Improving Policy Optimization: Algorithms and Foundations

WANG, Baoxiang

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science and Engineering

The Chinese University of Hong Kong
July 2020

Thesis Assessment Committee

Professor Pheng Ann Heng (Chair)

Professor Siu On Chan (Thesis Supervisor)

Professor Lin Shi (Committee Member)

Professor Minming Li (External Examiner)

Abstract of thesis entitled:

Improving Policy Optimization: Algorithms and Foundations
Submitted by WANG, Baoxiang
for the degree of Doctor of Philosophy
at The Chinese University of Hong Kong in July 2020

Reinforcement learning (RL) studies algorithmic approaches to optimizing the policy in sequential decision processes. The recent success of RL in a variety of applications has demonstrated its usefulness but also leaves room for improvements. In particular, the process of optimizing the policy requires extensive input of information with large sample complexity, while simultaneously suffers from the lack of information given the usual sparsity of rewards. In this thesis, we discuss algorithmic approaches to addressing both sides of the issue. We also look deeper beyond the algorithms into the foundations of reinforcement learning that could lead to inventive improvements in policy optimization in the future.

The effort to reduce the sample complexity has been consistent since the inception of RL. This leads to a significant line of research that improves policy optimization by statistical variance reduction methods. Following the line of research, we study

efficient variance reduction methods by decomposing the policy estimator over the high-dimensional action space. We show that such a decomposition strictly reduces the variance while conditionally preserving the unbiasedness. The idea was implemented using both heuristics and sound algorithms on learning and testing variable partitions. Rigorously, the bipartition of the action space is almost optimal, while the algorithm generalizes to settings with multiple partitions and functions over general groups. Empirical studies demonstrate the effectiveness in synthetic settings and high-dimensional OpenAI Gym's MuJoCo continuous control tasks.

On the side of utilizing the input information, various methods have been proposed to address the famous problem of credit assignment, especially in applications and real-world tasks. We study the canonical RL application of visual attention, which mimics the human vision on image processing. While previous works have achieved both superior speed and preciseness in object recognition, it is known to not scale to practical tasks due to the sparsity of the rewards, leaving open problems towards credit assignment. We explore both a heuristic algorithm and a method based on asymmetric actor-critic for reward shaping. The new mechanism stabilizes the learning and improves the convergence on large images. In both tasks of noised handwritten digits recognition and diabetic retinopathy screening, our algorithms demonstrate significant efficiency and accuracy improvements.

Beyond all these algorithmic studies we discuss the last but most impactful work on the foundations of reinforcement learning, which explores from the perspective of optimal control what a value function is truly like. We analyze the Gambler's problem, a simple reinforcement learning problem where the gambler has the chance to double or lose the bets until the target is reached. Despite being an early example in the RL textbook, interesting patterns of the value function have been observed but without further investigation. We provide the exact formula for the optimal value function, showing that it is in fact one of the generalized Cantor functions with pathological properties like fractal, self-similarity, singularity, and non-rectifiability. With the findings uncharted in years of reinforcement learning research, our analyses could lead to great insights into value function approximation, gradient-based algorithms, and Q-learning, in real applications and implementations.

強化學習（RL）研究在順序決策過程中優化策略的算法。近年來，RL在各種應用中的成功證實了它的實用性，但也展示出了改進的餘地。特別地，優化策略的過程具有較高的樣本複雜度並且需要大量信息輸入，而由於稀疏獎勵等緣故又通常缺乏信息來源。在本文中，我們討論了信息使用和信息輸入兩邊的算法與改進。我們還從本質上更深入地研究強化學習系統。這些基礎研究可能會在將來導致策略優化方面的創造性提升。

自RL被提出以來，大量的工作集中在減少策略優化的樣本複雜性，帶來了基於統計方差減少方法的重要研究方向。據此思路，我們通過分解高維動作空間上的策略梯度估計來實現有效的方差減少算法。我們證明，此分解嚴格地減小了方差，同時有條件地保留了估計的無偏性。該想法可以通過啓發式算法或嚴格的變量拆分算法實現。我們證明，動作空間的二拆分是幾乎最優的，而算法還可推廣到多拆分的場景和任意群上的函數拆分。實驗表明，在我們設計的環境和OpenAI Gym的MuJoCo高維連續控制任務中，該方法效果出色。

輸入信息的利用較多的被建模爲信用分配問題，在應用場景和實際任務中特別常見。我們研究了RL在計算視覺上模仿人眼對圖像處理的經典應用。儘管先前的工作在目標識別方面擁有較好的速度和精確性，但由於獎勵的稀疏性其無法擴展到實際任務中。我們提出基於啓發式算法和基於非對稱動作評價算法的獎勵設置方法。新的獎勵機制在大圖像上穩定了學習過程並提高了收斂性。在帶噪手寫數字識別和糖尿病性視網膜病變篩查這兩項任務中，我們的算法均顯示出顯著的效率和準確性提升。

最後，我們討論最具有影響力的工作RL基礎工作。該工作從最佳控制的角度探討了價值函數的未被發現的特性。我們分析經典的強化學習例子，賭徒問題。儘管該問題是RL教科書中的經典例子，但其中可觀察到奇特的價值函數曲線，且書中沒有進一步解釋。我們求解了最優價值函數的精確公式，證明它是具有分形，自相似，奇異性和non-rectifiability等病態性質的廣義Cantor函數。該發現可以爲價值函數近似，基於梯度的RL算法，和價值學習的實際應用和實現帶來深刻的見解。

Acknowledgement

This thesis is ever possible due to the enormous support from many people. They have my eternal gratitude.

I would express my utmost appreciation to my advisor, Siu On Chan, from whom I got extensive support and guidance, for my research, my career, and my life. From the void to this very moment, Siu On has led me through the long trial to the very boundary of the horizons of knowledge. He was more than kind and patient with his immature student, despite all the mistakes I have made. Without Siu On, my Ph.D. journey could not be completed.

I would express how deeply I am grateful to Andrej Bogdanov, who greatly helped my Ph.D. study and influenced my research style and my value, in a way like a mentor but more than that. He has been sharing the mind of solving problems and the excitement during this process, which I have borrowed for many years and will maintain for good.

I would also like to thank Sam Zheng, for taking me as a research intern at Siemens Research in 2016 and continuing mentoring me after that. We have spent a large amount of time

discussing ideas in reinforcement learning and applications, and many of them later form the basis of this thesis.

I very much thank Nidhi Hegde and Matthew Taylor, for taking me as a student researcher at RBC and University of Alberta. Despite that the winter in Edmonton was cold and dark, many brilliant ideas emerge and problems are solved in this warm community. The year I spent with you is a well worth one.

I would like to thank my thesis committee - Pheng Ann Heng, Lin Shi, and Minming Li, for their suggestions and comments on research over all these years. I am lucky to have many long-term collaborators - Shuai Li, Jiajin Li, Kenny Young, and Cuiyun Gao, with whom conducting research is more like a cake than like a chore. I am grateful that they trust me for my research skill and my research taste. During the years I have spent here, I am glad that I have many friends - Qi Dou, Siyao Guo, Ruitong Huang, Xin Huang, Yuting Ke, Chengyu Lin, Weiwen Liu, Jincheng Mei, Yangchen Pan, Xiaojuan Qi, Xingjian Shi, Qingyun Sun, Christopher Williamson, Chenjun Xiao, Jiani Zhang, Yihan Zhang, Hong Zhou, Ying Zhu, who make my Ph.D. journey flourish and enjoyable.

To my parents and grandparents.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Reinforcement learning (RL) [1, 2] studies sequential interactions between intelligent agents and environments [3, 4, 5, 6, 7, 8]. The topic was initially established in the 1980s, while recent breakthroughs in deep learning and related techniques have largely pushed forward the area in recent years. A few milestone results have been developed, including the works on the game of Go [9, 10, 11], the algorithmic approaches on the Atari Learning Environment [12, 13], and some other achievements [14, 15, 16]. Soon after, reinforcement learning managed to prevail in a larger set of settings, including games with partial information like Poker [17, 18, 19, 20], and complex tasks like robotics [21, 22], autonomous driving [23, 24, 25], and collaborative games [26, 27, 28]. These results lead to significant lines of research in the broad area of RL. The foundations of the modern reinforcement learning area lie in two major topics. On the one hand, reinforcement learning seeks the optimal policy

or value function in these tasks, following the objective of optimal control [29, 30, 31, 32, 33] but in a manner of tabula rasa [34, 35, 36, 37, 38]. On the other hand, inspired by psychology and cognitive science [39, 40], RL learns the behavior of humans and its underlying mechanism to achieve artificial general intelligence [41]. Reinforcement learning was initially developed as an interdisciplinary study of the combination of them, while absorbing both topics as it demonstrates its potential and generality [2, 38, 42, 43].

In this thesis, we focus on policy optimization, leaving the ideas from psychology and cognitive science to my future research. Optimizing the policy can be roughly categorized into two classes: Model-free reinforcement learning and model-based reinforcement learning [36, 42, 44, 45, 46, 47]. The former follows the basic idea of trial and error, where the intelligent agent learns from its previous experiences without a knowledge a priori about the environment [40, 48, 49, 50, 51]. The agent thus has to sensibly choose the actions it takes since the actions will have an impact on the environment which in turn decides what states the agent will explore in the trajectories. Therefore it induces a set of important problems like balancing exploration and exploitation [52, 53], policy improvement, and credit assignment, and so forth. This leads to a few natural lines of reinforcement learning research, forming the mainstream of the last decade's research. Model-based RL can be slightly counterintuitive, allowing the agent to have oracle access to a model that simulates

the environment it is interacting with [54, 55, 56]. Despite this model-based RL can help to understand the learning process, either with or without the presence of a model, and eventually obtain characterizations of the solution which is eventually helpful even for model-free learning.

The line of studies on trial-and-error reinforcement learning works very well in an ideal task, where simply averaging the cumulative rewards will obtain an estimation of how good an action is and how a policy could be optimized [36, 45, 46]. This is typically achieved by repeatedly generating rollouts of the task according to the current policy in an exploratory manner. However this vanilla method will not scale up, as the space the averaging is taking over is too large for an efficient estimate [13, 57]. Apart from being exponential in the size (discrete) or dimension (continuous) of the action space, the sample complexity is also growing up to an exponential factor to the horizon, depending on how sparse the reward signals are. In the extreme case of episodic rewards, it can be regarded as that the credits of the entire trajectory are aggregated as one single sample, equivalently making the dimension of the decision variables scale up linearly with the horizon [58, 59, 60]. In all, this sample complexity can be astronomically large, making it impossible for any existing computing facility to generate a comparable number of trials. The natural question is then asked by the community that how can we make this seemingly simple idea work in more complicated tasks. This has been the core topic in reinforcement

learning.

There are roughly two possible approaches to improving policy optimization. One is to use samples more efficiently and the other is to obtain better samples within the same scale of computing power. The former leads to the area of sample complexity reduction in reinforcement learning, which attracts a major flow of attention from the community [46, 57, 61, 62, 63, 64, 65, 66]. It usually generates relatively more rigorous results, having an explicit or empirical demonstration of the tradeoff between variance reduction and bias if any [67, 68, 69, 70, 71]. The latter, on the other hand, usually involves heuristics and algorithms who rely more on specific tasks [72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85]. In this thesis, we will cover both parts of the topic with several methods. Though all works are originally motivated by policy optimization, some of them could stand out as independent contributions in learning theory, algorithms, statistics, and machine learning [86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96].

As we have discussed, sample complexity reduction has been the most prevailing topic since the inception of reinforcement learning. From temporal-difference method [42, 97], policy gradient theorem and algorithm REINFORCE [46], actor-critic [45, 61], to advantage actor-critic (A2C) [57] and some more recent advances, these methods are based on a variety of tools in statistical variance reduction methods. Following the line of research, we study efficient variance reduction methods by decomposing the estimator of the policy update over the high-dimensional ac-

tion space, resulting in lower dimensionality for each component of the estimator [71, 98]. Algorithmically, this follows the idea of divide and conquer, where our techniques are general enough to be applied to a wider family of divide-and-conquer algorithms. We show that our proposed estimator strictly reduces the variance while under a mild condition preserves the unbiasedness. Armed with the new estimator, our new policy optimization algorithm demonstrates its effectiveness in a variety of tasks.

This mild condition is later abstracted as an independent problem of variable partitioning, which in turn is a fundamental and useful tool in math. Our learning algorithm establishes the method that finds the almost optimal bipartition for real functions. The algorithm generalizes to the case of multi-partitioning with an approximation factor that does not scale with the number of partitions. For functions over general groups the approximation factor is still guaranteed to be in $O(n^2)$ where n is the number of variables, even though it is shown by Bogdanov and me [98] that obtaining almost optimal partitions over general groups is NP-hard. It is worth noting that Bogdanov and I [98] also have a testing algorithm that finds the bipartitionability in $O(n^3)$ queries to the function with a lower bound at least $\Omega(n)$ queries. Plugging back our learning algorithm, the performance of the aforementioned policy optimization method is further improved, especially in high-dimensional OpenAI Gym's MuJoCo continuous control tasks.

The other side of policy optimization by increasing the ef-

fective sampling quality will also be discussed in this thesis. Primarily, the discussions will be anchored to applications, as in such works the methodology is usually depending on the exact tasks the agent will be conducting. We consider mostly a canonical application of reinforcement learning, where it mimics the human vision [99, 100] which deploys successive saccades instead of taking all pixels at parallel [101, 102, 103]. This, in general, achieves superior speed and better precision due to the fact that it skips most of the clutters [104, 105, 106, 107, 108]. On the negative side, the model is known to be hard to train with a slow and unstable learning process and possible divergence, leaving open problems towards credit assignment [42].

The cause of the drawback is that as we treat the entire process of sequential saccades as one trajectory and assign one evaluation for the output, from the perspective of reinforcement learning it suffers from an episodic reward signal. To better utilize the input information, we explore both a heuristic algorithm and a method based on asymmetric actor-critic for reward shaping [109]. The new mechanism stabilizes the learning and improves the convergence on large images. In both tasks of noisy handwritten digits recognition and diabetic retinopathy screening, our algorithms demonstrate significant efficiency and accuracy improvements.

Beyond all these algorithmic studies we discuss the last but most impactful work on the foundations of reinforcement learning [110]. This work assumes the full knowledge of the envi-

ronment and in lieu of learning analyzes the exact solution of the task, aiming to discover what a value function is truly like. Despite that this will fall into the category of optimal control and dynamical systems [111, 112], the exact solution will provide many greater insights into the foundations of reinforcement learning. The task is called the Gambler's problem, a simple reinforcement learning problem where the gambler has the chance to double or lose the bets until the target is reached. The problem is discussed as an early example in the RL textbook [2] and is used to demonstrate the value iteration method to obtain an numerical solution.

Though being an early example in the RL textbook, interesting patterns of the value function have been observed. Unfortunately the book does not provide us with addition investigation, leaving the explanation of the strange patterns as an open problem. We solve the problem, providing the exact formula for the optimal value function. Simple as it might seem, we show that it is in fact one of the generalized Cantor functions with several pathological properties including fractal, self-similarity, singularity (the derivative of the function is either zero or infinity), and non-rectifiability (the function cannot be written by elementary functions). The work gives an explicit description of the value function, leading to future research in optimization, approximation, and algorithm that take into considerations these characterizations uncharted for years. A solid understanding of the system can drive inventive improvements

for policy optimization in the future.

□ **End of chapter.**

Chapter 2

Background and Preliminaries

In this chapter we will briefly discuss the history of reinforcement learning and related areas. The position of this thesis will be given accordingly by the context. The remainder of this section will discuss general preliminaries and notations used in this thesis.

2.1 The history of reinforcement learning

Reinforcement learning is usually regarded as an interdisciplinary topic that emerged in the 1980s, though many topics who already consider the idea of reinforcement learning were raised earlier. In 1898, Thorndike's arguments about animal intelligence have already considered how intelligence is formed [40]. To be exact, it considers trial-and-error learning of animals, which is a basis of the major part of reinforcement learning. In 1938, Woodworth's results about experimental psychology further strengthen the study of intelligent behavior, which forms

a base of researching such intelligence of a computing machine [39]. One of the earliest studies of machine intelligence is conducted by Turing in 1948 [1], which borrows the idea of using the reinforcement signals such as pain stimulus and pleasure stimulus to drive the choice of actions. An important observation by Turing is that such a stimulus must have a permanent effect for the machine learner, which makes the idea stand out from later topics such as supervised learning.

To understand the history of reinforcement learning we first recall that it is in the intersection of optimal control and psychology, concerning machine intelligence under the way an animal or human is learning. The former, optimal control, can be traced back to the 1950s [113, 114]. In the early days, Markov decision processes have been formulated by Bellman, which is still the canonical formulation nowadays. It has been extensively developed in the next few decades, including approximation methods [115], methods for partially observable environments [116], and applications [117, 118, 119]. An important methodology, dynamic programming, was proposed back then in multiple settings to solve the Markov decision processes [34]. The method is still taking an important role in the modern days of computing algorithms, while in reinforcement learning its learning variant leads to the famous Q-learning who achieved superhuman performance in the Atari games in the past few years [12, 34, 35, 36, 37, 38, 47].

The other vein, which concerns its psychology background,

leads to trial-and-error learning. It is motivated by psychology studies that animals or humans learn by receiving reinforcement signals for their actions. Though the topic is closely aligned with the background, it receives insufficient attention before the 1970s from the community, who back to then was focusing on supervised learning. Starting in 1972, Klopf revived the thread of trial-and-error learning [49, 50, 51], which motivates the series of research by scientists including Sutton and Barto [48, 120, 121, 122]. Together with a few other academics, they are considered the founding people of the modern reinforcement learning partially based on time-difference methods and the actor-critic architecture [38, 42, 43, 45, 97, 123]. The line of research later led to successful applications on backgammon in 1992, resulting in heavy growth in attention to the area [124]. And then a decade later, together with improved computing facilities and tools like deep learning, a series of applications led by AlphaGo have proved the generality and usefulness of reinforcement learning [9, 10, 11].

This thesis positions itself in policy optimization, which is built upon the established modern reinforcement learning model in the 1980s by trial-and-error methods. We choose not to discuss advanced models via new findings in the intersection of computing science and cognitive science and psychology. We also leave out topics in optimal control [32, 56] and multi-arm bandits [125, 126, 127, 128, 129, 130, 131, 132, 133]. Though, our Section 5 can be categorized by its mathematical techniques

into optimal control and dynamical systems.

This thesis is based on a series of published conference papers by the author, including the works on decomposed policy gradient [71], variable partitioning [98], reinforcement learning computing vision [109], and the Gambler's problem [110]. Due to the author's intent to keep the thesis under a reasonable size, we will not include other interesting works conducted during my PhD study, including the works on differentially private reinforcement learning [134], meta-reinforcement learning [135], vectored inverse reinforcement learning [136], cascading bandits [130], and dynamic topic models [137]. The exact list of the author's publications can be found in Appendix B.

2.2 Preliminaries

We consider policy optimization formulated in the canonical setting of discrete-time Markov decision process (MDP), denoted as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma)$. The tuple includes $\mathcal{S} \in \mathbb{R}^m$ the m dimensional state space or $\mathcal{S} \in [m]$ the size-m discrete state space, where $[m]$ denotes $\{1, \ldots, m\}$; $\mathcal{A} \in \mathbb{R}^n$ the n dimensional action space or $\mathcal{A} \in [n]$ the size-n discrete action space; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ or $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ the stochastic or deterministic environment transition probability function, where $\Delta(\cdot)$ denotes the set of all random variables over the input space; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ or $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the stochastic or deterministic reward function; $\rho_0 \in \Delta(\mathcal{S})$ the initial state dis-

tribution or $\rho_0 \in \mathcal{S}$ the initial state; $\gamma \in [0, 1]$ the unnormalized discount factor. Policy optimization learns a stochastic policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ or a deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$, which is possibly parameterized by θ as π_θ, to decide what action the agent should take under a certain state. The agent starts at an initial state $s_0 \sim \rho_0$. Then at each step t, $t \geq 0$ the agent samples an action from $a_t \sim \pi(s_t)$, receives the reward $r_t \sim \mathcal{R}(s_t, a_t)$, and transitions to a subsequent state according to the Markovian dynamics $s_{t+1} \sim \mathcal{T}(s_t, a_t)$. The return is defined as the discounted cumulative reward as a random variable

$$R_t = \sum_{t=0}^{\infty} \gamma^t r_t.$$

The agent aims to maximize the expected return

$$J = \mathbb{E}_{s_t, a_t, r_t, t \geq 0}[\sum_{t=0}^{\infty} \gamma^t r_t].$$

Next we define a few important functions that are used in the thesis. Define the action-state value function

$$Q^\pi(s, a) = \mathbb{E}_{s_t, a_t, r_t, t \geq 0}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a]$$

to be the expected return of policy π at state s after taking action a. Also define the state-value function

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(s)}[Q^\pi(s, a)]$$

as the expected return given the initial state only, and the advantage function

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

as the difference between the action-value function and the state-value function. When the context is clear we omit the superscript π and write $Q(s, a)$, $V(s, a)$, and $A(s, a)$. Under the context of probably approximately correct learning, we use β instead of γ to denote the discount factor to avoid conflict with the notation of one-side testing probability γ.

□ **End of chapter.**

Chapter 3

Improving Sample Efficiency

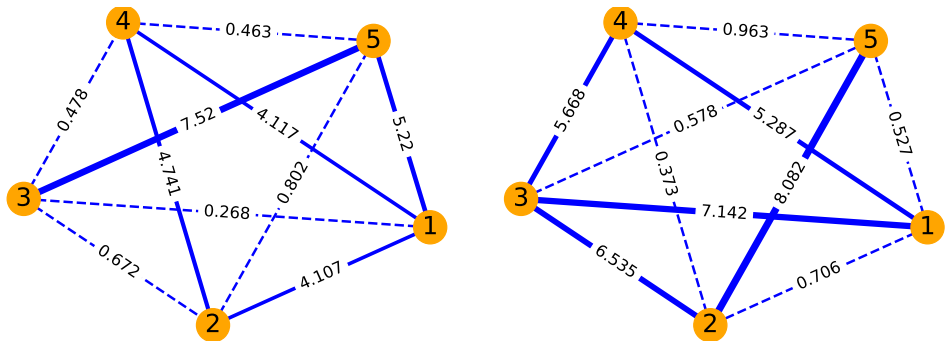
Policy gradient (PG) methods [46, 61] have been widely applied to various challenging problems including video games [57], robotics [138], and continuous control tasks [67, 68, 69]. It estimates the gradient of the expected cumulative reward directly from the rollouts of the agent trajectories. A major challenge of PG is the high variance of the gradient estimator. Since its inception, the community has been focusing on improving the PG estimator via a variety of variance reduction techniques [46, 57, 61, 62, 63, 64, 65, 66]. When dealing with high-dimensional continuous action spaces, methods based on the Rao-Blackwell theorem (RB) [139] demonstrate significant efficiency [70, 71, 140].

Motivated by the success of RB in high-dimensional spaces [140], we incorporate both RB and control variates (CV) into a unified framework. We present the action subspace dependent gradient (ASDG) estimator, which first breaks the original

high dimensional action space into several low dimensional action subspaces and replace the expectation (i.e., policy gradient) with its conditional expectation over subspaces (RB step) to reduce the sample space. A baseline function associated with each of the corresponding action subspaces is used to further reduce the variance (CV step). While ASDG is benefited from both RB and CV's ability to reduce the variance, we show that ASDG is unbiased under relatively weak conditions over the advantage function.

The key rationale of RB is to partition the action space into multiple subspaces, and estimate the conditional expectation respectively on each subspace. As shown later in this chapter, it is unbiased to decompose the PG estimator if the advantage function $A(s, a)$ can be partitioned into the direct sum $A(s, a_{(1)}) + \cdots + A(s, a_{(k)})$, where $a_{(j)}$ is the action subspace. Previous works [70, 71] have assumed the partition structure of the advantage function, which does not hold in general. Our aim is to learn this partition $\{a_{(j)}\}$ agnostically to elicit RB in the PG estimator.

We first rigorously formulate the decomposition of the action space. David et al. [141] have shown that a boolean function f has a direct sum decomposition $f(X, Y) = g(X) + h(Y)$ if and only if the dependence score $D = f(X, Y) - f(X', Y) - f(X, Y') + f(X', Y')$ is zero for all X, X', Y, Y'. We naturally extend this D to functions of real vectors and extend from testing $f = g + h$ to learning such g and h. Our analysis shows that



(a) Example 1 (b) Example 2

Figure 3.1: Illustrative examples of $a \in \mathbb{R}^5$. Each node represents one coordinate. Solid edges represent pairs of nodes with dependency where the line width scale with the dependency measure. The numbers on the edges are the exact dependency measure. Dashed lines represent nodes without dependency. The measure on the dashed line is nonzero which is analogous to the noise in the measure.

under the Monte-Carlo sampling of X, X', Y, Y', this estimator D is robust to the error. Namely, even if there exists a decomposition error of f, the characterization still holds approximately. Thus, we can find the direct sum decomposition of $A(s, a)$ and the corresponding action space partition $\{a_{(j)}\}$ by minimizing the expectation of D.

It suffices to find an efficient way to minimize the expectation of D. We propose two algorithms to achieve this. Our first algorithm is a greedy approach that leverages pairwise information. The algorithm builds a weighted complete graph at the beginning, where each node corresponds to one dimension and each edge is assigned the weight as the dependence score over the

Table 3.1: Comparisons of our algorithms with previous ones.

PG estimator	Variance	Partitioning	Guarantees	Limits
A2C [57]	CV	-	-	-
Wu et al. [70]	CV & RB	fully	no	$k = m$
POSA (ours) [71]	CV & RB	greedy	no	**no**
PE (ours) [98]	CV & RB	greedy	factor-$O(kn^2)$	**no**
SM (ours) [98]	CV & RB	**optimal**	**factor-4**	$k = 2$

two dimensions. The algorithm removes edges with the smallest weight until there are exactly k connected components. We show that the output partition is a factor-$O(kn^2)$ approximation of the best possible one, where n and k are the dimension and partition size, respectively. Our second algorithm works only when $k = 2$, but it improves the approximation to almost factor-4. The main insight behind this algorithm is that the dependence score is a submodular function of the bipartition. We can then find the bipartition efficiently by existing studies of submodular minimization algorithms [142, 143, 144]. Discussion on extending the algorithm to $k > 2$ is in the end of Section 3.4.5.

We present empirical results to corroborate our theoretical guarantee and to demonstrate the efficiency of the proposed algorithm. We evaluate our algorithm on both synthetic environments and a variety of high-dimensional continuous control tasks from OpenAI Gym. The algorithm consistently and significantly outperforms the baselines, and it has the best variance

reduction among the RB based methods. In practice, the algorithm is efficient and does not incur a notable computational cost.

Our contributions in this chapter are summarized as follows:

1. We propose the decomposed policy gradient algorithm. It elicits conditioning probability on the PG estimator and strictly reduces the variance of the estimator. The algorithm learns the PG decomposition agnostically, thus does not require assumptions on the action space structure.

2. We rigorously formulate the problem of partitioning the action space using a novel dependence score D. We show in Claim 11 and 12 that D is robust to error, namely, $\delta^p(\boldsymbol{X}, \boldsymbol{Y}) \leq D(\boldsymbol{X}, \boldsymbol{Y}) \leq 4\delta^p(\boldsymbol{X}, \boldsymbol{Y})$, where $\delta^p(\boldsymbol{X}, \boldsymbol{Y})$ is the ground truth of the partition error.

3. We propose a greedy-based algorithm such that it outputs a k-partition $\mathcal{P}$ efficiently. The partition error $\delta^p(\mathcal{P})$ is proved in Theorem 7 to be $O(kn^2)$.

4. We show that the expected error $\mathbb{E}[D]$ can be minimized efficiently with respect to the bipartition $\boldsymbol{X}, \overline{\boldsymbol{X}}$, as $\mathbb{E}[D(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$ is a submodular function, shown in Theorem 8.

3.1 Related works

3.1.1 Decomposition of action spaces

Several works share the similar high-level idea of decomposing the action space, though they implement the idea in preliminary approaches. [145] and [70] decompose the action space completely, where each subspace includes exactly one coordinate. Though the method demonstrates variance reduction on a variety of tasks, it relies on the assumption that all variables are independent with each other within the action space. Under a relatively weaker assumption, [71] uses the Hessian value to measure the pairwise dependency between variables and subsequently proposes a greedy-based algorithm. The algorithm depends on the heuristic that two variables are likely to be independent if the corresponding Hessian element is close to zero. While the Hessian element of two independent variables is zero, the converse does not hold. The comparisons between existing methods and our algorithm via heuristics (POSA), pairwise estimation (PE), and submodular minimization (SM) are summarized in Table 3.1.

Furthermore, previous approaches do not find the globally optimal partition, as they consider only local dependencies between pairs of coordinates. We show this local optimality on two examples of 5 coordinates in Figure 3.1. Both examples demonstrate a one-step MDP on $a \in \mathbb{R}^5$ where the objective is quadratic $a^T H a$. The edge between node i and j is weighted

by H_{ij}, and in this case partitioning A is equivalent to finding a minimum cut of the graph. In example 1 the optimal partition is $(1, 2, 4)$, $(3, 5)$ with 7.90 partition error. But the greedy algorithm yields the partition $(1, 3, 5)$, $(2, 4)$ with 11.66 partition error. Similarly, in example 2 the optimal partition is $(1, 3, 4)$, $(2, 5)$ with error 9.68 while the greedy algorithm finds $(1, 2, 3, 5)$, (4) with error 12.29. These examples are further explained in the experiments and in Appendix A.2.

3.1.2 Variance reduction methods

In practice, the vanilla policy gradient estimator is commonly estimated using Monte Carlo samples. A significant obstacle to the estimator is the sample efficiency. We review three prevailing variance reduction techniques in Monte Carlo estimation methods, including control variates, Rao-Blackwellization, and the reparameterization trick.

Control variates

Consider the case we estimate the expectation $\mathbb{E}_{p(x)}[h(x)]$ with Monte Carlo samples $\{x_i\}_{i=1}^{B}$ from the underlying distribution $p(x)$. Usually, the original Monte Carlo estimator has high variance, and the main idea of control variates is to find the proper baseline function $g(x)$ to partially cancel out the variance. A baseline function $g(x)$ with its known expectation over the dis-

tribution $p(x)$ is used to construct a new estimator

$$\hat{h}(x) = h(x) - \eta(g(x) - \mathbb{E}_p[g(x)]),$$

where η is a constant determined by the empirical Monte Carlo samples. The control variates method is unbiased but has a smaller variance $\mathrm{Var}(\hat{h}(x)) \leq \mathrm{Var}(h(x))$ at the optimal value $\eta^* = \frac{\mathrm{Cov}(h,g)}{\mathrm{Var}(g)}$.

Rao-Blackwellization

Though most of the recent policy gradient studies reduce the variance by control variates, the Rao-Blackwell theorem [139] decreases the variance significantly more than CV do, especially in high-dimensional spaces [140]. The motivation behind RB is to replace the expectation with its conditional expectation over a subset of random variables. In this way, RB transforms the original high-dimensional integration computation problem into estimating the conditional expectation on several low-dimensional subspaces separately.

Consider a simple setting with two random variable sets $\boldsymbol{A}$ and $\boldsymbol{B}$ and the objective is to compute the expectation $\mathbb{E}[h(\boldsymbol{A}, \boldsymbol{B})]$. Denote the conditional expectation $\hat{\boldsymbol{B}}$ as $\hat{\boldsymbol{B}} = \mathbb{E}[h(\boldsymbol{A}, \boldsymbol{B})|\boldsymbol{A}]$. The variance inequality

$$\mathrm{Var}(\hat{\boldsymbol{B}}) \leq \mathrm{Var}(h(\boldsymbol{A}, \boldsymbol{B}))$$

holds as shown in the Rao-Blackwell theorem. In practice, when $\boldsymbol{A}$ and $\boldsymbol{B}$ are in high dimensional spaces, the conditioning is

very useful and it reduces the variance significantly. The case of multiple random variables is hosted in a similar way.

Reparameterization trick

One of the recent advances in variance reduction is the reparameterization trick. It provides an estimator with lower empirical variance compared with the score function based estimators, as demonstrated in [146, 147]. Using the same notation as is in the control variates, we assume that the random variable x is reparameterized by $x = f(\theta, \xi), \xi \sim q(\xi)$, where $q(\xi)$ is the base distribution (e.g., the standard normal distribution or the uniform distribution). Under this assumption, the gradient of the expectation $\mathbb{E}_{p(x)}[h(x)]$ can be written as two identical forms i.e., the score function based form and reparameterization trick based form

$$\mathbb{E}_p[\nabla_\theta \log p(x)h(x)] = \mathbb{E}_q[\nabla_\theta f(\theta, \xi)\nabla_x h(x)]. \tag{3.1}$$

The reparameterization trick based estimator (the right-hand side term) has relatively lower variance. Intuitively, the reparameterization trick provides more informative gradients by exposing the dependency of the random variable x on the parameter θ.

3.1.3 Policy gradient methods

Previous attempts to reduce the variance mainly focus on the control variates method in the policy gradient framework (i.e.,

REINFORCE, A2C, Q-prop). A proper choice of the baseline function is vital to reduce the variance. The vanilla policy gradient estimator, REINFORCE [46], subtracts the constant baseline from the action-value function,

$$\nabla_\theta J(\theta)_{RF} = \mathbb{E}_\pi[\nabla_\theta \log \pi(a|s)(Q^\pi(s,a) - b)].$$

The estimator in REINFORCE is unbiased. The most important observation to conclude the unbiasedness is that the constant baseline function has a zero expectation with the score function. Motivated by this, the baseline function is set to be the value function $V^\pi(s)$ in the advantage actor-critic (A2C) method [57], as the value function can also be regarded as a constant under the policy distribution $\pi(a|s)$ with respect to the action a. Thus the A2C gradient estimator is

$$\begin{aligned}\nabla_\theta J(\theta)_{A2C} &= \mathbb{E}_\pi[\nabla_\theta \log \pi(a|s)(Q^\pi(s,a) - V^\pi(s))] \\ &= \mathbb{E}_\pi[\nabla_\theta \log \pi(a|s) A^\pi(s,a)].\end{aligned}$$

To further reduce the gradient estimate variance to acquire a zero-asymptotic variance estimator, [64] and [65] propose a general action dependent baseline function $b(s,a)$ based on the identity (3.1). Note that the stochastic policy distribution $\pi_\theta(a|s)$ is reparametrized as $a = f(\theta, s, \xi), \xi \sim q(\xi)$. We rewrite Equation (3.1) to get a zero-expectation baseline function as below

$$\mathbb{E}[\nabla_\theta \log \pi(a|s) b(s,a) - \nabla_\theta f(\theta, s, \xi) \nabla_a b(s,a)] = 0. \tag{3.2}$$

Incorporating with the zero-expectation baseline (3.2), the general action dependent baseline (GADB) estimator is formulated

as

$$\begin{aligned}\nabla_\theta J(\theta)_{GADB} = \mathbb{E}_\pi[\nabla_\theta \log \pi(a|s)(Q^\pi(s,a) - b(s,a)) \\ + \nabla_\theta f(\theta, s, \xi)\nabla_a b(s,a)].\end{aligned} \tag{3.3}$$

3.2 Decomposition of policy gradient

3.2.1 Construct the ASDG estimator

We present our action subspace dependent gradient (ASDG) estimator by applying RB on top of the GADB estimator. Starting with Equation (3.3), we rewrite the baseline function in the form of $b(s,a) = V^\pi(s) + c(s,a)$. The GADB estimator in Equation (3.3) is then formulated as

$$\begin{aligned}\nabla_\theta J(\theta)_{GADB} = \mathbb{E}_\pi[\nabla_\theta \log \pi(a|s)(A^\pi(s,a) - c(s,a)) \\ + \nabla_\theta f(\theta, s, \xi)\nabla_a c(s,a)].\end{aligned}$$

Assumption 1 (Advantage quadratic approximation)**.** *Assume that the advantage function $A^\pi(s,a)$ can be locally second-order Taylor expanded with respect to a at some point a^*, that is,*

$$\begin{aligned}A^\pi(a,s) \approx & A^\pi(a^*,s) + \nabla_a A^\pi(a,s)|_{a=a^*}^T (a-a^*) \\ & + \frac{1}{2}(a-a^*)^T \nabla_{aa} A^\pi(a,s)|_{a=a^*}(a-a^*).\end{aligned} \tag{3.4}$$

The baseline function $c(s,a)$ is chosen from the same family.

