}}{\alpha_{0:j-1}}\boldsymbol{Pm}q_{k-1} + \frac{1}{\alpha_{0:j-1}}E_{k-1},$$

as we assume that $q_0 = \Psi_0 = 0$. For $i = 1$, as $q_0 = \Psi_0 = 0$,

$$q_1(1,s) = r(1,s) + E_0(1,s) = -\frac{1-\gamma^1}{1-\gamma}\varepsilon = r(1,m) + E_0(1,m) = q_1(1,m)$$

$$q_1(2,m) = r(2,m) + E_0(2,m) = \varepsilon + \alpha\varepsilon = \frac{\alpha_{0:1}}{\alpha}\frac{1-\gamma^1}{1-\gamma}\varepsilon$$

$$q_1(1+l,s) = r(1+l,s) + E_0(1+l,s) < 0 \leq r(1+l,m) + E_0(1+l,m) = q_1(1+l,m).$$

Therefore, (2.26), (2.27) and (2.28) hold for $i = 1$.

Suppose that (2.26), (2.27) and (2.28) hold up to $i-1$ ($i>1$). First, note that

$$\begin{aligned}
q_i(i,m) &= \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\max\{q_{i-1}(i-1,s),q_{i-1}(i-1,m)\}+\frac{1}{\alpha_{0:i-1}}E_{i-1}(i,m)\\
&= -\gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\frac{1-\gamma^{i-1}}{1-\gamma}\varepsilon-\varepsilon-\frac{\alpha^{i-1}}{\alpha_{0:i-1}}\gamma\frac{1-\gamma^{i-1}}{1-\gamma}\varepsilon\\
&= -\varepsilon-\frac{1}{\alpha_{0:i-1}}\left(\alpha_{0:i-2}+\alpha^{i-1}\right)\gamma\frac{1-\gamma^{i-1}}{1-\gamma}\varepsilon,\\
&= -\frac{1-\gamma^{i}}{1-\gamma}\varepsilon,
\end{aligned}$$

where we used $\max\{q_{i-1}(i-1,s),q_{i-1}(i-1,m)\}=q_{i-1}(i-1,s)=q_{i-1}(i-1,m)$ and $\alpha_{0:i-2}+\alpha^{i-1}=\alpha_{0:i-1}$. Next, note that

$$\begin{aligned}
q_i(i,s) &= r(i,s)+\gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\max\{q_{i-1}(i,s),q_{i-1}(i,m)\}+\frac{1}{\alpha_{0:i-1}}E_{i-1}(i,s)\\
&= -2\frac{1-\gamma^{i}}{1-\gamma}\varepsilon+\gamma\frac{1-\gamma^{i-1}}{1-\gamma}\varepsilon+\varepsilon\\
&= -\frac{1-\gamma^{i}}{1-\gamma}\varepsilon,
\end{aligned}$$

where we used $q_{i-1}(i,s)<q_{i-1}(i,m)$ to obtain $\max\{q_{i-1}(i,s),q_{i-1}(i,m)\}=q_{i-1}(i,m)$. Therefore, (2.26) holds. Furthermore,

$$\begin{aligned}
q_i(i+1,m) &= \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\max\{q_{i-1}(i,s),q_{i-1}(i,m)\}+\frac{1}{\alpha_{0:i-1}}E_{i-1}(i+1,m)\\
&= \gamma\frac{1-\gamma^{i-1}}{1-\gamma}\varepsilon+\varepsilon+\frac{\alpha^{i}}{\alpha_{0:i-1}}\frac{1-\gamma^{i}}{1-\gamma}\varepsilon\\
&= \left(1+\frac{\alpha^{i}}{\alpha_{0:i-1}}\right)\frac{1-\gamma^{i}}{1-\gamma}\varepsilon\\
&= \frac{A_{i+1}}{\alpha_{0:i-1}}\frac{1-\gamma^{i}}{1-\gamma}\varepsilon,
\end{aligned}$$

where we again used $q_{i-1}(i,s)<q_{i-1}(i,m)$ to obtain $\max\{q_{i-1}(i,s),q_{i-1}(i,m)\}=$

$q_{i-1}(i,m)$. Thus, (2.27) holds. Finally, noting that $q_{i-1}(i+l-1,s) < q_{i-1}(i+l-1,m)$,

$$\begin{aligned} q_i(i+l,m) &= \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\max\{q_{i-1}(i+l-1,s), q_{i-1}(i+l-1,m)\} + \frac{1}{\alpha_{0:i-1}}E_{i-1}(i+l,m) \\ &= \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}q_{i-1}(i+l-1,m) + \frac{1}{\alpha_{0:i-1}}E_{i-1}(i+l,m) \\ &= \gamma^2\frac{A_{i-2}}{\alpha_{0:i-1}}q_{i-2}(i+l-2,m) + \frac{1}{\alpha_{0:i-1}}\left(E_{i-1}(i+l,m) + \gamma E_{i-2}(i+l-1,m)\right) \\ &\vdots \\ &= \frac{1}{\alpha_{0:i-1}}\left(E_{i-1}(i+l,m) + \gamma E_{i-2}(i+l-1,m) + \cdots + \gamma^{i-1}E_0(l+1,m)\right). \end{aligned}$$

Because $l \geq 1$, $E_{i-1-i}(i+l-i,m) \geq 0$, and thus, $q_i(l,m) \geq 0$. On the other hand,

$$\begin{aligned} q_i(i+l,s) &= r(i+l,s) + \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}\max\{q_{i-1}(i+l,s), q_{i-1}(i+l,m)\} + \frac{1}{\alpha_{0:i-1}}E_{i-1}(i+l,s) \\ &= r(i+l,s) + \gamma\frac{\alpha_{0:i-2}}{\alpha_{0:i-1}}q_{i-1}(i+l,m) \\ &= r(i+l,s) + \frac{1}{\alpha_{0:i-1}}\left(\gamma E_{i-2}(i+l,m) + \cdots + \gamma^{i-1}E_0(l+2,m)\right). \end{aligned}$$

Because $l \geq 1$, $E_{i-2-i}(i+l-i,m) = 0$, and thus, $q_i(i+l,m) = r(i+l,s) < 0$. Therefore, (2.28) holds. Given those results,

$$\lim_{i\to\infty}\text{r.h.s.} = \frac{2\gamma\varepsilon}{(1-\gamma)^2} = \lim_{i\to\infty}\|Q_* - Q^{\rho_i}\|_\infty$$

can be shown by following the proof sketch we have provided.

2.5.4 Proof of Proposition 2.2.4

Since

$$\begin{aligned} \ln\frac{\pi_i(a|x)}{\pi_{i-1}(a|x)} &= \beta\left[\alpha_{0:i-1}q_i(x,a) - \alpha_{0:i-2}q_{i-1}(x,a)\right] \\ &\quad - \beta\left[\boldsymbol{m}_\beta\left(\alpha_{0:i-1}q_i\right)(x) - \boldsymbol{m}_\beta\left(\alpha_{0:i-2}q_{i-1}\right)(x)\right], \end{aligned}$$

we have (note that the mellowmax is a non-expansion)

$$\left\|\sum_a \pi_i(a|\cdot)\ln\frac{\pi_i(a|\cdot)}{\pi_{i-1}(a|\cdot)}\right\|_\infty \leq 2\beta\left\|\alpha_{0:i-1}q_i - \alpha_{0:i-2}q_{i-1}\right\|_\infty.$$

By definition, $\alpha_{0:i-1}q_i = \alpha_{0:i-1}r + \gamma \boldsymbol{P}\boldsymbol{m}_\beta\left(\alpha_{0:i-2}q_{i-1}\right) + E_{i-1}$. Therefore,

$$\begin{aligned}
&\left\|\alpha_{0:i-1}q_i - \alpha_{0:i-2}q_{i-1}\right\|_\infty \\
&= \left\|\alpha^{i-1}r + \gamma \boldsymbol{P}\boldsymbol{m}_\beta\left(\alpha_{0:i-2}q_{i-1}\right) - \gamma \boldsymbol{P}\boldsymbol{m}_\beta\left(A_{i-2}q_{i-2}\right) + \varepsilon_{i-1} - (1-\alpha)E_{i-2}\right\|_\infty \\
&\leq \alpha^{i-1}r_{max} + \gamma\left\|\alpha_{0:i-2}q_{i-1} - \alpha_{0:i-3}q_{i-2}\right\|_\infty + 2\varepsilon.
\end{aligned}$$

By induction, it is easy to see that

$$\begin{aligned}
&\left\|\alpha_{0:i-1}q_i - \alpha_{0:i-2}q_{i-1}\right\|_\infty \\
&\leq \gamma^{i-1}\left\|q_1\right\|_\infty + 2(1+\gamma+\cdots+\gamma^{i-2})\varepsilon + (\alpha^{i-1}+\alpha^{i-2}\gamma+\cdots+\alpha\gamma^{i-2})r_{max} \\
&\leq 2\frac{1-\gamma^i}{1-\gamma}\varepsilon + r_{max}\sum_{j=0}^{i-1}\alpha^j\gamma^{i-j-1}.
\end{aligned}$$

As a result, $\left\|\sum_a \pi_i(a|\cdot)\ln\frac{\pi_i(a|\cdot)}{\pi_{i-1}(a|\cdot)}\right\|_\infty \leq 4\beta\left(\frac{1-\gamma^i}{1-\gamma}\varepsilon + r_{max}\sum_{j=0}^{i-1}\alpha^j\gamma^{i-j-1}\right)$.

2.5.5 Proof of Theorem 2.2.6

We prove Theorem 2.2.6. A basic strategy we take is almost same as the one we used in the proof of 2.2.2.

First, we show an upper bound of difference between Q-value functions of two policies.

Lemma 2.5.9. *For any pair of policies π and μ, the maximum difference between their Q-value functions is bounded by $\sqrt{2}\gamma V_{max}\delta^{1/2}/(1-\gamma)$, where $\delta = \max_s D_{KL}\left(\pi(\cdot|s)|\mu(\cdot|s)\right)$.*

Proof. We have

$$\begin{aligned}
Q_\pi - Q_\mu &= \gamma \boldsymbol{P}_\pi Q_\pi - \gamma \boldsymbol{P}_\mu Q_\mu = \gamma \boldsymbol{P}\left(\boldsymbol{\pi}Q_\pi - \boldsymbol{\mu}Q_\pi\right) + \gamma \boldsymbol{P}_\mu\left(Q_\pi - Q_\mu\right) \\
&= \gamma\left(\boldsymbol{I} - \gamma \boldsymbol{P}_\mu\right)^{-1}\boldsymbol{P}\left(\boldsymbol{\pi}Q_\pi - \boldsymbol{\mu}Q_\pi\right).
\end{aligned}$$

Therefore,

$$\begin{aligned}
\left\|Q_\pi - Q_\mu\right\|_\infty \leq \frac{\gamma}{1-\gamma}\left\|\boldsymbol{\pi}Q_\pi - \boldsymbol{\mu}Q_\pi\right\|_\infty &\leq \frac{\gamma}{1-\gamma}\max_{x\in\mathcal{X}}\sum_{a\in\mathcal{A}}\left|\left(\pi(a|x) - \mu(a|x)\right)Q_\pi(x,a)\right| \\
&\leq \frac{\gamma}{1-\gamma}V_{max}\max_{x\in\mathcal{X}}\sum_{a\in\mathcal{A}}\left|\pi(a|x) - \mu(a|x)\right|,
\end{aligned}$$

where the last inequality follows from Hölder's inequality and $\left\|Q_\pi\right\|_\infty \leq V_{max}$. By Pinsker's inequality, $\max_s\sum_a\left|\pi(a|s) - \mu(a|s)\right| \leq \sqrt{2}\delta^{1/2}$. In the consequence

$$\left\|Q_\pi - Q_\mu\right\|_\infty = \sqrt{2}\gamma\omega V_{max}\delta^{1/2}$$

□

The following lemma gives us a different upper bound for $Q^{\pi_i} - \psi_{K+1}$.