Assumption 2 (Variable partition)**.** *Assume that the row-switching transform of Hessian $\nabla_{aa} A^\pi(a,s)|_{a=a^*}$ is a block diagonal matrix $diag(M_1, \ldots, M_k)$, where $\sum_{k=1}^K dim(M_k) = m$.*

Based on Assumption 1 and 2, the advantage function $A^\pi(s,a)$ can be divided into K independent components

$$A^\pi(s,a) = \sum_{k=1}^{K} A_k^\pi(s, a_{(k)}),$$

where $a_{(k)}$ denotes the projection of the action a to the k-th action subspace corresponding to M_k. The baseline function $c(s,a)$ is divided in the same way.

Theorem 3 (ASDG Estimator)**.** *If the advantage function $A^\pi(s,a)$ and the baseline function $c(s,a)$ satisfy Assumption 1 and 2, the ASDG estimator $\nabla_\theta J(\theta)_{ASDG}$ is*

$$\begin{aligned}\nabla_\theta J(\theta)_{ASDG} = \sum_{k=1}^{K} \mathbb{E}_{\pi(a_{(k)}|s)}[\nabla_\theta \log \pi(a_{(k)}|s)(A^\pi(s,a_{(k)}) \\ - c(s,(a_{(k)}, \tilde{a}_{(-k)}))) - \nabla_\theta f_k(\theta,s,\xi)\nabla_{a_{(k)}} c_k(s,a_{(k)})],\end{aligned} \tag{3.5}$$

where $\nabla_\theta f(\theta,s,\xi) \in \mathbb{R}^{N_\theta \times m}$ is divided into K parts as $\nabla_\theta f = [\nabla_\theta f_1, ..., \nabla_\theta f_K]$ and N_θ is the dimension of θ.

Proof. Using the fact that

$$\mathbb{E}_{\pi(a|s)}[.] = \mathbb{E}_{\pi(a_{(k)}|s)}\mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)}[.],$$

where $a_{(-k)}$ represents the elements within a that are complementary to $a_{(k)}$. With the assumptions we have

$$\begin{aligned}&\nabla J(\theta)_{ASDG} \\ =&\mathbb{E}_{\pi(a_{(k)}|s)}\mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)}[(\nabla_\theta \log \pi(a_{(k)}|s) + \nabla_\theta \log \pi(a_{(-k)}|a_{(k)},s))\end{aligned}$$

$$
\begin{aligned}
&(A_k^\pi(s, a_{(k)}) + \sum_{i \neq k} A_i^\pi(s, a_{(i)}) - c_k(s, a_{(k)}) - \sum_{i \neq k} c_i(s, a_{(i)})) \\
&+ \sum_{k=1}^{K} \nabla_\theta f_k(s, a_{(k)}) \nabla_{a_{(k)}} c_k(s, a_{(k)})] \\
=&\mathbb{E}_{\pi(a_{(k)}|s)} \mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)} [\nabla_\theta \log \pi(a_{(k)}|s)(A_k^\pi - c_k) - \nabla_\theta f_k \nabla_{a_{(k)}} c_k] \\
&+ \mathbb{E}_{\pi(a_{(k)}|s)} \mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)} [\nabla_\theta \log \pi(a_{(k)}|s)(\sum_{i \neq k} A_i^\pi - \sum_{i \neq k} c_i)] \qquad (3.6) \\
&+ \mathbb{E}_{\pi(a_{(k)}|s)} \mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)} [\nabla_\theta \log \pi(a_{(-k)}|a_{(k)}, s)(A_k^\pi - c_k)] \qquad (3.7) \\
&+ \mathbb{E}_{\pi(a_{(k)}|s)} \mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)} [\nabla_\theta \log \pi(a_{(-k)}|a_{(k)}, s)((\sum_{i \neq k} A_i^\pi - \sum_{i \neq k} c_i)) \\
&- \sum_{i \neq k} \nabla_\theta f_i \nabla_{a_{(i)}} c_i] \\
\overset{(\clubsuit)}{=}&\mathbb{E}_{\pi(a_{(k)}|s)} [\nabla_\theta \log \pi(a_{(k)}|s)(A_k^\pi - c_k) \\
&- \nabla_\theta f_k \nabla_{a_{(k)}} c_k] \\
&+ \mathbb{E}_{\pi(a_{(-k)}|a_{(k)},s)} [\nabla_\theta \log \pi(a_{(-k)}|a_{(k)}, s)((\sum_{i \neq k} A_i^\pi - \sum_{i \neq k} c_i)) \\
&- \sum_{i \neq k} \nabla_\theta f_i \nabla_{a_{(i)}} c_i] \\
\overset{(\heartsuit)}{=}&\sum_{k=1}^{K} \mathbb{E}_{\pi(a_{(k)}|s)} [\nabla_\theta \log \pi(a_{(k)}|s)(A_k^\pi - c_k) - \nabla_\theta f_k \nabla_{a_{(k)}} c_k] \\
=&\sum_{k=1}^{K} \mathbb{E}_{\pi(a_{(k)}|s)} [\nabla_\theta \log \pi(a_{(k)}|s)(A_k^\pi + \sum_{i \neq k} A_i^\pi - c_k - \sum_{i \neq k} c_i) \\
&- \nabla_\theta f_k \nabla_{a_{(k)}} c_k]
\end{aligned}
$$

$$= \sum_{k=1}^{K} \mathbb{E}_{\pi(a_{(k)}|s)}[\nabla_\theta \log \pi(a_{(k)}|s)(A^\pi(s,a) - c(s, a_{(k)}, \tilde{a}_{(-k)}))$$
$$- \nabla_\theta f_k \nabla_{a_{(k)}} c_k], \tag{3.8}$$

where (♣) holds as term (3.6) and term (3.7) are equal to zero (using the property that the expectation of the score function is zero) and (♥) is expanded by induction. □

Our assumptions are relatively weak compared with previous studies on variance reduction for policy optimization. Different from the fully factorization policy distribution assumed in [70], our method relaxes the assumption to the constraints on the advantage function $A^\pi(s,a)$ with respect to the action space instead. Similar to that, we just use this assumption to obtain the structured factorization action subspaces to invoke the Rao-Blackwellization and our estimator does not introduce additional bias.

Connection with other works

If we assume the Hessian matrix of the advantage function has no block diagonal structure under any row switching transformation (i.e., $K = 1$), ASDG in Theorem 3 is the one inducted in [64] and [65]. If we otherwise assume that Hessian is diagonal (i.e., $K = m$), the baseline function $c(s, a_{(k)}, \tilde{a}_{(-k)})$ equals $\sum_{i \neq k} c_i(s, a_{(i)})$, which means that each action dimension is independent with its baseline function. Thus, the estimator in [70] is obtained.

Selection of the baseline functions $c(s, a)$

Two approaches exist to find the baseline function, including minimizing the variance of the PG estimator or minimizing the square error between the advantage function and the baseline function [64, 65]. Minimizing the variance is hard to implement in general, as it involves the gradient of the score function with respect to the baseline function parameter. In our work, we use a neural network advantage approximation as our baseline function by minimizing the square error. Under the assumption that the variance of reparametrization term $\nabla_\theta f_k(\theta, s, \xi)\nabla_{a_{(k)}} c_k(s, a_{(k)})$ is close to zero, the two methods yield the same result.

3.2.2 Variance reduction via Rao-Blackwellization

The following proposition is a standard argument on conditional probabilities (also known as Rao-Blackwellization), which indicates that the decomposed score function estimator has lower variance than the original one.

Proposition 4 (Conditioning and Rao-Blackwellization)**.** *Let* (X, Y) *have joint distribution* f *and let* $f(X, Y)$ *satisfy* $\mathrm{Var}[f(X, Y)] < +\infty$. *Define* $h(X') = \mathbb{E}[f(X, Y)|X = X']$ *for* $X, Y \sim \mathbb{P}_f$. *Suppose that* $X_i, Y_i \sim \mathbb{P}_f$. *Then, we have*

$$\mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} h(X_i)] \leq \mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} f(X_i, Y_i)].$$

Combining both the definition of the advantage decomposition and the variance reduction proposition, we have the follow-

ing lemma. The key observation to conclude the lemma is the property of the score function $\mathbb{E}_{p(X,Y)}[\nabla \log p(X,Y)] = 0$. We include the full proof in the appendix.

Proposition 5. *Suppose that the function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ satisfies zero partition error in Assumption 2 with two blocks $X \in \mathbb{R}^n$ and $Y \in \mathbb{R}^m$ (i.e., $f(X,Y) = f_X + f_Y$). $P(X,Y)$ is a distribution enjoying the independence structure $P(X,Y) = P(X)P(Y)$ and $\mathbb{E}_{P(X,Y)}[\|f(X,Y)\|^2] < +\infty$. Then, we have*

- *(Unbiased Estimator) Consider the estimation problem $\mathbb{E}_{P(X,Y)}[\nabla \log P(X,Y) f(X,Y)]$. Then*

$$\begin{aligned}\mathbb{E}_{P(X,Y)}[\nabla_\theta \log P(X,Y) f(X,Y)] =& \mathbb{E}_{P(X)}[\nabla_\theta \log P(X) f_X] \\ &+ \mathbb{E}_{P(Y)}[\nabla_\theta \log P(Y) f_Y].\end{aligned}$$

- *(Variance Reduction) For the two estimators $g(X,Y) = \nabla \log P(X,Y) f(X,Y)$ and $h(X,Y) = \nabla_\theta \log P(X) f_X + \nabla \log P(Y) f_Y$,*

$$\mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} h(X_i, Y_i)] \le \mathrm{Var}[\frac{1}{n}\sum_{i=1}^{n} g(X_i, Y_i)],$$

where n is the number of Monte-Carlo samples.

Proof. To verify the unbiasedness, observe that

$$\begin{aligned}\mathbb{E}_{P(X,Y)}[\nabla \log P(X,Y) f(X,Y)] =& \mathbb{E}_{P(X)P(Y)}[(\nabla \log P(X) \\ &+ \nabla \log P(Y))(f_X + f_Y)] \\ &= \mathbb{E}_{P(X)}[\nabla \log P(X) f_X]\end{aligned}$$

$$+ \mathbb{E}_{P(Y)}[\nabla \log P(Y) f_Y],$$

which holds by the property of the score function that $\mathbb{E}[\nabla \log P(X)] = 0$. Define the conditional estimator $\tau_X(X')$ as

$$\begin{aligned}\tau_X(X') =& \mathbb{E}_{P(Y)}[\nabla_\theta \log P_\theta(X,Y) f(X,Y) | X = X'] \\ =& \mathbb{E}_{P(Y)}[(\nabla \log P(X) + \nabla \log P((Y)))(f_X + f_Y) | X = X'] \\ =& \nabla \log P(X') f_{X'} + \mathbb{E}_{P(Y)}[\nabla \log P(Y) f_Y] \\ &+ \nabla_\theta \log P(X') \mathbb{E}_{P(Y)}[f_Y].\end{aligned} \tag{3.9}$$

Similarly, also define $\tau_Y(Y') = \mathbb{E}_{P(X)}[\nabla \log P(X,Y) f(X,Y) | Y = Y']$. Without loss of generality, we assume that $\mathrm{Var}[\tau_X(X')] < \mathrm{Var}[\tau_Y(Y')]$. Since the original estimator $g(X,Y)$ is sampled from the joint distribution $P(X,Y)$, the decomposed estimator $h(X,Y)$ does not introduce additional variance on top of $\mathrm{Var}[\tau_Y(Y')]$. Thus,

$$\mathrm{Var}[h(X,Y)] = \max(\mathrm{Var}[\tau_X(X')], \mathrm{Var}[\tau_Y(Y')]).$$

Combining with the Rao-Blackwellization theorem, we have $\max($ $\mathrm{Var}[\tau_X(X')], \mathrm{Var}[\tau_Y(Y')]) \leq \mathrm{Var}[g(X,Y)]$. The lemma follows. □

We substitute $p(X,Y)$ with $\pi(a_{(1)}, a_{(2)}|s)$, and $f(X,Y)$ with $A(a_{(1)}, a_{(2)}, s)$ in Proposition 5 and yield that the variance of $\nabla_\theta J(\theta)$ is strictly less than policy gradient estimator proposed in [57], [64] and [65].

Proposition 6. *When the advantage function can be decomposed into multiple components, the variance of the estimator* (3.5) *is strictly less than the Monte-Carlo policy gradient estimator proposed in [57], [64], and [65].*

Proof. Recall that

$$\nabla_\theta J(\theta) = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)}) - c(s, (a_{(j)}, \tilde{a}_{(-j)}))) - \nabla_\theta f_j(\theta, s, \xi)\nabla_{a_{(j)}} c_j(s, a_{(j)})].$$

We ignore the term $\nabla_\theta f_j(\theta, s, \xi)\nabla_{a_{(j)}} c_j(s, a_{(j)})$ as the variance of a reparameterization term is close to zero. Thus we focus on the following estimator

$$\sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A^\pi(s, a_{(j)}) - c(s, (a_{(j)}, \tilde{a}_{(-j)})))].$$

By the second statement of Proposition 5, we plug in the $f(a)$ as $A^\pi(s, a) - c(s, a)$. It concludes immediately the desired result. □

3.3 Heuristic approach using second-order advantage information

When implementing the ASDG estimator, Temporal Difference (TD) learning methods such as generalized advantage estimation (GAE) [148, 149] allow us to obtain the estimation $\hat{A}(s, a)$ based

on the value function $V^w(s)$ via

$$\hat{A}(s_t, a_t) = \sum_{t' \geq t}^{T} (\lambda\gamma)^{t'-t} \delta_{t'}, \tag{3.10}$$

where

$$\delta_t = \mathbb{E}[r_t + \gamma V^w(s_{t+1}) - V^w(s_t)], \tag{3.11}$$

and λ is the discount factor of the λ-return in GAE. GAE further reduces the variance and avoids the action gap at the cost of a small bias.

Obviously, we cannot obtain the second-order information $\nabla_{aa} A(s, a)$ with the advantage estimation in GAE identity (3.10). Hence, apart from the value network $V^w(s)$, we train a separate advantage network to learn the advantage information. The neural network approximation $A^\mu(s, a)$ is used to smoothly interpolate the realization values $\hat{A}(s, a)$, by minimizing the square error

$$\min_{\mu} ||\hat{A}(s, a) - A^\mu(s, a)||^2. \tag{3.12}$$

As shown in Assumption 2, we use the block diagonal matrix to approximate the Hessian matrix and subsequently obtain the structure information in the action space. In the above advantage approximation setting, the Hessian computation is done by first approximating the advantage realization value and then differentiating the advantage approximation to obtain an approximate Hessian. However, for any finite number of data points there exists an infinite number of functions, with arbitrarily

satisfied Hessian and gradients, which can perfectly approximate the advantage realization values [150]. Optimizing such a square error objective leads to unstable training and is prone to poor results. To alleviate this issue, we propose a novel wide & deep architecture [151] based advantage net. In this way, we divide the advantage approximator into two parts, including the quadratic term and the deep component, as

$$A^{\mu}(s,a) = \beta_1 \cdot A_{wide} + \beta_2 \cdot A_{deep},$$

where β_1 and β_2 are the importance weights. Subsequently, we make use of factorization machine (FM) model as our wide component

$$A_{wide}(s,a) = w_0(s) + w_1(s)^T a + w_2(s)w_2(s)^T \odot aa^T,$$

where $w_0(s) \in \mathbb{R}$, $w_1(s) \in \mathbb{R}^m$ and $w_2(s) \in \mathbb{R}^{m \times m'}$ are the coefficients associated with the action. Also, m' is the dimension of latent feature space in the FM model. Note that the Hadamard product $A \odot B = \sum_{i,j} A_{ij} B_{ij}$. To increase the signal-to-noise ratio of the second-order information, we make use of wide components Hessian $w_2(s)w_2(s)^T$ as our Hessian approximator in the heuristic algorithm. The benefits are two-fold. On the one hand, we can compute the Hessian via the forward propagation with low computational costs. On the other hand, the deep component involves large noise and uncertainties and we obtain stable and robust Hessian by excluding the deep component from calculating Hessian.

The Hessian matrix contains both positive and negative values. However, we concern only the pairwise dependency between the action dimensions, which can be directly represented by the absolute value of Hessian. For instance, considering a quadratic function $f(x) = a + b^T x + x^T C x, x \in \mathbb{R}^m$, it can be written as $f(x) = a + \sum_i b_i x_i + \sum_{i,j} C_{ij} x_i x_j$. The elements in the Hessian matrix satisfy $\frac{\partial^2 f(x)}{\partial x_i \partial x_j} = C_{ij}$. When C_{ij} is close to zero, x_i and x_j are close to be independent. Thus we can decompose the function $f(x)$ accordingly and optimize the components separately.

We modify the evolutionary clustering algorithm in [152] by using the absolute approximating Hessian $|w_2(s)w_2(s)^T|$ as the affinity matrix in the clustering task. In other words, each row in the absolute Hessian is regarded as a feature vector of that action dimension when running the clustering algorithm. With the evolutionary clustering algorithm, our policy optimization with second-order advantage information algorithm (POSA) is described in Algorithm 1.

3.4 Rigorous approach using variable partitioning

Divide-and-conquer methods like our decomposed policy gradient rely on the ability to identify independent sub-instances of a given instance, such as connected components of graphs and hypergraphs. When these are not available one looks for partitions into loosely related parts like small or sparse cuts. These

Algorithm 1 Policy optimization with second-order advantage information (POSA)

Input: Number of iterations N, number of value iterations M_w, batch size B, number of subspaces K, initial policy parameter θ, initial value and advantage parameters w and μ;
Output: Policy optimal parameter θ;
for each iteration n in $[N]$ **do**
    Collect a batch of trajectory data $\{s_t^{(i)}, a_t^{(i)}, r_t^{(i)}\}_{i=1}^{B}$;
    for M_θ iterations **do**
        Update θ by one SGD step using PPO with ASDG in Theorem (3);
    end for
    for M_w iterations **do**
        Update w and μ by minimizing $||V^w(s_t) - R_t||_2^2$ and $||\hat{A}(s_t, a_t) - A^\mu(s_t, a_t)||_2^2$ in one SGD step;
    end for
    Estimate $\hat{A}(s_t, a_t)$ using $V^w(s_t)$ by GAE (3.10);
    Calculate the action subspace partition $a_{(k)}$ based on the absolute Hessian $|w_2(s)w_2(s)^T|$ by the evolutionary clustering algorithm;
end for

classic problems and their variants remain at the forefront of algorithmic research [153, 154, 155, 156, 157, 158].

We study the related problem of function decomposition: Given a multivariate function $F(\boldsymbol{V})$ over n variables $\boldsymbol{V} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, we seek to partition the variables into k groups $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ so that F decomposes into a sum $F_1(\boldsymbol{X}_1) + \cdots + F_k(\boldsymbol{X}_k)$. In case an exact decomposition of this type is unavailable, we seek an approximate one under a suitable error metric. This algebraic partitioning question can be sensibly asked for any Abelian group. While some of our results are quite general, two particular cases of interest are addition over $\mathbb{Z}_2$ with respect to the Hamming metric and addition over reals with respect to the 2-norm.

As a multivariate function is an exponentially large object, it is sensible to model the input F to the partitioning problem as an oracle and allow query access to it. This departs from the common setup in (hyper)graph partitioning problems, where an explicit representation of the input is assumed to be available. While variable partitioning of real-valued functions under the 2-norm turns out to be closely related to hypergraph partitioning, the difference in input access models renders certain techniques developed for the latter (e.g., random contractions) inapplicable to our setting.

Our main results are algorithmic: We show that variable partitions can be learned agnostically.

Let $F(\boldsymbol{V})$ be a function from some product set to an Abelian

group G. A direct sum decomposition of F is a partition $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ of the set of variables $\boldsymbol{V}$ such that $F(\boldsymbol{V})$ is $F_1(\boldsymbol{X}_1) + \cdots + F_k(\boldsymbol{X}_k)$ for some functions $F_1, \ldots, F_k$. When the decomposition is imperfect, the decomposition error is measured by

$$\begin{aligned}\delta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) = \min_{F_1,\ldots,F_k} \|F(X_1, \ldots, X_k) \\ - F_1(X_1) - \cdots - F_k(X_k)\|,\end{aligned} \tag{3.13}$$

where $\|\cdot\| : G \to \mathbb{R}^+$ is a partial norm. The definition is given in Section 3.4.3; the main examples of interest are $G = \mathbb{Z}_q$ under the Hamming metric $\|F\| = \mathbb{P}(F(V) = 0)$ and $G = \mathbb{R}$ under the p-norm $\|F\|_p = \mathbb{E}\big[|F(V)|^p\big]^{1/p}$ for any $p \geq 1$ under some product measure. We seek an approximation of the best-possible partition, which minimizes the objective

$$\delta_2(F) = \min_{\boldsymbol{X}} \delta(\boldsymbol{X}, \overline{\boldsymbol{X}}), \tag{3.14}$$

for bipartition and

$$\delta_k(F) = \min_{\boldsymbol{X}_1,\ldots,\boldsymbol{X}_k} \delta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k). \tag{3.15}$$

for k-partition. (For p-norms over $\mathbb{R}$ we use the notations $\|\cdot\|_{\mathbb{R},p}$, $\delta_{\mathbb{R},2}(F)$, and $\delta_{\mathbb{R},k}(F)$.)

Theorem 7. *Let $\|\cdot\|$ be either 1) $\|\cdot\|_{\mathbb{R},p}$ assuming $\|F\|_{\mathbb{R},2p} = O(1)$, or 2) the Hamming metric over $\mathbb{Z}_q$. There is an algorithm that given parameters n, k, ε, γ, and oracle access to $F\colon \Sigma^n \to G$ outputs a k-partition $\mathcal{P}$ such that $\delta(\mathcal{P}) \leq O(kn^2)(\delta_k(F) + \varepsilon)$ with probability at least $1 - \gamma$. The algorithm makes $O(K^p n^2 \log(n/\gamma)/\varepsilon^{2p})$ queries to F and runs in time linear in the number of queries, for an absolute constant K.*

This algorithm is closely related to the heuristic ones used in the aforementioned empirical studies. However, it only guarantees optimality up to an $O(kn^2)$ approximation factor. While we do not know if an approximation factor of this magnitude is inevitable, we showed in [98] that obtaining a solution with additive error is NP-hard. The proofs are given in Section 3.5.

In contrast, our second algorithm obtains an additive error for bipartitions of real-valued functions under the 2-norm:

Theorem 8. *Let $F\colon \Sigma^n \to \mathbb{R}$ be a function with $\|F\|_{\mathbb{R},4} \leq 1$. There is an algorithm that given inputs n, ε, γ, and oracle access to F, runs in time $O(n^5 \log(n/\gamma)/\varepsilon^2)$ and outputs a bipartition $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ such that $\delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2 \leq \delta_{\mathbb{R},2}(F)^2 + \varepsilon$ with probability at least $1 - \gamma$.*

More generally, we show that it is possible to output a $\sqrt{2 - 2/k}$-approximate k-partition in time $\mathrm{poly}(n^k, k, 1/\varepsilon)$ (Corollary 20). For unbounded k finding a good approximation is ETH hard (Corollary 18).

Theorem 8 and Corollary 18 are based on an equivalence between variable partitioning under the 2-norm and hypergraph partitioning given in Proposition 17. The results are described and proved in Section 3.4.5.

As a consequence of Theorem 7, the property of being close to a k-partition is testable with $\tilde{O}(k^{2p} n^{4p+2}/\varepsilon^{2p})$ queries, shown in [98]. The query complexity of the tester can be somewhat improved:

Notation	Meaning	Notation	Meaning
$\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{V}$	variables	$\delta_k(F)$	optimal k-partition error
$\boldsymbol{X}, \boldsymbol{Y}, \overline{\boldsymbol{X}} \subseteq \boldsymbol{V}$	sets of variables	$D_F(\boldsymbol{X}, \boldsymbol{Y})$	dependence score
x, y, X, Y	assignments	$\|\cdot\|, \|\cdot\|_{\mathbb{R},p}$	partial norm and p-norm

Theorem 9. *k-partitionability is testable with one-sided error and $O(kn^3/\varepsilon)$ non-adaptive queries with respect to Hamming weight over $\mathbb{Z}_q$, and with $O(k^{2p}n^3/\varepsilon^{2p})$ non-adaptive queries with respect to the p-norm over $\mathbb{R}$ assuming $\|F\|_{2p} \leq 1$.*

We plug our partitioning algorithm back to reinforcement learning control. In this setting, the oracle is real-valued and as we adapt the 2-norm we use the submodularity cut algorithm described in Theorem 8.

We compare empirically with three previous approaches. The first baseline does not involve partitioning [46, 57]. The second baseline trivially partitions n variables into n subsets [70, 145]. The third baseline partitions the variables using heuristic methods [71]. The way [71] partitions the variables is to calculate the discrete estimate of the Hessian of the oracle. Then they remove from Hessian the elements with lowest absolute values, until it forms at least k connected components if the Hessian matrix is treated as the adjacency matrix.

The scores we attained on the tasks in the physics simulator are improved over these approaches, which is demonstrated in Section 3.6.3.

Algorithm 2 Policy optimization with variable partitioning

Input: Total number of samples T, batch size B, partition frequency M_p, number of value iterations M_w, initial policy parameter θ, initial value and advantage parameters w and μ;
Output: Optimized policy π_θ;
for each iteration j in $[T/B]$ **do**
    Collect a batch of trajectory data $\{s_t^{(i)}, a_t^{(i)}, r_t^{(i)}\}_{i=1}^B$;
    for M_θ iterations **do**
        Update θ by one gradient descent step using proximal policy gradient with the gradient estimator (3.5);
    end for
    for M_w iterations **do**
        Update w and μ by minimizing $\|V^w(s_t) - R_t\|_2^2$ and $\|\hat{A} - A^\mu(s_t, a_t)\|_2^2$ in one step;
    end for
    Estimate $\hat{A}(s_t, a_t)$ using $V^w(s_t)$ by generalized advantage estimator;
    if $j \equiv 0 \pmod{M_p}$ **then**
        Define random function $\xi(\boldsymbol{X})$ to be an estimation of $\mathbb{E}[D_A(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$;
        Run submodular minimization over $\boldsymbol{X}$ on $\xi(\boldsymbol{X})$;
        Assign $\boldsymbol{X}$ and $\bar{\boldsymbol{X}}$ to $a_{(1)}$ and $a_{(-1)}$ in (3.5), respectively;
    end if
end for

3.4.1 Ideas and techniques

Our Theorem 7 is inspired by algebraic property testing techniques. The starting point is the dual characterization of partitionability into sets $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ by the constraints $D_F(\boldsymbol{X}, \boldsymbol{Y}) = 0$, where $D_F = F(X, Y) - F(X', Y) - F(X, Y') + F(X', Y')$, for all assignments X, X' to $\boldsymbol{X}$ and Y, Y' to $\boldsymbol{Y}$. David et al. [141] apply this relation to random inputs towards testing whether a $\mathbb{Z}_2$-valued function F tensor decomposes into a direct sum. The acceptance probability of this test approximates the best decomposition to within a factor of 4 (Proposition 10).

Our partitioning algorithm estimates the dependence score $\|D_F(\boldsymbol{x}, \boldsymbol{y})\|$ on every pair of variables $\boldsymbol{x}, \boldsymbol{y}$ (keeping the rest fixed) to decide whether they should be partitioned or not. Here, $\|D_F\|$ is the probability that the test $D_F = 0$ fails for discrete groups like $\mathbb{Z}_2$. In general, it can represent any error metric satisfying the axioms in Section 3.4.3. The proof of Theorem 7 amounts to showing that a collection of single variable partitions $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{P}$ for which the local scores $\|D_F(\boldsymbol{x}, \boldsymbol{y})\|$ are small can be glued together into a single k-partition $\mathcal{P}$ with a small global score.

When F is real-valued and error is measured under the 2-norm, variable partitioning has a natural geometric interpretation. Functions that depend on different coordinates are orthogonal modulo their constant term, so the optimal decomposition with respect to a fixed partition $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ is given by

the projection of F onto the respective subspaces of functions. This yields an equality between the distance and the dependence score (3.18) for bipartitions and a generalization to k-partitions (Proposition 14). Variable partitioning for functions is then equivalent to hypergraph partitioning of their orthogonal decompositions (Proposition 17), with the cost of cut $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ given by $\frac{1}{4}\|D_F(\boldsymbol{x}, \boldsymbol{y})\|^2$.

This connection suggests the application of hypergraph partitioning algorithms that can be implemented with access to an approximate cut oracle,[1] leading to Theorem 8. On the negative side it reveals that approximately optimal partitions into a large number of components are hard to find (Corollary 18).

3.4.2 Relation to other learning and testing problems

A j-junta is a function that depends on at most j of its n variables. The problems of learning and testing juntas have been extensively studied [161, 162, 163, 164, 165, 166, 167]. While a j-junta is always $(n-j+1)$-partitionable, the two problems are technically incomparable. Moreover, juntas are usually studied in the regime where the junta size j is significantly smaller than the number of variables n and are therefore partitionable into many (mostly trivial) components. In this work we are

[1] Several state-of-the-art algorithms for cuts in graphs and hypergraphs rely on random contractions [159, 160, 157]. In particular, Rubinstein et al. [158] showed that $\tilde{O}(n)$ queries to an *exact* cut oracle and similar running time are sufficient to find the minimum cut. We do not know if comparable efficiency can be obtained with an approximate oracle.

mostly interested in partitions into two or a small number of components. Nevertheless, this connection between juntas and partitionable functions is used to prove the testing lower bound in our paper [98].

Dinur and Golubev [168] showed that the existence of decomposition with respect to a fixed k-partition (given as input) is testable with four queries and soundness error $\Omega(\delta)$. The case $k = 2$ was already analyzed by David et al. [141] (see Section 3.4.4).

3.4.3 Some additional definitions

Let $F(\boldsymbol{V})$ be a function from some product set to an Abelian group G. In general we will assume that the variables $\boldsymbol{V}$ take values in some set Σ endowed with a product measure which is efficiently sampleable. The quality of the partition $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ of $\boldsymbol{V}$ is measured by $\delta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ given in (3.13), where $\|\cdot\| : G \to \mathbb{R}^+$ can be any functional satisfying the following three axioms:

1. $\|0\| = 0$;
2. $\|F_1 + F_2\| \leq \|F_1\| + \|F_2\|$;
3. $\mathbb{E}[\|F(X, \cdot)\| \mid X] \leq \|F\|$ for any set of variables X of F.

Our goal is to approximately optimize $\delta_2(F)$ in (3.14) and $\delta_k(F)$ in (3.15).

Our algorithms are based on the following dependence estimator inspired by the rank-1 test of [141]. Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two disjoint sets of variables. The dependence estimator $D_F(\boldsymbol{X}, \boldsymbol{Y})$ is the random variable

$$D_F = F(X, Y, Z) + F(X', Y', Z) - F(X', Y, Z) - F(X, Y', Z),$$

where X, X' are independent samples of the $\boldsymbol{X}$ variable, Y, Y' are independent samples of the $\boldsymbol{Y}$ variable, and Z is a random sample of the remaining variables. If F decomposes into a direct sum that partitions the $\boldsymbol{X}$ and $\boldsymbol{Y}$ variables then D_F equals zero. Conversely, $\|D_F\|$ measures the quality of the approximation.

In the analysis it will be convenient to use the notation $F \approx_\delta G$ for $\|F - G\|_p \leq \delta$. The following two facts are immediate consequences of axioms 2 and 3:

Triangle inequality: If $F \approx_\delta G$ and $G \approx_{\delta'} H$ then $F \approx_{\delta+\delta'} H$.

Fixing: If $F(X, Z) \approx_\delta G(X, Z)$ then $F(\underline{X}, Z) \approx_\delta G(\underline{X}, Z)$ for some fixed value $\underline{X}$.

3.4.4 Estimating the quality of a partition

In this section we show that $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|$ is an approximate estimator for the quality $\delta(\boldsymbol{X}, \boldsymbol{Y})$ of a decomposition, namely

$$\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq \|D_F(\boldsymbol{X}, \boldsymbol{Y})\| \leq 4 \cdot \delta(\boldsymbol{X}, \boldsymbol{Y}). \tag{3.16}$$

The proof is given in Claims 11 and 12 below. As $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|$ can be estimated efficiently from oracle access to F (Claim 13),

we obtain an algorithm for estimating the quality of a partition to within a factor of 4 in general, and exactly for the 2-norm over $\mathbb{R}$.

Proposition 10. *Let $\|\cdot\|$ be either 1) $\|\cdot\|_{\mathbb{R},p}$ assuming $\|F\|_{\mathbb{R},2p} = O(1)$, or 2) the Hamming metric over $\mathbb{Z}_q$. There is an algorithm that given a bipartition $\boldsymbol{X}, \boldsymbol{Y}$ of the variables and parameters $\varepsilon, \gamma > 0$, outputs a value $\hat{\delta}$ such that*

$$\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq \hat{\delta} \leq 4 \cdot \delta(\boldsymbol{X}, \boldsymbol{Y}) + \varepsilon,$$

with probability at least $1 - \gamma$ from $K^p \log(1/\gamma)/\epsilon^{2p}$ queries to F in time linear in the number of queries, for an absolute constant K.

The value of $\delta(\boldsymbol{X}, \boldsymbol{Y})$ is known to be NP-hard to calculate exactly over $\mathbb{Z}_2$ under the Hamming metric given explicit access to the truth-table of F [169]. Therefore some approximation factor is unavoidable for algorithms running in time polynomial in n and $1/\varepsilon$ unless BPP is in NP. On the positive side Karpinski and Schudy [170] give a fully polynomial-time randomized approximation scheme for this special case. Their algorithm requires at least linear time but it is plausible that a sublinear-time variant can be obtained. However, it appears unrelated to the dependence score D_F which plays an essential role in the results to follow.

The analysis of D_F applies to any pair of disjoint subsets $\boldsymbol{X}$, $\boldsymbol{Y}$ that do not necessarily partition all the variables. In this

more general setting distance is measured by the formula

$$\delta(\boldsymbol{X}, \boldsymbol{Y}) = \min_{A,B} \|F(X, Y, Z) - A(X, Z) - B(Y, Z)\|. \quad (3.17)$$

Claim 11 (Completeness of D_F)**.** *For all disjoint* $\boldsymbol{X}$*,* $\boldsymbol{Y}$*,* $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\| \leq 4 \cdot \delta(\boldsymbol{X}, \boldsymbol{Y})$.

Proof. By definition of $\delta(\boldsymbol{X}, \boldsymbol{Y})$ there exists a decomposition of the form

$$F(X, Y, Z) = A(X, Z) + B(Y, Z) + D(X, Y, Z),$$

where $\|D(X, Y, Z)\| = \delta(\boldsymbol{X}, \boldsymbol{Y})$. In the expansion of D_F all the A and B terms cancel out, leaving

$$\begin{aligned} \|D_F(\boldsymbol{X}, \boldsymbol{Y})\| =& \|D(X, Y, Z) + D(X', Y', Z) - D(X, Y', Z) \\ & - D(X', Y, Z)\| \\ \leq& \|D(X, Y, Z)\| + \|D(X', Y', Z)\| \\ & + \|D(X, Y', Z)\| + \|D(X', Y, Z)\| \\ =& 4\delta(\boldsymbol{X}, \boldsymbol{Y}). \end{aligned}$$

□

Soundness for Boolean functions under the uniform measure was proved by David et al. [141]. We reproduce their proof under a more general setting.