Lemma 2.5.10. *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. Let δ_i be an upper bound of $\max_x D_{KL}(\pi_i(\cdot|x)|\pi_{i-1}(\cdot|x))$. The following lower bound for $Q_{\pi_i} - q_{i+1}$ holds for any non-negative integer i:*

$$\begin{aligned} &Q_{\pi_i} - q_{i+1} \\ &\geq -\frac{1}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j \boldsymbol{P}_{i,i-j+1}E_{i-j} - \frac{\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} - \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma}\sum_{j=0}^{i-1}\gamma^j\frac{\alpha_{0:i-j-1}}{\alpha_{0:i}}\delta_{i-j}^{1/2}, \end{aligned} \tag{2.29}$$

where $\sum_{j=1}^{0} Q_j$ means a constant function whose value is 0 for any sequence of functions Q_j, and

$$\boldsymbol{P}_{i,i-j+1} := \begin{cases} \boldsymbol{I} & \text{for } j = 0 \\ \boldsymbol{P}_{\pi_i}\boldsymbol{P}_{\pi_{i-1}}\cdots\boldsymbol{P}_{\pi_{i-j+2}}\boldsymbol{P}_{\pi_{i-j+1}} & \text{for } 1 \leq j \leq i \end{cases}$$

Proof. For any non-negative integer $i \geq 0$,

$$\begin{aligned} Q_{\pi_i} - q_{i+1} &= \gamma\boldsymbol{P}_{\pi_i}Q_{\pi_i} - \gamma\boldsymbol{P}_{\mu_i}\left(\frac{\alpha_{0:i-1}}{\alpha_{0:i}}q_i + \frac{\alpha^i}{\alpha_{0:i}}q_0\right) - \frac{E_i}{\alpha_{0:i}} \\ &\geq \gamma\frac{\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\left(Q^{\pi_{i-1}} - q_i\right) - \frac{E_i}{\alpha_{0:i}} - \frac{\gamma V_{max}}{\alpha_{0:i}}\alpha^i + \gamma\frac{\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\left(Q_{\pi_i} - Q^{\pi_{i-1}}\right) \\ &\geq \gamma\frac{\alpha_{0:i-1}}{\alpha_{0:i}}\boldsymbol{P}_{\pi_i}\left(Q^{\pi_{i-1}} - q_i\right) - \frac{E_i}{\alpha_{0:i}} - \frac{\gamma V_{max}}{\alpha_{0:i}}\alpha^i - \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma}\frac{\alpha_{0:i-1}}{\alpha_{0:i}}\delta_i^{1/2}. \end{aligned}$$

(The first and last term disappear if $i = 0$.) It is clear that the claim holds for $i = 0$. It is not difficult to prove the claim by induction with the aid of the above inequality. □

By combining Lemmas 2.5.6 and 2.5.10, the following proposition is obtained.

Proposition 2.5.11. *Suppose sequences of policies $(\pi_i)_{i\in\mathbb{Z}_+}$ in the update (2.9), $(\mu_i)_{i\in\mathbb{Z}_+}$ in (2.11) and functions $(q_i)_{i\in\mathbb{Z}_+}$ defined in Lemma 2.5.4. Let δ_i denote an upper bound of $\max_x D_{KL}(\pi_i(\cdot|x)|\pi_{i-1}(\cdot|x))$. The following point-wise upper bound for $Q_* - Q_{\pi_i}$ holds for any non-negative integer K:*

$$\begin{aligned} Q_* - Q_{\pi_i} \leq \frac{1}{\alpha_{0:i}}\sum_{j=1}^{i}\gamma^j\left(\boldsymbol{P}_{i,i-j+1}E_{i-j} - (\boldsymbol{P}_*)^j E_{i-j}\right) + \frac{2\gamma V_{max}}{\alpha_{0:i}}\sum_{j=0}^{i}\gamma^j\alpha^{i-j} \\ + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)}\ln|\mathcal{A}| + \frac{\sqrt{2}\gamma^2}{1-\gamma}V_{max}\sum_{j=0}^{i-1}\gamma^j\frac{\alpha_{0:i-j-1}}{\alpha_{0:i}}\delta_{K-k}^{1/2}, \end{aligned} \tag{2.30}$$

where $\boldsymbol{P}_{i,i-j+1}$ are defined in Lemma 2.5.10, and $\sum_{j=1}^{0} Q_j$ means a constant function whose value is 0 for any sequence of functions Q_j.

Now we prove Theorem 2.2.5. From Proposition 2.5.11 and by noting that $|Q_*(x,a)-$

$$Q_{\pi_i}(x,a)| = Q_*(x,a) - Q_{\pi_i}(x,a),$$

$$\begin{aligned}
&\|Q_* - Q_{\pi_i}\|_\infty \\
&= \max_{x,a} \left(Q_* - Q_{\pi_i}\right)(x,a) \\
&= \max_{x,a} \left(Q_* - q_{i+1} - \left(Q_{\pi_i} - q_{i+1}\right)\right)(x,a) \\
&\leq \max_{x,a \in \mathcal{X}\times\mathcal{A}} \sum_{j=1}^{i} \frac{\gamma^j}{\alpha_{0:i}} \left(\boldsymbol{P}_{i,i-j+1} E_{i-j} - \left(\boldsymbol{P}_*\right)^j E_{i-j}\right)(x,a) \\
&\qquad + \frac{2\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} + \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln|\mathcal{A}| + \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma} \sum_{j=0}^{i-1} \gamma^j \frac{\alpha_{0:i-j-1}}{\alpha_{0:i}} \delta_{i-j}^{1/2}.
\end{aligned}$$

Because $\|\boldsymbol{P}_{i,i-j+1} Q\|_\infty \leq \|Q\|_\infty$ for any $Q \in \mathcal{Q}$,

$$\begin{aligned}
\|Q_* - Q_{\pi_i}\|_\infty \leq 2\gamma \sum_{j=1}^{i} \gamma^j \left\| \frac{E_{i-j}}{\alpha_{0:i}} \right\|_\infty + \frac{2\gamma V_{max}}{\alpha_{0:i}} \sum_{j=0}^{i} \gamma^j \alpha^{i-j} \\
+ \frac{\gamma(1-\gamma^i)}{\alpha_{0:i}\beta(1-\gamma)} \ln|\mathcal{A}| + \frac{\sqrt{2}\gamma^2 V_{max}}{1-\gamma} \sum_{j=0}^{i-1} \gamma^j \frac{\alpha_{0:i-j-1}}{\alpha_{0:i}} \delta_{i-j-1}^{1/2}.
\end{aligned}$$

Loosening it by replacing $\alpha_{0:i-j-1}/\alpha_{0:i}$ with 1, we conclude the proof of Theorem 2.2.5.

A proof of L^p-norm performance bounds for ACVI-Q and Ψ is similar to that for ATD(λ) (Theorem 1.7.4). However we omit it because it is notationally very cluttered.

Chapter 3

Noise-Tolerant Policy Evaluation via Gap-Increasing Operator

As explained in Section 1.6, policy evaluation is a key problem in RL because two of the most fundamental algorithms called AC and PI require a value function for policy improvement (Sutton and Barto, 2018). As examples, recent popular deep RL algorithms called deep deterministic policy gradient (DDPG), Actor-Critic with Experience Replay (ACER), Asynchronous Advantage Actor-Critic (A3C) are based on AC (Lillicrap et al., 2016; Wang et al., 2016; Mnih et al., 2016). However, current policy evaluation algorithms are unsatisfactory since they are either inefficient or prone to noise originating from stochastic rewards and state transition.

For example, a multi-stage lookahead algorithm called R(λ) is efficient in that it is off-policy, uses low-variance updates thanks to truncated importance sampling ratios, and allows control of bias-variance trade-off (Munos et al., 2016). It has achieved state-of-the-art performance on different kinds of RL tasks (Wang et al., 2016). However, R(λ) is prone to noise, as shown later by error propagation analysis (Section 3.1.1) and experiments (Section 3.1.2).

A simple approach to handle noise is to use partial updates using a learning rate (Sutton and Barto, 2018). We call such an approach Learning-Rate-based (LR-based). As we argue in Section 3.2, its learning is unsatisfactorily slow.

To maintain both noise-tolerance and learning efficiency, we propose a new policy evaluation algorithm, called GRAPE, combining R(λ) and gap-increasing operator (explained later). Theoretical analysis shows that GRAPE is noise-tolerant without significantly sacrificing learning speed and efficiency of R(λ). The theoretical analysis also includes a comparison of GRAPE to R(λ) with a learning rate, which emphasizes GRAPE's capacity to learn faster than R(λ) with a learning rate. Finally, we demonstrate experimentally that our algorithm outperforms R(λ) in noisy environments.

The following is a list contributions of the present chapter.

- Proposing a new multi-stage lookahead off-policy policy evaluation algorithms, GRAPE and its variant called RGRAPE, based on gap-increasing operators.

- Providing error propagation analysis of the new algorithms that elucidates their noise-tolerance and faster convergence than the LR-based approach.

- Providing preliminary experimental results on the new algorithms that support our theoretical argument.

This chapter is organized as follows: in Section 3.1, we explain R(λ). Furthermore we provide error propagation analysis of R(λ) in Section 3.1.1, which implies R(λ)'s proneness to noise. We confirm the theoretical argument by a simple experiment in Section 3.1.2. Section 3.2 explains that a simple approach to noise by partial value updates with a learning rate causes unsatisfactorily slow learning. To overcome this issue, we propose GRAPE in Section 3.3 and motivate it by a intuitive argument Section 3.3.1. In Section 3.1.1, we provide error propagation analysis of GRAPE justifying the intuition on GRAPE. In Section 3.4.1, we explain a practical implementation of GRAPE. In Section 3.5, we provide experimental results on GRAPE. In Section 3.5.1, policy evaluation performance of GRAPE is compared to that of R(λ). In Section 3.5.2, performance of GRAPE combined with a variant of TRPO is compared to that of R(λ).

3.1 Retrace and Approximate Retrace

In addition to TD(λ), many policy evaluation algorithms have been proposed (Sutton and Barto, 2018). Retrace (R(λ)) algorithm described below provides a unified view of them (Munos et al., 2016).

Suppose a target policy π, the Q-value function of which we want to estimate, and behavior policy μ, with which data are collected. Let $\rho(x,a)$ denote the importance sampling ratio $\pi(a|x)/\mu(a|x)$, which is assumed to be well-defined. An operator $\boldsymbol{P}_{\mu d}$: $\mathcal{B}(\mathcal{X}\times\mathcal{A}) \to \mathcal{B}(\mathcal{X}\times\mathcal{A})$ is defined such that

$$(\boldsymbol{P}_{\mu d}Q)(x,a) := \mathbb{E}^{\mu}[d(X_1,A_1)Q(X_1,A_1)|X_0=x,A_0=a],$$

where d is a real-valued Borel-measurable function from $\mathcal{X}\times\mathcal{A}$ to $[0,\rho(x,a)]$. Munos et al. (2016) has shown that an operator $\boldsymbol{R}^{\lambda}_{\mu d}$ shown below is a contraction around Q_π:

$$Q_{i+1} := \boldsymbol{R}^{\lambda}_{\mu d}Q_i := Q_i + \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_{\mu d}\right)^{-1}\left(\boldsymbol{B}_\pi Q_i - Q_i\right), \tag{3.1}$$

where $\lambda\in[0,1]$, and $\left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_{\mu d}\right)^{-1} := \sum_{t=0}^{\infty}\lambda^t\gamma^t\left(\boldsymbol{P}_{\mu d}\right)^t$. Thus Q_i uniformly converges to Q_π by Banach's fixed point theorem.[1]

Depending on the function d, various algorithms are reconstructed. For example, tree-backup is obtained when $d(x,a)=\pi(a|x)$, while TD(λ) with importance sampling is obtained when $d(x,a)=\rho(x,a)$ (Precup et al., 2000). Particularly, Munos et al. (2016) proposed $d(x,a)=\min\{1,\rho(x,a)\}$ and called the resultant algorithm R(λ). When we mean this choice of d, we use c to differentiate from other choices.