Claim 12 (Soundness of D_F)**.** *For all disjoint* $\boldsymbol{X}$*,* $\boldsymbol{Y}$*,* $\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq \|D_F(\boldsymbol{X}, \boldsymbol{Y})\|$.

Proof. Let $\varepsilon = \|D_F(\boldsymbol{X}, \boldsymbol{Y})\|$. Then

$$F(X, Y, Z) \approx_\varepsilon F(X, Y', Z) - F(X', Y, Z) - F(X', Y', Z).$$

We can fix values $\underline{X}'$ and $\underline{Y}'$ for which

$$\begin{aligned} F(X,Y,Z) \approx_\varepsilon & F(\underline{X}',\underline{Y}',Z) - F(X,\underline{Y}',Z) - F(\underline{X}',Y,Z) \\ = & A(X,Z) + B(Y,Z), \end{aligned}$$

where $A(X,Z) = F(\underline{X}',\underline{Y}',Z) - F(X,\underline{Y}',Z)$ and $B(Y,Z) = F(\underline{X}',Y,Z)$. □

Proposition 10 now follows from inequality (3.16) and the following claim, which states that $\|D_F\|$ can be estimated by sampling in the cases of interest. See Appendix A.1 for the proof.

Claim 13. *Assuming $\|F\|_{\mathbb{R},2p} \leq 1$, the value $\|F\|_{\mathbb{R},p}$ can be estimated within ε from $K^p \log(1/\gamma)/\epsilon^{2p}$ (random) queries to F in linear time with probability $1-\gamma$ for some absolute constant K.*

Exact partitioning under the 2-norm

Since computing the optimal partition is in general NP-complete, we do not expect to replace the inequalities in (3.16) with an equality. However, in the special case of real-valued functions with 2-norm, the estimate becomes exact:

$$\|D_F(\boldsymbol{X},\boldsymbol{Y})\|_{\mathbb{R},2} = 2 \cdot \delta_{\mathbb{R},2}(\boldsymbol{X},\boldsymbol{Y}). \tag{3.18}$$

This equality is a consequence of the following characterization of $\delta_{\mathbb{R},2}$, which applies more generally to k-partitions:

Proposition 14. *Assuming $\mathbb{E}[F] = 0$, the k-partition $F_i(X_i) = \mathbb{E}[F|X_i]$ achieves the minimum for $\delta_{\mathbb{R},2}(\boldsymbol{X}_1,\ldots,\boldsymbol{X}_k)$.*

In particular it follows that $\delta_{\mathbb{R},2}$ takes the value

$$\delta_{\mathbb{R},2}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) = \mathbb{E}[(\overline{F} - \mathbb{E}[\overline{F}|X_1] - \cdots - \mathbb{E}[\overline{F}|X_k])^2], \quad (3.19)$$

where $\overline{F} = F - \mathbb{E}[F]$. To derive identity (3.18) it remains to verify that when $k = 2$, the right-hand side of (3.19) is a quarter of $\|D_F\|^2$:

Fact 15. $\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|^2_{\mathbb{R},2} = 4 \cdot \mathbb{E}[(\overline{F} - \mathbb{E}[\overline{F}|X] - \mathbb{E}[\overline{F}|Y])^2]$.

Armed with this fact we prove the proposition.

Proof of Proposition 14. First assume $F(\boldsymbol{X}, \boldsymbol{Y})$ is bivariate. Let $A(\boldsymbol{X})$ be any function. The inequality $\mathbb{E}[(\mathbb{E}[F|X] - A(X))^2] \geq 0$ can be rewritten as

$$\mathbb{E}[(F - \mathbb{E}[F|X])^2] \leq \mathbb{E}[(F - A(X))^2], \quad (3.20)$$

stating that the orthogonal projection of F onto the subspace of functions that depend only on $\boldsymbol{X}$ in 2-norm is $\mathbb{E}[F|X]$.

Now let $F(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ be k-variate. Assume $\mathbb{E}[F] = 0$ and $\mathbb{E}[F_i(X_i)] = 0$ for all i. Then $\mathbb{E}[F_i(X_i)|X_j] = 0$ for all $i \neq j$. Applying inequality (3.20) for k times in succession together with this fact, we obtain

$$\begin{aligned}
&\mathbb{E}[(F - F_1(X_1) - \cdots - F_{k-1}(X_{k-1}) - F_k(X_k))^2] \\
\geq &\mathbb{E}[(F - F_1(X_1) - \cdots - F_{k-1}(X_{k-1}) \\
&\quad - \mathbb{E}[F - F_1(X_1) - \cdots - F_{k-1}(X_{k-1})|X_k])^2] \\
= &\mathbb{E}[(F - F_1(X_1) - \cdots - F_{k-1}(X_{k-1}) - \mathbb{E}[F|X_k])^2] \\
&\vdots
\end{aligned}$$

$$\geq \mathbb{E}[(F - \mathbb{E}[F|X_1] - \cdots - \mathbb{E}[F|X_k])^2]$$

as desired. Finally, by orthogonality the optimal decomposition must satisfy $\sum \mathbb{E}[F_i(X_i)] = 0$ so the assumption $\mathbb{E}[F_i(X_i)] = 0$ can be made without loss of generality. □

By orthogonality, equation (3.19) can also be written in the following forms:

$$\begin{aligned}\delta_{\mathbb{R},2}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) =& \mathbb{E}\left[\overline{F}^2\right] - \sum\nolimits_{i=1}^{k} \mathbb{E}\left[\mathbb{E}[\overline{F}|X_i]^2\right] \\ =& \mathbb{E}_X\left[\overline{F}(X)^2\right] \\ & - \sum_{i=1}^{k} \mathbb{E}_{X,X'}\left[\overline{F}(X_{-i}, X_i)\overline{F}(X'_{-i}, X_i)\right],\end{aligned} \tag{3.21}$$

where (X_{-i}, X_i) is the input whose i-th variable takes value X_i and j-th variable takes value X'_j for $j \neq i$. As all these terms can be efficiently estimated, we obtain the following algorithm for estimating the quality of a given k-partition:

Proposition 16. *There is an algorithm that given a k-partition $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ of the variables and parameters $\varepsilon, \gamma > 0$, outputs a value $\hat{\delta}$ such that*

$$|\hat{\delta}^2 - \delta_{\mathbb{R},2}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)^2| \leq \varepsilon,$$

with probability at least $1 - \gamma$ from $O(k \log(k/\gamma)/\epsilon^4)$ queries to F in time linear in the number of queries.

3.4.5 Partitioning real-valued functions under the 2-norm and policy gradient algorithms

The problem of partitioning real-valued functions under the 2-norm is closely related to the well-studied problem of hypergraph partitioning. To explain this connection we recall the Efron-Stein decomposition of real-valued functions over product sets. The Efron-Stein decomposition of a function $F\colon \Sigma^n \to \mathbb{R}$ (under some product measure) is the unique decomposition of the form

$$F(x) = \sum\nolimits_{S\subseteq[n]} \hat{F}_S \cdot F_S(x),$$

where $\hat{F}_S$ are real coefficients and F_S are functions satisfying the following properties:

1. F_S depends on the variables in S only;
2. $\mathbb{E}[F_S|x_{S'}] = 0$ for any $S \not\subseteq S'$, where $x_{S'}$ is a fixing of all variables in S';
3. $\mathbb{E}[F_S^2] = 1$.

In particular, properties 1 and 2 imply that $\mathbb{E}[F_S F_T] = 0$ when $S \neq T$, and so $\mathbb{E}[F^2] = \sum_S \hat{F}_S^2$ by property 3.

Proposition 17. *Given $F\colon \Sigma^n \to \mathbb{R}$, let H be the hypergraph whose vertices are the variables of F and whose hyperedges S have weight $\hat{F}_S^2$ for every subset S. The cost of the k-cut $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ in H equals $\delta_{\mathbb{R},2}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)^2$.*

Proof. We may assume $\mathbb{E}[F] = 0$ and use expression (3.21) to evaluate $\delta_{\mathbb{R},2}$. The first term equals $\mathbb{E}[F^2] = \sum_S \hat{F}_S^2$. The rest of

the terms have the form $\mathbb{E}[\mathbb{E}[F|X_I]^2] = \mathbb{E}[F(X_{-I}, X_I)F(X'_{-I}, X_I)]$ for subsets I of variables. Plugging in the Efron-Stein decomposition of F we have

$$\mathbb{E}[\mathbb{E}[F|X_I]^2] = \sum_{S,T} \hat{F}_S \hat{F}_T \mathbb{E}[F_S(X_{-I}, X_I)F_T(X'_{-I}, X_I)].$$

By property 2, the terms in the summation in which $S \neq T$ evaluate to zero. Among the rest, if the set S contains any variable i outside I then

$$\mathbb{E}[F_S(X_{-I}, X_I)F_T(X'_{-I}, X_I)] = \mathbb{E}\big[F_S(X_{-I}, X_I) \\ \mathbb{E}[F_T(X'_{-I}, X_I)|X, X'_{-i}]\big]$$

and the inside expectation evaluates to zero by property 2. Therefore the only surviving terms are those where $S = T$ and $S \subseteq I$, by which

$$\mathbb{E}[\mathbb{E}[F|X_I]^2] = \sum\nolimits_{S\subseteq I} \hat{F}_S^2.$$

By (3.21),

$$\begin{aligned}\delta_{\mathbb{R},2}(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) &= \sum\nolimits_S \hat{F}_S^2 - \sum\nolimits_{i=1}^k \sum\nolimits_{S\subseteq \boldsymbol{X}_i} \hat{F}_S^2 \\ &= \sum\nolimits_{S \not\subseteq \boldsymbol{X}_i \text{ for any } i} \hat{F}_S^2.\end{aligned}$$

The last quantity is the desired value of the cost of k-cut in H. □

When $\Sigma = \{-1, 1\}$ under uniform measure, the functions F_S do not depend on F and equal the Fourier characters $\chi_S(x) = \prod_{i\in S} x_i$, allowing us to embed instances of hypergraph partitioning into variable partitioning.

Corollary 18. *Assume there is an algorithm A that given oracle access to $F\colon \{-1,1\}^n \to \mathbb{R}$ under uniform measure outputs a k-variable partition of cost at most $C \cdot \delta_2(F) + \varepsilon$ in time $t(n,k,\varepsilon)$. Then given a hypergraph with n vertices and m hyperedges with a k-cut of value opt, it is possible to output a k-cut of value $C \cdot opt$ in time $mn \cdot t(n,k,1/m)$.*

Chekuri and Li [155] give a reduction from hypergraph k-cut to densest-k-subgraph. Manurangsi [156] shows that the latter is hard to approximate to within $O(n^{1/(\log\log n)^c})$ assuming the exponential-time hypothesis, implying inapproximability of the same order for $\delta_{\mathbb{R},2}(F)$.

On the positive side, Proposition 17 can be used to obtain variable partitioning algorithms from hypergraph partitioning ones. The conversion is not direct as hypergraph partitioning assumes explicit access to the hypergraph. Klimmek and Wagner [171] observed that submodularity of the hypergraph cut function $\xi(\boldsymbol{X}) = \delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2$ allows for efficient minimization from *exact* oracle access. To derive Theorem 8 we extend the analysis to approximate oracle access.

The following proposition is an analysis of Queyranne's symmetric submodular minimization algorithm [172] for an approximate input oracle. We say g is ε-submodular if $g(\boldsymbol{XYZ}) - g(\boldsymbol{XZ}) - g(\boldsymbol{YZ}) + g(\boldsymbol{Z}) \leq \varepsilon$ for all disjoint subsets $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$.

Proposition 19 (Queyranne's algorithm with an approximate oracle)**.** *There is an algorithm that given oracle access to a sym-*

metric ε-submodular g, makes $O(n^3)$ oracle queries and outputs a nontrivial subset $\boldsymbol{X}$ such that $g(\boldsymbol{X})$ is within $n\varepsilon/2$ of the minimum of g.

Algorithm 3 Variable bipartitioning for real functions

1: **Output:** partition $\mathcal{P}$
2: Define random function $\xi(\boldsymbol{X})$ to be an estimation of $\mathbb{E}[D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$;
3: Run symmetric submodular minimization over $\boldsymbol{X}$ on $\xi(\boldsymbol{X})$;[2]
4: Output partition $\mathcal{P} = (\boldsymbol{X}, \overline{\boldsymbol{X}})$;

Theorem 8. *Let $F\colon \Sigma^n \to \mathbb{R}$ be a function with $\|F\|_{\mathbb{R},4} \leq 1$. There is an algorithm that given inputs n, ε, γ, and oracle access to F, runs in time $O(n^5 \log(n/\gamma)/\varepsilon^2)$ and outputs a bipartition $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ such that $\delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2 \leq \delta_{\mathbb{R},2}(F)^2 + \varepsilon$ with probability at least $1 - \gamma$.*

Proof of Theorem 8. By Proposition 16, $\delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2$ can be estimated to within error ε/Kn with $O(\log(n/\gamma)n^2/\varepsilon^2)$ queries to F with probability $1 - K\gamma/n^3$ for any constant K. This estimator implements an $\varepsilon/2n$-approximate oracle to $\delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2$ with probability $1 - \gamma$ with respect to an algorithm that makes at most Kn^3 queries. In particular, with probability $1 - \gamma$, the output of the oracle is $\varepsilon/4n$-close to the value of the submodular function $\delta^2_{\mathbb{R},2}$ at all points queried by Queyranne's algorithm and also at the minimum of $\delta^2_{\mathbb{R},2}$. Since from the algorithm's perspective it is interacting with a symmetric ε/n-submodular function g, it outputs a partition such that $g(\boldsymbol{X}, \overline{\boldsymbol{X}})$ is within $\varepsilon/2$ of the

[2]For example, Queyranne's algorithm [172] solves symmetric submodular minimization.

minimum of g. By the triangle inequality, $\delta_{\mathbb{R},2}(\boldsymbol{X}, \overline{\boldsymbol{X}})^2$ is within $\varepsilon/2 + 2\varepsilon/4n \leq \varepsilon$ close to the minimum of $\delta^2_{\mathbb{R},2}$. □

Saran and Vazirani's approximation algorithm [173] for multiway k-cut (with fixed terminals) can be viewed as a reduction from multiway k-cut to multiway 2-cut. The reduction works given access to approximate s-t-cut oracles, where s and t are designated terminals that must be split by the cut. The corresponding cut function $\delta_{\mathbb{R},2}(\boldsymbol{s}\boldsymbol{X}, \boldsymbol{t}\overline{\boldsymbol{X}})$, where $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ is now a partition of $\boldsymbol{V} - \{\boldsymbol{s}, \boldsymbol{t}\}$, is still submodular but no longer symmetric.

Therefore, we desire an analogue of Proposition 19 for general (not necessarily symmetric) submodular minimization [142, 174]. Blais et al. [175] (Algorithm 2 and Corollary 5.4 in their paper) have proposed such an algorithm under the context of tolerant junta testing, by investigating the Lovász extension [142] and a separation oracle [176] for the optimization. Given an inexact oracle to a submodular function with up to $\mathrm{poly}(\varepsilon/n)$ estimation error, they provide an algorithm to minimize the function with up to an ε optimization error in time $\mathrm{poly}(n/\varepsilon)$, leading to our algorithm for k-partitioning.

Corollary 20. *Let $F\colon \Sigma^n \to \mathbb{R}$ be a function with $\|F\|_{\mathbb{R},4} \leq 1$ and $\|F\|_\infty \leq 1$. There is an algorithm that given inputs n, k, ε, γ, and oracle access to F, runs in time $O(k^2 n^k \,\mathrm{poly}(n/\varepsilon) \log(1/\gamma))$ and outputs a k-partition $\mathcal{P}$ such that $\delta_{\mathbb{R},2}(\mathcal{P})^2 \leq (2-2/k)\delta_{\mathbb{R},2}(F)^2 + \varepsilon$ with probability $1 - \gamma$.*

Algorithm 4 Variable k-partitioning for real functions

1: **Input:** number of sets k in the partition, $k \geq 3$;
2: **Output:** partition $\mathcal{P}$;
3: Define random function $\xi'(\boldsymbol{X}, \boldsymbol{s}, \boldsymbol{t})$ to be an estimation of $\mathbb{E}[D_F(\boldsymbol{sX}, \boldsymbol{V} \setminus \boldsymbol{sX})^2]$, where $\boldsymbol{s}, \boldsymbol{t} \notin \boldsymbol{X}$;
4: **for** Set $\boldsymbol{W}$ of k vertices out of $\binom{n}{k}$ combinations **do**
5: Run multiway-k-cut given k terminals $\boldsymbol{W}$ and obtain $\mathcal{P}_{\boldsymbol{W}}$, where the cost of $\boldsymbol{s}$-$\boldsymbol{t}$-cut is treated as $\min_{\boldsymbol{X}} \xi'(\boldsymbol{X}, \boldsymbol{s}, \boldsymbol{t})$ for any $s, t \in \boldsymbol{W}$;
6: **end for**
7: Output the partition $\mathcal{P}_{\boldsymbol{W}}$ with the minimum cost over all $\boldsymbol{W}$;

It remains to prove Proposition 19.

Claim 21. *Let g be ε-submodular. Assume there exists $\boldsymbol{x} \in \boldsymbol{W}$ such that for all $\boldsymbol{Y} \subseteq \boldsymbol{W} \setminus \boldsymbol{x}$ and $\boldsymbol{u} \notin \boldsymbol{W}$,*

$$g(\boldsymbol{W}) + g(\boldsymbol{u}) \leq g(\boldsymbol{W} \setminus \boldsymbol{Y}) + g(\boldsymbol{Yu}) + \delta.$$

If $\boldsymbol{x}'$ maximizes $g(\boldsymbol{Wu}) - g(\boldsymbol{u})$ among all $\boldsymbol{u} \notin \boldsymbol{W}$ then

$$g(\boldsymbol{Wx}') + g(\boldsymbol{u}) \leq g(\boldsymbol{Wx}' \setminus \boldsymbol{Y}) + g(\boldsymbol{Yu}) + (\delta + \varepsilon).$$

Proof. If $\boldsymbol{x} \notin \boldsymbol{Y}$ then

$$\begin{aligned} g(\boldsymbol{Wx}') + g(\boldsymbol{u}) &\leq \big(g(\boldsymbol{W}) - g(\boldsymbol{W} \setminus \boldsymbol{Y}) + g(\boldsymbol{Wx}' \setminus \boldsymbol{Y})\big) + g(\boldsymbol{u}) + \varepsilon \\ &= g(\boldsymbol{Wx}' \setminus \boldsymbol{Y}) + \big(g(\boldsymbol{W}) - g(\boldsymbol{W} \setminus \boldsymbol{Y}) + f(\boldsymbol{u})\big) + \varepsilon \\ &\leq g(\boldsymbol{Wx}' \setminus \boldsymbol{Y}) + g(\boldsymbol{Yu}) + (\delta + \varepsilon) \end{aligned}$$

Otherwise, $\boldsymbol{x} \notin \boldsymbol{W} \setminus \boldsymbol{Y}$ and

$$\begin{aligned} g(\boldsymbol{Wx}') + g(\boldsymbol{u}) &\leq g(\boldsymbol{Wu}) + g(\boldsymbol{x}') \\ &\leq \big(g(\boldsymbol{W}) - g(\boldsymbol{Y}) + g(\boldsymbol{Yu})\big) + g(\boldsymbol{x}') + \varepsilon \end{aligned}$$

$$= \big(g(\boldsymbol{W}) + g(\boldsymbol{x}') - g(\boldsymbol{Y})\big) + g(\boldsymbol{Y}\boldsymbol{u}) + \varepsilon$$
$$\leq g(\boldsymbol{W}\boldsymbol{x}' \setminus \boldsymbol{Y}) + g(\boldsymbol{Y}\boldsymbol{u}) + (\delta + \varepsilon) \qquad \square$$

Proof of Proposition 19. Queyranne's algorithm Q^g is recursive. If $n = 2$ the unique partition is output. Otherwise, starting from an arbitrary singleton set $\boldsymbol{W}_1$, the algorithm sets $\boldsymbol{W}_{i+1} = \boldsymbol{W}_i\boldsymbol{x}_i$, where $\boldsymbol{x}_i$ maximizes $g(\boldsymbol{W}_i\boldsymbol{u}) - g(\boldsymbol{u})$ among all $\boldsymbol{u} \notin \boldsymbol{W}_i$. The algorithm then contracts the elements $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$ into $\boldsymbol{x}_{n-1}\boldsymbol{x}_n$ and outputs the smaller value of $Q^g(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-2}, \boldsymbol{x}_{n-1}\boldsymbol{x}_n)$ and $g(\boldsymbol{x}_n)$.

We prove by induction on n that the output of Q^g is $(n-1)\varepsilon/2$-close to the minimum of g. The base case $n = 2$ is clear. Now assume this is true for inputs of size $n-1$. If the minimum partition of g doesn't split $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$ then the claim follows by inductive assumption.

Otherwise, we show that $g(\boldsymbol{x}_n)$ is within $(n-1)\varepsilon/2$-close to the minimum of g. Applying Claim 21 iteratively to the sets $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_{n-1}$, we conclude that

$$g(\boldsymbol{W}_{n-1}) + g(\boldsymbol{x}_n) \leq g(\boldsymbol{W}_{n-1} \setminus \boldsymbol{Y}) + g(\boldsymbol{Y}\boldsymbol{x}_n) + (n-1)\varepsilon$$

for all $\boldsymbol{Y}$ that do not contain $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n-1}$. Applying symmetry this inequality can be rewritten as $g(\boldsymbol{x}_n) \leq g(\boldsymbol{x}_n Y) + (n-1)\varepsilon/2$. As $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$ are split in the optimal solution it must be of

[2]Multiway-k-cut can be solved by, for example, Saran and Vazirani's algorithm [173]. The function $\xi'(\boldsymbol{X}, \boldsymbol{s}, \boldsymbol{t})$ is still submodular over $\boldsymbol{X}$ but not necessarily symmetric. The value $\min_{\boldsymbol{X}} \xi'(\boldsymbol{X}, \boldsymbol{s}, \boldsymbol{t})$ can be computed by general submodular minimization algorithms like [142] and [174].

type $\boldsymbol{x}_n Y$ for some Y excluding $\boldsymbol{x}_{n-1}$, so $g(\boldsymbol{x}_n)$ is $(n-1)/2\varepsilon$ close to the minimum as desired. □

3.5 Generalized variable partitioning

In this section we present our alternative partitioning algorithm, which is general enough to work on any normed group G assuming it is possible to efficiently estimate the quantity $\|D_F(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})\|$. The algorithm outputs an $O(kn^2)$ approximation to the optimal partition in time polynomial in n, k, and $1/\varepsilon$.

The algorithm is based on the pairwise estimates of dependency over sets of single variables. The intuition behind the algorithm is that if the dependency between $\boldsymbol{x}$ and $\boldsymbol{y}$ is low, then these two variables should be assigned to different partitions. Therefore the algorithm keeps asserting such "in different partitions" for the pairs with the lowest dependency estimates, until the k-partitioning can be clearly observed from the assertions. It is worth noting that this idea of the algorithm has been used in previous works in reinforcement learning control [70] in a heuristic way.

Proposition 22. *Assuming* $e(\boldsymbol{x}, \boldsymbol{y}) \leq \hat{e}(\boldsymbol{x}, \boldsymbol{y}) \leq e(\boldsymbol{x}, \boldsymbol{y}) + \varepsilon$ *for all* $\boldsymbol{x}$ *and* $\boldsymbol{y}$*,*

$$\delta(\mathcal{P}) \leq (8k - 10)n^2(4\delta_k(F) + \varepsilon). \tag{3.22}$$

When $k = 2$, the leading constant $8k - 10 = 6$ can be improved to 1 by using Claim 26 instead of Claim 23 and the fact

Algorithm 5 Approximate partitioning via pairwise estimates

Input: Number of sets k in partition;
Output: Partition $\mathcal{P}$;
For every pair of variables $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{V}$, find estimate $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$ for $e(\boldsymbol{x}, \boldsymbol{y}) = \|D_F(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})\|$;
Create a weighted graph with vertices $\boldsymbol{V}$ and weights $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$;
Order the edges in increasing weight;
repeat
    Remove the edge with the smallest weight;
until The graph has exactly k connected components

that at most $n^2/4$ pairs cross the partition.

If the estimates $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$ are obtained by empirical averaging, we obtain Theorem 7.

3.5.1 Proof of Proposition 22 and Theorem 7

For a partition $\mathcal{P}$ of the variables, let $\Delta(\mathcal{P}) = \sum \delta(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})$, where the sum is taken over all pairs that cross the partition. We will deduce Theorem 7 from the following bound on $\delta(\mathcal{P})$.

Claim 23. *For every k-partition $\mathcal{P}$, $\delta(\mathcal{P}) \leq (16k - 20)\Delta(\mathcal{P})$.*

The following fact is immediate from the definitions of δ. The proof of this claim is delayed to the end of this section.

Fact 24. *For any partition $(\boldsymbol{U}, \overline{\boldsymbol{U}})$ such that $\boldsymbol{X} \subseteq \boldsymbol{U}$ and $\boldsymbol{Y} \subseteq \overline{\boldsymbol{U}}$, $\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq \delta(\boldsymbol{U}, \overline{\boldsymbol{U}})$.*

Now we prove Proposition 22 and Theorem 7, assuming the correctness of Claim 23.

Proof of Proposition 22. By Claim 11 and Fact 24, all edges $(\boldsymbol{x}, \boldsymbol{y})$ in the optimal partition must satisfy $e(\boldsymbol{x}, \boldsymbol{y}) \leq 4\delta_2(F)$. By our assumption on the quality of the approximations,

$$\hat{e}(\boldsymbol{x}, \boldsymbol{y}) \leq 4\delta_2(F) + \varepsilon. \tag{3.23}$$

Since the algorithm removes edges in increasing order of weight, all the edges that cross the output partition $\mathcal{P}$ must also satisfy this inequality. Then

$$\begin{aligned}
\delta(\mathcal{P}) &\leq (16k-20)\Delta(\mathcal{P}) && \text{by Claim 23,} \\
&\leq (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y} \text{ cross } \mathcal{P}} e(\boldsymbol{x}, \boldsymbol{y}) && \text{by Claim 12,} \\
&\leq (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y} \text{ cross } \mathcal{P}} \hat{e}(\boldsymbol{x}, \boldsymbol{y}) \\
&\leq (16k-20)\sum\nolimits_{\boldsymbol{x},\,\boldsymbol{y} \text{ cross } \mathcal{P}} 4\delta_2(F) + \varepsilon && \text{by (3.23),} \\
&\leq (8k-10)n^2 \cdot (4\delta_2(F) + \varepsilon).
\end{aligned}$$

The last inequality holds because there are at most $\binom{n}{2} \leq n^2/2$ pairs of variables crossing the partition. □

Theorem 7. *Let $\|\cdot\|$ be either 1) $\|\cdot\|_{\mathbb{R},p}$ assuming $\|F\|_{\mathbb{R},2p} = O(1)$, or 2) the Hamming metric over $\mathbb{Z}_q$. There is an algorithm that given parameters n, k, ε, γ, and oracle access to $F\colon \Sigma^n \to G$ outputs a k-partition $\mathcal{P}$ such that $\delta(\mathcal{P}) \leq O(kn^2)(\delta_k(F) + \varepsilon)$ with probability at least $1 - \gamma$. The algorithm makes $O(K^p n^2 \log(n/\gamma)/\varepsilon^{2p})$ queries to F and runs in time linear in the number of queries, for an absolute constant K.*

Proof. By Proposition 10 we can estimate an edge $e(\boldsymbol{x}, \boldsymbol{y})$ up to ε-error using $O(K^p \log(n/\gamma)/\epsilon^{2p})$ samples with probability at

least $1-\gamma/n$, for some absolute constant K. Therefore with n^2 times the amount of samples, which is $O(K^p n^2 \log(n/\gamma)/\varepsilon^{2p})$, we can estimate all edges up to ε-error with probability at least $1-\gamma$. Then by Proposition 22 we have $\delta(\mathcal{P}) \leq (8k-10)n^2(4\delta_k(F)+\varepsilon)$ with probability at least $1-\gamma$. □

We first prove Claim 23 in the case of bipartitions ($k = 2$). This is Claim 26 below. We use $\boldsymbol{X}\boldsymbol{X}'$ to denote the union of the variable sets $\boldsymbol{X}$ and $\boldsymbol{X}'$.

Claim 25. *For disjoint sets of variables $\boldsymbol{X}, \boldsymbol{X'}, \boldsymbol{Y}$, $\delta(\boldsymbol{X}\boldsymbol{X'}, \boldsymbol{Y}) \leq \delta(\boldsymbol{X}, \boldsymbol{Y}) + 2\delta(\boldsymbol{X}', \boldsymbol{Y})$.*

Proof. Assume that

$$F(X, X', Y) \approx_\delta A(X, X') + B(X', Y) \qquad \text{and}$$
$$F(X, X', Y) \approx_{\delta'} A'(X, X') + B'(X, Y).$$

By the triangle inequality,

$$A(X, X') + B(X', Y) \approx_{\delta+\delta'} A'(X, X') + B'(X, Y).$$

Fix $X'(Z) = \underline{X}'(Z)$. Writing $C(X) = A(X, \underline{X}') - A'(X, \underline{X}')$ and $D(Y') = B(\underline{X}', Y')$, we get that

$$B'(X, Y) \approx_{\delta+\delta'} C(X) + D(Y).$$

By the triangle inequality (with the second equation), we get that

$$F(X, X', Y) \approx_{\delta+2\delta'} A'(X, X') + C(X) + D(Y). \qquad \square$$

Claim 26. *For every bipartition $\boldsymbol{X}, \overline{\boldsymbol{X}}$ of the variables, $\delta(\boldsymbol{X}, \overline{\boldsymbol{X}}) \leq 4 \cdot \Delta(\boldsymbol{X}, \overline{\boldsymbol{X}})$.*

Proof. By Claim 25,

$$\delta(\boldsymbol{X}'\{\boldsymbol{x}\}, \{\boldsymbol{y}\}) \leq \delta(\boldsymbol{X}', \{\boldsymbol{y}\}) + 2\delta(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})$$

for all $\boldsymbol{X}' \subseteq \boldsymbol{X} \setminus \{x\}$ and $\boldsymbol{y}$. Applying this inequality iteratively we conclude that $\delta(\boldsymbol{X}, \{\boldsymbol{y}\}) \leq 2\sum_{\boldsymbol{x} \in \boldsymbol{X}} \delta(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})$. Also by Claim 25

$$\delta(\boldsymbol{X}, \boldsymbol{Y}'\{\boldsymbol{y}\}) \leq \delta(\boldsymbol{X}, \boldsymbol{Y}') + 2\delta(\boldsymbol{X}, \boldsymbol{Y}'\{\boldsymbol{y}\}),$$

so $\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq 2\sum_{\boldsymbol{y} \in \boldsymbol{Y}} \delta(\boldsymbol{X}, \{\boldsymbol{y}\})$. Combining the two inequalities we obtain the desired conclusion. □

To extend the proof to larger k and obtain Claim 23, we generalize the first inequality in this sequence to k-partitions.

Claim 27. *For every $2k$-partition $(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k, \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k)$,*

$$\delta(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_k, \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_k) \leq 2\delta(\boldsymbol{Y}_1\boldsymbol{Z}_1, \ldots, \boldsymbol{Y}_k\boldsymbol{Z}_k) + 3\delta(\boldsymbol{Y}_1 \ldots \boldsymbol{Y}_k, \boldsymbol{Z}_1 \ldots \boldsymbol{Z}_k).$$

Proof. Assume that

$$F(V) \approx_\delta F_1(Y_1, Z_1) + \cdots + F_t(Y_t, Z_t)$$
$$F(V) \approx_{\delta'} A(Y_1, \ldots, Y_t) + B(Z_1, \ldots, Z_t).$$

By the triangle inequality

$$A(Y_1, \ldots, Y_t) + B(Z_1, \ldots, Z_t) \approx_{\delta+\delta'} F_1(Y_1, Z_1) + \cdots + F_t(Y_t, Z_t).$$

Fixing $Z_1, \ldots, Z_t$ to values $\underline{Z}_1, \ldots, \underline{Z}_t$ we get the decomposition

$$A(Y_1, \ldots, Y_t) \approx_{\delta+\delta'} F_1(Y_1, \underline{Z}_1) + \cdots + F_t(Y_t, \underline{Z}_t) - B(\underline{Z}_1, \ldots, \underline{Z}_t).$$

and similarly

$$B(Z_1, \ldots, Z_t) \approx_{\delta+\delta'} F_1(\underline{Y}_1, Z_1) + \cdots + F_t(\underline{Y}_t, Z_t) - A(\underline{Y}_1, \ldots, \underline{Y}_t).$$

Plugging these into the second equation gives the desired decomposition. □

Proof of Claim 23. We assume that k is a power of two and prove by induction that $\delta(\mathcal{P}) \leq c_k \Delta(\mathcal{P})$, where c_k is the sequence $c_{2k} = 2c_k + 12$, $c_2 = 4$. The base case $k = 2$ follows from Claim 26. Assume the claim holds for k and apply Claim 27 to $\mathcal{P}$. By inductive assumption and Claim 26,

$$\delta(\mathcal{P}) \leq 2 \cdot c_k \Delta(\boldsymbol{Y}_1 \boldsymbol{Z}_1, \ldots, \boldsymbol{Y}_k \boldsymbol{Z}_k) + 3 \cdot 4 \Delta(\boldsymbol{Y}_1 \ldots \boldsymbol{Y}_k, \boldsymbol{Z}_1 \ldots \boldsymbol{Z}_k).$$

Since $\mathcal{P}$ is a refinement of both these partitions, it follows that $\delta(\mathcal{P}) \leq (2c_k + 12)\Delta(\mathcal{P}) = c_{2k}\Delta(\mathcal{P})$, concluding the induction.

The recurrence finds $c_k = 8k - 12$, proving the claim when k is a power of two. When it is not, the same reasoning applies to the closest power of two exceeding k (by taking some of the sets in the partition to be empty), which is at most $2k - 1$, proving the desired bound. □

3.6 Experiments

3.6.1 Implementation of heuristic methods

Our policy optimization algorithm is built on top of proximal policy optimization (PPO) [67, 177] where the advantage realization value is estimated by GAE [149]. Our code is available at https://github.com/wangbx66/Action-Subspace-Dependent. We use a policy network for PPO and a value network for GAE that have the same architecture as is in [57, 67]. We utilize a third network which estimates the advantage function $A^{\mu}(s, a)$ smoothly by solving Equation (3.12) to be our baseline function $c(s, a)$. The network computes the advantage and the Hessian matrix approximator $w_2(s)w_2(s)^T$ by a forward propagation. It uses the wide & deep architecture borrowed from the field of recommendation system [151, 178, 179, 180, 181, 182, 183, 184, 185]. For the wide component, the state is mapped to $w_1(s)$ and $w_2(s)$ through two-layer MLPs, both with size 128 and $\tanh(\cdot)$ activation. The deep component A_{deep} is a three-layer MLPs with size 128 and $\tanh(\cdot)$ activation. Our other parameters are consistent with those in [67] except that we reduce the learning rate by ten times (i.e., $3 \cdot 10^{-4}$) for more stable comparisons.

3.6.2 Experiments of heuristic methods

We demonstrate the sample efficiency and the accuracy of ASDG estimator and POSA algorithm (Algorithm 1) in terms of both

performance and variance. ASDG is compared with several of the state-of-the-art gradient estimators.

Baselines

Action dependent factorized baselines (ADFB) [70] assumes fully factorized policy distributions, and uses $A(s, (\bar{a}_{(k)}, a_{(-k)}))$ as the k-th dimensional baseline. The subspace $a_{(k)}$ is restricted to contain only one dimension, which is the special case of ASDG with $K = m$.