We are more interested in analyzing approximate version of R(λ)

$$Q_{i+1} := \boldsymbol{R}^{\lambda}_{\mu c}Q_i := Q_i + \left(\boldsymbol{I}-\gamma\lambda\boldsymbol{P}_{\mu c}\right)^{-1}\left(\boldsymbol{B}_\pi Q_i - Q_i\right) + \varepsilon_i, \tag{3.2}$$

[1] The following generalization of R(λ) works too as $\|Q_\pi - Q_{i+1}\|_\infty \le \gamma\|Q_\pi - Q_i\|_\infty$ holds: $Q_{i+1} := \sum_{j=1}^{J} p_j \boldsymbol{R}^{\lambda}_{\mu_j c}Q_i$, where μ_j is j-th behavior policy, and $p_j\in[0,1], \sum_j p_j = 1$. This generalization fits better to a case wherein data is collected with multiple policies.

where $\varepsilon_i \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$. We call this algorithm Approximate Retrace (AR(λ)). As is the case with other ADP algorithms, we frequently omit the qualifier "approximate".

3.1.1 Error Propagation Analysis of Retrace

Munos et al. (2016) has proven the convergence of exact R(λ). We aim at proving the following theorem that generalizes their result.

Theorem 3.1.1 (An L^∞-Norm Error Bound for AR(λ))**.** *For AR(λ) in Equation (3.2), the following holds:*

$$\|Q_\pi - Q_i\|_\infty \leq \gamma^i \|Q_\pi - Q_0\|_\infty + \textstyle\sum_{j=0}^{i-1} \gamma^j \|\varepsilon_{i-j-1}\|_\infty . \tag{3.3}$$

Remark 3.1.1. *As argued in Remark 1 of (Munos et al., 2016), a contraction modulus (γ in Theorem 3.1.1) of R(λ) is smaller when π and μ are close. In other words, γ is the worst-case modulus.*

Remark 3.1.2. *It is possble to exchanging $\|Q_\pi - Q_i\|_\infty$ on the left hand side with $\|V_\pi - \boldsymbol{\pi} Q_i\|_\infty$ or $\|A_\pi - (Q_i - \boldsymbol{\pi} Q_i)\|_\infty$. In the former case, use that $\|V_\pi - \boldsymbol{\pi} Q_i\|_\infty = \|\boldsymbol{\pi}(Q_\pi - Q_i)\|_\infty \leq \|Q_\pi - Q_i\|_\infty$. In the latter case, use that $\|A_\pi - (Q_i - \boldsymbol{\pi} Q_i)\|_\infty \leq \|V_\pi - \boldsymbol{\pi} Q_i\|_\infty + \|Q_\pi - Q_i\|_\infty \leq 2\|Q_\pi - Q_i\|_\infty$.*

The lemma below can be proven in a way similar to Lemma 1.7.1. Combining this lemma and the fact that $\|\boldsymbol{N} Q\|_\infty \leq \|Q\|_\infty$ proven by Munos et al. (2016), Theorem 3.1.1 is proven in a way similar to Corollary 1.7.2.

Lemma 3.1.2 (A Point-Wise Error Bound for AR(λ))**.** *For AR(λ) in Equation (3.2), the following holds:*

$$Q_\pi - Q_i = \gamma^i \boldsymbol{N}^i (Q_\pi - Q_0) - \textstyle\sum_{j=0}^{i-1} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j-1},$$

where $\boldsymbol{N}$ is an operator from $\mathcal{B}(\mathcal{X} \times \mathcal{A})$ to itself defined by $(\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_{\mu c})^{-1} (\boldsymbol{P}_\pi - \lambda \boldsymbol{P}_{\mu c})$.

As shown in the following proposition, the error bound (3.3) is not improvable. It can be proven by setting π and μ such that they have disjoint supports.

Proposition 3.1.3 (Tightness of the L^∞-Norm Error Bound (3.3) for AR(λ))**.** *The L^∞-Norm Error Bound (3.3) for AR(λ) is tight meaning that there exists an MDP, a pair of policies μ, π and a sequence $(\varepsilon_i)_{i \in \mathbb{Z}_+}$ of error functions such that the error bound holds with equality.*

The error bound (3.3) implies AR(λ)'s proneness to noise. Indeed the right hand side is a sum of L^∞-norm of error functions. A simple experiment in Section 3.1.2 confirms the theoretical analysis.

3.1.2 Retrace's Proneness to Noise

The error bound (3.3) shows that $\|Q_\pi - Q_i\|_\infty$ is governed by a discounted sum of L^∞-norm of error functions. Therefore the effect of errors accumulates. Given this result,

a natural question is whether we can do better if error functions satisfy a certain condition.

We are interested in a model-free case, in which $\boldsymbol{R}^{\lambda}_{\mu c} Q_i$ is estimated based on samples obtained through interactions with the environment. In model-free case, it is expected that error functions $\varepsilon_j \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$ satisfy

$$\frac{1}{\sum_{j=0}^{k} \alpha^{i-k}} \left| \sum_{j=0}^{i} \alpha^{i-j} \varepsilon_j(x,a) \right| \approx 0 \tag{3.4}$$

for sufficiently large $\alpha \in (0,1]$ and $i \in \mathbb{Z}_+$. We call such error functions as *noisy error functions* (or simply noises). The error bound (3.3) shows that $\mathrm{R}(\lambda)$ is unfortunately not able to handle noises.

In order to confirm whether $\mathrm{R}(\lambda)$ indeed suffers from noises, we have carried out a simple experiment with DP updates in a environment called 8×8 FrozenLake (see Figure 3.1) as explained in Experiment 3.1.1. In Section 3.5, we present experimental results in more realistic model-free case.

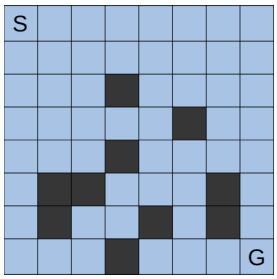


Figure 3.1: 8×8 FrozenLake. Blue grids are slippery but safe states, while black grids are terminal states with no reward. An agent obtains a reward 1 when it reaches to a goal, G, (bottom right) from a start state S (top left). At a slippery state, an action to go one direction (e.g., left) results in, with equal probability, going to one of directions except its opposite (e.g., right).

Experiment 3.1.1 (8×8 FrozenLake)**.** *This experiment for AR(λ) is done as follows: first, μ and π are sampled from a Dirichlet distribution with concentration parameters all set to 1. From those policies, matrices $\boldsymbol{P}_{\mu c}$ and $\boldsymbol{P}_{\pi}$ are constructed. Using $\boldsymbol{P}_{\mu c}$, $\boldsymbol{P}_{\pi}$ and an expected reward function r, a function Q_{i+1} is computed as a sum of $\boldsymbol{R}^{\lambda}_{\mu c} Q_i$ and Gaussian noise $\varepsilon_i(x,a) \sim N(0,\sigma)$. The discount factor γ and λ are set to 0.99 and $\lambda = 0.8$, respectively. Similar results are obtained regardless of their values. The standard deviation $\sigma \in \{0.0, 0.2, 0.4, 0.6, 0.8\}$ is varied to investigate the effect of noise intensity. An initial function is $Q_0(x,a) \sim N(0,1)$. The experiment for AR(λ) with a learning rate (explained in Section 3.2) is done similarly except that Q_{i+1} is computed as $\eta \boldsymbol{R}^{\lambda}_{\mu c} Q_i + (1-\eta) Q_i + \eta \varepsilon_i$.*

To measure the performance of $\mathrm{AR}(\lambda)$, we used Normalized Mean Squared Error (NMSE). Let e_i be

$$e_i := \frac{1}{|\mathcal{X} \times \mathcal{A}|} \sum_{(x,a) \in \mathcal{X} \times \mathcal{A}} \left(A_\pi(x,a) - A_i(x,a) \right)^2, \tag{3.5}$$

where $A_i(x,a) := Q_i(x,a) - \sum_{a \in \mathcal{A}} \pi(a|x) Q_i(x,a)$. NMSE is defined by e_i / e_0. The division by e_0 is to simply remove effects of initialization. The reason why we use A_π and A_i is as follows: first note that adding a state-dependent function to Q_i makes no difference on $A_\pi - A_i$, whereas it of course does on $Q_\pi - Q_i$; given that policy updates

are insensitive to the addition of a state-dependent function, this performance measure is more appropriate than the one computed based on $Q_\pi - Q_i$.

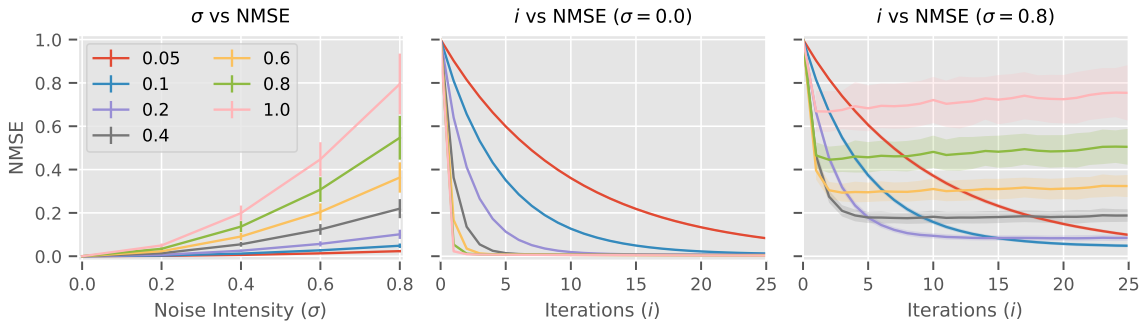


Figure 3.2: DP experiment results in 8×8 FrozenLake. Experimental results with R(λ) (left panel) and R(λ) with a learning rate (right panel) using DP updates. For details, see Experiment 3.1.1. Lines indicate the median of NMSE (lower is better) over 100 experiments, and the shaded area shows the 95 percentile. Colors indicate noise intensity σ. Note that the vertical axis is in log-scale.

The left panel of Figure 3.2 visualizes asymptotic performance (after $2,000$ iterations) of AR(λ) measured by NMSE with varying noise intensity. AR(λ)'s result corresponds to the pink line, i.e., the one indicated by the legend as 1.0. (Other lines correspond to results of AR(λ) with a learning rate explained later.) The middle and right panels show learning curves when σ is 0.0 and 0.8, respectively. In the middle panel NMSE quickly decreases to 0 after 1 or 2 iterations, whereas in the right panel NMSE gradually increases after a small quick decrease. These results show that as the noise intensity σ increases, AR(λ)'s performance quickly degrades. In other words, it indeed suffers from noises.

3.2 Slow Learning Due to a Learning Rate

As we have discussed now, AR(λ) is prone to noises. A simple approach to handle noise is to use a learning-rate. We call such an approach LR-based. For example, the update rule of TD(λ) with $\lambda = 0$ and a learning rate is given by

$$Q_{i+1} := \eta_i \odot \boldsymbol{B}_\pi Q_i + (1 - \eta_i) \odot Q_i, \tag{3.6}$$

where $\eta_i : \mathcal{X} \times \mathcal{A} \to [0, 1]$ is a learning rate, and $\odot$ is element-wise multiplication, i.e., $((1 - \eta_i) \odot Q_i)(x, a) = (1 - \eta_i(x, a))\, Q_i(x, a)$. This generalized notion of a learning rate is frequently used in theoretical analysis (Bertsekas and Tsitsiklis, 1996; Singh et al., 2000; Even-Dar and Mansour, 2004). It includes various algorithms such as the online TD(0) when $k = t$, and $\eta_i(x, a) \neq 0$ iff x and a is visited at time t.

The LR-based approach attains, as expected, noise-tolerance. For simplicity, let us assume that $Q_0(x, a) = 0$ and $\eta_i(x, a) = \eta \in (0, 1]$ for all k and state-action pairs

$(x, a) \in \mathcal{X} \times \mathcal{A}$. Let us suppose that due to noise, the update (3.6) becomes

$$Q_{i+1} := \eta \left(\boldsymbol{B}_\pi Q_i + \varepsilon_i \right) + (1 - \eta) Q_i = \eta \sum_{j=0}^{i} (1 - \eta)^j \left(\boldsymbol{B}_\pi Q_{i-j} + \varepsilon_{i-j} \right). \tag{3.7}$$

Because $Q_\pi = \eta \sum_{j=0}^{i} (1-\eta)^j Q_\pi + (1-\eta)^{i+1} Q_\pi$,

$$\left\| Q_\pi - Q_{i+1} \right\|_\infty \leq \eta\gamma \sum_{j=0}^{i} (1 - \eta)^j \left\| Q_\pi - Q_{i-j} \right\|_\infty + \left\| \eta E_i \right\|_\infty + (1 - \eta)^{i+1} V_{max},$$

where $E_i := \sum_{j=0}^{i} (1-\eta)^j \varepsilon_{i-j}$. By induction, it is easy to show that an upper bound of the right hand side consists of $\left\| \eta E_j \right\|, j \in [0 : i]$ and some constant. As $\eta E_j(s, a)$ is an exponentially weighted average of noises $\varepsilon_{i-j}(s, a)$, it is expected that $\eta E_j \approx 0$ for a sufficiently small learning rate η.

Note that the update rule (3.7) can be rewritten as $Q_{i+1} = \eta \sum_{j=0}^{i} \boldsymbol{\Gamma}^j r$, where $\boldsymbol{\Gamma} := (1 - \eta) \boldsymbol{I} + \eta\gamma \boldsymbol{P}_\pi$. Using it, we can derive the convergence rate of LR-based approach. As $Q_\pi = \eta \sum_{j=0}^{i} \boldsymbol{\Gamma}^j r + \boldsymbol{\Gamma}^{i+1} Q_\pi$, we can deduce that

$$\left\| Q_\pi - Q_{i+1} \right\| \leq \left(1 - \eta(1 - \gamma) \right)^{i+1} V_{max}. \tag{3.8}$$

Because this upper bound holds with equality when $\boldsymbol{P}_\pi$ is an identity operator $\boldsymbol{I}$, it is not improvable. (For example, $\boldsymbol{P}_\pi = \boldsymbol{I}$ when an environment has only one state and action). Therefore, the convergence rate is $O((1-\eta(1-\gamma))^K)$. Considering that $\gamma \approx 1$ and $\eta \approx 0$ in many cases, $1-\eta(1-\gamma)$ is close to 1. Thus, the LR-based is noise-tolerant at the sacrifice of learning efficiency.

To confirm this argument, we conducted a simple experiment using DP updates of AR(λ) with a learning rate. (In Section 3.5, we present experimental results in more realistic model-free case.) The left panel of Figure 3.2 visualizes asymptotic performance (after 2,000 iterations) with varying noise intensity. Learning rates used are indicated by different colors as in the legend. The middle and right panel show learning curves when σ is 0.0 and 0.8, respectively. In those panels, as the learning rate decreases, the decay of NMSE slows, while the final result becomes better. These results illustrate the tolerance of the LR-based to noise as well as its unsatisfactorily slow learning.

3.3 Gap-Increasing Operators for Policy Evaluation

In Section 3.2 we discussed noise-tolerance of the LR-based at the sacrifice of learning efficiency. Is it possible to tame noise while maintaining efficiency? In this section, we affirmatively answer the question with a gap-increasing policy evaluation algorithm, called Gap-increasing RetrAce Policy Evaluation (GRAPE), inspired by single-stage lookahead control algorithms called AL and DPP (Baird III, 1999; Azar et al., 2012; Rawlik, 2013; Bellemare et al., 2016), which are shown to be noise-tolerant (Azar et al., 2012; Kozuno et al., 2019).

Approximate AL has the following update rule (Bellemare et al., 2016):

$$Q_{i+1} := \boldsymbol{B}Q_i + \alpha\left(Q_i - \boldsymbol{m}Q_i\right) + \varepsilon_i, \tag{3.9}$$

where an initial function Q_0 is an element of $\mathcal{B}(\mathcal{X}\times\mathcal{A})$, and $\alpha \in [0,1]$ is a coefficient of the advantage term $Q_i - \boldsymbol{m}Q_i$. As we prove later in Chapter 2, the sequence $(Q_i)_{i\in\mathbb{Z}_+}$ uniformly converges to $V_* + \frac{1}{1-\alpha}A_*$ in exact case. As $A_* = Q_* - V_*$ is value differences of actions, we call it action-gaps. The coefficient α of the advantage term controls "gap-increasingness" of the operator $Q \to \boldsymbol{B}Q + \alpha\left(Q_i - \boldsymbol{m}Q_i\right)$.

We have shown in (Kozuno et al., 2019) that not only AL but also other control algorithms using gap-increasing operators are noise-tolerant. Thus it is expected that policy evaluation algorithms using gap-increasing operators have noise-tolerance too.

Being inspired by gap-increasing operators, we propose RGRAPE. Suppose target and behavior policies π, μ and two positive real numbers $\alpha, \lambda \in [0,1]$. RGRAPE's update rule is the following:

$$Q_{i+1} := \boldsymbol{B}_\pi Q_i + \gamma\lambda\left(\boldsymbol{I} - \gamma\lambda\boldsymbol{P}_{\mu c}\right)^{-1}\boldsymbol{P}_{\mu c}\left(\boldsymbol{B}_\pi Q_i - Q_i + \alpha A_{i-1}\right) + \alpha A_i + \varepsilon_i, \tag{3.10}$$

where A_i is defined as $Q_i - \boldsymbol{\pi}Q_i$ for $i \notin \{-1, 0\}$ and a constant function taking 0 otherwise. Note that A_{i-1} is used in $\gamma\lambda\left(\boldsymbol{I} - \gamma\lambda\boldsymbol{P}_{\mu c}\right)^{-1}\boldsymbol{P}_{\mu c}\left(\boldsymbol{B}_\pi Q_i - Q_i + \alpha A_{i-1}\right)$, while A_i is used at the second to the last term. This non-trivial subtlety, which is difficult to predict from the update rule of AL, seems to be essential for theoretically proving noise-tolerance.

A slightly different form of RGRAPE with the following update is also possible:

$$Q_{i+1} := \boldsymbol{B}_\pi Q_i + \gamma\lambda\left(\boldsymbol{I} - \gamma\lambda\boldsymbol{P}_{\mu c}\right)^{-1}\boldsymbol{P}_{\pi}\left(\boldsymbol{B}_\pi Q_i - Q_i\right) + \alpha A_i + \varepsilon_i, \tag{3.11}$$

We call this variant as GRAPE. Note that $\boldsymbol{P}_\pi$ is used in $\gamma\lambda\left(\boldsymbol{I} - \gamma\lambda\boldsymbol{P}_{\mu c}\right)^{-1}\boldsymbol{P}_{\pi}$ rather than $\boldsymbol{P}_{\mu c}$. As a result, A_{i-1} disappears.

The price to pay for the implementation simplicity of GRAPE is the use of an importance sampling ratio in $\boldsymbol{P}_\pi$ instead of truncated importance sampling ratio $c(x,a) = \min\{1, \pi(a|x)/\mu(a|x)\}$. Because an importance sampling ratio is used only at one time step, we expect no significant difference between GRAPE and RGRAPE. Accordingly we have carried out experiments with only GRAPE and presented results in Section 3.5.

3.3.1 Motivation for the Gap-Increasing Approach

In Section 3.4, we provide error propagation analysis of GRAPE and RGRAPE. Before that, we intuitively explain why the gap-increasing approach may work well.

The update rule (3.7) reveals that the basic idea of the LR-based approach is mitigating the effect of noises by taking exponentially weighted sum of $\boldsymbol{B}_\pi Q_{i-j} + \varepsilon_{i-j}$. If $\boldsymbol{B}_\pi Q_{i-j}$ were equal to $\boldsymbol{B}_\pi^{i-j} Q_0$ (ignoring errors), such approach would yield a fast but noise-tolerant algorithm.

Concretely the following algorithm will have a faster convergence while keeping

noise-tolerance similar to that of the LR-based approach:

$$\kappa'_{i+1} := \boldsymbol{B}_\pi \kappa'_i + \varepsilon_i = \sum_{j=0}^{i} \gamma^j \boldsymbol{P}_\pi^j \left(r + \varepsilon_{i-j}\right) + \gamma^{i+1} \boldsymbol{P}_\pi^{i+1} \kappa'_0 \tag{3.12}$$

$$\kappa_{i+1} = \frac{1}{\sum_{k=0}^{i} \alpha^k} \sum_{j=0}^{i} \alpha^j \kappa'_{i-j+1} = \frac{1}{\sum_{k=0}^{i} \alpha^k} \sum_{j=0}^{i} \alpha^j \left(\boldsymbol{B}_\pi \kappa'_{i-j} + \varepsilon_i\right), \tag{3.13}$$

where κ_{i+1} is actually used an estimate of Q_π, and $\alpha \in [0, 1]$. However it is cumbersome to store both κ'_i and κ_i. It turns out that GRAPE and RGRAPE are almost equivalent to this algorithm while storing only Q_i.

Suppose that $\lambda = 0$ in the update rules of RGRAPE (3.10) and GRAPE (3.11) for simplicity. To see the equivalence, note that the advantage term A_{i-1} in $\boldsymbol{B}_\pi Q_i = \boldsymbol{B}_\pi(\boldsymbol{B}_\pi Q_{i-1} + \varepsilon_{i-1} + \alpha A_{i-1})$ will disappear. Thus we deduce that

$$\boldsymbol{B}_\pi Q_i + \varepsilon_i = \sum_{j=0}^{i} \gamma^j \boldsymbol{P}_\pi^j \left(r + \varepsilon_{i-j}\right) + \gamma^{i+1} \boldsymbol{P}_\pi^{i+1} Q_0,$$

which is in the same form as that of κ'_{i+1}. Furthermore the update rules (3.10) and (3.11) can be rewritten as

$$Q_{i+1} := \boldsymbol{B}_\pi Q_i + \alpha A_i + \varepsilon_i = \sum_{j=0}^{i} \alpha^j \left(\boldsymbol{B}_\pi Q_{i-j} + \varepsilon_{i-j}\right) + \alpha^{i+1} Q_0 - \boldsymbol{\pi} \sum_{j=0}^{i} \alpha^{j+1} Q_{i-j}.$$

Because $\boldsymbol{\pi} \sum_{j=0}^{i} \alpha^{j+1} Q_{i-j}$ is not important for policy improvement, we may ignore it. As a result we can see that $Q_{i+1} / \sum_{j=0}^{i} \alpha^j$ is essentially κ_{i+1}.