Generalized advantage dependent baselines (GADB) [64, 65] uses a general baseline function $c(s, a)$ which depends on the action. It does not utilize Rao-Blackwellization and is our special case when $K = 1$.

Synthetic high-dimensional action spaces

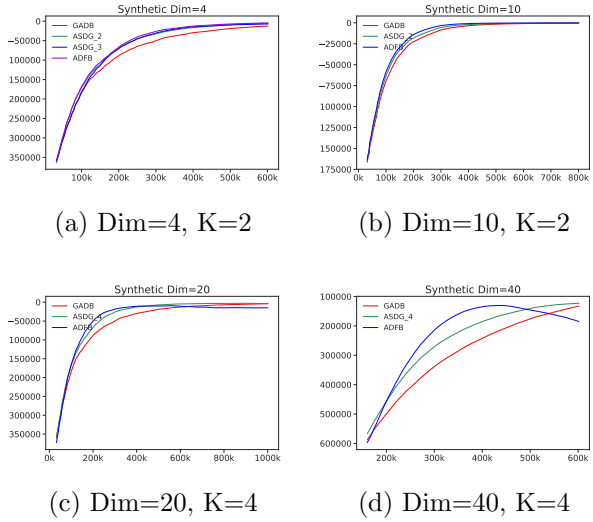
We design a synthetic environment with a wide range of action space dimensions and explicit action subspace structures to test the performance of Algorithm 1 and compare that with previous studies. The environment is a one-step MDP where the reward $r(s, a) = \sum_{k=1}^{K} a_{(k)}^T H_k a_{(k)} + \epsilon$ does not depend on the state s (e.g., ϵ is a random noise). In the environment, the action is partitioned into K independent subspaces with a stationary Hessian of the advantage function. Each of the subspace can be regarded as an individual agent. The environment setting satisfies both Assumption 1 and 2.

Figure 3.2 shows the results on the synthetic environment for ASDG with different dimensions m and number of subspaces K. The legend *ASDG_K* stands for our ASDG estimator with K blocks partitions. For environments with relatively low dimensions such as (a) and (b), all of the algorithms converge to the same point because of the simplicity of the settings. Both ASDG and ADFB (that incorporates RB) outperform GADB significantly in terms of sample efficiency while ADFB is marginally better than ASDG. For high dimensional settings such as (c) and (d), both ASDG and GADB converge to the same point with high accuracy. Meanwhile, ASDG achieves the convergence significantly faster because of its efficiency. ADFB, though having better efficiency, fails to achieve the competitive accuracy.

We observe an ideal balance between accuracy and efficiency. On the one hand, ASDG trades marginal accuracy for efficiency when efficiency is the bottleneck of the training, as is in (a) and (b). On the other hand, ASDG trades marginal efficiency for accuracy when accuracy is relatively hard to achieve, as is in (c) and (d). ASDG's tradeoff results in the combination of both the merits of its extreme cases.

We also demonstrate that the performance is robust to the assumed K value in (a) when accuracy is not the major difficulty. As is shown in (a), the performance of ASDG is only decided by its sample efficiency, which is monotonically increasing with K. However in complicated environments, an improper selection of K may result in the loss of accuracy. Hence, in general, ASDG

performs best overall when the K value is set to the right value instead of the maximum.



(a) Dim=4, K=2 (b) Dim=10, K=2

(c) Dim=20, K=4 (d) Dim=40, K=4

Figure 3.2: Learning curve for synthetic high-dimensional continuous control tasks, varying from 4 to 40 dimensions. At high dimensions, our ASDG estimator provides an ideal balance between the accuracy (i.e., GADB) and efficiency (i.e., ADFB).

OpenAI Gym's MuJoCo Environments

We present the results of the proposed POSA algorithm with ASDG estimator on common benchmark tasks. These tasks and experiment settings have been widely studied in the deep reinforcement learning community [186, 63, 70, 64]. We test POSA on several environments with high action dimensions, namely Walker2d-V1, Hopper-V1, HalfCheetah-V1, and Ant-V1, shown in Figure 3.3 and Figure 3.4. In general, ASDG outperforms ADFB and GADB consistently but performs ex-

traordinarily well for HalfCheetah-V1. Empirically, we find the block diagonal assumption 2 for the advantage function is minimally violated, and that may be one of the reasons for its good performance.

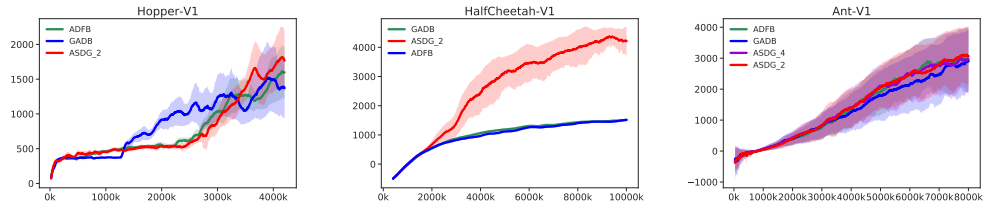


Figure 3.3: Comparison between two baselines (ADFB, GADB) and our ASDG estimator on various OpenAI Gym's Mujoco continuous control tasks, including Hopper-V1 (dim=3), HalfCheetah-V1 (dim=6) and Ant-V1 (dim=8). Our ASDG estimator performs consistently the best across all these tasks.

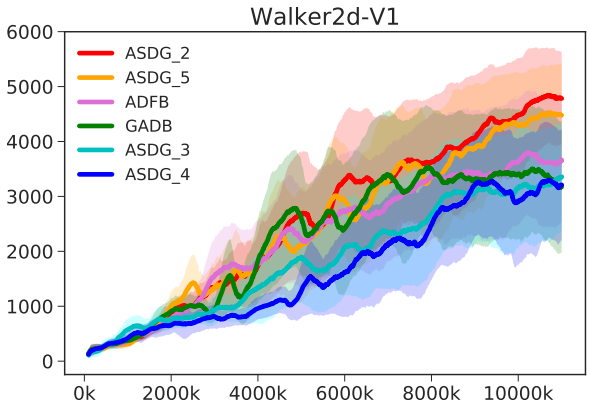


Figure 3.4: The choices of action subspace number K in the Walker2d-V1 environment.

To investigate the choice of K, we test all the possible K values in Walker2d-V1. The optimal K value is supposed to be

between its extreme $K = 1$ and $K = m$ cases. Empirically, we find it effective to conduct a grid search. We consider an automatic approach to finding the optimal K value an interesting future work.

3.6.3 Experiments of rigorous methods

We compare our first algorithm by pairwise estimates (PE) and our second algorithm by submodular minimization (SM) with the aforementioned existing approaches. A2C [57] is the baseline approach in reinforcement learning who does not leverage variable partition. It uses control variates as the primary variance reduction technique. Other methods [70, 71] partition the control variable so as to reduce the variance in the Monte-Carlo estimation by Rao-Blackwellization [139]. For the discussion on variance reduction we refer the readers to the paper cited above.

Now we study the performance in terms of both the correctness and the optimality on graph cuts on weighted graphs. Correctness notes the number of times the algorithm outputs exactly the optimal partition, while optimality describes the average of the ratio of the partition error and the optimal partition error, over all the independent runs. This will illustrate the difference between greedy-based algorithms like [71] and our first algorithm, and submodular minimization based algorithms like our second algorithm. Note that submodular minimization always finds the optimal partition.

#Nodes n	$n = 5$	$n = 10$	$n = 20$	$n = 40$	$n = 100$
Submodular	-	-	-	-	-
Greedy (correctness)	7753	6271	4226	2380	1101
Greedy (optimality)	1.060	1.203	1.408	1.352	1.250

Table 3.2: Performance of the greedy algorithm on variable partition.

Then Table 3.3 compares the partitioning algorithms when the oracle is a quadratic function $a^T H_0 a$ for some random H_0. In this case our second algorithm SM also incurs an error per Theorem 8, but the error in practice is shown to be small enough. It has constantly the best empirical performance in both correctness and optimality.

Since we only replaced heuristic partitioning with our partitioning algorithm in reinforcement learning, it is reasonable that our more accurate partitions will improve reinforcement learning.

#Nodes n	$n = 5$	$n = 10$	$n = 20$	$n = 40$	$n = 100$
Li and Wang [71] (correctness)	7553	5651	2929	1251	400
PE (correctness)	7709	6108	4001	2020	918
SM (correctness)	**9896**	**9630**	**9243**	**8193**	**6802**
Li and Wang [71] (optimality)	1.150	1.281	1.508	1.501	1.290
Wu et al. [70] (optimality)	9.049	13.54	20.96	34.42	72.55
PE (optimality)	1.075	1.277	1.452	1.400	1.281
SM (optimality)	**1.020**	**1.028**	**1.101**	**1.110**	**1.025**

Table 3.3: Comparisons of the algorithms on variable partition.

Finally we plug our algorithms into reinforcement learning control, replacing the partitioning steps in [71]. The tasks we are testing on are standard tasks in reinforcement learning by the MuJoCo physics simulator. This includes training a simplified model of ant, cheetah, or human to run as fast as possible. The score is the cumulative reward over time, where the reward is the speed less the energy cost (which is $0.001\|a\|_2^2$). The control variables a are the forces applied to the joints. We refer to [187] for the exact simulator settings.

We have conducted experiments on all eight environments from MuJoCo that has the action dimension higher than one, shown in Figure 3.5 below. In the figure the x-axis is the number of Monte-Carlo sample updates, which can be regarded as the time elapsed on the training, while the y-axis is the score attained by the model. Our second algorithm (SM) has achieved the highest score among most of these tasks, which agrees with our theoretical finding.

□ **End of chapter.**

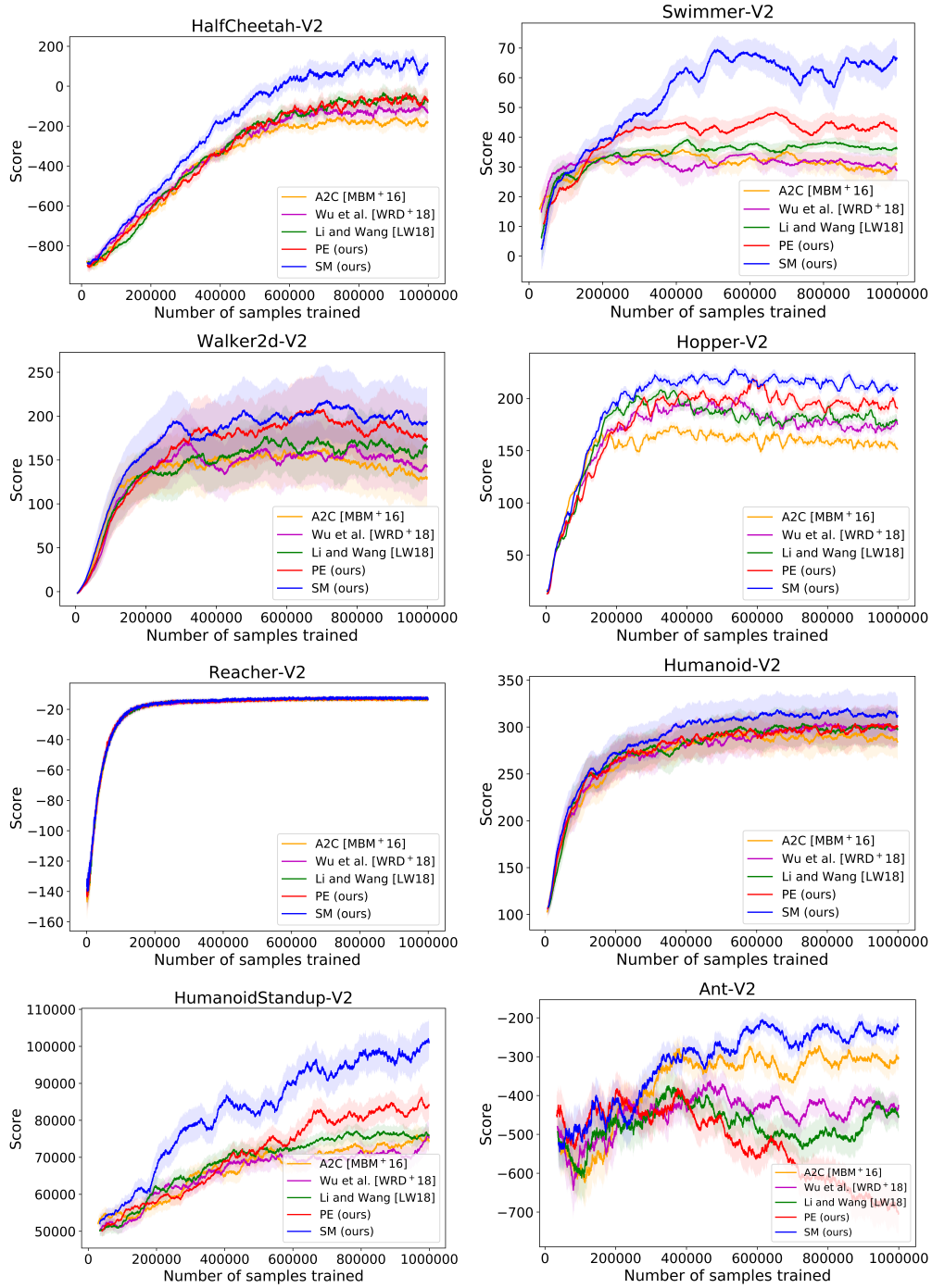


Figure 3.5: Empirical comparisons on MuJoCo high-dimensional control tasks. Each curve is averaged over 10 independent experiments.

Chapter 4

Improving Sample Quality

On the other side of improving sample efficiency lies the improvement of sample quality. Obtaining better samples in equivalence will result in more accurate estimates and desired performance in learning. In general, there is not a way to increase such a quality universally for all tasks in model-free policy optimization. Though, many methods share common underlying ideas so they are usually categorized into famous classes. These include reward shaping, hierarchical learning, learning from demonstration, curriculum learning, and meta-learning. Despite that little universal guarantees can be given, this topic can be useful under specific contexts of applications.

Primarily, our discussion will be anchored to applications, where we consider mostly a canonical application of reinforcement learning of mimicking the human vision by deploying successive glimpses without taking all pixels at once. In many tasks like object recognition, the attention-based methods achieve su-

perior speed and more robustness since during the process it managed to skip most of the clutters. However, it is very hard to train the model when the size of the image scales up. The reward signals are simply too sparse, in particular, one per episode without regard to the size of the image and the horizon of the glimpse sequence.

To better utilize the input information, we explore both a heuristic algorithm and a method based on asymmetric actor-critic for reward shaping. The new mechanisms can stabilize the learning process and make the algorithm converge on very large images. The heuristic method employs a k-maximum aggregation layer and a shaping of reward based on these k maximum activations. The latter borrows from partial observable MDP to handle the asymmetry between the agent state and the environment state. Both methods demonstrate significant efficiency and accuracy improvement over existing approaches, on two independent tasks of computing vision.

4.1 Computing vision by recurrent attention

Object existence determination (ED) focuses on deciding if certain visual patterns exist in an image.[1] As the basis of many computing vision tasks, ED's quality affects further processing such as locating certain patterns (apart from telling the exis-

[1]Certain literature, for example in [188], may refer the problem using the terminology *object detection*. More commonly, object detection implies both deciding the existence of the patterns and subsequently locating them if so.

tence), segmentation of certain patterns, object recognition, and object tracking in consecutive image frames. However, while ED is conducted by humans rapidly and effortlessly [99, 100], the performance of computer vision algorithms are surprisingly poor especially when the image is of large size and low quality. Hence, it is desirable to develop efficient and noise-proof systems to deal with object detection tasks with large and noisy images.

In fact, the way humans process images is not similar to recent prevailing approaches such as detecting objects via convolution networks (ConvNet) and residual networks [101]. Instead of taking all pixels from the image in parallel, humans perform sequential interactions with the image. Humans may recursively deploy visual attentions and perform glimpses on selective locations to acquire information. At the end of the processing, information from all past locations and glimpses is gathered together to make the final decision. Such behavior accomplishes ED tasks efficiently, especially for large images as it depends only on the number of saccades. Meanwhile, as the approach selectively learns to skip the clutter,[2] it tends to be less sensitive to noise compared with those that take all pixels into the computation.

The process can be naturally interpreted as a reinforcement learning (RL) task where each image represents an environment. At the beginning of the process, the agent conducts an action which is represented by a 2-dimension Cartesian coordi-

[2]Clutter are the irrelevant features of the visual environment, discussed in RAM [104].

nate. When the environment receives the action, it calculates the retina-like representation of the image at the corresponding location, and returns that representation to the agent as the agent's observation. Repeatedly until the last step, the agent predicts the detection result based on the trajectory and receives the evaluation of its prediction as the reward signal. It is important to note that the agent has never had access to the full image directly. Instead, it carefully chooses its actions in order to get the desired partial observations of the internal states of the environment.

Recurrent attention models (RAM) [104] are the first computational models to imitate the process with a reinforcement learning algorithm. The success of RAM leads to enormous studies on attention-based computer vision solutions [107, 108, 189]. However, RAM and their extensions [105, 106] are designed to solve object recognition tasks such as handwritten digit classification. Those models largely ignore the trajectory information which causes massively delayed rewards. Indeed in RAM, the reward function is associated with only the last step of the process and is otherwise zero. The actions before that, which deploy the attention for the model, do not receive direct feedback and are therefore not efficiently learned. Especially in ED (and in general, object detection) tasks, delayed rewards fail to provide reinforcement signals to the choice of locations when the glimpse at certain locations may provide explicit information for the existence of the object.

We present recurrent existence determination models (RED), which inherit the advantage of RAM that the attention is only deployed on locations that are deemed informative. Our approach involves a new observation setting which allows the agent to have access to explicit visual patches. Unlike previous trails which blur the pixels that are far from the saccade location, we acquire the exact patches which help to detect the existence of specific patterns. We employ gated recurrent units (GRU) [190] to encode the historical information acquired by the agent and generates temporary predictions at each time step. The temporary predictions over the time horizon are then aggregated via a novel k-maximum aggregation layer, which averages the k-greatest value to compute the final decision. It allows the rewards to be backpropagated to the early and middle stages of the processing directly apart from through the recurrent connections of the GRU. It provides immediate feedback which guides the agent to allocate its attention, and therefore addresses the issues caused by delayed rewards.

RED is evaluated empirically on both synthetic datasets, *Stained MNIST*, and real-world datasets. Stained MNIST is a set of handwritten digits from MNIST. Additionally, the resolution has been enlarged and each digit may be added dot stains around the writings. The dataset is designed to compare the performance of RED and existing algorithms on images with high-resolution settings. The results show that attention based models run extraordinarily faster than traditional, ConvNet-based

methods [102, 103], while having better accuracy as well. Experiments on real-world dataset show superior speed improvement and competitive accuracy on retinopathy screening, compared to existing approaches. This also demonstrates that our algorithm is practical enough to be applied to real-world systems.

4.2 Preliminaries

4.2.1 The REINFORCE method and recurrent attention models

In an episode of reinforcement learning, at each time step t, the agent takes an action a_t from the set $\mathcal{A}_t$ of feasible actions. Receiving the action from the agent, the environment updates its internal state, and returns an observation x_t and a scalar reward r_t to the agent accordingly. In most of the problems, the observation does not fully describe the internal state of the environment, and the agent has to develop its policy using only the partial observations of the state. This process continues until the time horizon T. Let $R_t = \sum_{t'=1}^{t'=t} r_{t'}$ denote the cumulative rewards up to time t, the policy is trained to maximize the expectation $\mathbb{E}[R_T]$ of the return. Let π_θ be the policy function, parameterized by θ, the REINFORCE algorithm [46, 57] estimates the policy gradient using

$$g = \mathbb{E}_\pi[\nabla_\theta \log \pi(a_t|s_t)(R_t - b_t)], \tag{4.1}$$

where b_t is an baseline function for variance reduction.

It is common in RL to use x_t as the state s_t. However, consider that x_t are small patches in our setting, the information in a single x_t is insufficient. Ideally, the decision of the action is based on the trajectory $\tau_t = (a_1, x_1, r_1, \ldots, a_{t-1}, x_{t-1}, r_{t-1})$ which includes past actions, observations, and rewards. To handle the growing dimensionality of τ_t, the agent maintains an internal state s_t which encodes the trajectory using a recurrent neural network (RNN), and updates it repeatedly until the end of the time horizon[3]. In this way, the action is decided by the policy function, based only on the internal state s_t of the agent. Note that the full state of the environment is τ_t and the image to be processed, and the agent observes τ_t only.

The training of RL repeats the above process from step 1 to step T for a certain number of episodes. At the beginning of each episode, the agent resets its internal state while the environment resets its internal state as well. The model parameters are maintained across multiple episodes and are updated gradually as the occurrence of the reward signals at the end of each episode. Note that in RAM and RED, different from general online learning framework, the agent does not receive the reward signal in the middle stages of an episode. Hence the reward signals are inevitably heavily delayed, and RED need to address temporal credit assignment problem [42], which evaluates individual action within a sequence of actions according to a single

[3]In our paper, s_t is defined to be the state of the agent instead of the state of the environment.

reinforcement signal.

4.2.2 Glimpse and retina-like representations

A retina-like representation is the visual signal humans receive when glimpsing at a point of an image. The visual effect is that regions close to the focused location tend to retain their original, high-resolution form, while regions far from the focused location are blurred and passed to the human brain in their low-resolution form. In RAM and RED, the environment calculates the retina-like representations and returns it as the observation. Existing approaches to mimic such visual effects have been used in RAM and RAM's variants. They can be categorized into two classes: soft attention [108, 191] and hard attention [104, 191, 192].

Soft attention [189] applies a filter centered at the focused location. It imitates human behaviors, which downsamples the image gradually as it approaches far away from the focused point, resulting in a smooth representation. The approach is fully differentiable and is hence amenable to be trained straightforwardly using neural networks together with gradient descent. Despite those merits, soft attention is in general computationally expensive as it involves the filtering operation over all pixels, which deviates from the idea of RED and RAM to only examine parts of the image and subsequently making the process relatively inefficient.

Hard attention, on the other hand, extracts pixels with predefined sample rates. Fewer pixels are extracted as the region approaches further away from the focused location, making the process cost only constant time. Hard attention fits the idea of RED well though it is non-differentiable as it indexes the image and extracts pixels. To address the non-differentiability, we develop our training algorithm via policy gradient and use the rollouts of the attention mechanism to estimate the gradient. Formally, let x_t be a list of c channels and the i-th channel extracts the squared region centered at a_t with size $n_i \times n_i$, and down sample the patch to $n_1 \times n_1$. The channels are incorporated with the location information (known as the what and where pathways) with the patch information by adding the linear transformation $\tanh(W_{xa}a_t)$ of a_t. Note that the value of each entry W_{xa} will be restricted to be relatively small compared to the pixel values to retain the original patch information.

4.2.3 Convolutional gated recurrent units

RAM and RED use an RNN to encode the trajectory and update the states of the agent. While both long short-term memory (LSTM) and GRU are prevailing RNN implementations in sequential data processing [190], GRU is preferred to LSTM in RED. The reason is that the input passes through the unit explicitly when a GRU merges the memory cell state and the output state in an LSTM unit. This explicit information helps

to make the temporary detection decisions and enables our design of the k-maximum aggregation layer. Meanwhile, with the merge, the agent updates its internal state more efficiently. The speed improvement is critical especially for real-time applications such as surveillance anomaly detection when an instant detection decision is required.

We use convolutional GRU, a variant of GRU where the matrix product operations between the output state s_t, the input x_t, and the model parameters are replaced with convolution operations, and s_t and x_t are kept in their 2-dimension matrix shapes [193, 194]. A graphical illustration of GRU is shown in Figure 4.1, where lines in orange represent convolution operation. The gate mechanism in a convolutional GRU is formulated as

$$\begin{aligned} z_t &= \sigma(W_{zh} * s_{t-1} + W_{zx} * x_t) \\ v_t &= \sigma(W_{rh} * s_{t-1} + W_{rx} * x_t) \\ \tilde{s}_t &= \tanh(W_{sh} * (v_t \circ s_{t-1}) + W_{sx} * x_t) \\ s_t &= (1 - z_t)s_{t-1} + z_t\tilde{s}_t, \end{aligned} \tag{4.2}$$

where $*$ denotes convolution, $\circ$ denotes Hadamard product, $\sigma(\cdot)$ denotes the sigmoid function, and z_t and v_t are the update gate and the reset gate, respectively. W_{zh}, W_{zx}, W_{rh}, W_{rx}, W_{sh}, and W_{sx} are trainable parameters. Convolutional GRU retains the spatial information in the output state so that temporary detection decisions can be made well before the end of an episode.

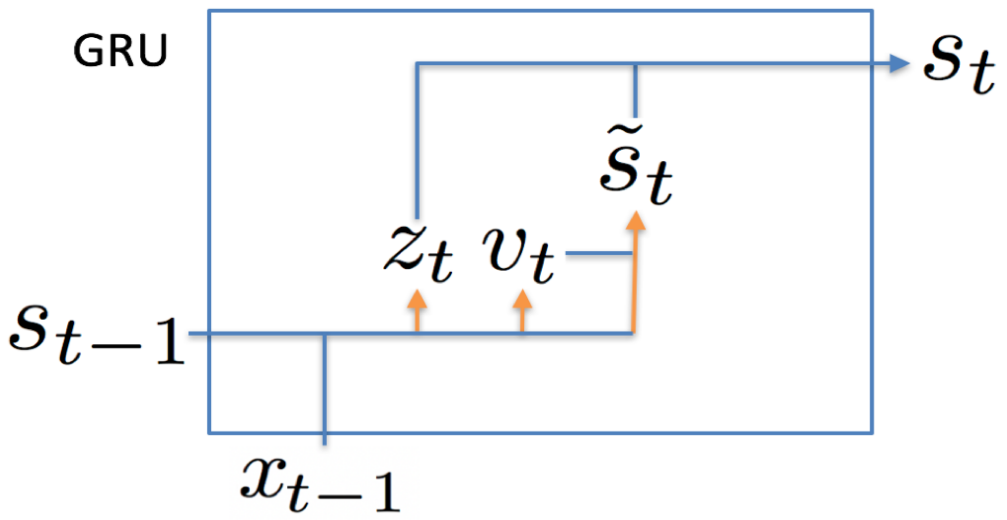


Figure 4.1: Illustration of convolutional gated recurrent units.

4.3 Recurrent existence determination

In this section we discuss the three main components of RED, namely, the attention mechanism, the k-maximum aggregation layer, and the policy gradient estimator. Taking together, an illustrative graph of our model is shown in Figure 4.2, where the arrows denote forward propagation.

4.3.1 Attention mechanism in RED

We formulate the attention mechanism within each of the episodes, that is, within the processing of one image. Let $\mathcal{I}$ denote the image, the agent has x_0, which is the low-resolution form of $\mathcal{I}$, as the initial observation. The state s_0 of the agent is initialized as the zero vector. Repeatedly, at each time step t, the agent

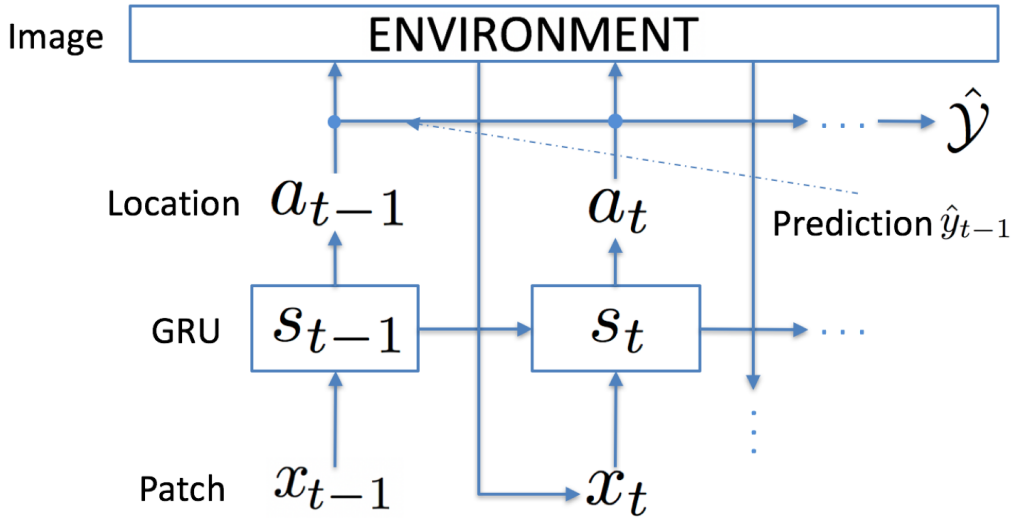


Figure 4.2: The attention mechanism and the reward mechanism in our proposed RED model.

calculates its action $a_t \in \mathbb{R}^2$, according to

$$a_t = \tanh(W_{as} s_t) + \epsilon_t, \tag{4.3}$$

where W_{as} is a trainable parameter of the model and ϵ_t is a random noise to improve exploration. The action a_t refers to a Cartesian Coordinate on the image, with $(-1, -1)$ corresponding to the bottom-left corner of $\mathcal{I}$ and $(1, 1)$ corresponding to the top-right corner of $\mathcal{I}$. Each entry of ϵ_t is sampled from a normal distribution with a fixed standard deviation of β, independently. The environment returns the retina-like representation x_t via the hard attention model described in Section 4.2.2.

The agent employs a single convolutional GRU and uses the output state s_t as the agent's state, defined in Equation (4.2). The state s_t has the same shape $n_1 \times n_1$ as each channel of the observation, which is ensured by the convolution operation

in Equation (4.2). We take advantage of GRU that the reset gate v_t in Equation (4.2) controls the choice between long-term dependencies and short-term observations. The former is important for exploring future attention deployment within an episode, while the latter is important for exploiting currently available information to make temporary decisions. By training over a large number of episodes, the agent learns to balance exploration and exploitation from the reinforcement signals by updating its gate parameters.

With Equation (4.2), Equation (4.3), and the hard attention mechanism, each rollout is computed in constant time with respect to the image size as T and n_i are fixed. As a result, a trained RED model is able to make predictions very efficiently.

4.3.2 Prediction aggregation

We present the framework to generate temporary predictions and subsequently aggregate the temporary predictions into the final prediction, i.e. the detection of the patterns. At each time step t, the agent has access to the output state s_t of the GRU which carries the information from the current patch x_t. Based on s_t the agent makes a temporary prediction $\hat{\mathcal{Y}}$ using a feed-forward network followed by a non-linear operation $\hat{y}_t = \frac{1}{2}(1 + \tanh(W_{ys}s_t))$, where $\hat{y}_t$ is the estimated probability that the object exists in $\mathcal{I}$.

The temporary predictions are aggregated over time using

our newly proposed k-maximum aggregation layer. The layer calculates the weighted average of the top k largest values among $\hat{y}_{t_0}, \ldots, \hat{y}_T$, where $t_0 \geq 1$ is a fixed threshold of the model. The output $\hat{\mathcal{Y}}$ of the k-maximum layer is formulated as

$$\hat{\mathcal{Y}} = \frac{1}{Z} \sum_{t \in K} (1 - \gamma^t) \hat{y}_t, \tag{4.4}$$

where $K = k\text{-argmax}_{t_0 \leq t \leq T}\{\hat{y}_t\}$ is the set of the indexes of the top k-largest temporary predicted probabilities, and $Z = \sum_{t \in K}(1 - \gamma^t)$ is the normalizer to guarantee $0 \leq \hat{\mathcal{Y}} \leq 1$. In Equation (4.4) we elaborate a time discount factor $1 - \gamma^t$ which assigns a larger value toward the late stages of the process than the early stages of the process, where γ is fixed through the process. The factor γ is a trade-off between RAM where all previous steps are used to benefit the prediction at the end of the episode and majority voting where all observation contributes to the binary determination.

The advantage of using the k-maximum layer is to guide the model to balance between exploration and exploitation.[4] Considering that only steps t with top k largest $\hat{y}_t$ are taken into account in the final prediction, the model has a sufficient number of time steps to explore different locations on $\mathcal{I}$ and does not need to worry about affecting the final prediction. In fact, exploring the context of the image is important to collect information and locate the detection objective in late stages. The

[4]It also helps to address the problem of vanishing gradient at the same time, though, it is out of the scope of this paper.

time discount factor further reinforces that by assigning larger weights toward late stages, which encourages the agent to explore at the early stages of the process and exploit at the late stages of the process.

Viewing our proposed prediction aggregation mechanism from an RL perspective, it addresses the credit assignment problem [42]. Existing studies on applications via policy learning, e.g. [57, 71, 135], commonly equally assign the feedback of an episode toward all actions the agent has made. The large variance of estimating the quality of a single action using the outcome of the entire episode is neutralized by training the agent for millions of episodes. However, in our settings the state of the environment is diverse as each different image $\mathcal{I}$ corresponds to a unique initial state of the environment. The variance cannot be reduced by simply training on a large dataset of images without a fixed observation function with respect to $\mathcal{I}$. In this way, our proposed aggregation mechanism is necessary to help the algorithm to converge and it is the key component for RED to make detection decisions.

4.3.3 Policy gradient estimation

In this section we derive the estimator of the policy gradient. It is feasible to apply the policy gradient theorem [61, 2, 104], but since we know the exact formulation of the reward function we can largely reduce the variance by incorporating this infor-

mation. To achieve this, we derive the estimator specifically for RED from scratch by taking the derivative of the expected cumulative regret, defined as the negative reward [130].

Let W denote the set of trainable parameters including θ, W_{as}, W_{xa}, W_{ys} and the trainable parameters in the GRU, in Equation (4.2). Also let $\mathcal{Y} \in \{0, 1\}$ be the ground truth of the detection result, where 0 and 1 correspond to the existence and non-existence of the object, respectively. Define a rollout $\hat{\tau_T}$ of the trajectory within an episode to be a sample drawn from the probability distribution $\mathbb{P}(\tau_T|\pi_\theta(\cdot))$. During training, the agent generates its rollouts $\hat{\tau}_T$ and predictions $\hat{\mathcal{Y}}$ on an iterator of $(\mathcal{I}, \mathcal{Y})$ pairs, where each pair of the image and the ground truth corresponds to one episode of RL. Define the regret L_T to be the squared error between the predicted probability and the ground truth

$$L_T = (\hat{\mathcal{Y}} - \mathcal{Y})^2. \tag{4.5}$$

The model updates W after the conclusion of each episode, when it receives a reward signal $r_T = 1 - L_T$. In this case, $L_T + R_T = L_T + r_T = 1$.

We utilize similar arguments in the policy gradient theorem to address the non-differentiability. Let $\boldsymbol{a} = (\hat{a}_1, \ldots, \hat{a}_T)$ be the sequence of actions in $\hat{\tau}_T$, we have the expected regret

$$\mathbb{E}[L_T|W] = \sum\nolimits_{\boldsymbol{a}} \mathbb{P}(\boldsymbol{a}|W)(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})^2, \tag{4.6}$$

where the deterministic variable $\hat{\mathcal{Y}}_{\boldsymbol{a}}$ denotes the model's counterfactual prediction under the condition that $\boldsymbol{a}$ is sampled with

probability one. Since there is no randomness involved on the environment side, the expectation above is calculated over the actions only. Taking the derivative with respect to W, the gradient is

$$\begin{aligned}\nabla_W \mathbb{E}[L_T|W] = \mathbb{E}_{\boldsymbol{a}\sim\mathbb{P}(\boldsymbol{a}|W)}[(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})^2 \\ \nabla_W \log \mathbb{P}(\boldsymbol{a}|W) + 2(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})\nabla_W \hat{\mathcal{Y}}_{\boldsymbol{a}}],\end{aligned} \tag{4.7}$$

where the immediate partial derivative from the chain rule of Equation (4.3) is

$$\nabla_W \log \mathbb{P}(a_t|W) = \frac{1}{\beta^2} \cdot (a_t - \mathbb{E}[a_t|W]) s_{t-1}^T. \tag{4.8}$$

Further, deduct from the regret the baseline function

$$b_T = \mathbb{E}_{\boldsymbol{a}\sim\mathbb{P}(\boldsymbol{a}|W)}[(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})^2], \tag{4.9}$$

which calculates the expected regret from the rollouts, for variance reduction. By doing this we account only the difference between the actual reward and the baseline function. Note that the baseline function introduces no bias into the expectation in Equation (4.7), while it is used to reduce the variance when estimating the policy gradient using the Monte-Carlo samples $\boldsymbol{a}$. At the end of each episode, update W according to

$$\begin{aligned}W \leftarrow\ & W - \alpha \mathbb{E}_{\boldsymbol{a}\sim\mathbb{P}(\boldsymbol{a}|W)}[((\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})^2 - b_T) \\ & \nabla_W \log \mathbb{P}(\boldsymbol{a}|W) + 2(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})\nabla_W \hat{\mathcal{Y}}_{\boldsymbol{a}}],\end{aligned} \tag{4.10}$$

where α is the learning rate.