From this result, we can understand that GRAPE and RGRAPE are almost equivalent to the algorithm with the update rules (3.12) and (3.13), and hence, we can expect that the gap-increasing approach works better than the LR-based approach.

3.4 Error Propagation Analysis of GRAPE and RGRAPE

In this Section 3.4, we carry out error propagation analysis of RGRAPE. As analysis of RGRAPE is notationally simpler but almost same as that of GRAPE, we omit analysis of GRAPE. However note that similar results hold for GRAPE too. For readability, all proofs are deferred to Section 3.8.

We begin with some shorthand notations. First we define

$$E_i := \sum_{j=0}^{i} \alpha^j \varepsilon_{i-j} \tag{3.14}$$

for all i. Furthermore we use a shorthand notation

$$\alpha_{j:i} := \begin{cases} \sum_{k=j}^{i} \alpha^k & \text{if } i \geq j \\ 0 & \textit{otherwise} \end{cases} \tag{3.15}$$

for two integers $i, j \in \mathbb{Z}$. We also recall that $\boldsymbol{N}$ is an operator from $\mathcal{B}(\mathcal{X} \times \mathcal{A})$ to itself defined by $(\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_{\mu c})^{-1} (\boldsymbol{P}_\pi - \lambda \boldsymbol{P}_{\mu c})$ (see Lemma 3.1.2).

As for the convergence of GRAPE and RGRAPE, the following theorem holds.

Theorem 3.4.1 (Uniform Convergence of Exact GRAPE and RGRAPE). *When there are no errors, the following holds for GRAPE and RGRAPE:*

$$\lim_{i\to\infty} \frac{1}{\alpha_{0:i-1}} A_i = A^\pi \text{ and } \lim_{i\to\infty} \frac{1}{\alpha_{0:i-1}} Q_i = A^\pi + (1-\alpha)V^\pi,$$

where the convergence is uniform. Moreover, the convergence rates of GRAPE and RGRAPE are by $O(\sum_{j=0}^{i} \delta^{i-j}\alpha^j/\alpha_{0:i})$ *and* $O(\sum_{j=0}^{i} \gamma^{i-j}\alpha^j/\alpha_{0:i})$*, respectively, where* $\delta = \gamma\,(1 - \lambda\,(1-\gamma))$.

Interestingly, while a fixed point of previous policy evaluation algorithms is Q_π, GRAPE's fixed point is $V_\pi(x) + A_\pi(x,a)/(1-\alpha)$ when $\alpha \neq 1$. Thus in GRAPE, A_π is enhanced by a factor of $1/(1-\alpha)$. This is the reason why we call GRAPE as *gap-increasing* Retrace; Q-value differences, or action-gaps, are increased. In case of AL, its fixed point is $V_*(x) + A_*(x,a)/(1-\alpha)$, which is indicative of the point to which GRAPE converges.

This gap-increasing property might be beneficial when RL is applied to a system operating at a fine time scale, as argued in (Baird III, 1999; Bellemare et al., 2016). Briefly, in such a situation, changes of states caused by an action at one time step are small. Consequently, so are action-gaps. Hence, a function approximator combined with a previous policy evaluation algorithm mainly approximates V_π rather than A_π (because it tries to minimize error between $Q_\pi = V_\pi + A_\pi$ and an estimated Q-value function). However, A_π is the one truly required to improve a policy.

When updates are not exact, the following L^∞-norm error bounds for GRAPE and RGRAPE hold.

Theorem 3.4.2 (An L^∞-Norm Error Bound for RGRAPE). *Recall* δ *defined in Theorem 3.4.1. For GRAPE, the following holds:*

$$\left\| A_\pi - \frac{1}{\alpha_{0:i-1}} A_i \right\|_\infty \leq 2\Delta_i \left\| Q_\pi - Q_0 \right\|_\infty + 2 \sum_{j=0}^{i-1} \delta^j \left\| \frac{1}{\alpha_{0:i-1}} E_{i-j-1} \right\|_\infty,$$

where $\Delta_i := \dfrac{1}{\alpha_{0:i-1}} \displaystyle\sum_{j=0}^{i-1} \delta^{i-j}\alpha^j$. *For RGRAPE, the following holds:*

$$\left\| A_\pi - \frac{1}{\alpha_{0:i-1}} A_i \right\|_\infty \leq 2\Gamma_i \left\| Q_\pi - Q_0 \right\|_\infty + 2 \sum_{j=0}^{i-1} \gamma^j \left\| \frac{1}{\alpha_{0:i-1}} E_{i-j-1} \right\|_\infty,$$

where $\Gamma_i := \dfrac{1}{\alpha_{0:i-1}} \sum_{j=0}^{i-1} \gamma^{i-j}\alpha^j$.

Let us discuss RGRAPE's noise-tolerance. It can be seen from $\|E_{i-1}/\alpha_{0:i-1}\|_\infty$ in Theorem 3.4.2. As $E_{i-1}/\alpha_{0:i-1} = \sum_{j=0}^{i-1} \alpha^j \varepsilon_{i-j-1}/\alpha_{0:i-1}$, RGRAPE shows noise-tolerance similar to the LR-based approach. Furthermore the noise-tolerance of LR-based and RGRAPE is expected to approximately coincide when $\alpha = 1 - \eta$. In numerical experiments, we indeed observed the coincidence.

We also argue the ineffectiveness of increasing the number of samples in each update. Suppose that $\varepsilon_j(x, a), i \in [0\,{:}\,i]$ are i.i.d. random variables whose mean and variance are 0 and 1, respectively. Then $E_{i-1}(x, a)/\alpha_{0:i-1}$ has a variance $(1 + \alpha^2 + \cdots + \alpha^{2i})/\alpha_{0:i-1}^2$. It converges to approximately 0.025 when $\alpha = 0.95$, while it is 1 when $\alpha = 0$. Thus a higher α leads to a significantly smaller variance. Although $\varepsilon_j(x, a), i \in [0\,{:}\,i]$ are not i.i.d. in practice, a similar result is expected to hold in model-free setting, in which updates are estimated from samples. To attain a variance of ε_j as small as 0.025, around forty times more samples are required ($1/0.025 \approx 40$).

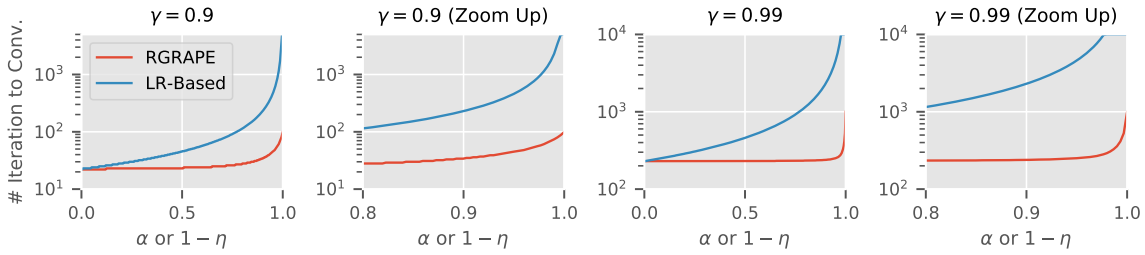


Figure 3.3: Convergence rate comparison of RGRAPE and R(λ). The red lines show results with RGRAPE, while the blue lines show results with R(λ) with a learning rate. In each panel, the vertical axis is the number of iterations i at which Γ_i for RGRAPE or $(1 - \eta(1 - \gamma))^{i+1}$ for R(λ) with a learning rate becomes less than 0.1. Note that vertical axes are in log scale. The horizontal axis is α for RGRAPE or $1 - \eta$ for R(λ) with a learning rate. As explained in the main text, noise-tolerance of RGRAPE and R(λ) are approximately equal when $\alpha = 1 - \eta$. γ is shown on top of each panel.

Maximum noise-tolerance is obtained when $\alpha = 1$. However the convergence RGRAPE with $\alpha = 1$ is extremely slower than that with α smaller than 0.95. In Figure 3.3, we have visualized the number of iterations i by which $\sum_{j=0}^{i-1} \gamma^{i-j}\alpha^j/\alpha_{0:i-1}$ becomes less than 0.01 (red lines). As is seen, the number of iterations starts to quickly increase around $\alpha = 0.975$. Yet RGRAPE show several times faster convergence than that of R(λ) with a learning rate.

Finally we argue what happens if $\varepsilon_k(s, a)$ are not noise, and averaging has no effect. Then using the triangle inequality, we have

$$\|A^\pi - A_i\| \le o(1) + \frac{2}{\alpha_{0:i-1}} \sum_{j=0}^{i-1} \gamma^j \sum_{k=0}^{i-j-1} \alpha^k \|\varepsilon_{i-j-k-1}\|_\infty \propto \sum_{j=0}^{i-1} c_{i-j-1} \|\varepsilon_{i-j-1}\|_\infty ,$$

where we ignored $o(1)$ term in the right hand side and defined

$$c_{i-j-1} := \frac{1}{\alpha_{0:i-1}} \sum_{k=0}^{j} \alpha^{j-k}\gamma^k. \tag{3.16}$$

This coefficient determines how quickly effects of past errors decay. Note that c_{i-j-1} is the coefficient of $\|\varepsilon_{i-j-1}\|_\infty$. Figure 3.4 visualizes the coefficient clearly illustrating enlarged and lessened effect of the past ($j \approx i-1$) and recent errors ($j \approx 0$) for a large α, respectively.

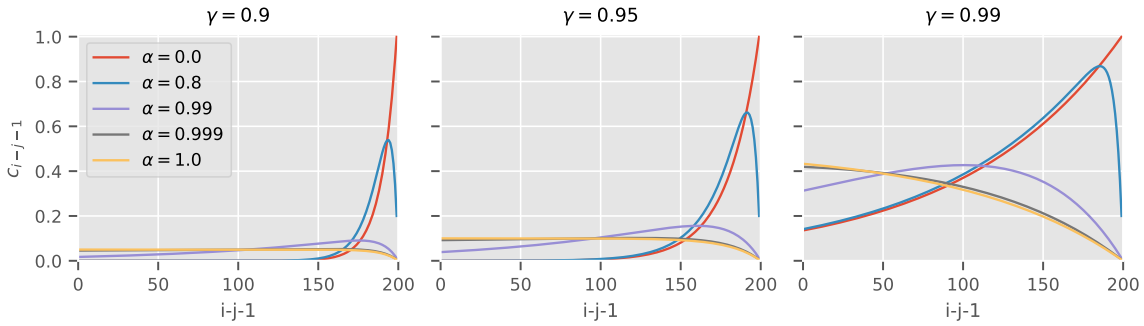


Figure 3.4: Error decay of RGRAPE. Lines show the coefficient (3.16) with various α as in the legend. γ is shown on top of each panel. The horizontal axis is $i-j-1$, where $i = 200$. Therefore the value at $i-j-1 = 200$ shows strength of ε_{200}'s effect on $A_\pi - A_i$. As clearly shown, effects of errors at early iterations lingers if α is high. However if each error function has the same L^∞-norm, the net effect of errors is the same across different α as argued in the main text.

From Figure 3.4 one may wonder whether the net effect of errors might be large in RGRAPE. To see that this is not the case, let us suppose for simplicity that $\|\varepsilon_j\| = \varepsilon$, and that $\alpha = 1$, which must show a drastic difference from a case with $\alpha = 0$. Then

$$\lim_{i\to\infty} \|A^\pi - A_i/\alpha_{0:i-1}\| \le \frac{2\varepsilon}{1-\gamma}.$$

The same asymptotic bound is obtained when $\alpha = 0$, i.e., when AR(λ) is used; thus, the net effect of errors is unchanged.