To estimate the expectation in Equation (4.10), the agent generates a rollout $\hat{\tau}_T$ which samples $\boldsymbol{a}$ according to $\boldsymbol{a} \sim \mathbb{P}(\boldsymbol{a}|W)$.

The expectation is then estimated using the generated $\boldsymbol{a}$ value by Equation (4.8) and REINFORCE's back-propagation [195]. Note that the second part $2(\hat{\mathcal{Y}}_{\boldsymbol{a}} - \mathcal{Y})\nabla_W \hat{\mathcal{Y}}_{\boldsymbol{a}}$ of the gradient is useful, though, it is sometimes ignored in previous studies [104]. It connects the regret to the early stages which allows the regret signal to be back-propagated directly to those steps and to guide the exploitation of the agent. It can be regarded as a retrospective assignment of the credits after the rollout has been fully generated, equivalently making the reward r_t in RED no longer 0 when $t < T$ during the training phase, which addresses the issues caused by delayed rewards.

4.4 Experiments

4.4.1 Stained MNIST

We first test and compare RED on our synthetic dataset, *stained MNIST*, with a variety of baseline methods. Stained MNIST contains a set of handwritten digits, which have very high resolution and much thinner writings than the original MNIST does. Each digit may be associated with multiple *stains* on the edge of its writing, which are dot-shaped regions with high tonal value. The algorithms are required to predict if such stains exist in the images. The task is very challenging as the image resolution is very high while the writings are thin and unclear. Hence it is hard to locate the stains or recognize the stains from the

writings.

Stained MNIST is constructed by modifying the original MNIST dataset as follows. Each image from MNIST is first resized to 7168×7168 by bilinear interpolation, and rescaled to 0 to 1 tonal value. The enlarged images are then smoothed using a Gaussian filter with a 20×20 kernel. After that, it calculates the central differences of each pixel and finds out the set C of pixels with 0.2 or larger gradient. The tonal values of pixels that are within 500 pixels of C are set to 0. This operation makes the writings of the digits much thinner in the high-resolution images. After removing those pixels, the gradient of each pixel is calculated again, and 10 to 15 stains with radius 12 are randomly added at pixels with high gradient.

The hyper-parameters of RED are set to be $c = 3, n_1 = 18, n_2 = 36, n_3 = 54$ for attention mechanism and $\gamma = 0.95, k = 25, t_0 = 10$ for prediction aggregation, through a random search on a training subset. The search over $\gamma \in [0.9, 0.98]$, $k \in [15, 30]$, and $t_0 \in [10, 50]$ does not observe significant difference on the performance. Accordingly, the patch size x_t is set to $n_1 \times n_1$ as the input of the GRU. The horizon is fixed to $T = 350$, where no significant improvement can be observed by further increasing it. When estimating the baseline function b_T, 15 instances are sampled and are averaged over. When evaluating RED, we remove the stochastic components ϵ_t in computing the actions.

We compare both the accuracy and the average runtime to make a prediction by RED with the baseline approaches, includ-

Table 4.1: Comparisons of RED with different baseline approaches on Stained MNIST.

Approach	Runtime (s)	Accuracy (test)
RED	0.06	84.43%
Random $\boldsymbol{a}$	0.06	51.79%
RAM	0.06	62.35%
ConvNet-3	1.95	81.49%
ConvNet-4	3.30	82.92%

ing RAM with the same set of parameters, a 3-layer ConvNet, a 4-layer ConvNet, and RED where the attention $\hat{a}_t$ is uniformly randomly selected from $\mathcal{A}_t = [-1, 1]^2$. The last baseline is used to show the necessity of the learned attention mechanism. As shown in Table 4.1, RED significantly outperforms all baselines in terms of accuracy, and all attention-based models have better speed compared with ConvNet-based algorithms.

4.4.2 Diabetic retinopathy screening

Diabetic retinopathy (DR) [196] is among the leading causes of blindness in the working-age population of the developed world. Its consequence of vision loss is effectively prevented by population-wise DR screening, where automatic and efficient DR screening is an interesting problem in medical image analysis [197, 198, 199]. The screening process is to detect abnormality from the fundus photographs, which are generally in high resolution and are noisy due to the photo-taking procedure. The

high-resolution, low signal-to-noise ratio, and the need for efficient population-wise screening agree with the characterizations of our proposed RED model, which motivates us to test the model on this task. We test and compare the performance using a dataset publicly available on Kaggle.[5] While the images are originally rated with five levels, we consider level 0 and 1 as negative results $\mathcal{Y} = 0$ and level 2, 3 and 4 as positive results $\mathcal{Y} = 1$. The results are shown in Table 4.2, where the same hyper-parameters are used as is in the stained MNIST experiment.

The performance of our RED approach is compared with RAM and ConvNet with both four layers and five layers. Also, we test ConvNet with fractional max-pooling layers [103] and cyclic pooling layers [102] which have solid performances on the Kaggle challenge. We re-implement their approach with 4 and 5 layers (*ConvNet-4+* and *ConvNet-5+*) and the comparisons are shown in Table 4.2.

Our RED approach achieves extraordinary speed performance while demonstrating competitive accuracy. Notably, compared with the ConvNet-based methods which usually take many seconds to process each image, RED provides a way to trade marginal accuracy to significant speed improvement. That could be critical especially for the DR screening tasks designed to be used on population-wise datasets while requiring timely results. Apart from the speed improvement, it is worth noting that RED is

[5] https://www.kaggle.com/c/diabetic-retinopathy-detection

Table 4.2: Comparisons of RED with different baseline approaches on diabetic retinopathy screening.

Approach	Runtime (s)	Accuracy (test)
RED	0.04	91.55%
Random $\boldsymbol{a}$	0.04	53.44%
RAM	0.04	81.35%
ConvNet-4	2.32	90.61%
ConvNet-4+	2.32	91.97%
ConvNet-5	2.92	91.84%
ConvNet-5+	2.92	92.29%

also light weighted, where the number of parameters needed is relatively low to process only small patches at any time step. The experiments on DR screening demonstrate that our RED method is practical enough to be applied to real-world systems.

4.4.3 Intuitive demonstration of trajectories

To understand the policy that deploys the agent's attention, we present a graphical demonstration of the trajectory, which imitates the way humans process existence detection tasks. As shown in Figure 4.3 top, the trained agent predicts if patterns related to DR exist in a fundus image. To observe the trajectory, we put the limit $T \rightarrow \infty$ on the time horizon while keeping the stochastic components ϵ in Equation (4.3). We then illustrate the distribution of the attentions, in the form of a heat map, in Figure 4.3 bottom.

We first observe that the majority the attention deployed are concentrated in the bottom right part of the image, which coincides with the lesion patterns (yellow stains on the fundus image). Within the small blue box marked on Figure 4.3 top concentrates 30 out of the first 250 saccades. It shows the ability of the trained model to locate regions of interest and to deploy its limited attention resource selectively. Notably, only 4 of them happen in the first 100 time steps, and the density of attention for $T \to \infty$ becomes even higher (≥ 9 heat value on Figure 4.3 bottom). On the other hand, we observe that the model tends to deploy its attentions on around the blood vessels especially at the early stages of the process. Such behavior helps the agent to gain information about the context of the image and locate the region of interest in the later stages of the process. Also, it is worth noting that the agent does not get stuck in a small region even when we set the time horizon to be arbitrarily large. Instead, the agent keeps exploring the image indefinitely. The way the agent automatically balance exploitation and exploration is what we have been expecting an RL algorithm to learn.

4.5 Asymmetric actor-critic

We present recurrent existence determination, a novel RL algorithm for existence detection. RED imitates the attention mechanism that humans elaborate to process object detection both efficiently and precisely, yielding similar characterizations

as desired. RED employs hard attention which boosts the test-time speed while the non-differentiability introduced by the attention mechanism is addressed via policy optimization. We propose the k-maximum aggregation layer and other components in RED which help to solve the delayed reward problem and automatically learns to balance exploration and exploitation. Experimental analysis shows significant speed and accuracy improvement compared with previous approaches, on both synthetic and real-world datasets.

One of the plausible future directions is to further address the delayed reward problem, by adding a value network as the critic the actor-critic method [2, 57]. The critic will give the agent immediate feedback for any action it takes, using the estimation of the action-state value function. In this case, as the environment is partially observable, the actor-critic need to be asymmetric where the critic will have access to the full image [195, 196, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210]. The critic network is expected to be a proper replacement of the aggregation layer in this thesis with an improved performance.

□ **End of chapter.**

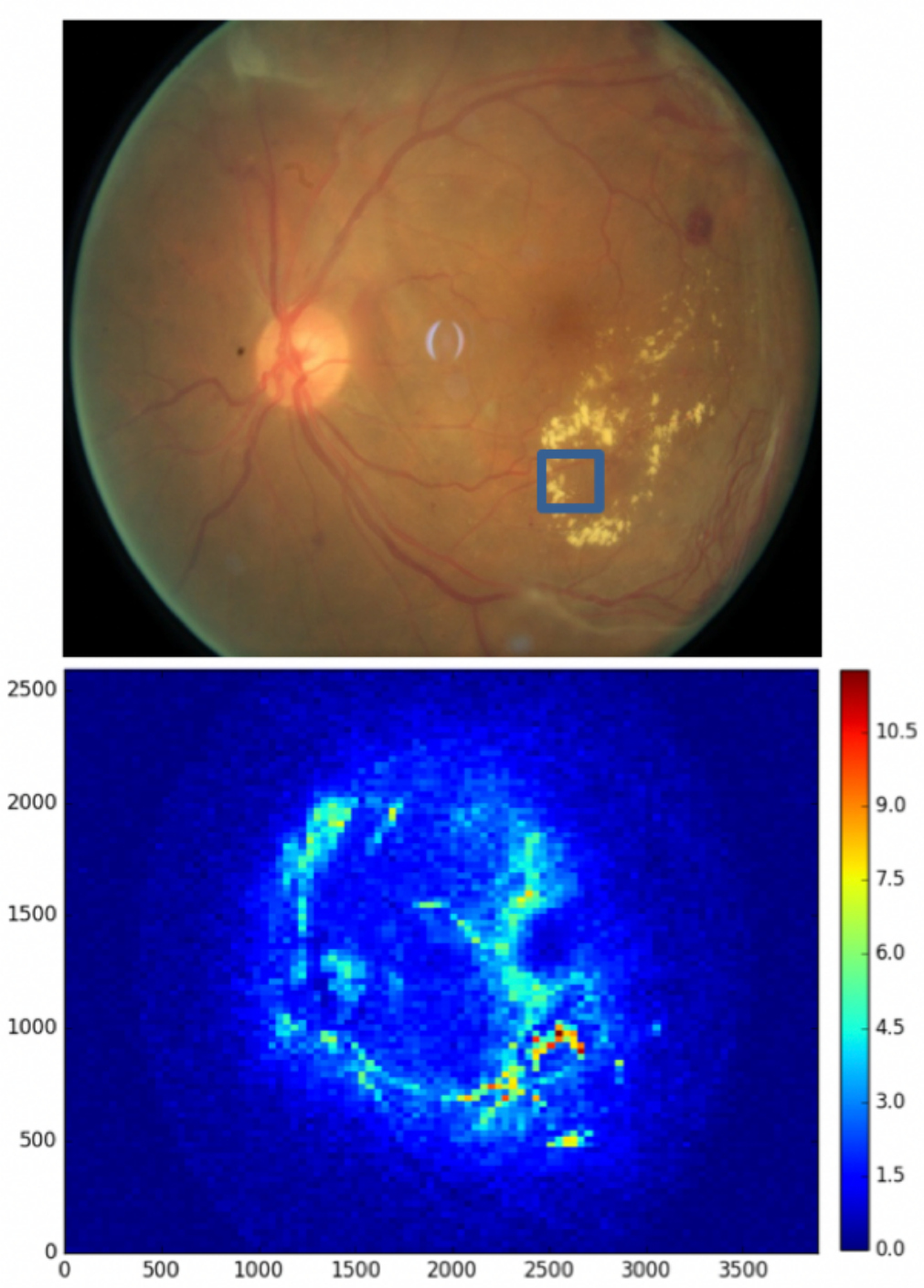


Figure 4.3: Distribution of the deployed attentions in a rollout of the recurrent existence determination model.

Chapter 5

The Foundations

As we have discussed in the introduction and the history of reinforcement learning, RL origins from the intersection of optimal control and cognitive science. The former formulates the problem of sequential decision making while the latter describes what a learning process should be like. This intersection positions the very core of reinforcement learning into the model-free, trial-and-error style of learning algorithms. But what we are discussing in this section is the foundations of such a process, that is, what the solution is truly like and what characterizations it possesses thereafter. We discard both the model-free and trial-and-error constraints, and arm ourselves with fundamental mathematical tools to investigate the analytical solution - the one the learning algorithms desire to converge to.

We analyze the Gambler's problem, a simple reinforcement learning problem where the gambler has the chance to double or lose the bets until the target is reached. This is an early example

introduced in the reinforcement learning textbook by [2], where they mention an interesting pattern of the optimal value function with high-frequency components and repeating non-smooth points. It is however without further investigation. We provide the exact formula for the optimal value function for both the discrete and the continuous cases. Though simple as it might seem, the value function is pathological: fractal, self-similar, derivative taking either zero or infinity, and not written as elementary functions. It is in fact one of the generalized Cantor functions, where it holds a complexity that has been uncharted thus far. Our analyses could provide insights into improving value function approximation, gradient-based algorithms, and Q-learning, in real applications and implementations.

5.1 Solving the Gambler's problem

We analytically investigate a deceptively simple problem, the Gambler's problem, introduced in the reinforcement learning textbook by [2] in Example 4.3. The problem setting is natural and simple enough that little discussion was given in the book apart from a numerical solution by value iteration. A close inspection will however show that the problem, as a representative of the entire family of Markov decision processes (MDP), involves a level of complexity and curiosity uncharted in years of reinforcement learning research.

The problem discusses a typical double-or-nothing casino game,

where the gambler places multiple rounds of betting. The gambler gains the bet amount upon winning a round or loses the bet upon losing the round, with probability $1-p \leq 0.5$ and $p \geq 0.5$, respectively. The game terminates when the gambler's capital reaches either the target or 0.

In each round, the gambler must decide what portion of the capital to stake. In the discrete setting the target and the bet amount must be integers, but both can be real numbers in the continuous setting. The problem is formulated as an MDP (see Section 5.1.1 for a preliminary), where state s is the current capital and action a is the bet amount. The reward is $+1$ when the state reaches the target state, and zero otherwise. Our goal is to solve the optimal value function of the MDP.

We first give the solution to the discrete Gambler's problem. Denote $N \in \mathbb{N}^+$ as the target capital, $n \in \mathbb{N}^+$ as the current capital (which is the state in the discrete setting), $p > 0.5$ as the probability of losing a bet, and γ as the discount factor. The special case of $N = 100$, $\gamma = 1$ corresponds to the original setting described in Sutton and Barto's book.

Proposition 28. *Let* $0 \leq \gamma \leq 1$ *and* $p > 0.5$. *The optimal value function* $z(n)$ *is* $v(n/N)$ *in the discrete setting of the Gambler's problem, where* $v(\cdot)$ *is the optimal value function under the continuous case defined in Theorem 39.*

The above statement reduces the discrete problem to the continuous problem by a uniform discretization. The rest of the

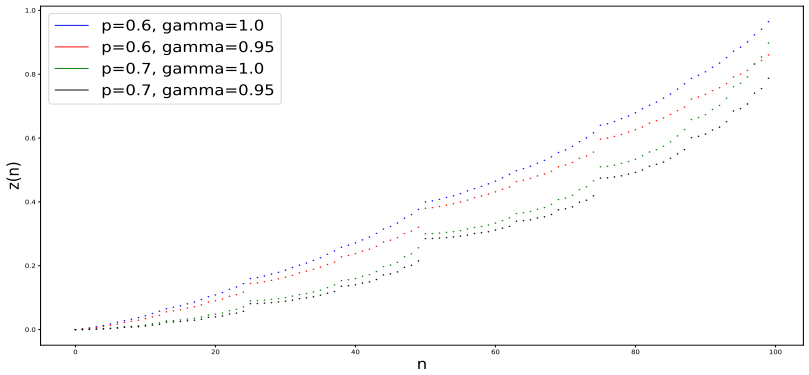


Figure 5.1: The optimal state-value function of the discrete Gambler's problem.

discussion will be on the more general continuous setting.

In the continuous setting, let the target capital be 1 without loss of generality. The state space is then $[0, 1]$, and the action space is $0 < a \leq \min\{s, 1 - s\}$ at state s, meaning that the bet can be any fraction of the current capital as long as the capital after winning does not exceed 1 (the target). We give the optimal state-value function below and some intuitive descriptions later in this section.

Theorem 39. *Let $0 \leq \gamma \leq 1$ and $p > 0.5$. Under the continuous setting of the Gambler's problem, the optimal state-value function is $v(1) = 1$, and*

$$v(s) = \sum_{i=1}^{\infty} (1-p)\gamma^i b_i \prod_{j=1}^{i-1} ((1-p) + (2p-1)b_j) \tag{5.1}$$

for $0 \leq s < 1$, where $s = 0.b_1 b_2 \ldots b_\ell \ldots_{(2)}$ is the binary representation of the state s.

Next, we solve the Bellman equation of the continuous Gambler's problem, that is, $f(0) = 0$, $f(1) = 1$, and

$$f(s) = \max_{0<a\leq \min\{s,1-s\}} (1-p)\gamma f(s+a) + p\gamma f(s-a), \qquad (5.2)$$

for a real function f. In the strictly discounted case $0 \leq \gamma < 1$, the solution of the Bellman equation is the optimal value function $f(s) = v(s)$ (Proposition 48).

This uniqueness does not hold in general. If the reward is not discounted, the solution of the Bellman equation is either the optimal value function, or a constant function larger than 1.

Theorem 49. *Let $\gamma = 1$, $p > 0.5$, and $f(\cdot)$ be a real function on $[0, 1]$. $f(s)$ solves the Bellman equation if and only if either*

- *$f(s)$ is $v(s)$ defined in Theorem 39, or*
- *$f(0) = 0$, $f(1) = 1$, and $f(s) = C$ for all $0 < s < 1$, for some constant $C \geq 1$.*

Under the most complicated case of $\gamma = 1$, $p = 0.5$, the problem involves midpoint concavity and Cauchy's functional equation [211, 212]. The measurable functions that solve the Bellman equation are $f(s) = C's + B'$, $s \in (0, 1)$, for $C' + B' \geq 1$. Additionally, there exists non-constructive, non Lebesgue measurable solutions under Axiom of Choice (Theorem 54).

Though the description of the Gambler's problem seems natural and simple, Theorem 39 shows that its simpleness is deceptive. The optimal value function is fractal, self-similar, and

non-rectifiable (see Corollary 41 and Lemma 35). It is thus not smooth on any interval, which can be unexpected when a significant line of reinforcement learning studies is based on function approximation like discretization and neural networks. The value function (5.1) can neither be simplified into a formula of elementary functions, which introduces additional difficulties to analyses of algorithms. The function is monotonically increasing with $v(0) = 0$ and $v(1) = 1$, but its derivative is 0 almost everywhere, which is counterintuitive. This is known as singularity, a famous pathological property of functions. $v(s)$ is continuous almost everywhere but not absolutely continuous. Also when γ is strictly smaller than 1, it is discontinuous on a dense and compact set of infinitely many points. These properties indicate that assumptions like smoothness, continuity, and approximability are not satisfied in this problem. In general, it is reasonable to doubt if these assumptions can be imposed in reinforcement learning. To better understand the pathology of $v(s)$, we analogize it to the Cantor function, which is well known in analysis as a counterexample of many seemingly true statements [213]. In fact, $v(s)$ is a generalized Cantor function, where the above descriptions are true for both $v(s)$ and the Cantor function.

Intuitive descriptions of $v(s)$. All the statements above require the definition of $v(s)$. In fact, in this paper, $v(s)$ is important enough such that its definition will not change with the context. The function cannot be written as a combination of elementary functions. Nevertheless, we give an intuitive way to

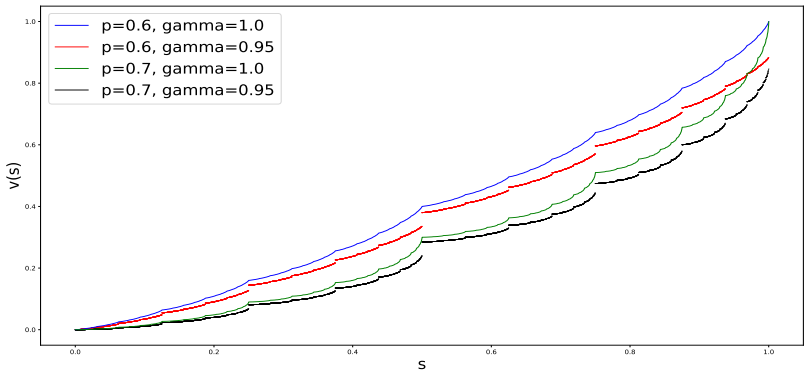


Figure 5.2: The optimal state-value function of the continuous Gambler's problem.

understand the function for the original, undiscounted problem. The function can be regarded as generated by the following iterative process. First we fix $v(0) = 0$ and $v(1) = 1$, and compute

$$v(\frac{1}{2}) = pv(0) + (1-p)v(1) = (1-p).$$

Here, $v(\frac{1}{2})$ is the weighted average of the two "neighbors" $v(0)$ and $v(1)$ that have been already evaluated. Further, the same operation applies to $v(\frac{1}{4})$ and $v(\frac{3}{4})$, where $v(\frac{1}{4}) = pv(0) + (1-p)v(\frac{1}{2}) = (1-p)^2$ and $v(\frac{3}{4}) = pv(\frac{1}{2}) + (1-p)v(1) = (1-p) + p(1-p)$, and so forth to $v(\frac{1}{8})$, $v(\frac{3}{8})$, etc. This process evaluates $v(s)$ on the dense and compact set $\bigcup_{\ell \geq 1} G_\ell$ of the dyadic rationals, where $G_\ell = \{k2^{-\ell} \mid k \in \{1, \ldots, 2^\ell - 1\}\}$. With the fact that $v(s)$ is monotonically strictly increasing, these dyadic rationals determine the function $v(s)$ uniquely.

This iterative process can also be explained from the analytical formula of $v(s)$. Starting with the first bit, a bit of 0 will

not change the value, while a bit of 1 will add $(1-p)\prod_{j=1}^{i-1}((1-p)+(2p-1)b_j)$ to the value. This term can also be written as $(1-p)((1-p)^{\#\mathbf{0}\text{ bits}}\cdot p^{\#\mathbf{1}\text{ bits}})$, where the number of bits is counted over all previous bits. The value $(1-p)^{\#\mathbf{0}\text{ bits}}\cdot p^{\#\mathbf{1}\text{ bits}}$ decides the gap between two neighbor existing points in the above process, when we insert a new point in the middle. This insertion corresponds to the iteration on G_ℓ over ℓ.

We provide high resolution plots of $z(n), N = 100$ and $v(s)$ in Figure 5.1 and Figure 5.2, respectively. The non-smoothness and the self-similar fractal patterns can be clearly observed from the figures. Also, $v(s)$ is continuous when $\gamma = 1$ while $v(s)$ is not continuous on infinitely many points when $\gamma < 1$. In fact, when $\gamma < 1$, the function is discontinuous on the dyadic rationals $\bigcup_{\ell\geq 1} G_\ell$ while continuous on its complement, as we will rigorously show later.

Self-similarity. The function $v(s)$ on $[\bar{s}, \bar{s}+2^{-\ell}]$ for any $\bar{s} = 0.b_1b_2\ldots b_{\ell(2)}$, $\ell \geq 1$ is self-similar to the function itself on $[0,1]$. Let $s = 0.b_1b_2\ldots b_\ell\ldots_{(2)} \in [\bar{s}, \bar{s}+2^{-\ell}]$, this can be observed by

$$\begin{aligned} v(s) &= \sum_{i=1}^{\infty}(1-p)\gamma^i b_i \prod_{j=1}^{i-1}((1-p)+(2p-1)b_j) \\ &= \sum_{i=1}^{\ell}(1-p)\gamma^i b_i \prod_{j=1}^{i-1}((1-p)+(2p-1)b_j) \\ &\quad + \gamma^\ell(\prod_{j=1}^{\ell}((1-p)+(2p-1)b_j)) \end{aligned} \tag{5.3}$$

$$\times \sum_{i=1}^{\infty}(1-p)\gamma^i b_{\ell+i}\prod_{j=1}^{i-1}((1-p)+(2p-1)b_{\ell+j})$$

$$= v(\bar{s}) + \gamma^\ell \prod_{j=1}^{\ell}((1-p)+(2p-1)b_j)\cdot v(2^\ell(s-\bar{s})). \quad (5.4)$$

The self-similarity can be compared with the Cantor function [213, 214], which uses the ternary of s instead in the formula. The Cantor function is self-similar to itself on $[\bar{s}, \bar{s}+3^{-\ell}]$, when $\bar{s} = 0.b_1 b_2 \ldots b_{\ell(3)}$ and $b_\ell \neq 1$. Both $v(s)$ and the Cantor function can be uniquely described by their self-similarity, the monotonicity, and the boundary conditions.

Optimal policies. It is immediate by Theorem 39 and Lemma 35 that

$$\pi(s) = \min\{s, 1-s\}$$

is one of the (Blackwell) optimal policies. Here, Blackwell optimality is defined as the uniform optimality under any $0 \leq \gamma \leq 1$. This policy agrees with the intuition that under a game that is in favor of the casino ($p > 0.5$), the gambler desires to bet the maximum to finish the game in as little cumulative bet as possible. In fact, the probability of reaching the target is the expected amount of capital by the end of the game, which is negative linear to the cumulative bet.

The optimality is not unique though, for example, $\pi(\frac{15}{32}) = \frac{1}{32}$ is also optimal (for any γ). Under $\gamma = 1$ the original, undiscounted setting, small amount of bets can also be optimal. Namely, when s can be written in finitely many bits

$s = b_1 b_2 \dots b_{\ell(2)}$ in binary (assume $b_\ell = 1$), $\pi(s) = 2^{-l}$ is also an optimal policy. This policy is by repeatedly rounding the capital to carryover the bits, keeping the game to be within at most ℓ rounds of bets.

5.1.1 Preliminaries

We use the canonical formulation of the discrete-time Markov decision process (MDP), denoted as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \rho_0, \gamma)$. That includes $\mathcal{S}$ the state space, $\mathcal{A}$ the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^+$ the stochastic transition probability function, $r : \mathcal{S} \to \mathbb{R}$ the reward function, $\rho_0 \in \mathcal{S}$ the initial state, and $\gamma \in [0, 1]$ the unnormalized discount factor. A deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ is a map from the state space to the action space. In this problem, $\mathcal{T}(s, a, s-a)$ and $\mathcal{T}(s, a, s+a)$ are p and $1-p$ respectively for $s \in \mathcal{S}$, $a \in \mathcal{A}$, and $\mathcal{T}(\cdot)$ is otherwise 0.

Our goal is to solve the optimal value function of the Gambler's problem, which is the probability that the gambler reaches the target from an initial state. The definition of the state-value function of an MDP with respect to state s and policy π is

$$f^\pi(s) = \mathbb{E}[\sum_t \gamma^t r_t \Big| s_0 = s, a_t = \pi(s_t), r_t = r(s_t), \\ s_{t+1} \sim \mathcal{T}(s_t, a_t), t = 0, 1, \dots].$$

When $\pi^*(\cdot)$ is one of the optimal policies, $f^{\pi^*}(s)$ is the optimal state-value function. Despite that multiple optimal policies can exist, this optimal state-value function is unique [2, 3].

5.1.2 Implications

Our results indicate hardness on reinforcement learning [215, 216, 217] and revisions of existing reinforcement learning algorithms. It is worth noting that similar patterns of fractal and self-similarity have been observed empirically, for example in [218] for the Mountain Car problem. With these characterizations being observed in simple problems like Mountain Car and the Gambler's problem, our results are expected to be generalized to a variety of reinforcement learning settings.

The first implication is naturally on value function approximation, which is a developed topic in reinforcement learning [219, 220]. By the fractal property of the optimal value function, that representation of such function must be inexact [221]. When discretization is used for value function representation, the approximation error is at least $O(1/N)$, where N is the number of bins.

Proposition 46. *When $N \in \mathbb{N}^+$, $N \geq 4$ is a power of 2, let $\bar{v}_1(s)$ be piecewise constant on any of the intervals $s \in (k/N, (k+1)/N)$, $k = 0, \ldots, N-1$, then*

$$\int_s |v(s) - \bar{v}_1(s)|\, ds \geq \frac{1}{N} \frac{(2-\gamma)(1-p)\gamma}{1-p\gamma} + o(\frac{1}{N}).$$

Alternatively, when a subclass of L-Lipschitz continuous functions is used to represent $v(s)$, this error is then at least $(1/L) \cdot (1-p)^2\gamma^2(1-\gamma)^2/(4-4p\gamma)$ by the discontinuity of $v(s)$ (Proposition 47). It is worth remarking that despite this specific lower

bound diminishes when γ is 1, the approximation error is nonzero for an arbitrarily large L under $\gamma = 1$, as the derivative of $v(s)$ can be infinite (Fact 43).

Notably, neural networks are within this family of functions, where the Lipschitz constant L is determined by the network architecture. By the proposition, it is not possible to obtain the optimal value function when a neural network is used, albeit the universal approximation theorem [213, 222, 223].

The second implication is by Theorem 39 and Fact 43 that the derivative of $v(s)$ is

$$\lim_{\Delta s \to 0^+} \frac{v(s+\Delta s)}{\Delta s} = 0,$$

$$\lim_{\Delta s \to 0^-} \frac{v(s+\Delta s)}{\Delta s} = \begin{cases} +\infty, & \text{if } s = 0 \text{ or } s \in \bigcup_{\ell \geq 1} G_\ell, \\ 0, & \text{otherwise.} \end{cases}$$

This imposes that the value function's derivative must not be exactly obtained, as it is 0 almost everywhere, except on the dyadic rationals G_ℓ, where it has a left derivative of infinity and a right derivative of 0. Algorithms relying on $\partial v(s)/\partial s$ and $\partial Q(s,a)/\partial a$ [69, 224, 225, 226, 227, 228, 229], where $Q(s,a)$ is the action-value function [2], can suffer from the estimation error or even have unpredictable behavior.

In practice, the boolean implementation of float numbers can further increase this error, as all points s implemented are in G_ℓ for some ℓ. A precise evaluation requires all these derivatives to be infinity when a Lebesgue derivative is used (the average of left

and right derivatives), which cannot be obtained in a computer system.

The third implication is on Q-learning [12, 47, 230], by Theorem 49 and its supporting lemmas. It is proved that when $\gamma = 1$, Q-learning has multiple converging points, as the Bellman equation has multiple solutions, namely $v(s)$ and

$$f(0) = 0, f(1) = 1, \text{ and } f(s) = C \text{ for all } 0 < s < 1,$$

for some constant $C \geq 1$. Therefore, even when the Q-learning algorithm converges, it may not converge to the optimal value function $v(s)$. In fact, as the solution can be either the ground truth of the optimal value function, or a large constant function, it is easier to approximate a constant function than the optimal value function, resulting in a relatively lower Bellman error when converging to the large constant.

This challenges Q-learning under $\gamma = 1$ when the return (cumulative reward) is unbiased. Though the artificial γ is originally introduced to prevent the return from diverging, it can be also necessary to prevent the algorithm from converging to a large constant in Q-learning, which is not desired.

5.2 Discrete Case

The analysis of the discrete case of the Gambler's problem will give an exact solution. It will also explain the reason the plot on the book has a strange pattern of repeating spurious points.

The discrete case can be described by the following MDP: The state space is $\{0, \ldots, N\}$; the action space at n is $\mathcal{A}(n) = \{0 < a \leq \min\{n, N-n\}\}$; the transition from state n and action a is $n-a$ and $n+a$ with probability p and $1-p$, respectively; the reward function is $r(N) = 1$ and $r(n) = 0$ for $0 \leq n \leq N-1$. The MDP terminates at $n \in \{0, N\}$. We use a time-discount factor of $0 \leq \gamma \leq 1$, where the agent receives $\gamma^T r(N)$ rewards if the agent reaches the state $n = N$ at time T.

Let $z(n)$, $n \in \mathcal{N}, 0 \leq n \leq N$, be the value function. The exact solution below of the discrete case is relying on Theorem 39, our main theorem which describes the exact solution of the continuous case. This theorem will be discussed and proved later in Section 5.4.1.

Proposition 28. *Let $0 \leq \gamma \leq 1$ and $p > 0.5$. The optimal value function $z(n)$ is $v(n/N)$ in the discrete setting of the Gambler's problem, where $v(\cdot)$ is the optimal value function under the continuous case defined in Theorem 39.*

Proof. We first verify the Bellman equation. By the definition of $v(\cdot)$ we have

$$\begin{aligned} z(n) &= v(n/N) \\ &= \max_{0<a\leq \min\{n/N, 1-n/N\}} p\gamma\ v(n/N - a) + (1-p)\gamma\ v(n/N + a) \\ &\geq \max_{0<a\leq \min\{n/N, 1-n/N\}, Na\in\mathbb{N}} p\gamma\ v(n/N - a) \\ &\quad + (1-p)\gamma\ v(n/N + a) \end{aligned}$$

$$= \max_{0<a\leq\min\{n,N-n\},a\in\mathbb{N}} p\gamma\ z(n-a) + (1-p)\gamma\ z(n+a).$$

Meanwhile let $a^* = \min\{n, N-n\}$, Corollary 40 suggests that

$$\begin{aligned} z(n) &= v(n/N) \\ &= p\gamma\ v((n-a^*)/N) + (1-p)\gamma\ v((n+a^*)/N) \\ &= p\gamma\ z(n-a^*) + (1-p)\gamma\ v(n+a^*) \\ &\leq \max_{0<a\leq\min\{n,N-n\},a\in\mathbb{N}} p\gamma\ z(n-a) + (1-p)\gamma\ z(n+a). \end{aligned}$$

Therefore $z(n) = \max_{0<a\leq\min\{n,N-n\},a\in\mathbb{N}} p\gamma\ z(n-a) + (1-p)\gamma\ z(n+a)$ as desired.

We then show that $z(n) = v(n/N)$ is the unique function that satisfies the Bellman equation. The proof is similar to the proof of Lemma 29, but the arguments will be relatively easier, as both the state space and the action space are discrete. Let $f(n)$ also satisfy the Bellman equation. We desire to prove that $f(n)$ is identical to $z(n)$.

Define $\delta = \max_{1\leq n\leq N-1} f(n) - z(n)$. This maximum must exist as there are finitely many states. Then define the non-empty set $S = \{n \mid f(n) - z(n) = \delta, 1 \leq n \leq N-1\}$. For any $n' \in S$ and $a' \in \arg\max_{1\leq n\leq\min\{n',N-n'\}} p\gamma\ f(n'-a) + (1-p)\gamma\ f(n'+a)$, we have

$$\begin{aligned} f(n') &= p\gamma\ f(n'-a') + (1-p)\gamma\ f(n'+a') \\ &\overset{(\heartsuit)}{\leq} p\gamma\ (z(n'-a')+\delta) + (1-p)\gamma\ (z(n'+a')+\delta) \\ &\leq p\gamma\ z(n'-a') + (1-p)\gamma\ z(n'+a') + \delta \\ &\leq z(n') + \delta \end{aligned}$$

$$= f(n').$$

As the equality holds, by the equality of (♥) we have $n' - a' \in S$ and $n' + a' \in S$.

Now we specify some $n_0 \in S$ and $a_0 \in \arg\max_{1 \le n \le \min\{n_0, N-n_0\}} p\gamma\, f(n_0 - a) + (1-p)\gamma\, f(n_0 + a)$. Then, we have $n_0 - a_0 \in S$. Denote $n_1 = n_0 - a_0$ and, recursively, $a_t \in \arg\max_{1 \le n \le \min\{n_t, N-n_t\}} p\gamma\, f(n_t - a) + (1-p)\gamma\, f(n_t + a)$ and $n_{t+1} = n_t - a_t$, $t = 1, 2, \ldots$; Since $a_t \ge 1$ and $n_t \in \mathcal{N}$, there must exist a T such that $n_T = 0$. Therefore, $\delta = f(n_T) - z(n_T) = 0$.

By the same argument $\bar{\delta} = \max_{1 \le n \le N-1}\ z(n) - f(n) = 0$. Therefore, $z(n)$ and $f(n)$ are identical, as desired.

As $z(n)$ is the unique function that satisfies the Bellman equation, it is the optimal value function of the problem. □

Proposition 28 indicates the discretization of the problems yields the discrete, exact evaluation of the continuous value function at $0, 1/N, \ldots, 1$. If we omit the learning error, the plots on the book and by the open source implementation [231] are the evaluation of the fractal $v(s)$ at $0, 1/N, \ldots, 1$. This explains the strange appearance of the curve in the figures.

5.3 Setting

We formulate the continuous Gambler's problem as a Markov decision process (MDP) with the state space $\mathcal{S} = [0, 1]$ and the action space $\mathcal{A}(s) = (0, \min\{s, 1-s\}]$, $s \in (0, 1)$. Here $s \in \mathcal{S}$

represents the capital the gambler currently possesses and the action $a \in \mathcal{A}(s)$ denotes the amount of bet. Without loss of generality, we have assumed that the bet amount should be less than or equal to $1 - s$ to avoid the total capital to be more than 1. The consecutive state s' transits to $s - a$ and $s + a$ with probability $p \geq 0.5$ and $1 - p$ respectively. The process terminates if $s \in \{0, 1\}$ and the agent receives an episodic reward $r = s$ at the terminal state. Let $0 \leq \gamma \leq 1$ be the discount factor.

Let $f : [0, 1] \to \mathbb{R}$ be a real function. For $f(s)$ to be the optimal value function, the Bellman equation for the non-terminal and terminal states are

$$f(s) = \max_{a \in \mathcal{A}(s)} p\gamma\ f(s-a) + (1-p)\gamma\ f(s+a) \quad \text{for any} \quad s \in (0, 1), \tag{A}$$

and

$$f(0) = 0, \quad f(1) = 1. \tag{B}$$

It can be shown (later in Lemma 29 and Lemma 30) that a function satisfying (AB) must be lower bounded by 0. A reasonable upper bound is 1, as the value function is the probability of the gambler eventually reaching the target, which must be between 0 and 1. It is also reasonable to assume the continuity of the value function at $s = 0$. Otherwise an arbitrary small amount of starting capital will have at least a constant probability of reaching the target 1.[1] Consequently the expectation

[1]This continuity assumption is only for a better organization of the settings. The

of capital at the end of the game is greater than the starting capital, which contradicts $p \geq 0.5$. The bounded version (X) of the problem leads to the optimal value function.

$$0 \leq \gamma \leq 1,\ p > 0.5,\ f(s) \leq 1 \text{ for all } s,\ f(s) \text{ is continuous on } s = 0. \tag{X}$$

Respectively, the unbounded version (Y) of the problem leads to the solutions of the Bellman equation.

$$0 \leq \gamma \leq 1,\ p > 0.5. \tag{Y}$$

The results extend for $p = 0.5$ in general, except an extreme corner case of $\gamma = 1$, $p = 0.5$, where the monotonicity in Lemma 30 will not apply. This case (Z) involves arguments over measurability and the belief of Axiom of Choice, which we will discuss at the end of Section 5.4.

$$\gamma = 1,\ p = 0.5,\ f(s) \text{ is unbounded.} \tag{Z}$$

We are mostly interested in two settings: the first setting (ABX) and its solution in Theorem 39 discuss a set of necessary conditions of $f(s)$ being the optimal value function of the Gambler's problem. As we show later the solution of (ABX) is unique, this solution must be the optimal value function. The second setting (ABY) and its solutions in Proposition 48 and Theorem 49 discuss all the functions that satisfy the Bellman equation. These functions are the optimal points that value iteration and Q-learning algorithms may converge to. (ABZ) is

Gambler's problem can be solved without this assumption by solving (AB), as described in Section 5.4.2.

interestingly connected to some foundations of mathematics like the belief of axioms, and is discussed in Theorem 54.

5.4 Analysis

5.4.1 Analysis of the Gambler's problem

In this section we show that $v(s)$ defined below is a unique solution of the system (ABX). Since the optimal state-value function must satisfy the system (ABX), $v(s)$ is the optimal state-value function of the Gambler's problem. This statement is rigorously proved in Theorem 39.

Let $0 \leq \gamma \leq 1$ and $p > 0.5$. We define $v(1) = 1$, and

$$v(s) = \sum_{i=1}^{\infty} (1-p)\gamma^i b_i \prod_{j=1}^{i-1} ((1-p) + (2p-1)b_j) \tag{5.1}$$

for $0 \leq s < 1$, where $s = 0.b_1 b_2 \dots b_\ell \dots_{(2)}$ is the binary representation of s. It is obvious that the series converges for any $0 \leq s < 1$.

The notation $v(s)$ will always refer to the definition above in this thesis and will not change with the context. We use the notation $f(s)$ to denote a general function or a general solution of a system, which varies according to the required properties.

Let the set of dyadic rationals be

$$G_\ell = \left\{ k2^{-\ell} \mid k \in \left\{1, \dots, 2^\ell - 1\right\}\right\} \tag{5.5}$$

such that G_ℓ is the set of numbers that can be represented by at most ℓ binary bits. The general idea to verify the Bellman

equation (AB) is to prove

$$v(s) = \max_{a \in G_\ell \cap \mathcal{A}(s)} (1-p)\gamma\ v(s+a) + p\gamma\ v(s-a) \text{ for any } s \in G_\ell$$

by induction on $\ell = 1, 2, \ldots$, and generalize this optimality to the entire interval $s \in (0, 1)$.

It then suffices to show the uniqueness of $v(s)$ that solves the system (ABX). This is proved by assuming the existence of a solution $f(s)$ and then deriving the identity $f(s) = v(s)$, conditioning on the Bellman property that $v(s)$ satisfies (AB). For presentation purposes, the uniqueness is discussed first.

As an overview, Lemma 29, 30, and 31 describe the characterizations of the system (ABX). Claim 32, Lemma 33, 35, and 36 describe the properties of $v(s)$.

Lemma 29 (Uniqueness under existence)**.** *Let $f(s) : [0, 1] \to \mathbb{R}$ be a real function. If $v(s)$ and $f(s)$ both satisfy (ABX), then $v(s) = f(s)$ for all $0 \leq s \leq 1$.*

Proof. We prove the lemma by contradiction. Assume that $f(s)$ is also a solution of the system such that $f(s)$ is not identical with $v(s)$ at some s. Define $\delta = \sup_{0<s<1} f(s) - v(s)$. As $f(2^{-1}) \geq (1-p)\gamma\ f(1) + p\gamma\ f(0) = (1-p)\gamma = v(2^{-1})$, we have $\delta \geq 0$.

We show that δ cannot be zero by contradiction. If δ is zero, as $v(s)$ and $f(s)$ are not identical, there exists an s such that $f(s) < v(s)$. In this case, let $\bar{\delta} = \sup_{0<s<1} v(s) - f(s)$. Then we choose $\bar{\epsilon} = (1-p\gamma)\bar{\delta}$ and specify s_0 such that $v(s_0) - f(s_0) > \bar{\delta} - \bar{\epsilon}$.

Let $a_0 = \min\{s_0, 1 - s_0\}$, we have

$$\begin{aligned} v(s_0) &= (1-p)\gamma\ v(s_0 - a_0) + p\gamma\ v(s_0 + a_0) \\ &\leq (1-p)\gamma\ f(s_0 - a_0) + p\gamma\ f(s_0 + a_0) + p\gamma\bar{\delta} \\ &\leq f(s_0) + p\gamma\bar{\delta}. \end{aligned}$$

The above inequality is due to the fact that at least one of $s_0 - a_0 = 0$ and $s_0 + a_0 = 1$ must hold. Thus at least one of $v(s_0 - a_0) - f(s_0 - a_0)$ and $v(s_0 + a_0) - f(s_0 + a_0)$ must be zero. The inequality contradicts $v(s_0) - f(s_0) > \bar{\delta} - \bar{\epsilon}$. Hence, δ cannot be zero. We discuss under $\delta > 0$ for the rest of the proof.

Case (I): $\gamma < 1$. In this case, we choose $\epsilon = (1 - \gamma)\delta$. By the definition of δ we specify s_0 such that $f(s_0) > v(s_0) + \delta - \epsilon$. In fact, the existence of s_0 is by the condition $\gamma < 1$. Let $a_0 \in \arg\max_{a \in \mathcal{A}(s_0)} p\gamma\ f(s_0 - a) + (1-p)\gamma\ f(s_0 + a)$. We have

$$\begin{aligned} f(s_0) &= p\gamma\ f(s_0 - a_0) + (1-p)\gamma\ f(s_0 + a_0) \\ &\leq p\gamma\ (v(s_0 - a_0) + \delta) + (1-p)\gamma\ (v(s_0 + a_0) + \delta) \\ &= p\gamma\ v(s_0 - a_0) + (1-p)\gamma\ v(s_0 + a_0) + \gamma\delta \\ &\leq v(s_0) + \delta - \epsilon. \end{aligned}$$

The inequality $f(s_0) \leq v(s_0) + \delta - \epsilon$ contradicts $f(s_0) > v(s_0) + \delta - \epsilon$. Hence, the lemma is proved for the case $\gamma < 1$.

Case (II): $\gamma = 1$. When there exists an s' such that $f(s') - v(s') = \delta$, we show the contradiction. Let $S = \{s | f(s) - v(s) = \delta, 0 < s < 1\} \neq \varnothing$. For any $s' \in S$ and $a' \in \arg\max_{a \in \mathcal{A}(s')} p\gamma\ f(s' - a) + (1-p)\gamma\ f(s' + a)$, we have

$$f(s') = p\gamma\ f(s' - a') + (1-p)\gamma\ f(s' + a')$$

$$
\begin{aligned}
&= p\ f(s'-a') + (1-p)\ f(s'+a') \\
&\overset{(\heartsuit)}{\leq} p\ (v(s'-a')+\delta) + (1-p)\ (v(s'+a')+\delta) \\
&= p\ v(s'-a') + (1-p)\ v(s'+a') + \delta \\
&\overset{(\diamondsuit)}{\leq} v(s') + \delta.
\end{aligned}
$$

Thus, the equality of ($\heartsuit$) and ($\diamondsuit$) must hold. We specify $s_0 \in S$, and by the equality of ($\heartsuit$) we have $f(s_0 - a_0) = v(s_0 - a_0) + \delta$, thus $s_0 - a_0 \in S$. Let $s_1 = s_0 - a_0$, and we recursively specify an arbitrary $a_t \in \arg\max_{a \in \mathcal{A}(s_t)} (1-p)\gamma\ f(s_t + a) + p\gamma\ f(s_t - a)$ and $s_{t+1} = s_t - a_t$, for $t = 1, 2, \ldots$, until $s_T = 0$ for some T, or indefinitely if such an s_T does not exist. If s_T exists and the sequence $\{s_t\}$ terminates at $s_T = 0$, then $f(s_T) = v(s_T) + \delta = \delta$ by ($\heartsuit$), which contradicts the boundary condition $f(s_T) = f(0) = 0$.

We next show the existence of T. When there exists t and ℓ such that $s_t \in G_\ell$, by Corollary 34 we have $s_{t+1} \in G_\ell$ and inductively $s_{t'} \in G_\ell$ for all $t' \geq t$. Consider that $\{s_t\}$ is strictly decreasing and there are finitely many elements in G_ℓ, $\{s_t\}$ cannot be infinite. Otherwise $s_t \notin G_\ell$ for any $t, \ell \geq 1$. Then by Corollary 37 the uniqueness of the optimal action we have $s_{t+1} = 2s_t - 1$ if $s_t \geq \frac{1}{2}$, and $s_{t+1} = 0$ if $s_t \leq \frac{1}{2}$. After finitely many steps of $s_{t+1} = 2s_t - 1$ we will have $s_t = 0$ for some t.

It amounts to show the existence of s' such that $f(s') - v(s') = \delta$. By Lemma 36 we have the continuity of $v(s)$. Lemma 30 indicates the monotonicity of $f(s)$ on $[0, 1)$. The upper bound

$f(s) \leq f(1)$ in (X) extends this monotonicity to the closed interval $[0, 1]$. Then by Lemma 31 we have the continuity of $f(s)$ on $(0, 1]$. By (X) this extends to $[0, 1]$. Thus we have the continuity of $f(s) - v(s)$, and consequently the existence of $\max_{0 \leq s' \leq 1} f(s') - v(s')$. As $f(0) - v(0) = f(1) - v(1) = 0$ and $\delta > 0$, this maximum must be attained at some $s' \in (0, 1)$. Therefore we have the existence of $\max_{0 < s' < 1} f(s') - v(s')$, which concludes the lemma. □

Lemma 30 (Monotonicity)**.** *Let $\gamma = 1$ and $p > 0.5$. If a real function $f(s)$ satisfies (AB) then $f(s)$ is monotonically increasing on $[0, 1)$.*

Proof. We prove the claim by contradiction. Assume that there exists $s_1 < s_2$ where $f(s_1) > f(s_2)$. Denote $\Delta s = s_2 - s_1 > 0$ and $\Delta f = f(s_1) - f(s_2) > 0$. By induction we have

$$f(s_2 - 2^{-\ell}\Delta s) - f(s_2) \geq p^{\ell}\Delta f$$

for an arbitrary integer $\ell \geq 1$. Then when $s_2 + 2^{-\ell}\Delta s < 1$, by $f(s_2) \geq pf(s_2 - 2^{-\ell}\Delta s) + (1 - p)f(s_2 + 2^{-\ell}\Delta s)$,

$$\begin{aligned} f(s_2 + 2^{-\ell}\Delta s) &\leq \frac{1}{1-p} f(s_2) - \frac{p}{1-p} f(s_2 - 2^{-\ell}\Delta s) \\ &= f(s_2) + \frac{p}{1-p}(f(s_2) - f(s_2 - 2^{-\ell}\Delta s)) \\ &\leq f(s_2) + f(s_2) - f(s_2 - 2^{-\ell}\Delta s). \end{aligned}$$

This concludes $f(s_2 + 2^{-\ell}\Delta s) - f(s_2) \leq f(s_2) - f(s_2 - 2^{-\ell}\Delta s)$. By induction we have

$$f(s_2 + k2^{-\ell}\Delta s) - f(s_2 + (k-1)2^{-\ell}\Delta s)$$

$$\leq f(s_2 + (k-1)2^{-\ell}\Delta s) - f(s_2 - (k-2)2^{-\ell}\Delta s)$$

for $k = 1, 2, \ldots,$ when $s_2 + k2^{-\ell}\Delta s < 1$. We sum this inequality over k and get

$$\begin{aligned} f(s_2 + k2^{-\ell}\Delta s) - f(s_2) &\leq k(f(s_2) - f(s_2 - 2^{-\ell}\Delta s)) \\ &\leq -kp^{\ell}\Delta f. \end{aligned}$$

By letting $k = 2^n$, $\ell = n + n_0$, $s_2 + 2^{-n_0}\Delta s < 1$, and $n \to +\infty$, we have $s_2 + k2^{-\ell}\Delta s < 1$ and $-kp^{\ell}\Delta f \to -\infty$. The arbitrarity of n indicates the non-existence of $f(s_2 + k2^{-\ell}\Delta s)$, which contradicts the existence of the solution $f(\cdot)$. □

Lemma 31 (Continuity)**.** *Let $\gamma = 1$ and $p \geq 0.5$. If a real function $f(s)$ is monotonically increasing on $(0, 1]$ and it satisfies (AB), then $f(s)$ is continuous on $(0, 1]$.*

Proof. We show the continuity by contradiction. Suppose that there exists a point $s' \in (0, 1)$ such that $f(s)$ is discontinuous at s', then there exists $\epsilon, \delta > 0$ where $f(s' + \epsilon_1) - f(s' - \epsilon_2) \geq \delta$ for any $\epsilon_1 + \epsilon_2 = \epsilon$. Then, by

$$f(s' - \frac{1}{4}\epsilon) \geq p\ f(s' - \epsilon) + (1 - p)\ f(s' + \frac{2}{4}\epsilon),$$

we have

$$f(s' - \frac{1}{4}\epsilon) - f(s' - \epsilon) \geq (1 - p)\delta/p.$$

Similarly, for $k = 1, 2, \ldots,$

$$f(s' - \frac{1}{4^k}\epsilon) - f(s' - \frac{1}{4^{k-1}}\epsilon) \geq (1 - p)\delta/p.$$

Let $k > ((1-p)\delta/p)^{-1}$, we have $f(s' - \frac{1}{4^k}\epsilon) - f(s' - \epsilon) \geq 1$. This contradicts with the fact that $f(s)$ is bounded between 0 and 1. The continuity follows on $(0, 1)$.

If the function is discontinuous on $s = 1$, then there exists $\epsilon, \delta > 0$ where $f(1) - f(1 - \epsilon_1) \geq \delta$ for any $\epsilon_1 \leq \epsilon$. The same argument holds by observing

$$f(1 - \frac{1}{2^{k-1}}\epsilon) \geq p\ f(1 - \frac{1}{2^k}\epsilon) \geq f(1).$$

The lemma follows. □

When $f(s)$ is only required to be monotonically increasing on $(0, 1)$, the continuity still holds but only on $(0, 1)$.

For simplicity define

$$Q_v(s, a) = p\gamma\ v(s - a) + (1 - p)\gamma\ v(s + a). \tag{5.6}$$

As $v(s)$ is the optimal state-value function (to be proved later in Theorem 39), $Q_v(s, a)$ is in fact the optimal action-value function [2, 3].

Recall that G_ℓ is the set of dyadic rationals $\{k2^{-\ell} \mid k \in \{1, \ldots, 2^\ell - 1\}\}$.

Claim 32. *For any* $s = 0.b_1 b_2 \ldots b_{\ell(2)} \in G_\ell \cup \{0\}$,

$$\begin{aligned} v\left(s + 2^{-(\ell+1)}\right) - v(s) =& (1-p)\gamma^{\ell+1} \prod_{j=1}^{\ell} \left((1-p) + (2p-1)b_j\right) \\ \leq& (1-p)p^\ell \gamma^{\ell+1}. \end{aligned} \tag{5.7}$$

For any $s = 0.b_1 b_2 \dots b_{k(2)} \in G_\ell$ *with* $b_k = 1$ *and* $1 \le k \le \ell$,

$$\begin{aligned} & v(s) - v\left(s - 2^{-(\ell+1)}\right) \\ \ge & p^{\ell-k+1}(1-p)\gamma^{\ell+1} \prod_{j=1}^{k-1} \left((1-p) + (2p-1)b_j\right). \end{aligned} \tag{5.8}$$

Also,

$$v(1) - v\left(1 - 2^{-(\ell+1)}\right) \ge p^{\ell+1}\gamma^{\ell+1}. \tag{5.9}$$

The equality of (5.8) *and* (5.9) *holds if and only if* $\gamma = 1$.

Proof. Inequality (5.7) and (5.9) are obtained by the definition of $v(s)$. To derive inequality (5.8), denote $k = \max\{1 \le i \le \ell : b_i = 0\}$ and then

$$\begin{aligned} & v(s) - v\left(s - 2^{-(\ell+1)}\right) \\ = & (1-p)\gamma^k \prod_{j=1}^{k-1}((1-p) + (2p-1)b_j) \\ & - \sum_{i=k+1}^{\ell+1} (1-p)\gamma^i \cdot 1 \cdot \prod_{j=1}^{k-1}((1-p) + (2p-1)b_j) \cdot (1-p) \cdot \prod_{j=k+1}^{i-1} p \\ = & (1-p)\gamma^k \prod_{j=1}^{k-1}((1-p) + (2p-1)b_j) \\ & \cdot \left(1 - (1-p)\sum_{i=k+1}^{\ell+1} \gamma^{i-k} p^{i-k-1}\right) \\ = & (1-p)\gamma^k \prod_{j=1}^{k-1}((1-p) + (2p-1)b_j) \\ & \cdot \left(1 - (1-p)\gamma\frac{1-(\gamma p)^{\ell-k+1}}{1-\gamma p}\right) \end{aligned}$$

$$\geq (1-p)\gamma^k \prod_{j=1}^{k-1}((1-p)+(2p-1)b_j)\cdot\left(1-\left(1-(\gamma p)^{\ell-k+1}\right)\right)$$

$$= (1-p)p^{\ell-k+1}\gamma^{\ell+1}\prod_{j=1}^{k-1}((1-p)+(2p-1)b_j).$$

The arguments are due to the fact that $s-2^{-(\ell+1)} = 0.b_1b_2\ldots b_{k-1}$ $0_k 1_{k+1}\ldots 1_{\ell+1(2)}$ and the inequality is by $(1-p)\gamma \leq 1-\gamma p$. □

Lemma 33. *Let $\ell \geq 1, 0 < \gamma \leq 1, 0.5 \leq p < 1$. For any $s \in G_\ell$,*

$$\max_{a\in(G_{\ell+1}\setminus G_\ell)\cap\mathcal{A}(s)} Q_v(s,a) \leq \max_{a\in G_\ell\cap\mathcal{A}(s)} Q_v(s,a).$$

Proof. **Case (I):** First we prove that for $\ell > 1$, any $s \in G_\ell$, $a \in G_\ell \cap \mathcal{A}(s)$, $a > 2^{-\ell}$, and $s + a < 1$,

$$Q_v\left(s, a-2^{-(\ell+1)}\right) \leq \max\left\{Q_v(s,a), Q_v\left(s,a-2^{-\ell}\right)\right\}.$$

Note that in this case, $a - 2^{-(\ell+1)} \in G_{\ell+1}\cap\mathcal{A}(s)$ and $a-2^{-\ell} \in G_\ell\cap\mathcal{A}(s)$.

Let $s - a = 0.c_1c_2\ldots c_{\ell(2)} = 0.c_1c_2\ldots 0_k 1_{k+1}\ldots 1_{\ell(2)}$, where $k = \max\{1 \leq i \leq \ell : c_i = 0\}$ is the index of the last 0 bit in $s-a$. Such k must exist since $0 \leq s-a \leq 1-3\times 2^{-\ell} < 1$. Similarly, let $s + a = 0.d_1d_2\ldots d_{\ell(2)} = 0.d_1d_2\ldots 1_{k'(2)}$ where $k' = \max\{1 \leq i \leq \ell : d_i = 1\}$ is the index of the last 1 bit in $s+a$. Such k' must exist since $3\times 2^{-\ell} \leq s+a < 1$. Also, $s + a - 2^{-(\ell+1)} = 0.d_1d_2\ldots 0_{k'}1_{k'+1}\ldots 1_{\ell+1(2)}$.

To prove $Q_v\left(s, a-2^{-(\ell+1)}\right) \leq Q_v(s,a)$, it is equivalent to proving

$$v(s+a) - v\left(s+a-2^{-(\ell+1)}\right)$$

$$\geq \frac{p}{1-p}\left(v\left(s-a+2^{-(\ell+1)}\right)-v(s-a)\right).$$

Then by applying inequality (5.8), (5.9) and inequality (5.7) in Claim 32 on the LHS and RHS respectively, it suffices to prove

$$\begin{aligned} & p^{\ell-k'}(1-p)\prod_{j=1}^{k'-1}((1-p)+(2p-1)d_j) \\ \geq & \prod_{j=1}^{\ell}((1-p)+(2p-1)c_j) \\ = & \, p^{\ell-k}(1-p)\prod_{j=1}^{k-1}((1-p)+(2p-1)c_j). \end{aligned}$$

Let $M_c = c_1+\cdots+c_{k-1}, M_d = d_1+\cdots+d_{k'-1}$ be the number of 1s in $\{c_1,\ldots,c_k\}, \{d_1,\ldots,d_{k'}\}$ respectively. Then $Q_v\left(s, a-2^{-(\ell+1)}\right) \leq Q_v(s,a)$ holds when $p=0.5$ or $p>0.5, M_c+k \geq M_b+k'$.

To prove $Q_v\left(s, a-2^{-(\ell+1)}\right) \leq Q_v\left(s, a-2^{-\ell}\right)$, it is equivalent to proving

$$\begin{aligned} & v\left(s-a+2^{-\ell}\right)-v\left(s-a+2^{-(\ell+1)}\right) \\ \geq & \frac{1-p}{p}\left(v\left(s+a-2^{-(\ell+1)}\right)-v\left(s+a-2^{-\ell}\right)\right). \end{aligned}$$

Note that $s-a+2^{-\ell} = 0.c_1c_2\ldots1_{k(2)}$ and $s+a-2^{-\ell} = 0.d_1d_2\ldots0_{k'}1_{k'+1}\ldots1_{\ell(2)}$. Then by inequality (5.8), (5.9) and inequality (5.7) on the LHS and RHS respectively, it suffices to prove

$$p^{\ell-k+2}\prod_{j=1}^{k-1}((1-p)+(2p-1)c_j)$$

$$\geq (1-p)^2 p^{\ell-k'} \prod_{j=1}^{k'-1} ((1-p) + (2p-1)d_j).$$

Then $Q_v\left(s, a - 2^{-(\ell+1)}\right) \leq Q_v\left(s, a - 2^{-\ell}\right)$ holds when $p = 0.5$ or $p > 0.5, M_c + k' + 2 \geq M_d + k$.

As at least one of $M_c + k' + 1 \geq M_d + k$ and $M_d + k \geq M_c + k' + 1$ holds, thus $Q_v\left(s, a - 2^{-(\ell+1)}\right) < \max\left\{Q_v(s,a), Q_v\left(s, a - 2^{-\ell}\right)\right\}$.

We cover two corner cases for the completeness of the proof.

Case (II): Next we prove for $\ell \geq 1$, any $s \in G_\ell$, $a \in G_\ell \cap \mathcal{A}(s)$ and $s + a = 1$,

$$Q_v\left(s, a - 2^{-(\ell+1)}\right) \leq Q_v(s,a).$$

Similar to above, it is equivalent prove

$$v(1) - v\left(1 - 2^{-(\ell+1)}\right) \geq \frac{p}{1-p}\left(v\left(s - a + 2^{-(\ell+1)}\right) - v(s-a)\right).$$

Note that by Claim 32,

$$\begin{aligned} v(1) - v\left(1 - 2^{-(\ell+1)}\right) &\geq p^{\ell+1}\gamma^{\ell+1} = \frac{p}{1-p} \cdot (1-p)p^{\ell}\gamma^{\ell+1} \\ &\geq \frac{p}{1-p}\left(v\left(s - a + 2^{-(\ell+1)}\right) - v(s-a)\right), \end{aligned}$$

which concludes the proof.

Case (III): Last we prove for $\ell > 1$, any $s \in G_\ell$ and $a = 2^{-\ell}, s < 1 - 2^{-\ell}$.

When $s = 0.b_1 b_2 \ldots 0_m 1_{m+1} \ldots 1_{\ell(2)}$ with $1 \leq m < \ell$, to prove $Q_v\left(s, 2^{-(\ell+1)}\right) \leq Q_v\left(s, 2^{-\ell}\right)$, it is equivalent to proving

$$v\left(s + 2^{-\ell}\right) - v\left(s + 2^{-(\ell+1)}\right)$$

$$\geq \frac{p}{1-p}\left(v\left(s-a+2^{-(\ell+1)}\right)-v(s-a)\right).$$

In this case, $s+2^{-\ell} = 0.b_1b_2\ldots 1_{m(2)}, s-2^{-\ell} = 0.b_1b_2\ldots 0_m 1_{m+1}\ldots 1_{\ell-1(2)}$ and $M_c = M_d, k = k' = m$, thus $M_d+k \geq M_c+k'$, which concludes the proof similar to the first part of Case (I).

When $s = 0.b_1b_2\ldots 1_{m'}0_{m'+1}\ldots 0_{\ell(2)}$ with $1 \leq m' < \ell$, let $M_b = b_1 + \ldots b_{m'}$. Then,

$$\begin{aligned}
&Q_v\left(s,2^{-m'}\right)-Q_v\left(s,2^{-(\ell+1)}\right)\\
&=(1-p)\gamma(v(s+2^{-m'})-v(s-2^{-m'}))\\
&\quad-(1-p)\gamma(v(s)-v(s-2^{-m'}))\\
&\quad-(1-p)\gamma(v(s+2^{-\ell})-v(s))-p\gamma(v(s-2^{-\ell})-v(s-2^{-m'}))\\
&\geq(1-p)\gamma(p\gamma)^{M_b}((1-p)\gamma)^{m'-2-M_b}(1-p)\gamma\\
&\quad-(1-p)\gamma(p\gamma)^{M_b}((1-p)\gamma)^{m'-1-M_b}(1-p)\gamma\\
&\quad-(1-p)\gamma(p\gamma)^{M_b+1}((1-p)\gamma)^{\ell-2-M_b}(1-p)\gamma\\
&\quad-p\gamma(p\gamma)^{M_b}((1-p)\gamma)^{m'-M_b}(1+(p\gamma)+\ldots(p\gamma)^{\ell-m'-1})(1-p)\gamma\\
&=p^{M_b}(1-p)^{m'-M_b}\gamma^{m'}-p^{M_b}(1-p)^{m'+1-M_b}\gamma^{m'+1}\\
&\quad-p^{M_b+1}(1-p)^{\ell-M_b}\gamma^{\ell+1}\\
&\quad-p^{M_b+1}(1-p)^{m'+1-M_b}\gamma^{m'+2}(1-(p\gamma)^{\ell-m'})/(1-p\gamma)\\
&\geq p^{M_b}(1-p)^{m'-M_b}\gamma^{m'}-p^{M_b}(1-p)^{m'+1-M_b}\gamma^{m'+1}\\
&\quad-p^{M_b+1}(1-p)^{\ell-M_b}\gamma^{\ell+1}\\
&\quad-p^{M_b+1}(1-p)^{m'-M_b}\gamma^{m'+1}(1-(p\gamma)^{\ell-m'})\\
&\geq\ -p^{M_b+1}(1-p)^{\ell-M_b}\gamma^{\ell+1}-p^{M_b+1}(1-p)^{m'-M_b}\gamma^{m'+1}(-(p\gamma)^{\ell-m'})\\
&\geq 0. \qquad \square
\end{aligned}$$

The arguments in the proof that either $M_c+k \geq M_d+k'+1$

or $M_d + k' \geq M_c + k$ must hold is tight for integers M_c and M_d. This is the case for $a \in G_{\ell+1} \setminus G_\ell$. When $a \notin G_{\ell+1}$, this sufficient condition becomes even looser. The lemma imposes G_ℓ to be the only set of possible optimal actions, given $s \in G_\ell$.

Corollary 34. *Let $\ell \geq 1$. For any $s \in G_\ell$,*

$$\arg\max_{a \in \mathcal{A}(s)} Q_v(s,a) \subseteq G_\ell.$$

Now we verify the Bellman property on $\bigcup_{\ell \geq 1} G_\ell$.

Lemma 35. *Let $\ell \geq 1$. For any $s \in G_{\ell+1}$,*

$$\min\{s, 1-s\} \in \arg\max_{a \in G_{\ell+1} \cap \mathcal{A}(s)} Q_v(s,a).$$

Proof. We prove the lemma by induction on ℓ. When $\ell = 1$, it is obvious since G_1 has only one element. The base case $\ell = 2$ is also immediate by exhausting $a \in \{2^{-1}, 2^{-2}\}$ for $s = 2^{-1}$. Now we assume that for any $s \in G_\ell$, $\min\{s, 1-s\} \in \arg\max_{a \in G_\ell \cap \mathcal{A}(s)} Q_v(s,a)$. We aim to prove this lemma for $\ell + 1$.

For $s \in G_\ell$, by Lemma 33, $\arg\max_{a \in G_{\ell+1} \cap \mathcal{A}(s)} Q_v(s,a) \subseteq G_\ell$. Then by the induction assumption, $\min\{s, 1-s\} \in \arg\max_{a \in G_\ell \cap \mathcal{A}(s)} Q_v(s,a) \subseteq \arg\max_{a \in G_{\ell+1} \cap \mathcal{A}(s)} Q_v(s,a)$. Hence, the lemma holds for $s \in G_\ell$. We discuss under $s \in G_{\ell+1} \setminus G_\ell$ for the rest of the proof.

We start with two inductive properties of $v(s)$ to reduce the problem from $s \in G_{\ell+1}$ to $s' \in G_\ell$, where s' is either $2s$ or $2s-1$. For any $s \geq 2^{-1}$, that is, $s = 0.c_1 c_2 \ldots c_{\ell+1(2)} \in G_{\ell+1}$ with $c_1 = 1$,

$$v(s) = \sum_{i=1}^{\ell} (1-p)\gamma^i c_i \prod_{j=1}^{i-1} ((1-p) + (2p-1)c_j)$$

$$
\begin{aligned}
&= (1-p)\gamma + \sum_{i=2}^{\ell}(1-p)\gamma^i c_i \prod_{j=1}^{i-1}((1-p)+(2p-1)c_j) \\
&= (1-p)\gamma + \sum_{i=1}^{\ell-1}(1-p)\gamma^{i+1} c_{i+1}((1-p)+(2p-1)c_1) \\
&\quad \cdot \prod_{j=1}^{i-1}((1-p)+(2p-1)c_{j+1}) \\
&= (1-p)\gamma + p\gamma\ v(0.c_2 \ldots c_{\ell+1_{(2)}}) \\
&= (1-p)\gamma + p\gamma\ v(2s-1).
\end{aligned}
$$

Similarly, for any $s < 2^{-1}$, that is, $s = 0.c_1c_2 \ldots c_{\ell+1_{(2)}} \in G_{\ell+1}$ with $c_1 = 0$,

$$
\begin{aligned}
v(s) &= \sum_{i=1}^{\ell-1}(1-p)^2\gamma^{i+1} c_{i+1} \prod_{j=1}^{i-1}((1-p)+(2p-1)c_{j+1}) \\
&= (1-p)\gamma v(2s).
\end{aligned}
$$

Armed with the properties, we split the discussion into four cases $2^{-1}+2^{-2} \le s < 1$, $2^{-1} \le s < 2^{-1}+2^{-2}$, $2^{-1}-2^{-2} < s < 2^{-1}$, and $0 < s \le 2^{-1} - 2^{-2}$.