3.4.1 Practical Implementation

We explain how to efficiently implement GRAPE and RGRAPE in model-free setting.

Update Estimation Based on Samples

We first discuss how to estimate update targets of GRAPE and RGRAPE in model-free setting. Sample estimate of RGRAPE's update can be straightforwardly given by

$$r_0 + \gamma \boldsymbol{\pi} Q_i(x_1) + \sum_{t=0}^{\infty} \gamma^{t+1} \lambda^{t+1} \prod_{u=1}^{t+1} c(x_u, a_u) \delta_{t+1} + \alpha A_i(x_0, a_0), \tag{3.17}$$

where $\delta'_t := r_t + \gamma \boldsymbol{\pi} Q_i(x_{t+1}) - Q_i(x_t, a_t) + \alpha A_{i-1}(x_t, a_t)$, and $a_t \sim \mu(\cdot|x_t)$.

GRAPE needs some tricks to reduce variances of update target estimates. First it is a bad idea to estimate $\sum_{a \in \mathcal{A}} \pi(a|x) Q_i(x, a)$ by $Q_i(x, a), a \sim \pi(\cdot|x)$ or $\rho(x, a) Q_i(x, a), a \sim \mu(\cdot|x)$, where ρ is a importance sampling ratio $\rho(x, a) := \pi(a|x)/\mu(a|x)$. The reason is that the variances of such estimators tend to be high. Indeed from Lemma 3.8.1, we have $Q_{i+1} = \alpha_{0:i} q_{i+1} - \alpha_{1:i} \boldsymbol{\pi} q_i$. Furthermore the last paragraph before Theorem 3.4.1 explains that $q_i \approx Q_\pi$ holds. Accordingly we can expect that $Q_i \approx \alpha_{0:i} Q_\pi - \alpha_{1:i} V_\pi$. Suppose that it holds with equality. Then, the variance of $Q_i(x, a), a \sim \pi(\cdot|x)$, for example, is given by

$$\begin{aligned}
\mathbb{V} Q_i(x, \cdot) &= \sum_{a \in \mathcal{A}} \pi(a|x) \left[\left(Q_i(x, a) - \sum_{b \in \mathcal{A}} \pi(b|x) Q_i(x, b) \right)^2 \right] \\
&= \alpha_{0:i}^2 \sum_{a \in \mathcal{A}} \pi(a|x) A_\pi(x, a)^2.
\end{aligned}$$

Thus it is proportional to $\alpha_{0:i}^2$, which is extremely large when $\alpha \approx 1$. For example, it is i^2 when $\alpha = 1$.

Next let us consider the variance of $\rho(x_t, a_t) \left(r_t + \gamma \boldsymbol{\pi} Q_i(x_{t+1}) - Q_i(x_t, a_t) \right)$ given x_t. For simplicity, assume that $Q_i = \alpha_{0:i} Q_\pi - \alpha_{1:i} V_\pi$. Then we have

$$\sum_{a_t \in \mathcal{A}} \mu(a_t|x_t) \mathbb{E} \left[\rho(x_t, a_t) \left(R_t + \gamma \boldsymbol{\pi} Q_i(X_{t+1}) - Q_i(x_t, a_t) \right) | x_t, a_t \right] = 0.$$

Letting $\kappa(x_t, a_t)$ denote a control variate such that $\sum_{a_t \in \mathcal{A}} \mu(a_t|x_t) \rho(x_t, a_t) \kappa(x_t, a_t) = 0$,

$$\begin{aligned}
&\sum_{a_t \in \mathcal{A}} \mu(a_t|x_t) \mathbb{E} \left[\rho(x_t, a_t)^2 \left(R_t + \gamma \boldsymbol{\pi} Q_i(X_{t+1}) - Q_i(x_t, a_t) - \kappa(x_t, a_t) \right)^2 \middle| x_t, a_t \right] \\
&\propto \sum_{a_t \in \mathcal{A}} \mu(a_t|x_t) \left(2 \alpha_{1:i} \rho(x_t, a_t)^2 \kappa(x_t, a_t) A_\pi(x_t, a_t) + \rho(x_t, a_t)^2 \kappa(x_t, a_t)^2 \right).
\end{aligned}$$

The last line is minimized when $-\kappa(x_t, a_t) = \alpha_{1:i} A_\pi(x_t, a_t) \approx \alpha A_i(x_t, a_t)$. Accordingly, given a trajectory $(x_0, a_0, r_0, x_1, a_1, r_1, \ldots)$ wherein $a_t \sim \mu(\cdot|x_t)$, one of the most straightforward and reasonable estimator is

$$r_0 + \gamma \boldsymbol{\pi} Q_i(x_1) + \sum_{t=0}^{\infty} \gamma^{t+1} \lambda^{t+1} \prod_{u=1}^{t} c(x_u, a_u) \rho(x_{t+1}, a_{t+1}) \delta_{t+1} + \alpha A_i(x_0, a_0), \tag{3.18}$$

where $\delta_t := r_t + \gamma \boldsymbol{\pi} Q_i(x_{t+1}) - Q_i(x_t, a_t) + \alpha A_i(x_t, a_t)$, and $\prod_{u=1}^{0} c(x_u, a_u) = 1$.

Policy Improvement

For policy improvement, we have used a simple variant of TRPO by Schulman et al. (2015). Its policy updates are given by

$$\pi_{k+1}(a|x) = \frac{\pi_k(a|x) \exp\left(\beta A^{\pi_k}(x, a)\right)}{\sum_{b \in \mathcal{A}} \pi_k(b|x) \exp\left(\beta A^{\pi_k}(x, b)\right)}, \tag{3.19}$$

with $\pi_0(a|x) = 1/|\mathcal{A}|$, where $\beta \in (0, \infty)$. In real implementation, A^{π_k} is estimated by each algorithm. While writing the thesis, we found that this algorithm is MPO (Abdolmaleki et al., 2018) using KL divergence regularization rather than constraint. Therefore we omit its derivation.

Efficient Target Computation

To reduce the number of computations, it is recommended to compute the update targets (3.17) and (3.18) backwards. Algorithms 4 and 5 explain how to do it. It is an approximation in a sense that the sum $\sum_{t=0}^{T} \gamma^{t+1} \lambda^{t+1} \prod_{u=1}^{t+1} c(x_u, a_u) \delta_{t+1}$ is used (i.e., the sum is from $t = 0$ to T).

Algorithm 4 Compute RGRAPE Target

Require: Contiguous samples $(x_t, a_t, r_t, x_{t+1}, \mu_t, d_t)_{t \in [0:T]}$, iteration index i, current and old value functions Q_i, Q_{i-1}, and a target policy π.

$b_{T+1} \leftarrow 0$.
for t from T to 0 **do**
  $\rho \leftarrow \pi(a_t|s_t)/\mu_t$, $c \leftarrow \min\{1, \rho\}$.
  $B_\pi Q_i \leftarrow r_t + \gamma(1 - d_t) \sum_{b \in \mathcal{A}} \pi(b|x_{t+1}) Q_i(x_{t+1}, b)$.
  $A_{i-1} \leftarrow Q_{i-1}(x_t, a_t) - \sum_{b \in \mathcal{A}} \pi(b|x) Q_{i-1}(x_t, b)$ if $i \neq 0$ or $i \neq 1$ otherwise 0.
  $A_i \leftarrow Q_i(x_t, a_t) - \sum_{b \in \mathcal{A}} \pi(b|x) Q_i(x_t, b)$ if $i \neq 0$ otherwise 0.
  $Q'_t \leftarrow B_\pi Q_i + \alpha A_i + \gamma \lambda b_{t+1}$.
  $b_t \leftarrow c\left(B_\pi Q_i - Q_i(s_t, a_t) + \alpha A_{i-1}\right) + \gamma \lambda c b_{t+1}$ if $d_t \neq 0$ otherwise 0.
end for
return Targets $(Q'_0, \ldots, Q'_T)$.

3.5 Numerical Experiments

In order to confirm theoretical results, we have carried out experiments in several environments. We provide experimental results in this Section 3.5.

3.5.1 Policy Evaluation Performance Comparison in NChain

The first experiment is conducted to compare policy evaluation performance of GRAPE and R(λ). We carried out it in an environment called NChain as explained in Experi-

Algorithm 5 Compute GRAPE Target

Require: Contiguous samples $(x_t, a_t, r_t, x_{t+1}, \mu_t, d_t)_{t\in[0:T]}$, iteration index i, current value function Q_i, and a target policy π.

$b_{T+1} \leftarrow 0$.
for t from T to 0 **do**
  $\rho \leftarrow \pi(a_t|s_t)/\mu_t$, $c \leftarrow \min\{1, \rho\}$.
  $B_\pi Q_i \leftarrow r_t + \gamma(1-d_t)\sum_{b\in\mathcal{A}}\pi(b|x_{t+1})Q_i(x_{t+1}, b)$.
  $A_i \leftarrow Q_i(x_t, a_t) - \sum_{b\in\mathcal{A}}\pi(b|x)Q_i(x_t, b)$ if $i \neq 0$ otherwise 0.
  $Q'_t \leftarrow B_\pi Q_i + \alpha A_i + \gamma\lambda b_{t+1}$.
  $b_t \leftarrow \rho\left(B_\pi Q_i - Q_i(s_t, a_t) + \alpha A_i\right) + \gamma\lambda c b_{t+1}$ if $d_t \neq 0$ otherwise 0.
end for
return Targets $(Q'_0, \ldots, Q'_T)$.

ment 3.5.1.

Experiment 3.5.1 (Model-Free Policy Evaluation Experiment in NChain)**.** *NChain is a larger, stochastic version of an environment in Example 6.2 Random Walk of (Sutton and Barto, 2018). The environment is a horizontally aligned linear chain of twenty states in which an agent can move left or right at each time step. However, with a small probability called slip prob (≤ 0.5), the agent moves to an opposite direction. The agent can get a small positive reward when it reaches the right end of the chain. The left and right ends of the chain are terminal states. This environment is suitable for investigating noise-tolerance of algorithms as it allows the control of stochasticity by slip prob.*

The experiments in NChain are conducted as follows: one trial consists of $200,000$ *interactions, i.e. time steps, of an agent with an environment. At each time step, the agent takes an action $a \sim \mu(\cdot|x)$ given a current state x. Then, it observes a subsequent state y with an immediate reward r. If the state transition is to a terminal state, an episode ends, and the agent starts again from a random initial state. The interactions are divided into multiple blocks. One block consists of $N = 250$ time steps. After each block, the agent update its value function using N samples of the state transition data $(x, a, r, y, \mu(a|x), d)$ in the block, where $d = 1$ if the transition is to a terminal state otherwise* 0*. After each block, the agent is reset to the start state. Ψ_0 is initialized to $\Psi_0(x, a)$. $\pi(\cdot|x)$ and $\mu(\cdot|x)$ are sampled from $|\mathcal{A}|$-dimensional Dirichlet distribution with all concentration parameters set to* 1*. The discount factor is* 0.99*, and λ is varied.*

Figure 3.5 visually compares GRAPE and R(λ) with a learning rate. λ is set to 0. In all panels, there is a clear tendency that increasing either α or η leads to decreased NMSE, except $\eta = 0.01$. Asymptotic NMSE of GRAPE and R(λ) closely match when $\alpha = 1 - \eta$. We note that GRAPE with $\alpha = 0.99$ shows strong noise-tolerance with reasonably fast learning. Because the number of samples in one update is fixed, this result shows significantly more efficient learning by GRAPE. Due to page limitations, we omit experimental results in which the number of samples in one update is $N = 2000$. However, we note that GRAPE with a frequent update with $N = 250$ worked better in terms of the number of samples, in accordance with our theory.