When $s \ge 2^{-1} + 2^{-2}$, As $a \le 1 - s$, we have $s - a \ge 2^{-1}$ and $s + a \ge 2^{-1}$. Hence, the first bit after the decimal of $s - a$ and $s + a$ is 1. Hence,

$$
\begin{aligned}
Q_v(s,a) &= p\gamma\ v(s-a) + (1-p)\gamma\ v(s+a) \\
&= (1-p)\gamma^2 + p\gamma\ (p\gamma\ v(2s-2a-1)) \\
&\quad + (1-p)\gamma\ v(2s+2a-1) \\
&= (1-p)\gamma^2 + p\gamma\ (p\gamma\ v((2s-1)-2a)
\end{aligned}
$$

$$\begin{aligned} &+ (1-p)\gamma\ v((2s-1)+2a)) \\ &= (1-p)\gamma^2 + p\gamma\ Q_v(2s-1, 2a). \end{aligned}$$

As $2s-1 \in G_\ell$ and $2a \in G_\ell$, by the induction assumption the maximum of $Q_v(2s-1, 2a)$ is obtained at $a = 1-s$. Hence, $1-s \in \arg\max_{a \in G_{\ell+1} \cap \mathcal{A}(s)} Q_v(s,a)$ as desired.

When $2^{-1} \leq s < 2^{-1} + 2^{-2}$, if $s - a \geq 2^{-1}$, then the first bit after the decimal of $s-a$ and $s+a$ is 1 and the lemma follows the same arguments as the above case. Otherwise, if $s - a < 2^{-1}$, we have

$$\begin{aligned} Q_v(s,a) &= p\gamma\ v(s-a) + (1-p)\gamma\ v(s+a) \\ &= (1-p)^2\gamma^2 + p(1-p)\gamma^2\ v(2s-2a) \\ &\quad + p(1-p)\gamma^2\ v(2s+2a-1) \\ &= (1-p)\gamma\ (p\gamma\ v((2s-2^{-1}) - (2a-2^{-1})) \\ &\quad + (1-p)\gamma\ v((2s-2^{-1}) + (2a-2^{-1}))) \\ &\quad + (1-p)(2p-1)\gamma^2\ v(2s+2a-1) + (1-p)^2\gamma^2 \\ &= (1-p)\gamma\ Q_v(2s-2^{-1}, 2a-2^{-1}) \\ &\quad + (1-p)(2p-1)\gamma^2\ v(2s+2a-1) + (1-p)^2\gamma^2. \end{aligned}$$

As $2s-2^{-1} \in G_\ell$ and $2a-2^{-1} \in G_\ell$ whenever $l \geq 2$, by the induction assumption $Q_v(2s-2^{-1}, 2a-2^{-1})$ obtains its maximum at $a = 1-s$. By Claim 32, $v(s)$ is monotonically increasing on G_ℓ for any $\ell \geq 2$. Hence, $v(2s+2a-1)$ obtains the maximum at the maximum feasible a, which is $a = 1-s$. Since both terms take their respective maximum at $a = 1-s$, we conclude that $1-s \in \arg\max_{a \in G_{\ell+1} \cap \mathcal{A}(s)} Q_v(s,a)$ as desired.

The other two cases, $2^{-1} - 2^{-2} < s < 2^{-1}$ and $0 < s \leq 2^{-1} - 2^{-2}$, follow similar arguments. The lemma follows. □

Lemma 36. *Both $v(s)$ and $v'(s) = \max_{a \in \mathcal{A}(s)} Q_v(s, a)$ are continuous at s if there does not exist an ℓ such that $s \in G_\ell$.*

Proof. We first prove the continuity of $v(s)$. For $s = b_1 b_2 \ldots b_\ell \ldots_{(2)}$, $s \notin G_\ell$ indicates that for any integer N there exists $n_1 \geq N$ such that $b_{n_1} = 1$ and $n_0 \geq N$ such that $b_{n_0} = 0$. The monotonicity of $v(s)$ is obvious from Equation (5.1) that flipping a 0 bit to a 1 bit will always yield a greater value. For any $s - 2^{-N} \leq s' \leq s + 2^{-N}$, we specify n_1 and n_0 such that $s - 2^{-n_1} \leq s' \leq s + 2^{-n_0}$. By the monotonicity of $v(s)$ we have

$$
\begin{aligned}
& v(s) - v(s') \leq v(s) - v(s - 2^{-n_1}) \\
= & (1-p)\gamma^{n_1} \prod_{j=1}^{n_1-1} ((1-p) + (2p-1)b_j) \cdot (1 + \sum_{i=n_1+1}^{\infty} \gamma^{i-n_1} b_i p \\
& \prod_{j=n_1+1}^{i-1} ((1-p) + (2p-1)b_j)) \\
& - (1-p)\gamma^{n_1} \prod_{j=1}^{n_1-1} ((1-p) + (2p-1)b_j) \sum_{i=n_1+1}^{\infty} \gamma^{i-n_1} b_i (1-p) \\
& \prod_{j=n_1+1}^{i-1} ((1-p) + (2p-1)b_j) \\
= & (1-p)\gamma^{n_1} \prod_{j=1}^{n_1-1} ((1-p) + (2p-1)b_j) \cdot (1 \\
& + \sum_{i=n_1+1}^{\infty} \gamma^{i-n_1} b_i (2p-1) \prod_{j=n_1+1}^{i-1} ((1-p) + (2p-1)b_j))
\end{aligned}
$$

$$\leq (1-p)\gamma^{n_1}p^{n_1-1}\cdot(1+\sum_{i=n_1+1}^{\infty}\gamma^{i-n_1}(2p-1)p^{n_1-i-1})$$
$$\leq 2(1-p)\gamma^N p^{N-1}.$$

And similarly,

$$\begin{aligned} v(s)-v(s') &\geq v(s)-v(s+2^{-n_0}) \\ &\geq -(1-p)\gamma^{n_0}p^{n_0-1}\cdot(1+\sum_{i=n_0+1}^{\infty}\gamma^{i-n_0}(2p-1)p^{n_0-i-1}) \\ &\geq -2(1-p)\gamma^N p^{N-1}. \end{aligned}$$

Hence, $|v(s)-v(s')|$ is bounded by $2(1-p)\gamma^N p^{N-1}$ for $s-2^{-N}\leq s'\leq s+2^{-N}$. As $2(1-p)\gamma^N p^{N-1}$ converges to zero when N approaches infinity, $v(s)$ is continuous as desired.

We then show the continuity of $v'(s)=\max_{a\in\mathcal{A}(s)}Q_v(s,a)$. We first argue that $v'(s)$ is monotonically increasing. In fact, for $s'\geq s$ and $0<a\leq\min\{s,1-s\}$, either $0<a\leq\min\{s',1-s'\}$ or $0<a+s-s'\leq\min\{s',1-s'\}$ must be satisfied. Therefore $a\in\mathcal{A}(s)$ indicates at least one of $a\in\mathcal{A}(s')$ and $a+s-s'\in\mathcal{A}(s')$.

Let a' be a or $a+s-s'$ whoever is in $\mathcal{A}(s')$, we have both $s'+a'\geq s+a$ and $s'-a'\geq s-a$. Specifying a such that $v'(s)=Q_v(s,a)$, we have

$$v'(s')\geq Q_v(s',a')\geq v'(s).$$

The monotonicity follows.

Let $s=b_1b_2\ldots b_\ell\ldots_{(2)}$. Similarly, for any N, specify $n_1\geq N$ such that $b_{n_1}=1$ and $n_0\geq N+2$ such that $b_{n_0}=0$. Also

let $s_0 = b_1 b_2 \dots b_{N(2)}$. Then for the neighbourhood set $s_0 - 2^{-(N+1)} \leq s' \leq s_0 + 2^{-(N+1)}$, $v'(s) = v(s)$ for both the ends of the interval $s_0 - 2^{-(N+1)}, s_0 + 2^{-(N+1)} \in G_{N+1}$. $|v'(s) - v'(s')|$ is then bounded by $|v(s_0 - 2^{-(N+1)}) - v(s_0 + 2^{-(N+1)})|$. According to Claim 32, this value converges to zero when N approaches infinity. The continuity of $v'(s)$ follows. □

The continuity of $v(s)$ extends to the dyadic rationals $\bigcup_{\ell \geq 1} G_\ell$ when $\gamma = 1$, which means that $v(s)$ is *continuous everywhere* on $[0, 1]$ under $\gamma = 1$. It is worth noting that similar to the Cantor function, $v(s)$ is *not absolutely continuous*. In fact, $v(s)$ shares more common properties with the Cantor function, as they both have *a derivative of zero almost everywhere*, both their values go from 0 to 1, and their range is every value in between 0 and 1.

The continuity of $v'(s) = \max_{a \in \mathcal{A}(s)} Q_v(s, a)$ indicates that the optimal action is uniquely $\min\{s, 1 - s\}$ on $s \notin G_\ell$. This optimal action agrees with the optimal action we specified on $s \in G_\ell$ in Lemma 35, which makes $\pi(s) = \min\{s, 1 - s\}$ an optimal policy for every state (condition on that $v(s)$ is the optimal value function, which will be proved later).

Corollary 37. *If $s \notin G_\ell$ for any $\ell \geq 1$,*

$$\arg\max_{a \in \mathcal{A}(s)} \; Q_v(s, a) = \{\min\{s, 1 - s\}\}.$$

Lemma 38. *$v(s)$ is the unique solution of the system (ABX).*

Proof. Let $v'(s) = \max_{a \in \mathcal{A}(s)} Q_v(s, a)$. As per Lemma 35 we have $v(s) = v'(s)$ on the dyadic rationals $\bigcup_{\ell \geq 1} G_\ell$. Since $\bigcup_{\ell \geq 1} G_\ell$ is dense and compact on $(0, 1)$, $v(s) = v'(s)$ holds whenever both $v(s)$ and $v'(s)$ are continuous at s. By Lemma 36 $v(s)$ and $v'(s)$ are continuous for any s if there does not exist an $\ell \geq 1$ such that $s \in G_\ell$, which then indicates $v(s) = v'(s)$ on the complement of $\bigcup_{\ell \geq 1} G_\ell$. Therefore $v(s) = v'(s)$ is satisfied on $(0, 1)$, which verifies the Bellman property (AB). The boundary conditions (X) holds obviously. Finally as per Lemma 29, $v(s)$ is the unique solution to the system of Bellman equation and the boundary conditions. □

Theorem 39. *Let* $0 \leq \gamma \leq 1$ *and* $p > 0.5$. *Under the continuous setting of the Gambler's problem, the optimal state-value function is* $v(1) = 1$ *and* $v(s)$ *defined in Equation* (5.1) *for* $0 \leq s < 1$.

Proof. As the optimal state-value function must satisfy the system (ABX) and $v(s)$ is the unique solution to the system, $v(s)$ is the optimal state-value function. □

Corollary 40. *The policy* $\pi(s) = \min\{s, 1 - s\}$ *is (Blackwell) optimal.*

It is worth noting that when $\gamma = 1$ and $s \in G_\ell \setminus G_{\ell-1}$ for some ℓ, then $\pi'(s) = 2^{-\ell}$ is also an optimal policy at s.

Theorem 39 and the proof of Lemma 35 also induce the following statement that the optimal value function $v(s)$ is fractal and self-similar. The derivation of the corollary is in the introduction.

Corollary 41. *The curve of the value function $v(s)$ on the interval $[k2^{-\ell}, (k+1)2^{-\ell}]$ is similar (in geometry) to the curve of $v(s)$ itself on $[0,1]$, for any integer $\ell \geq 1$ and $0 \leq k \leq 2^{\ell} - 1$.*

Some other notable facts about $v(s)$ are as below:

Fact 42. *The expectation*

$$\int_0^1 v(s)ds = (1-p)\gamma = v(\frac{1}{2}).$$

Fact 43. *The derivative*

$$\begin{aligned} \lim_{\Delta s \to 0^+} \frac{v(s+\Delta s)}{\Delta s} =& 0, \\ \lim_{\Delta s \to 0^-} \frac{v(s+\Delta s)}{\Delta s} =& \begin{cases} +\infty, & \text{if } s = 0 \text{ or } s \in \bigcup_{\ell \geq 1} G_\ell, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{5.10}$$

Fact 44. *The length of the arc $y = v(s)$, $0 \leq s \leq 1$ equals 2.*

In fact, any singular function (zero derivative almost everywhere) has an arc length of 2 if it goes from $(0,0)$ to $(1,1)$ monotonically [232]. This can be intuitively understood as that the curve either goes horizontal, when the derivative is zero, or vertical, when the derivative is infinity. Therefore the arc length is the Manhattan distance between $(0,0)$ and $(1,1)$, which equals 2.

Fact 45.

$$\arg\min_{0 \leq s \leq 1} v(s) - s = \{\frac{2}{3}\}.$$

It is natural that by the fractal characterization of $v(s)$ the approximation must be inexact. The following two propositions give quantitative lower bounds on such approximation errors.

Proposition 46. *When $N \in \mathbb{N}^+$, $N \geq 4$ is a power of 2, let $\bar{v}_1(s)$ be piecewise constant on any of the intervals $s \in (k/N, (k+1)/N)$, $k = 0, \ldots, N-1$, then*

$$\int_s |v(s) - \bar{v}_1(s)|\, ds \geq \frac{1}{N} \frac{(2-\gamma)(1-p)\gamma}{1-p\gamma} + o(\frac{1}{N}).$$

Proof. When N is a power of 2, for $k \in \{0, \ldots, N-1\}$, the curve of $v(s)$ on each of the intervals $(k/N, (k+1)/N)$ is self-similar to $v(s)$ itself on $(0, 1)$. We consider this segment of the curve. By Equation (5.3),

$$v(s) = v(\bar{s}) + \gamma^\ell \prod_{j=1}^{\ell} ((1-p) + (2p-1)b_j) \cdot v(2^\ell(s - \bar{s})),$$

where $\ell = \log_2 N$, $\bar{s} = k/N$ and $s = 0.b_1 b_2 \ldots b_\ell \ldots_{(2)} \in (\bar{s}, \bar{s} + \frac{1}{N})$.

Let $\Delta s = 1/N$, $\Delta y = v((k+1)/N) - v(k/N)$, we have for $0 < s < \Delta s$

$$v(\bar{s} + s) = v(\bar{s}) + v(s/\Delta s)\Delta y.$$

As $v(s)$ is monotonically increasing on every interval $(k/N, (k+1)/N)$, the minimum over $\bar{y}$

$$\int_{s=\bar{s}}^{\bar{s}+\Delta s} |v(s) - \bar{y}|\, ds$$

is obtained when $\bar{y}_{\bar{s}} = v(\bar{s} + \frac{1}{2}\Delta s)$ (intuitively, the median of $v(s)$ on the interval). This results in an approximation error of

$$\min_{\bar{y}} \int_{s=\bar{s}}^{\bar{s}+\Delta s} |v(s) - \bar{y}|\, ds$$

$$
\begin{aligned}
&= \int_{s=\bar{s}}^{\bar{s}+\frac{1}{2}\Delta s} v(\bar{s}+\frac{1}{2}\Delta s) - v(s)\ ds \\
&\quad + \int_{s=\bar{s}+\frac{1}{2}\Delta s}^{\bar{s}+\Delta s} v(s) - v(\bar{s}+\frac{1}{2}\Delta s)\ ds \\
&= -\int_{s=\bar{s}}^{\bar{s}+\frac{1}{2}\Delta s} v(s)\ ds + \int_{s=\bar{s}+\frac{1}{2}\Delta s}^{\bar{s}+\Delta s} v(s)\ ds \\
&= -\frac{1}{2}\Delta s \int_{s=0}^{1} v(\bar{s}) + (v(\bar{s}+\frac{1}{2}\Delta s) - v(\bar{s}))v(s)\ ds \\
&\quad + \frac{1}{2}\Delta s \int_{s=0}^{1} v(\bar{s}+\frac{1}{2}\Delta s) + (v(\bar{s}+\Delta s) - v(\bar{s}+\frac{1}{2}\Delta s))v(s)\ ds \\
&= \frac{1}{2}\Delta s((1-(1-p)\gamma)(v(\bar{s}+\frac{1}{2}\Delta s) - v(\bar{s})) \\
&\quad + (1-p)\gamma(v(\bar{s}+\Delta s) - v(\bar{s}+\frac{1}{2}\Delta s))).
\end{aligned}
$$

This error is then summed over $\bar{s} = 0, 1/N, \ldots, (N-1)/N$ such that

$$
\begin{aligned}
&\sum_{\bar{s}=0/N}^{(N-1)/N} \int_{s=\bar{s}}^{\bar{s}+\Delta s} |v(s) - \bar{y}_{\bar{s}}|\, ds \\
&\geq \frac{1}{2N}((1-(1-p)\gamma)v(\frac{N-\frac{1}{2}}{N}) + (1-p)\gamma(1-v(\frac{1}{2N}))) \\
&= \frac{1}{2N}((1-(1-p)\gamma)\frac{(1-p)\gamma}{1-p\gamma}(1-(p\gamma)^{\log_2 N+1}) \\
&\quad + (1-p)\gamma(1-((1-p)\gamma)^{\log_2 N+1})) \\
&= N^{-1}\frac{(2-\gamma)(1-p)\gamma}{1-p\gamma} - N^{-1+\log_2 p\gamma}\frac{(1-(1-p)\gamma)p(1-p)\gamma^2}{2(1-p\gamma)} \\
&\quad - N^{-1+\log_2(1-p)\gamma}\frac{(1-p)^2\gamma^2}{2} \\
&= \frac{1}{N}\frac{(2-\gamma)(1-p)\gamma}{1-p\gamma} + o(\frac{1}{N}).
\end{aligned}
$$

□

An error bound in $O(1/N)$ can be generated to any integer N, as we can relax N to $2^{\lfloor \log_2 N \rfloor - 1}$ so that at least one self-similar segment of size $1/N$ is included in each interval.

For Lipschitz continuous functions like neural networks, the following proposition shows an approximation error lower bound in $O(1/L)$, where L is the Lipschitz constant.

Proposition 47. *Let $L \geq (1-p)\gamma(1-\gamma)/(1-p\gamma)$. If $\bar{v}_2(s)$ is Lipschitz continuous on $s \in (0,1)$ where L is the Lipschitz constant, then*

$$\int_s |v(s) - \bar{v}_2(s)|\, ds \geq \frac{1}{L} \frac{(1-p)^2\gamma^2(1-\gamma)^2}{4(1-p\gamma)}.$$

Proof. We consider $v(\frac{1}{2}) = (1-p)\gamma$ and

$$\lim_{s \to \frac{1}{2}^-} v(s) = \frac{(1-p)^2\gamma^2}{1-p\gamma}.$$

When $0 < \gamma < 1$, we have

$$v(\frac{1}{2}) - \lim_{s \to \frac{1}{2}^-} v(s) = \frac{(1-p)\gamma(1-\gamma)}{1-p\gamma} > 0.$$

Denote $h = (1-p)\gamma(1-\gamma)/(1-p\gamma)$. By the monotonicity of $v(s)$, using $\bar{v}_2(s)$ to approximate $v(s)$ has an error at least $\int_s |\xi(s) - \bar{v}_2(s)|\, ds$, where $\xi(s)$ denotes the step function on $[0,1]$,

$$\xi(s) = \begin{cases} 0 & 0 \leq s < \frac{1}{2}, \\ h & \frac{1}{2} \leq s \leq 1. \end{cases}$$

In this case, the optimal $\bar{v}_2(s)$ is

$$\bar{v}_2(s) = \begin{cases} 0 & 0 \leq s < \frac{1}{2} - \frac{h}{2L}, \\ \frac{h}{2} + L(s - \frac{1}{2}) & \frac{1}{2} - \frac{h}{2L} \leq s \leq \frac{1}{2} + \frac{h}{2L}, \\ h & \frac{1}{2} + \frac{h}{2L} < s \leq 1. \end{cases}$$

Hence, we have

$$\int_s |v(s) - \bar{v}_2(s)|\, ds \geq \int_s |\xi(s) - \bar{v}_2(s)|\, ds \geq \frac{h^2}{4L},$$

as desired. □

5.4.2 Analysis of the Bellman equation

We have proved that $v(s)$ is the optimal value function in Theorem 39, by showing the existence and uniqueness of the solution of the system (ABX). However, the condition (X) is derived from the context of the Gambler's problem. It is rigorous enough to find the optimal value function, but we are also interested in solutions purely derived from the MDP setting. Also, algorithmic approaches such as Q-learning [12, 47, 230] optimize the MDP by solving the Bellman equation, without eliciting the context of the problem. Studying such systems will help to understand the behavior of these algorithms. In this section, we inspect the system of Bellman equation (AB) of the Gambler's problem. We aim to solve the general case (ABY) where $p > 0.5$ and the corner case (ABZ) where $p = 0.5$.

When $p > 0.5$, the value function $v(s)$ is obviously still a solution of the system (ABY) without condition (X). The natu-

ral question is if there exist any other solutions. The answer is two-fold: When $\gamma < 1$, $f(s) = v(s)$ is unique; when $\gamma = 1$, the solution is either $v(s)$ or a constant function at least 1. This indicates that algorithms like Q-learning have constant functions as their set of converging points, apart from $v(s)$. As $v(s)$ is harder to approximate due to the non-smoothness, a constant function in fact induces a smaller approximation error and thus has a better optimality for Q-learning with function approximation.

Generating this result to a general MDP with episodic rewards is immediate, as functions of a large constants solve such an MDP. This indicates that Q-learning may have more than one converging points and may diverge from the optimal value function under $\gamma = 1$. This leads to the need of γ, which is artificially introduced and biases the learning objective. More generally, the Bellman equation may have a continuum of finite solutions in an infinite state space, even with $\gamma < 1$. Some studies exist on the necessary and sufficient conditions for a solution of the Bellman equation to be the value function [233, 234, 235], though the majority of this topic remains open.

The discussions above are supported by a series of rigorous statements. We begin with the following proposition that when the discount factor is strictly less than 1, the solution toward the Bellman equation is the optimal value function.

Proposition 48. *When $\gamma < 1$, $v(s)$ is the unique solution of*

the system (ABY).

Proof. The uniqueness has been shown in Lemma 29 for the system (ABX). When $\gamma < 1$ it corresponds to case (I), where neither the upper bound $f(s) \leq 1$ nor the continuity at $s = 0$ in condition (X) is used. Therefore Lemma 29 holds for (ABY) under $\gamma < 1$, and Lemma 38 follows as desired. □

This uniqueness no longer holds under $\gamma = 1$.

Theorem 49. *Let $\gamma = 1$, $p > 0.5$, and $f(\cdot)$ be a real function on $[0, 1]$. $f(s)$ satisfies the Bellman equation (ABY) if and only if either*

- *$f(s)$ is $v(s)$ defined in Theorem 39, or*
- *$f(0) = 0$, $f(1) = 1$, and $f(s) = C$ for all $0 < s < 1$, for some constant $C \geq 1$.*

Proof. It is obvious that both $f(s)$ defined above are the solutions of the system. It amounts to show that they are the only solutions.

Without the bound condition (X), the function $f(s)$ is not necessarily continuous on $s = 0$ and $s = 1$ and is not necessarily monotonic on $s = 1$. Therefore the same arguments in the proof of Lemma 29 will not hold. However, the arguments can be extended to (Y) by considering the limit of $f(s)$ when s approaches 0 and 1.

By Lemma 31 the function is continuous on the open interval $(0,1)$. Let

$$C_0 = \lim_{s\to 0^+} f(s), \quad C_1 = \lim_{s\to 1^-} f(s).$$

Then by Lemma 30, $0 \le C_0 \le f(s) \le C_1$ for $s \in (0,1)$. Here we eliminate the possibility of $C_0 = +\infty$ and $C_1 = +\infty$. This is because if there is a sequence of $s_t \to 0$ such that $f(s_t) > t$, then we have $f(\frac{1}{2}) \ge p\ f(s_t) + (1-p)\ f(1-s_t) \ge (1-p)t$ for any t. Then $f(\frac{1}{2})$ does not exist. Similar arguments show that C_1 cannot be $+\infty$.

Now specify a sequence $a_t \to \frac{1}{2}$, $a_t < \frac{1}{2}$, such that $C_0 \le f(\frac{1}{2} - a_t) \le C_0 + \frac{1}{t}$ and $C_1 - \frac{1}{t} \le f(\frac{1}{2} + a_t) \le C_1$. Then we have

$$\begin{aligned} f(\frac{1}{2}) &\ge p\ f(\frac{1}{2} - a_t) + (1-p)\ f(\frac{1}{2} + a_t) \\ &\ge p\ C_0 + (1-p)\ C_1 - \frac{1}{t}. \end{aligned}$$

As t is arbitrary we have $f(\frac{1}{2}) \ge pC_0 + (1-p)C_1$. By induction on ℓ it holds on $s \in \bigcup_{\ell\ge 1} G_\ell$ that

$$f(s) \ge C_0 + (C_1 - C_0)v(s).$$

By Lemma 31 (the continuity of $f(s)$ and $v(s)$ under $\gamma = 1$), this lower bound extends beyond the dyadic rationals to the entire interval $(0,1)$. Define $\tilde{f}(s) = C_0 + (C_1 - C_0)v(s)$ for $s \in (0,1)$, $\tilde{f}(0) = C_0, \tilde{f}(1) = C_1$. It is immediate to verify that for any $C_1 > C_0 \ge 0$, $\tilde{f}(s)$ solves the system (AY) (without (B) the boundary conditions).

If $C_1 - C_0 \ne 0$, by Lemma 29 Case (II) $\tilde{f}(s)$ on $(0,1)$ is the unique solution of the system (AY), given monotonicity, conti-

nuity, and the lower bound $\tilde{f}(s)$. With the boundary conditions (B), we have $0 = \tilde{f}(0) = C_0$ and $1 = \tilde{f}(1) = C_1$, therefore $f(s) = v(s)$. This case $C_1 - C_0 \neq 0$ induces the first possible solution.

It amounts to determine $f(s)$ when $C_1 - C_0 = 0$, that is, when $0 \leq C_0 = f(s) = C_1$ for $s \in (0, 1)$ (by Lemma 30). It suffices to prove that $C_0 < 1$ is not possible. In fact, if $C_0 < 1$ then $f(\frac{3}{4}) < p\ f(\frac{1}{2}) + (1-p)f(1)$, which contradicts (A). Then, $f(s) = C_0$ for some $C_0 \geq 1$ is the only set of solutions when $C_1 - C_0 = 0$, as desired. □

The fact that a large constant function can also be a solution toward the Bellman equation can be extended to a wide range of MDPs. The below proposition lists one of the sufficient conditions but even without this condition it is likely to hold in practice.

Proposition 50. *For an arbitrary MDP with episodic rewards where every state has an action to transit to a non-terminal state almost surely, $f(s) = C$ for all non-terminal states s is a solution of the Bellman equation system for any C greater than or equal to the maximum one-step reward.*

Proof. The statement is immediate by verifying the Bellman equation. □

The rest of the section discusses the Gambler's problem under $p = 0.5$, where the gambler does not lose capital by betting

in expectation. In this case, the optimal value function is still $v(s)$ by similar arguments of Theorem 39. It is worth noting that when $\gamma = 1$, Theorem 39 indicates $v(s) = s$. This agrees with the intuition that the gambler does not lose their capital by placing bets in expectation, therefore the optimal value function should be linear to s. Proposition 48 also holds that $v(s)$ is the unique solution of the Bellman equation, given $\gamma < 1$. The remaining problem is to find the solution of the Bellman equation under $\gamma = 1$ and $p = 0.5$. This corresponds to the system (ABZ).

When $p = 0.5$, condition (A) implies midpoint concavity such that for all $a \in \mathcal{A}(s)$,

$$f(s) \geq \frac{1}{2} f(s - a) + \frac{1}{2} f(s + a), \tag{5.11}$$

where the equality must hold for some a. As Lemma 30 no longer holds, a solution $f(s)$ may have negative values for some s. Though, if it does not have a negative value, it must be concave, and thus linear by condition (A) (will be proved later in Theorem 54). It suffices to satisfy $f(s) \geq s$ for any s. Therefore the non-negative solution is $f(0) = 0$, $f(1) = 1$, and $f(s) = C's + B'$ on $0 < s < 1$ for some constants $C' + B' \geq 1$.

If $f(s)$ does have a negative value at some s, then the midpoint concavity (5.11) does not imply concavity. In this case, by recursively applying (5.11) we can show that the set $\{(s, f(s)) \mid s \in (0, 1)\}$ is dense and compact on $(0, 1) \times \mathbb{R}$. Then the function becomes pathological, if it exists. Despite this, the following

lemma shows that $f(s)$ needs to be positive on the rationals $\mathbb{Q}$.

Lemma 51. *Let $f(s)$ satisfy (ABZ). If there exists $0 \leq s^- < s^+ \leq 1$ and a constant C such that $f(s^-), f(s^+) \geq C$, then $f(s) \geq C$ for all $s \in \{s^- + w(s^+ - s^-) \mid w \in \mathbb{Q}, 0 \leq w \leq 1\}$.*

Proof. The statement is immediate for $w \in \{0, 1\}$. For $0 < w < 1$ we prove the lemma by contradiction. Let $f(s^- + w(s^+ - s^-)) < C$ for some $w \in \mathbb{Q}$ while $0 < w < 1$. We define $s_0 = s^- + w(s^+ - s^-)$ and $s_{t+1} = 2s_t - s^-$ for $s_t < \frac{1}{2}(s^- + s^+)$ and $s_{t+1} = 2s_t - s^+$ for $s_t > \frac{1}{2}(s^- + s^+)$, respectively. s_{t+1} will be undefined if $s_t = \frac{1}{2}(s^- + s^+)$. Since $w \in \mathbb{Q}$, let $w = m/n$ where m and n are integers and the greatest common divisor $\gcd(m, n) = 1$. Then $(s_t - s^-)/(s^+ - s^-) = m_t/n$, where $m_t = 2^t m \bmod n$. As $\mathbb{Z}_n$ is finite, $\{s_t\}_{t \geq 0}$ can only take finitely many values. Thus either the sequence $\{s_t\}$ is periodic, or it terminates at some $s_t = \frac{1}{2}(s^- + s^+)$.

Then we show that $f(s_t)$ is strictly decreasing by induction. Assume that $f(s_0) > \cdots > f(s_t)$. When $s_t < \frac{1}{2}(s^- + s^+)$, by (5.11) we have $f(s_t) \geq \frac{1}{2}f(s^-) + \frac{1}{2}f(s_{t+1})$, which indicates that $f(s_{t+1}) - f(s_t) \leq f(s_t) - f(s^-) < f(s_0) - f(s^-) < 0$. When $s_t > \frac{1}{2}(s^- + s^+)$, by (5.11) we have $f(s_t) \geq \frac{1}{2}f(s_{t+1}) + \frac{1}{2}f(s^+)$, which indicates $f(s_{t+1}) - f(s_t) \leq f(s_t) - f(s^+) < f(s_0) - f(s^+) < 0$. The base case $f(s_1) < f(s_0)$ holds as at least one of $f(s_0) \geq \frac{1}{2}f(s^-) + \frac{1}{2}f(s_1)$ and $f(s_0) \geq \frac{1}{2}f(s_1) + \frac{1}{2}f(s^+)$ must be true. Thus we conclude that $f(s_t)$ is strictly decreasing.

If the sequence terminates at some $s_t = \frac{1}{2}(s^- + s^+)$, then

$f(s_t) < f(s_1) < C$, which contradicts $f(s_t) = f(\frac{1}{2}(s^- + s^+)) \geq \frac{1}{2}f(s^-) + \frac{1}{2}f(s^+) \geq C$. Otherwise s_t is periodic and indefinite. Denote the period as T we have $f(s_{t+T}) < f(s_t)$, which indicates $f(s_t) < f(s_t)$ as a contradiction. □

Lemma 51 agrees with the statement that the midpoint concavity indicates rational concavity. The below statements give some insights into the irrational points.

Lemma 52. *Let $f(s)$ satisfy (ABZ). If there exists an $\bar{s} \in \mathbb{R}\setminus\mathbb{Q}$ such that $f(\bar{s}) \geq 0$, then $f(s) \geq 0$ for all $s \in \{w\bar{s}+u \mid w, u \in \mathbb{Q}, 0 \leq w, u, \leq 1, w + u \leq 1\}$.*

Proof. Specify $s^- = \bar{s}$ and $s^+ = 1$ in Lemma 51, we have $f(\bar{s} + \frac{u}{w+u}(1 - \bar{s})) \geq 0$ whenever $0 \leq \frac{u}{w+u} \leq 1$ and $\frac{u}{w+u} \in \mathbb{Q}$. This is satisfied when $0 \leq w, u \leq 1$, $w + u > 0$, $w, u \in \mathbb{Q}$. Specifying $s^- = 0$ and $s^+ = \bar{s}+\frac{u}{w+u}(1-\bar{s})$ in Lemma 51, we have $f(w\bar{s}+u) = f((w + u)(\bar{s} + \frac{u}{w+u}(1 - \bar{s}))) \geq 0$ whenever $w + u \leq 1$. Thus $f(w\bar{s} + u) \geq 0$ for $0 < w, u < 1$, $w, u \in \mathbb{Q}$, and $0 < w + u \leq 1$. Since the case $w = u = 0$ is immediate, the statement follows with $s \in \{w\bar{s} + u \mid w, u \in \mathbb{Q}, 0 \leq w, u, \leq 1, w + u \leq 1\}$. □

Corollary 53. *Let $f(s)$ satisfy (ABZ). If there exists an $\bar{s} \in \mathbb{R}\setminus\mathbb{Q}$ such that $f(\bar{s}) < 0$, then $f(w\bar{s})$ is monotonically decreasing with respect to w for $w \in \mathbb{Q}$, $1 \leq w < 1/\bar{s}$.*

Lemma 52 and Corollary 53 indicate that when there exists a negative or positive value, infinitely many other points (that are not necessarily its neighbors) must have negative or

positive values as well. It is intuitive that the solution with a negative value, if it exists, must be complicated and pathological. In fact, Sierpiński has shown that a midpoint concave but non-concave function is not Lebesgue measurable and is non-constructive [211, 212], so does $f(s)$ if it solves (ABZ) while having a negative value.

Such an $f(s)$ exists if and only if we assume Axiom of Choice [211, 212, 236]. We consider the vector space by field extension $\mathbb{R}/\mathbb{Q}$. With the axiom specify a basis $\mathbb{B} = \{1\} \cup \{g_i\}_{i\in\mathcal{I}}$, known as a Hamel basis. With this basis $\mathbb{B}$ every real number can be written uniquely as a linear combination of the elements in $\mathbb{B}$ with rational coefficients. Therefore, denote a real number s uniquely as a vector $(w, w_i)_{i\in\mathcal{I}}$, $w, w_i \in \mathbb{Q}$, such that $s = w + \sum_{i\in\mathcal{I}} w_i g_i$. We correspond each $f(s) = f'(w, \{w_i\}_{i\in\mathcal{I}})$ uniquely to f' defined on $\mathbb{Q}^{|\mathbb{B}|}$ and use the two spaces interchangeably.

The solution $f(s)$ to (5.11) is any concave function f' on the vector space $\mathbb{R}/\mathbb{Q}$. Based on this solution we extend the system (5.11) to (ABZ). To this end, $f(s)$ needs to attain the equality in (5.11) at every s, which holds if and only if for every $s = w+\sum_{i\in\mathcal{I}} w_i g_i$ there exists $s^1 = w^1+\sum_{i\in\mathcal{I}} w_i^1 g_i$ and $s^2 = w^2+\sum_{i\in\mathcal{I}} w_i^2 g_i$ such that $f'(\lambda w^1 + (1-\lambda)w^2, \{\lambda w_i^1 + (1-\lambda)w_i^2\}_{i\in\mathcal{I}})$ is $\lambda f(s^1) + (1-\lambda)f(s^2)$ for any $0 < \lambda < 1$ (intuitively, local linearity of f' on at least one direction everywhere). By specifying a $\mathbb{B}$, the condition can be met if $f(s) = f'(w, \{w_i\}_{i\in\mathcal{I}}) = \alpha(w_j) + \beta(\overline{w}_j)$ for some $w_j \in \{w\} \cup \{w_i \mid i \in \mathcal{I}\}$, where α is a linear function, β is a concave function, and $\overline{w}_j$ denotes $\{w\} \cup$

$\{w_i \mid i \in \mathcal{I}\} \setminus \{w_j\}$. When w_j is w, $f(s)$ is in fact $w + \beta(\{w_i\}_{i\in\mathcal{I}})$ for some concave function β. This is equivalent to specifying a function $\omega(s) : \mathbb{R} \to \mathbb{Q}$ such that ω maps reals to rationals additively $\omega(s_1 + s_2) = \omega(s_1) + \omega(s_2)$ and ω is not constantly zero, and then write $f(s)$ as $\omega(s) + \beta_1(s - \omega(s))$ for some concave real function β_1. Otherwise if w_j is not w, $f(s)$ is in the aforementioned form with the boundary conditions (B).