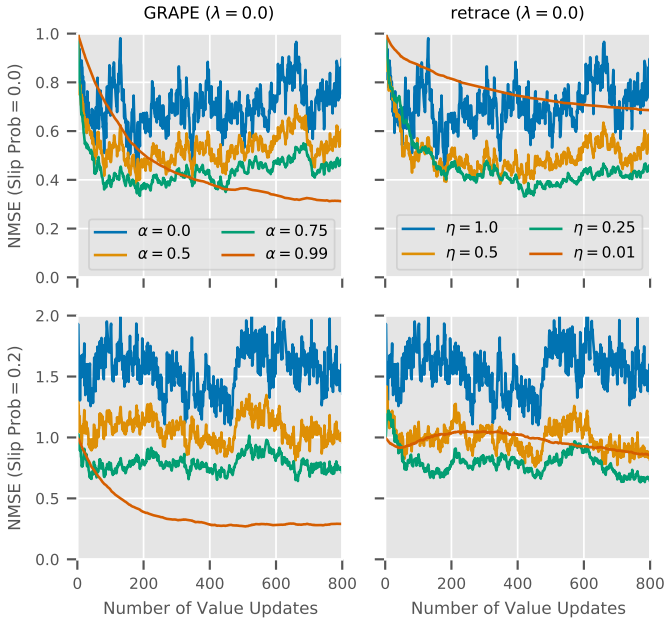


Figure 3.5: Policy evaluation performance of GRAPE and R(λ) with a learning rate in NChain. The horizontal axes show the number of value updates. The vertical axes show NMSE (lower is better). Lines show mean performance over twenty-four trials. For visibility, we omit error bars. The first row shows results of GRAPE and R(λ) when slip prob is 0.0. α and η are indicated by the legends. λ is fixed to 0.0. The second row is the same except that slip prob is increased to 0.2.

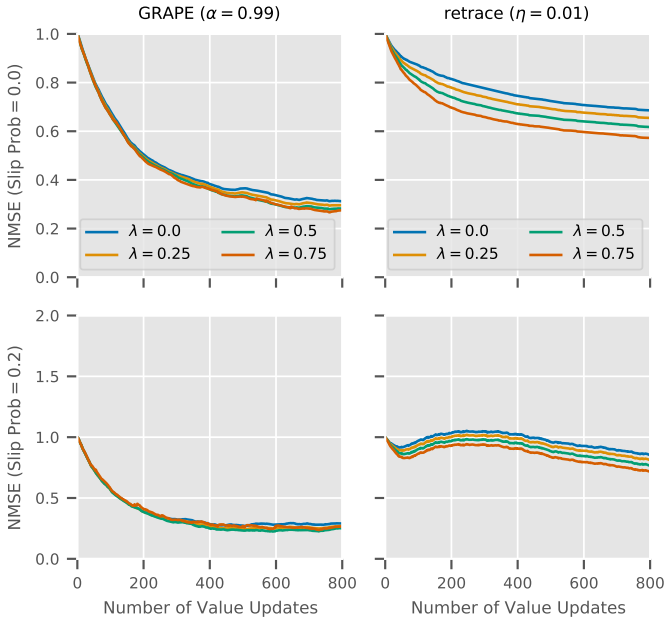


Figure 3.6: Policy evaluation performance of GRAPE and R(λ) with a learning rate in NChain using various λ. The figure is same as Figure 3.5 except that α of GRAPE and η of R(λ) are fixed to 0.99 and 0.01, respectively. These values are chosen so that the effect of λ is most visible.

Figure 3.6 illustrates the effect of changing λ. In GRAPE, there is a slight improvement by increasing λ, whereas in R(λ), there is a clear tendency that increasing λ improves learning. A possible reason implied by our theory is that δ is much smaller than $\alpha = 0.99$; thus, the convergence rate is almost determined by α.

3.5.2 Control Performance Comparison in FrozenLake

Next we have carried out model-free control experiments in 8×8 FrozenLake as explained in Experiment 3.5.2 to investigate the usefulness of GRAPE.

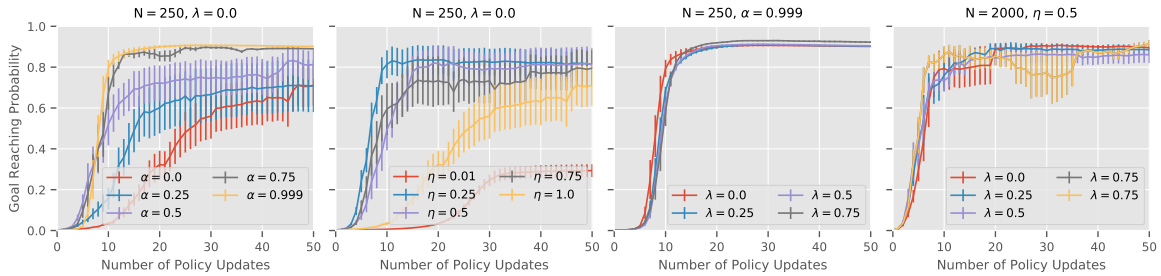


Figure 3.7: Control Task Performance comparison of GRAPE and R(λ) with a learning rate in FrozenLake. The horizontal axes show the number of policy updates. The vertical axes show goal-reaching probability (higher is better) computed by DP. An optimal α is almost 1. The number of samples N used for updating value functions is indicated on top of each panel. Lines show mean performance over six trials. Error bars show standard error. The first and second (from left to right) panels show effects of α in GRAPE and η in R(λ) with a learning rate, respectively. N and λ are fixed to 250 and 0, respectively, as shown on top of those panels. The third panel shows the effect of λ in GRAPE with $\alpha = 0.999$. The last panel shows the effect of λ in R(λ) with a learning rate $\eta = 0.5$ when $N = 2,000$. ($\eta = 0.5$ has performed best when $N = 2,000$ in contrast to a case $N = 250$.)

Experiment 3.5.2 (Model-Free Control Experiment in 8×8 FrozenLake)**.** *The experiments in FrozenLake are done as follows: one trial consists of* $5,000,000$ *interactions. At each time step, the agent takes an action* $a \sim \pi_k(\cdot|x)$*, which is repeatedly updated through the trial, given a current state* x*. Then, it observes a subsequent state* y *with an immediate reward* r*. If the state transition is to a terminal state, an episode ends, and the agent starts again from the start state. The state transition data* $(x, a, r, y, \pi_k(a|x), d)$ *are stored in a buffer* $\mathcal{D}$*, of size* $500,000$*. Every* $N = \{250, 2000\}$ *(fixed through the trial) time steps, the agent updates its value function using* N *contiguous samples from the buffer* $\mathcal{D}$*. Every* $100,000$ *time steps, the agent updates its policy according to a rule explained below.* $\beta \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$ *are tried for each parameter set* (α, λ, N) *(or* (η, λ, N) *when R(*λ*) with a learning rate is used), and we selected one that yielded the highest asymptotic performance.* Q_0 *is initialized to* $Q_0(x, a)$*.* π_0 *and* μ_0 *are initialized to* $\pi_0(a|x) = \mu_0(a|x) = 1/|\mathcal{A}|$*. For policy improvement, the variant of TRPO* (3.19) *is used.*

Figure 3.7 shows the result. The first and second (from left to right) panels show effects of α and η, respectively. It is possible to see a clear tendency of performance increase by increased α. Particularly, GRAPE with $\alpha = 0.999$ outperforms R(λ) with any learning rate. The third panel shows the effect of λ in GRAPE with $\alpha = 0.999$. A slightly better asymptotic performance is seen for $\lambda = 0.75$. However, its effect is not clear. The last panel shows the effect of λ in R(λ) with a learning rate $\eta = 0.5$ when $N = 2,000$. ($\eta = 0.5$ performed best when $N = 2,000$ in contrast to a case $N = 250$.) In this case, when λ is either 0 or 0.25, R(λ)'s asymptotic performance matches that of GRAPE with $\alpha = 0.999$. However, note that eight times more data are used in one update. Moreover, the learning of R(λ) with a learning rate is unstable compared to that of GRAPE with $\alpha = 0.999$.

3.6 Related Research

Before concluding this chapter, we provide a quick review of related results to clarify our contributions compared to existing works.

For policy evaluation, there are many algorithms. Arguably TD(λ) is the most well-known policy evaluation algorithm (Sutton and Barto, 2018). Interestingly Konidaris et al. (2011) showed that TD(λ) can be derived, under several assumptions, from a perspective of maximum likelihood estimation. Based on this result, they proposed a variance reduced version of TD(λ) called TD$_\gamma$, which also removes the tuning of λ. However TD$_\gamma$ assumes that n-step return and $n+1$-th step return are independent, which is clearly false. Ω-return algorithm is introduced by Thomas et al. (2015) as a further improved version of TD(λ). These improvements are orthogonal to the idea of GRAPE and can be combined with it.

As for an off-policy version of TD(λ), Precup et al. (2000) proposed an algorithm called Tree-Backup, which uses only a numerator of the importance sampling ratio. Later Munos et al. (2016) proposed R(λ), which unifies a variety of off-policy policy evaluation algorithms.

As we explained, GRAPE is obtained by combining R(λ) and AL (Baird III, 1999; Bellemare et al., 2016). To the best of our knowledge, this is a novel idea. Furthermore we provided an error bound of GRAPE using techniques of error propagation analysis (Munos, 2005, 2007; Farahmand, 2011; Scherrer et al., 2015), and showed the noise-tolerance of GRAPE with a faster convergence than the learning rate based algorithm. This result is also novel and makes clear the benefit of gap-increasing policy evaluation algorithms.

3.7 Conclusion

In this Chapter 3, we proposed a new policy evaluation algorithm called GRAPE. GRAPE is shown to be efficient and noise-tolerant by both theoretical analysis and experimental evidence. GRAPE has been compared to a state-of-the-art policy evaluation algorithm called R(λ). GRAPE demonstrated significant gains in performance and stability.

Though our theoretical analysis is valid even for continuous action space, we only tested GRAPE in environments with a finite action space. Extending GRAPE to a continuous action space is an important research direction. Also, we did not carry out sample complexity analysis. In order to further understand the algorithm, we need it.

3.8 Proofs

As analysis of RGRAPE is notationally simpler but almost same as that of GRAPE, we omit analysis of GRAPE. However note that the almost same results hold for GRAPE too.

We first prove the following lemma.

Lemma 3.8.1. *Let q_0 be a constant function taking 0, and*

$$\alpha_{0:i} q_{i+1} := \sum_{j=0}^{i} \alpha^j \left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i+1-j} Q_0 + \sum_{j=0}^{i} \gamma^j \boldsymbol{N}^j E_{i-j}.$$

For RGRAPE in Equation (3.11), $Q_{i+1} = \alpha_{0:i} q_{i+1} - \alpha_{1:i} \boldsymbol{\pi} q_i$ holds.

Proof of Lemma 3.8.1. We prove that

$$Q_{i+1} = \left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i+1} Q_0 + \sum_{j=0}^{i} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j} + \alpha A_i \tag{3.20}$$

by induction. For $i = 0$, it clearly holds. Now assume that it holds up to i. Then

$$\begin{aligned} Q_{i+1} &= Q_i - \alpha A_{i-1} + \left(\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_{\mu c}\right)^{-1} \left(\boldsymbol{B}_\pi Q_i - Q_i + \alpha A_{i-1}\right) + \alpha A_i + \varepsilon_i \\ &\overset{\text{(a)}}{=} \boldsymbol{R}_{\mu c}^{\lambda} \left(\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i} Q_0 + \sum_{j=0}^{i-1} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j-1} \right) + \alpha A_i + \varepsilon_i \\ &\overset{\text{(b)}}{=} \left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i+1} Q_0 + \sum_{j=0}^{i-1} \gamma^{j+1} \boldsymbol{N}^{j+1} \varepsilon_{i-j-1} + \alpha A_i + \varepsilon_i, \end{aligned}$$

where we used $Q_i = \left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i} Q_0 + \sum_{j=0}^{i-1} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j-1} + \alpha A_{i-1}$ at (a), and

$$\begin{aligned} \boldsymbol{R}_{\mu c}^{\lambda} (f + g) &= f + g + \left(\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_{\mu c}\right)^{-1} \left(\boldsymbol{B}_\pi (f + g) - f - g\right) \\ &= \boldsymbol{R}_{\mu c}^{\lambda} f + \gamma \left(\boldsymbol{I} - \gamma\lambda \boldsymbol{P}_{\mu c}\right)^{-1} \left(\boldsymbol{P}_\pi - \lambda \boldsymbol{P}_{\mu c}\right) g \\ &= \boldsymbol{R}_{\mu c}^{\lambda} f + \gamma \boldsymbol{N} g \end{aligned}$$

at (b). As $\sum_{j=0}^{i-1} \gamma^{j+1} \boldsymbol{N}^{j+1} \varepsilon_{i-j-1} + \varepsilon_i = \sum_{j=0}^{i} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j}$ holds, Equation (3.20) holds.

Next note that

$$\begin{aligned} A_i &= \left(\boldsymbol{I} - \boldsymbol{\pi}\right) \left(\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i} Q_0 + \sum_{j=0}^{i-1} \gamma^j \boldsymbol{N}^j \varepsilon_{i-j-1} \right) + \alpha A_{i-1} \\ &= \left(\boldsymbol{I} - \boldsymbol{\pi}\right) \sum_{j=0}^{i-1} \alpha^j \left(\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i-j} Q_0 + \sum_{k=0}^{i-j-1} \gamma^k \boldsymbol{N}^k \varepsilon_{i-j-k-1} \right) + \alpha^i A_0 \\ &= \left(\boldsymbol{I} - \boldsymbol{\pi}\right) \left(\sum_{j=0}^{i-1} \alpha^j \left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i-j} Q_0 + \sum_{j=0}^{i-1} \gamma^j \boldsymbol{N}^j E_{i-j-1} \right). \end{aligned}$$

As

$$\sum_{j=0}^{i}\gamma^j\boldsymbol{N}^j\varepsilon_{i-j}+\alpha\sum_{j=0}^{i-1}\gamma^j\boldsymbol{N}^jE_{i-j-1}=\sum_{j=0}^{i}\gamma^j\boldsymbol{N}^j\varepsilon_{i-j}+\sum_{j=0}^{i-1}\gamma^j\boldsymbol{N}^j\sum_{k=1}^{i-j}\alpha^k\varepsilon_{i-j-k}$$
$$=\sum_{j=0}^{i}\gamma^j\boldsymbol{N}^j\sum_{k=0}^{i-j}\alpha^k\varepsilon_{i-j-k},$$

we deduce that $Q_{i+1}=\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i+1}Q_0+\sum_{j=0}^{i}\gamma^j\boldsymbol{N}^j\varepsilon_{i-j}+\alpha A_i=\alpha_{0:i}q_{i+1}-\alpha_{1:i}\boldsymbol{\pi}q_i$. □

Remark 3.8.1. *The key step of the proof is showing that* Q_{i+1} *can be rewritten as* $\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i+1}Q_0+\sum_{j=0}^{i}\gamma^j\boldsymbol{N}^j\varepsilon_{i-j}+\alpha A_i$. *Because* $\boldsymbol{\pi}A_i=0$ *holds, the almost same result holds for GRAPE as noted before.*

By using Lemma 3.8.1, Theorem 3.4.1 can be immediately obtained. Indeed, from the definition of q_{i+1} in Lemma 3.8.1, it is immediately deduced that

$$\left\|Q_\pi-q_i\right\|_\infty=\left\|\frac{1}{\alpha_{0:i-1}}\sum_{j=0}^{i-1}\alpha^j\left(Q_\pi-\left(\boldsymbol{R}_{\mu c}^{\lambda}\right)^{i-j}Q_0\right)-\frac{1}{\alpha_{0:i-1}}\sum_{j=0}^{i-1}\gamma^j\boldsymbol{N}^jE_{i-j-1}\right\|_\infty$$
$$\leq\frac{1}{\alpha_{0:i-1}}\sum_{j=0}^{i-1}\gamma^{i-j}\alpha^j\left\|Q_\pi-Q_0\right\|_\infty+\sum_{j=0}^{i-1}\gamma^j\left\|\frac{1}{\alpha_{0:i-1}}E_{i-j-1}\right\|_\infty.$$

Thus we obtain Theorem 3.4.1.

Theorem 3.4.2 can be proven by using Lemma 3.8.1 and the inequality above.

Conclusion

RL is an important research field because many real-world decision making problems can be formulated as and solved by it. For a long time, RL field had focused on online algorithms with a linear model with fixed basis functions because RL algorithms combined with a non-linear function approximator may not converge (Tsitsiklis and Van Roy, 1997). However, after the successful application of AVI combined with a deep neural network to a set of complex video games (Mnih et al., 2015), many researchers started the development of efficient RL algorithms with deep neural networks. Such algorithms are now called deep RL algorithms. Despite many deep RL algorithms' success, it poses several challenges: how to reduce the amount of data required, how to stabilize the learning, how to train a neural network for multiple tasks without catastrophic forgetting, and so on.

In this thesis, I mainly focused on how to stabilize the learning while making it more efficient via theoretical analysis of gap-increasing algorithms. In particular, I formulated CVI and showed its equivalence to VI with the entropy and KL divergence regularization, on the contrary to previous works that consider only one of them (Azar et al., 2012; Schulman et al., 2015; Fox et al., 2016; Haarnoja et al., 2018, 2017; Song et al., 2019; Abdolmaleki et al., 2018). In contrast to most of those previous works, my analysis shows clear benefits of having regularizations, deepening the understanding of the regularization in RL. Leveraging the idea of CVI, I proposed GRAPE, a policy evaluation version of gap-increasing algorithms. I proved that GRAPE is tolerant of noise, similarly to CVI, and confirmed the theory with experiments.

While I proved some benefits of VI with the entropy and KL divergence regularizations, there are some open problems as listed below:

- *Is the sample complexity of gap-increasing algorithms minimax optimal?* In this thesis, I carried out error propagation analysis, wherein I did not consider how using the gap-increasing operator changes the error functions. Sample complexity analysis takes such changes into account and provides a deeper understanding of algorithms.

- *Do other regularizations, such as a Bregman divergence regularization, have similar property or any other benefits?* While the KL divergence is a type of the Bregman divergence, it is not clear if benefits of more general regularization with Bregman divergence (or any other divergences and probability metrics) exist and can be proven.

- *Does a policy regularization have any benefit in terms of exploration?* The exploration is surely a vital part of RL algorithm. However, error propagation analysis

cannot capture the aspect of exploration.

- *Is the regularization agnostic L^∞-norm performance bound for ACVI-Q and Ψ (bound (2.12)) tight for arbitrary β?* While I could show that it is tight when $\beta = \infty$, it is unclear if the bound is tight. A key to its proof is that in the worst case, a set of greedy policies may contain the best and worst policy, the latter of which is intentionally chosen to prove the tightness. When β is finite, this fact cannot be used.

To conclude, in this thesis, I proposed and theoretically analyzed new algorithms based on gap-increasing and softmax operators. Although some open problems remain, I believe that the results in this thesis are an important step towards a deep understanding of RL algorithms.

Bibliography

AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in Neural Information Processing Systems 19*, pages 1–8. 2007.

A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018.

K. Asadi and M. L. Littman. An alternative softmax operator for reinforcement learning. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning*, pages 243–252, 2017.

M. G. Azar, V. Gómez, and H. J. Kappen. Dynamic policy programming. *Journal of Machine Learning Research*, 13(1):3207–3245, 2012.

L. C. Baird III. *Reinforcement Learning Through Gradient Descent*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, US, May 1999.

M. G. Bellemare, G. Ostrovski, A. Guez, P. S. Thomas, and R. Munos. Increasing the action gap: New operators for reinforcement learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

D. P. Bertsekas and S. E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 1996.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Nashua, NH, USA, 1st edition, 1996.

J. A. Boyan and M. L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *Advances in Neural Information Processing Systems 6*, pages 671–678. 1994.

R. M. Dudley. *Real Analysis and Probability*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2nd edition, 2002.

E. Even-Dar and Y. Mansour. Learning rates for q-learning. *Journal of Machine Learning Research*, 5:1–25, Dec 2004.

A.-m. Farahmand. *Regularization in reinforcement learning*. PhD thesis, University of Alberta, 2011.

R. Fox, A. Pakman, and N. Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 202–211, 2016.

M. Geist, B. Scherrer, and O. Pietquin. A theory of regularized markov decision processes. *CoRR*, abs/1901.11275, 2019.

T. Haarnoja, H. Tang, P. Abbeel, and S. Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning*, pages 1352–1361, 2017.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning*, pages 1856–1865, 2018.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274, 2002.

J. Kober and J. Peters. *Reinforcement Learning in Robotics: A Survey*, pages 9–67. Springer International Publishing, 2014.

G. Konidaris, S. Niekum, and P. S. Thomas. TD_γ: Re-evaluating complex backups in temporal difference learning. In *Advances in Neural Information Processing Systems 24*, pages 2402–2410. 2011.

T. Kozuno, E. Uchibe, and K. Doya. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. In *Proceedings of Machine Learning Research*, volume 89, 2019.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

H. Mao, M. Alizadeh, I. Menache, and S. Kandula. Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, HotNets '16, pages 50–56, 2016.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of The Thirty-Third International Conference on Machine Learning*, pages 1928–1937, 2016.

A. Moore. *Efficient Memory-based Learning for Robot Control*. PhD thesis, Carnegie Mellon University, 1991.

R. Munos. Error bounds for approximate value iteration. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pages 1006–1011, 2005.

R. Munos. Performance Bounds in Lp norm for Approximate Value Iteration. *SIAM Journal on Control and Optimization*, 2007.

R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare. Safe and Efficient Off-Policy Reinforcement Learning. In *Proceedings of Twenty-Ninth Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

D. Precup, R. S. Sutton, and S. P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766, 2000.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

K. C. Rawlik. *On probabilistic inference approaches to stochastic optimal control*. PhD thesis, The University of Edinburgh, 2013.

A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

A. L. Samuel. Some studies in machine learning using the game of checkers. ii—recent progress. *IBM Journal of Research and Development*, 11(6):601–617, 1967.

B. Scherrer. Approximate policy iteration schemes: A comparison. In *Proceedings of the Thirty-First International Conference on Machine Learning*, pages 1314–1322, 2014.

B. Scherrer and B. Lesner. On the use of non-stationary policies for stationary infinite-horizon markov decision processes. In *Advances in Neural Information Processing Systems 25*, pages 1826–1834, 2012.

B. Scherrer, M. Ghavamzadeh, V. Gabillon, B. Lesner, and M. Geist. Approximate modified policy iteration and its application to the game of Tetris. *Journal of Machine Learning Research*, 16:1629–1676, 2015.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897, 2015.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3): 287–308, Mar 2000.

Z. Song, R. Parr, and L. Carin. Revisiting the softmax Bellman operator: New benefits and new perspective. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5916–5925, 2019.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, USA, second edition, 2018.

P. S. Thomas, S. Niekum, G. Theocharous, and G. Konidaris. Policy evaluation using the Ω-return. In *Advances in Neural Information Processing Systems 28*, pages 334–342. 2015.

J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas. Sample Efficient Actor-Critic with Experience Replay. In *International Conference on Learning Representations*, 2016.