While we have shown that under Axiom of Choice $f(s) = \alpha(w_j) + \beta(\overline{w}_j)$ is a set of solutions that can be described by the infinite-dimensional vector space $\mathbb{R}/\mathbb{Q}$ and its basis, we do not know if they are the only solutions. Nevertheless, combining our analysis and the literature we conclude the following statement about the system (ABZ).

Theorem 54. *Let $\gamma = 1$ and $p = 0.5$. A real function $f(s)$ satisfies (ABZ) if and only if either*

- *$f(s) = C's + B'$ on $s \in (0, 1)$, for some constants $C' + B' \geq 1$, or*
- *$f(s)$ is some non-constructive, not Lebesgue measurable function under Axiom of Choice.*

Proof. The first bullet corresponds to the function $f(s)$ that is non-negative on $[0, 1]$. By the midpoint concavity (5.11) and the fact that $f(s)$ is non-negative, $f(s)$ is concave on $[0, 1]$ [211, 212]. We specify $s_0 = \frac{1}{2}$. By (A) we have

$$f(s_0) = \frac{1}{2} f(s_0 - a_0) + \frac{1}{2} f(s_0 + a_0)$$

for some a_0. Since $f(s)$ is concave, $f(s)$ must be linear on the interval $[s_0 - a_0, s_0 + a_0]$. Consider the nonempty set

$$\mathcal{A}_1 = \left\{ a_0 \in \mathcal{A}(s_0) \;\middle|\; f(s_0) = \frac{1}{2} f(s_0 - a_0) + \frac{1}{2} f(s_0 + a_0) \right\}.$$

We show by contradiction that $\sup \mathcal{A}_1$ is $\frac{1}{2}$. If $a_1 = \sup \mathcal{A}_1 < \frac{1}{2}$, then by the continuity $a_1 \in \mathcal{A}_1$, where the continuity is implied by the convexity. This indicates that $f(s) = f(s_0) + \frac{f(s_0+a_1)-f(s_0-a_1)}{2a_1}(s - s_0)$ when $s_0 - a_1 \leq s \leq s_0 + a_1$. But as a_1 is the maximum element of $\mathcal{A}_1$, on at least one of the intervals $[0, s_0 - a_1)$ and $(s_0 + a_1]$ we have by the convexity $f(s) < f(s_0) + \frac{f(s_0+a_1)-f(s_0-a_1)}{2a_1}(s - s_0)$. Therefore, for at least one of $s \in \{s_0 - a_1, s_0 + a_1\}$ we have $f(s) > \frac{1}{2} f(s-a) + \frac{1}{2} f(s+a)$ for all a, which contradicts condition (A). Hence $\sup \mathcal{A}_1$ is $\frac{1}{2}$, which implies that $f(s)$ is linear on $(0, 1)$. Writing $f(s)$ as $C's + B'$, by the boundary condition (B) we have $C' + B' \geq 1$. It is immediate to verify that $f(s) = C's + B'$ with $C' + B' \geq 1$ is sufficient to satisfy the system (ABZ), and is thus the non-negative solution of the system.

If $f(s)$ is negative at some s, then by (5.11) and (B) $f(s)$ must not be concave. By [211, 212] such an $f(s)$ exists only if we assume Axiom of Choice, and is non-constructive and not Lebesgue measurable if it exists (some discussion on this function is given above the statement of this theorem). □

5.5 Summary and insights

We give a complete solution to the Gambler's problem, a simple and classic problem in reinforcement learning, under a variety of settings. We show that its optimal value function is very complicated and even pathological. Despite its seeming simpleness, these results are not clearly pointed out in previous studies.

Our contributions are the theoretical finding and the implications. It is worthy to bring the current results to start the discussion among the community. Indicated by the Gambler's problem, the current algorithmic approaches in reinforcement learning might underestimate the complexity. We expect more evidence could be found in the future and new algorithms and implementations could be brought out.

It would be interesting to see how these results of the Gambler's problem generalizes to other MDPs. Finding these characterizations of MDPs is in general an important step to understand reinforcement learning and sequential decision processes.

□ **End of chapter.**

Chapter 6

Conclusion

In this thesis we study policy optimization, the core of reinforcement learning where we seek trial-and-error methods for intelligent agents to learn in an interactive environment. We view the improvement of such a learning process in two perspectives of sample efficiency and sample quality. The former is studied systematically by rigorous approaches. The latter is investigated in applications by practically efficient heuristic algorithms. We also support these works with our research on the foundations by optimal control, which turns out to be an independent and innovative discovery. We leave many possible extensions and the involvement of cognitive science and psychology concepts to my future works.

6.1 Future works

Despite that our study on the improvement of sample efficiencies is a relatively complete series of works, from which two

independent theoretical contributions stand out. These results are general enough to have their implications on potential topics of future works.

Variable partitioning could be useful for a wide range of problems in statistics, algorithms, and learning theory. For reinforcement learning, it is particularly useful in the setting of multiple agents. It could be interesting to investigate how the partitions of the value function correspond to the separability between the agents. Further, partitioning the utility function can be an important tool to achieve decentralized training and decentralized execution. Variants of variable partitioning such as sparse partitioning and tolerate partitioning might lead to extensions in multi-agent reinforcement learning. The former can be used to achieve balanced divide-and-conquer algorithms, while the latter is related to communication between agents under decentralized execution.

The Gambler's problem has several implications that are new to the community. Following them immediately are the open problems in value function representation and the Q-learning algorithm. For value function representation, we might look at approaches to either obtain the ability to approximate fractal functions, or modify the objective so as to learn a surrogate instead of the optimal value function. The latter should then justify its guarantee of approximate optimality and demonstrate its empirical performance. For Q-learning, we look at variants of the algorithm that tend to converge to the optimal value func-

tion, without biasing the objective. Once a feasible algorithm is available for the non-discounted setting, more efforts can be put into scaling it towards complicated, real-world tasks.

□ **End of chapter.**

Appendix A

Proofs and Additional Details

A.1 Statistical claims

Claim 55. *Assume* $t, \hat{t} \geq 0$. *If* $t^p \leq \hat{t}^p \leq t^p + (\varepsilon/2)^p$ *then* $t \leq \hat{t} \leq t + \varepsilon$.

Proof. The left-hand inequalities are immediate. For the right-hand ones we consider two cases. If $t \leq \varepsilon/2$, then $\hat{t}^p \leq 2(\varepsilon/2)^p \leq \varepsilon \leq t + \varepsilon$. If $t > \varepsilon/2$ then

$$\hat{t} - t \leq \frac{\hat{t}^p - t^p}{t^{p-1}} \leq \frac{(\varepsilon/2)^p}{(\varepsilon/2)^{p-1}} \leq \varepsilon. \qquad \square$$

Claim 13. *Assuming* $\|F\|_{\mathbb{R},2p} \leq 1$, *the value* $\|F\|^p_{\mathbb{R},p}$ *can be estimated within* ε^p, *and* $\|F\|_{\mathbb{R},p}$ *can be estimated within* ε, *from* $K^p \log(1/\gamma)/\epsilon^{2p}$ *queries to* F *in linear time with probability* $1-\gamma$ *for some absolute constant* K.

Proof. By Chebyshev's inequality, $\mathbb{E}[|F|^p]$ can be estimated within an additive error of $(\varepsilon/2)^p$ by averaging $(2/\varepsilon)^{2p}$ samples with probability 3/4. The error can be improved to $1-\gamma$ by taking

the median value of $O(\log 1/\gamma)$ runs. The second bound follows from Claim 55. □

A.2 Additional experiment details

A.2.1 Two examples of the synthetic environment

The two examples are generated as analogies to practical RL environments, where the dependency between variables are sparse. Each edge has a 0.3 probability to be a dependency where the value is uniformly sampled from $[2, 10]$. Otherwise the edge is deemed noise and is uniformly sampled from $[0, 1]$. We make both the matrix tractable in the optimization by substituting an identity matrix. The matrices used in the experiments are $H^1 - 11.65\mathbb{I}$ and $H^2 - 14.98\mathbb{I}$.

$$H^1 = \begin{bmatrix} 0 & 7.14 & 6.53 & 5.67 & 0.58 \\ 7.14 & 0 & 0.71 & 5.29 & 0.52 \\ 6.53 & 0.71 & 0 & 0.37 & 8.08 \\ 5.67 & 5.29 & 0.37 & 0 & 0.96 \\ 0.58 & 0.53 & 8.08 & 0.96 & 0 \end{bmatrix},$$

$$H^2 = \begin{bmatrix} 0 & 0.27 & 0.67 & 0.48 & 7.52 \\ 0.27 & 0 & 4.11 & 4.12 & 5.22 \\ 0.67 & 4.11 & 0 & 4.74 & 0.80 \\ 0.48 & 4.12 & 4.74 & 0 & 0.46 \\ 7.52 & 5.22 & 0.80 & 0.46 & 0 \end{bmatrix}.$$

A.2.2 Understanding the synthetic environment

The Minimum Cut Problem (Min-cut) on a weighted undirected graph is a well-known problem whose objective is a submodular function. We show that the variable partition on our synthetic environment is equivalent to Min-cut. This also provides an intuitive understanding of our Theorem 8 and Algorithm 2.

Let $a_{\boldsymbol{X}}$ denote elements of the vector a whose index is in $\boldsymbol{X}$ and $H_{\boldsymbol{X},\boldsymbol{X}}$ denote elements of the matrix H whose row and column indexes are in $\boldsymbol{X}$. Then the variable partition problem of the synthetic environment is

$$\underset{\boldsymbol{X}}{\text{minimize}} \quad F(\boldsymbol{X}) = \mathbb{E}_a\big[a_{\boldsymbol{X}}^T H_{\boldsymbol{X},\boldsymbol{X}} a_{\boldsymbol{X}} + a_{\overline{\boldsymbol{X}}}^T H_{\overline{\boldsymbol{X}},\overline{\boldsymbol{X}}} a_{\overline{\boldsymbol{X}}}\big]. \tag{A.1}$$

Proposition 56. *Under Gaussian policies, minimizing the dependence score D over the bipartition $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ on the synthetic environment is equivalent to a Min-cut problem.*

Proof. Without loss of generality let $H = \mathbf{1}$ be an all-one matrix. This will correspond to an unweighted graph. For a general H we just have to change the weight of the edge (i, j) to H_{ij}. Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_j)$, $\boldsymbol{Y} = \overline{\boldsymbol{X}} = (\boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_n, \boldsymbol{y}_{j+1}, .., \boldsymbol{y}_n)$. We have

$$\begin{aligned} D_F(\boldsymbol{X}, \boldsymbol{Y}) &= F(X, Y) + F(X', Y') - F(X', Y) - F(X, Y') \\ &= (x_1 + \cdots + x_n)^2 + (y_1 + \cdots + y_n)^2 \\ &\quad + (x'_1 + \cdots + x'_n)^2 + (y'_1 + \cdots + y'_n)^2 \\ &\quad - (x_1 + \cdots + x_i + x'_{i+1} + \cdots + x'_n)^2 \end{aligned}$$

$$
\begin{aligned}
&- (y_1 + \cdots + y_j + y'_{j+1} + \cdots + y'_n)^2 \\
&- (x'_1 + \cdots + x'_i + x_{i+1} + \cdots + x_n)^2 \\
&- (y'_1 + \cdots + y'_j + y_{j+1} + \cdots + y_n)^2.
\end{aligned}
$$

Let $x_1 \dots x_i \circ x_{i+1} \dots x_n$ denote $\sum_{1 \le k \le i, i+1 \le l \le n} x_k x_l$. The above equals

$$
\begin{aligned}
D_F(\boldsymbol{X}, \boldsymbol{Y}) = {} & x_1 \dots x_i \circ x_{i+1} \dots x_n + y_1 \dots y_j \circ y_{j+1} ... y_n \\
& + x'_1 \dots x'_i \circ x'_{i+1} \dots x'_n + y'_1 \dots y'_j \circ y'_{j+1} \dots y'_n \\
& - x_1 \dots x_i \circ x'_{i+1} \dots x'_n - y_1 \dots y_j \circ y'_{j+1} \dots y'_n \\
& - x'_1 \dots x'_i \circ x_{i+1} \dots x_n - y'_1 \dots y'_j \circ y_{j+1} \dots y_n.
\end{aligned}
$$

Denoting $dx_k = x_k - x'_k$ and $dy_k = y_k - y'_k$, respectively, it is then equal to

$$D_F(\boldsymbol{X}, \boldsymbol{Y}) = dx_1 \dots dx_i \circ dx_{i+1} \dots dx_n + dy_1 \dots dy_j \circ dy_{j+1} \dots dy_n.$$

Since each x_k is a Gaussian random variable, we have $dx_k \sim \mathbb{N}(0, \sqrt{2}\sigma)$. Hence,

$$
\begin{aligned}
\mathbb{E}[D_F^2] &= \mathbb{E}[dx_1^2 \dots dx_i^2 \circ dx_{i+1}^2 \dots dx_n^2 + dy_1^2 \dots dy_j^2 \circ dy_{j+1}^2 \dots dy_n^2] \\
&= \mathbb{E}[dx_1^2]\mathbb{E}[dx_2^2](i(n-i) + j(n-j)) \\
&= 4\sigma^4(i(n-i) + j(n-j)),
\end{aligned}
$$

where $i(n-i)+j(n-j)$ is the number of edge cut by the partition $(\boldsymbol{X}, \boldsymbol{Y})$. Minimizing $D_F(\boldsymbol{X}, \boldsymbol{Y})$ is exactly equivalent to finding the solution of the Min-cut problem. □

□ **End of chapter.**

Appendix B

List of Publications

1. The Gambler's Problem and Beyond.
   Baoxiang Wang, Shuai Li, Jiajin Li and Siu On Chan.
   International Conference on Learning Representations (ICLR) 2020.

2. Learning and Testing Variable Partitions.[1]
   Andrej Bogdanov, Baoxiang Wang.
   Innovations in Theoretical Computer Science (ITCS) 2020.

3. Privacy-preserving Q-Learning with Functional Noise in Continuous Spaces.
   Baoxiang Wang, Nidhi Hegde.
   Advances in Neural Information Processing Systems (NeurIPS) 2019.

[1] Authors are listed in alphabetical order.

4. Recurrent Existence Determination Through Policy Optimization.
   Baoxiang Wang.
   International Joint Conference on Artificial Intelligence (IJCAI) 2019.

5. Metatrace Actor-Critic: Online Step-Size Tuning by Meta-Gradient Descent for Reinforcement Learning Control.
   Kenny Young, Baoxiang Wang, Matthew E. Taylor.
   International Joint Conference on Artificial Intelligence (IJCAI) 2019.

6. Beyond Winning and Losing: Modeling Human Motivations and Behaviors with Vector-valued Inverse Reinforcement Learning.
   Baoxiang Wang, Tongfang Sun, Xianjun Sam Zheng.
   Artificial Intelligence and Interactive Digital Entertainment (AIIDE) 2019.

7. Policy Optimization with Second-Order Advantage Information.[2]
   Jiajin Li, Baoxiang Wang.
   International Joint Conference on Artificial Intelligence (IJ-

[2]Authors are listed in alphabetical order.

CAI) 2018.

8. Contextual Combinatorial Cascading Bandit.
   Shuai Li, Baoxiang Wang, Shengyu Zhang, Wei Chen.
   International Conference on Machine Learning (ICML) 2016.

9. PAID: Prioritizing App Issues for Developers by Tracking User Reviews Over Versions.
   Cuiyun Gao, Baoxiang Wang, Pinjia He, Jieming Zhu, Yangfan Zhou, Michael R. Lyu.
   International Symposium on Software Reliability Engineering (ISSRE) 2015.

Bibliography

[1] A. Turing, "Intelligent machinery (1948)," *The Essential Turing*, pp. 395–432, 1948.

[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[3] C. Szepesvári, *Algorithms for reinforcement learning*. Morgan & Claypool Publishers, 2010.

[4] D. P. Bertsekas and J. N. Tsitsiklis, "Neuro-dynamic programming: An overview," in *Proceedings of 1995 34th IEEE Conference on Decision and Control*, vol. 1. IEEE, 1995, pp. 560–564.

[5] D. Bertsekas, "Distributed dynamic programming," *IEEE Transactions on Automatic Control*, vol. 27, no. 3, pp. 610–616, 1982.

[6] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[7] L. P. Kaelbling, "Hierarchical learning in stochastic domains: Preliminary results," in *International Conference on Machine Learning*, vol. 951, 1993, pp. 167–173.

[8] M. Sugiyama, *Statistical reinforcement learning: Modern machine learning approaches*. CRC Press, 2015.

[9] 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*, vol. 529, no. 7587, pp. 484–489, 2016.

[10] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[11] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering Chess and Shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

[12] 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*, vol. 518, no. 7540, p. 529, 2015.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[14] G. Zheng, F. Zhang, Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie, and Z. Li, "DRN: A deep reinforcement learning framework for news recommendation," in *Proceedings of the 27th International Conference on World Wide Web*, 2018, pp. 167–176.

[15] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[16] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representa-

tion and trading," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 653–664, 2016.

[17] M. Moravčík, M. Schmid, N. Burch, V. Lisỳ, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[18] N. Brown and T. Sandholm, "Superhuman AI for heads-up no-limit poker: Libratus beats top professionals," *Science*, vol. 359, no. 6374, pp. 418–424, 2018.

[19] M. Bowling, N. Burch, M. Johanson, and O. Tammelin, "Heads-up limit hold'em poker is solved," *Science*, vol. 347, no. 6218, pp. 145–149, 2015.

[20] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M. G. Bellemare, and M. Bowling, "The Hanabi challenge: A new frontier for AI research," *Artificial Intelligence*, vol. 280, p. 103216, 2020.

[21] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2017, pp. 31–36.

[22] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *International Conference on Robotics and Automation*. IEEE, 2017, pp. 3389–3396.

[23] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[24] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to real reinforcement learning for autonomous driving," *arXiv preprint arXiv:1704.03952*, 2017.

[25] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016.

[26] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[27] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooper-

ation and competition with deep reinforcement learning," *PloS One*, vol. 12, no. 4, 2017.

[28] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo, Q. Chen, Y. Yin, H. Zhang, T. Shi, L. Wang, Q. Fu, W. Yang, and L. Huang, "Mastering complex control in MOBA games with deep reinforcement learning," *arXiv preprint arXiv:1912.09729*, 2019.

[29] W. B. Powell, *Approximate dynamic programming: Solving the curses of dimensionality.* John Wiley & Sons, 2007, vol. 703.

[30] F. L. Lewis and D. Liu, *Reinforcement learning and approximate dynamic programming for feedback control.* John Wiley & Sons, 2013, vol. 17.

[31] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, *Handbook of learning and approximate dynamic programming.* John Wiley & Sons, 2004, vol. 2.

[32] D. P. Bertsekas, *Dynamic programming and optimal control.* Athena Scientific Belmont, MA, 2012, vol. 2.

[33] X.-R. Cao, "Stochastic learning and optimization-a sensitivity-based approach," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 3480–3492, 2008.

[34] R. Bellman and S. Dreyfus, "Functional approximations and dynamic programming," *Mathematical Tables and Other Aids to Computation*, pp. 247–251, 1959.

[35] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*, 1st ed. Athena Scientific Belmont, MA, 1996.

[36] C. J. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Cambridge, 1989.

[37] P. J. Werbos, "Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, no. 1, pp. 7–20, 1987.

[38] I. H. Witten, "An adaptive optimal controller for discrete-time Markov environments," *Information and Control*, vol. 34.4, pp. 286–295, 1977.

[39] R. S. Woodworth, B. Barber, and H. Schlosberg, *Experimental psychology.* Oxford and IBH Publishing, 1954.

[40] E. L. Thorndike, "Animal intelligence: An experimental study of the associative processes in animals." *The Psychological Review: Monograph Supplements*, vol. 2, no. 4, p. i, 1898.

[41] D. Crevier, *AI: The tumultuous history of the search for artificial intelligence.* Basic Books, Inc., 1993.

[42] R. S. Sutton, "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, University of Massachusetts Amherst, 1984.

[43] I. H. Witten, "Learning to control," Ph.D. dissertation, University of Essex, 1976.

[44] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman, "PAC model-free reinforcement learning," in *International Conference on Machine Learning*, 2006, pp. 881–888.

[45] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13, no. 5, pp. 835–846, 1983.

[46] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[47] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[48] A. G. Barto and R. S. Sutton, "Goal seeking components for adaptive intelligence: An initial assessment," University of Massachusetts Amherst, Tech. Rep., 1981.

[49] A. H. Klopf, *Brain function and adaptive systems: A heterostatic theory.* Air Force Cambridge Research Labo-

ratories, Air Force Systems Command, United States Air Force, 1972, no. 133.

[50] A. H. Klopf, "A comparison of natural and artificial intelligence," *ACM SIGART Bulletin*, no. 52, pp. 11–13, 1975.

[51] A. H. Klopf, *The hedonistic neuron: A theory of memory, learning, and intelligence*. Toxicology-Sci, 1982.

[52] J. G. March, "Exploration and exploitation in organizational learning," *Organization Science*, vol. 2, no. 1, pp. 71–87, 1991.

[53] A. K. Gupta, K. G. Smith, and C. E. Shalley, "The interplay between exploration and exploitation," *Academy of Management Journal*, vol. 49, no. 4, pp. 693–706, 2006.

[54] K. Doya, K. Samejima, K.-i. Katagiri, and M. Kawato, "Multiple model-based reinforcement learning," *Neural Computation*, vol. 14, no. 6, pp. 1347–1369, 2002.

[55] C. G. Atkeson and J. C. Santamaria, "A comparison of direct and model-based reinforcement learning," in *International Conference on Robotics and Automation*, vol. 4. IEEE, 1997, pp. 3557–3564.

[56] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal control*. John Wiley & Sons, 2012.

[57] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous

methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016.

[58] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. Van de Wiele, V. Mnih, N. Heess, and J. T. Springenberg, "Learning by playing-solving sparse reward tasks from scratch," *arXiv preprint arXiv:1802.10567*, 2018.

[59] J. Hare, "Dealing with sparse rewards in reinforcement learning," *arXiv preprint arXiv:1910.09281*, 2019.

[60] W. D. Smart and L. P. Kaelbling, "Effective reinforcement learning for mobile robots," in *International Conference on Robotics and Automation*, vol. 4. IEEE, 2002, pp. 3404–3410.

[61] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, 2000, pp. 1057–1063.

[62] S. Kakade, M. Wang, and L. F. Yang, "Variance reduction methods for sublinear reinforcement learning," *arXiv preprint arXiv:1802.09184*, 2018.

[63] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine, "Q-prop: Sample-efficient policy gradient with

an off-policy critic," in *International Conference on Learning Representations*, 2017.

[64] H. Liu, Y. Feng, Y. Mao, D. Zhou, J. Peng, and Q. Liu, "Action-dependent control variates for policy optimization via stein identity," in *International Conference on Learning Representations*, 2018.

[65] W. Grathwohl, D. Choi, Y. Wu, G. Roeder, and D. Duvenaud, "Backpropagation through the void: Optimizing control variates for black-box gradient estimation," in *International Conference on Learning Representations*, 2018.

[66] G. Tucker, S. Bhupatiraju, S. Gu, R. Turner, Z. Ghahramani, and S. Levine, "The mirage of action-dependent baselines in reinforcement learning," in *International Conference on Machine Learning*, 2018, pp. 5015–5024.

[67] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[68] A. Rajeswaran, K. Lowrey, E. V. Todorov, and S. M. Kakade, "Towards generalization and simplicity in continuous control," in *Advances in Neural Information Processing Systems*, 2017, pp. 6550–6561.

[69] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous con-

trol with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[70] C. Wu, A. Rajeswaran, Y. Duan, V. Kumar, A. M.Bayen, S. Kakade, I. Mordatch, and P. Abbeel, "Variance reduction for policy gradient with action-dependent factorized baselines," in *International Conference on Learning Representations*, 2018.

[71] J. Li and B. Wang, "Policy optimization with second-order advantage information," *arXiv preprint arXiv:1805.03586*, 2018.

[72] S. Schaal, "Learning from demonstration," in *Advances in Neural Information Processing Systems*, 1997, pp. 1040–1046.

[73] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[74] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys*, vol. 50, no. 2, pp. 1–35, 2017.

[75] M. Grześ and D. Kudenko, "Online learning of shaping rewards in reinforcement learning," *Neural Networks*, vol. 23, no. 4, pp. 541–550, 2010.

[76] A. D. Laud, "Theory and application of reward shaping in reinforcement learning," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2004.

[77] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," *arXiv preprint arXiv:1707.05300*, 2017.

[78] S. Narvekar, J. Sinapov, and P. Stone, "Autonomous task sequencing for customized curriculum design in reinforcement learning." in *IJCAI*, 2017, pp. 2536–2542.

[79] J. Z. Leibo, E. Hughes, M. Lanctot, and T. Graepel, "Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research," *arXiv preprint arXiv:1903.00742*, 2019.

[80] N. Schweighofer and K. Doya, "Meta-learning in reinforcement learning," *Neural Networks*, vol. 16, no. 1, pp. 5–9, 2003.

[81] A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine, "Meta-reinforcement learning of structured exploration strategies," in *Advances in Neural Information Processing Systems*, 2018, pp. 5302–5311.

[82] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of Artificial Intelligence Research*, vol. 13, pp. 227–303, 2000.

[83] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete Event Dynamic Systems*, vol. 13, no. 1-2, pp. 41–77, 2003.

[84] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *AAAI Conference on Artificial Intelligence*, 2017.

[85] H. M. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. Daumé III, "Hierarchical imitation and reinforcement learning," *arXiv preprint arXiv:1803.00590*, 2018.

[86] S. O. Chan, "Approximation resistance from pairwise-independent subgroups," *Journal of the ACM*, vol. 63, no. 3, pp. 1–32, 2016.

[87] B. Barak, S. O. Chan, and P. K. Kothari, "Sum of squares lower bounds from pairwise independence," in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, 2015, pp. 97–106.

[88] S. O. Chan, I. Diakonikolas, R. A. Servedio, and X. Sun, "Efficient density estimation via piecewise polynomial approximation," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, 2014, pp. 604–613.

[89] S. O. Chan, D. Papailliopoulos, and A. Rubinstein, "On the approximability of sparse PCA," in *Conference on Learning Theory*, 2016, pp. 623–646.

[90] K. J. Young, R. S. Sutton, and S. Yang, "Integrating episodic memory into a reinforcement learning agent using reservoir sampling," *arXiv preprint arXiv:1806.00540*, 2018.

[91] C. Sherstan, B. Bennett, K. Young, D. R. Ashley, A. White, M. White, and R. S. Sutton, "Directly estimating the variance of the λ-return using temporal-difference methods," *arXiv preprint arXiv:1801.08287*, 2018.

[92] C. Sherstan, D. R. Ashley, B. Bennett, K. Young, A. White, M. White, and R. S. Sutton, "Comparing direct and indirect temporal-difference methods for estimating the variance of the return." in *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, 2018, pp. 63–72.

[93] J. Mei, C. Xiao, C. Szepesvari, and D. Schuurmans, "On the global convergence rates of softmax policy gradient methods," *arXiv preprint arXiv:2005.06392*, 2020.

[94] C. Xiao, R. Huang, J. Mei, D. Schuurmans, and M. Müller, "Maximum entropy Monte-Carlo planning," in *Advances in Neural Information Processing Systems*, 2019, pp. 9520–9528.

[95] J. Mei, C. Xiao, R. Huang, D. Schuurmans, and M. Müller, "On principled entropy exploration in policy optimiza-

tion," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 3130–3136.

[96] C. Xiao, J. Mei, and M. Müller, "Memory-augmented Monte Carlo tree search," in *AAAI Conference on Artificial Intelligence*, 2018.

[97] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.

[98] A. Bogdanov and B. Wang, "Learning and testing variable partitions," in *Innovations in Theoretical Computer Science Conference*, 2020.

[99] A. Das, H. Agrawal, C. L. Zitnick, D. Parikh, and D. Batra, "Human attention in visual question answering: Do humans and deep networks look at the same regions?" *arXiv preprint arXiv:1606.03556*, 2016.

[100] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A survey," *arXiv preprint arXiv:1411.5878*, 2014.

[101] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[102] S. Dieleman, K. W. Willett, and J. Dambre, "Rotation-invariant convolutional neural networks for galaxy morphology prediction," *arXiv preprint arXiv:1503.07077*, 2015.

[103] B. Graham, "Fractional max-pooling," *arXiv preprint arXiv:1412.6071*, 2014.

[104] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Advances in Neural Information Processing Systems*, 2014, pp. 2204–2212.

[105] J. Ba, R. R. Salakhutdinov, R. B. Grosse, and B. J. Frey, "Learning wake-sleep recurrent attention models," in *Advances in Neural Information Processing Systems*, 2015, pp. 2593–2601.

[106] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *arXiv preprint arXiv:1412.7755*, 2014.

[107] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2678–2687.

[108] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[109] B. Wang, "Recurrent existence determination through policy optimization," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 3656–3662.

[110] B. Wang, S. Li, J. Li, and S. O. Chan, "The gambler's problem and beyond," *arXiv preprint arXiv:2001.00102*, 2019.

[111] S. Arimoto, S. Kawamura, F. Miyazaki, and S. Tamaki, "Learning control theory for dynamical systems," in *1985 24th IEEE Conference on Decision and Control*. IEEE, 1985, pp. 1375–1380.

[112] G. D. Birkhoff, *Dynamical systems*. American Mathematical Society, 1927, vol. 9.

[113] R. Bellman, "A Markovian decision process," *Journal of Mathematics and Mechanics*, pp. 679–684, 1957.

[114] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[115] J. Rust, "Numerical dynamic programming in economics," *Handbook of Computational Economics*, vol. 1, pp. 619–729, 1996.

[116] W. S. Lovejoy, "A survey of algorithmic methods for partially observed Markov decision processes," *Annals of Operations Research*, vol. 28, no. 1, pp. 47–65, 1991.

[117] D. J. White, "Real applications of Markov decision processes," *Interfaces*, vol. 15, no. 6, pp. 73–83, 1985.

[118] D. J. White, "Further real applications of Markov decision processes," *Interfaces*, vol. 18, no. 5, pp. 55–61, 1988.

[119] D. J. White, "A survey of applications of Markov decision processes," *Journal of the Operational Research Society*, vol. 44, no. 11, pp. 1073–1096, 1993.

[120] A. G. Barto, R. S. Sutton, and P. S. Brouwer, "Associative search network: A reinforcement learning associative memory," *Biological Cybernetics*, vol. 40, no. 3, pp. 201–211, 1981.

[121] A. G. Barto and R. S. Sutton, "Landmark learning: An illustration of associative search," *Biological Cybernetics*, vol. 42, no. 1, pp. 1–8, 1981.

[122] A. G. Barto and P. Anandan, "Pattern-recognizing stochastic learning automata," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 3, pp. 360–375, 1985.

[123] J. H. Holland, "Escaping brittleness: The possibilities of general-purpose learning algorithms applied to paral-

lel rule-based systems," *Machine Learning: An Artificial Intelligence Approach*, vol. 2, pp. 593–623, 1986.

[124] G. Tesauro, "TD-Gammon, a self-teaching backgammon program, achieves master-level play," *Neural Computation*, vol. 6, no. 2, pp. 215–219, 1994.

[125] W. Liu, S. Li, and S. Zhang, "Contextual dependent click bandit algorithm for web recommendation," in *International Computing and Combinatorics Conference*. Springer, 2018, pp. 39–50.

[126] S. Li, W. Chen, S. Li, and K.-S. Leung, "Improved algorithm on online clustering of bandits," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 2923–2929.

[127] S. Li, T. Lattimore, and C. Szepesvári, "Online learning to rank with features," in *International Conference on Machine Learning*, 2019, pp. 3856–3865.

[128] S. Li, Y. Abbasi-Yadkori, B. Kveton, S. Muthukrishnan, V. Vinay, and Z. Wen, "Offline evaluation of ranking policies with click models," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 1685–1694.

[129] S. Li and S. Zhang, "Online clustering of contextual cascading bandits," in *AAAI Conference on Artificial Intelligence*, 2018.

[130] S. Li, B. Wang, S. Zhang, and W. Chen, "Contextual combinatorial cascading bandits," in *International Conference on Machine Learning*, 2016, pp. 1245–1253.

[131] R. Huang, M. M. Ajallooeian, C. Szepesvári, and M. Müller, "Structured best arm identification with fixed confidence," in *International Conference on Algorithmic Learning Theory*, 2017, pp. 593–616.

[132] R. Huang, T. Lattimore, A. György, and C. Szepesvári, "Following the leader and fast rates in linear prediction: Curved constraint sets and other regularities," in *Advances in Neural Information Processing Systems*, 2016, pp. 4970–4978.

[133] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," *arXiv preprint arXiv:1511.03034*, 2015.

[134] B. Wang and N. Hegde, "Privacy-preserving q-learning with functional noise in continuous spaces," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 323–11 333.

[135] K. Young, B. Wang, and M. E. Taylor, "Metatrace actor-critic: Online step-size tuning by meta-gradient descent for reinforcement learning control," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 4185–4191.

[136] B. Wang, T. Sun, and X. S. Zheng, "Beyond winning and losing: Modeling human motivations and behaviors using inverse reinforcement learning," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 15, no. 1, 2019.

[137] C. Gao, B. Wang, P. He, J. Zhu, Y. Zhou, and M. R. Lyu, "PAID: Prioritizing app issues for developers by tracking user reviews over versions," in *2015 IEEE 26th International Symposium on Software Reliability Engineering*, 2015, pp. 35–45.

[138] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[139] G. Casella and C. P. Robert, "Rao-Blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.

[140] R. Ranganath, S. Gerrish, and D. Blei, "Black box variational inference," in *Artificial Intelligence and Statistics*, 2014, pp. 814–822.

[141] R. David, I. Dinur, E. Goldenberg, G. Kindler, and I. Shinkar, "Direct sum testing," *SIAM Journal on Computing*, vol. 46, no. 4, pp. 1336–1369, 2017.

[142] M. Grötschel, L. Lovász, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, no. 2, pp. 169–197, 1981.

[143] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Journal of Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 346–355, 2000.

[144] S. Iwata, L. Fleischer, and S. Fujishige, "A combinatorial strongly polynomial algorithm for minimizing submodular functions," *Journal of the ACM*, vol. 48, no. 4, pp. 761–777, 2001.

[145] I. Kostrikov, "PyTorch implementations of reinforcement learning algorithms," https://github.com/ikostrikov/pytorch-a2c-ppo-acktr, 2018.

[146] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[147] R. Ranganath, D. Tran, and D. Blei, "Hierarchical variational models," in *International Conference on Machine Learning*, 2016, pp. 324–333.

[148] T. Degris, M. White, and R. S. Sutton, "Off-policy actor-critic," *arXiv preprint arXiv:1205.4839*, 2012.

[149] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[150] Y. Li and R. E. Turner, "Gradient estimators for implicit models," *arXiv preprint arXiv:1705.07107*, 2017.

[151] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 2016, pp. 7–10.

[152] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2006, pp. 554–560.

[153] D. Karger and M. S. Levine, "Fast augmenting paths by random sampling from residual graphs," *SIAM Journal on Computing*, vol. 44, pp. 320–339, 03 2015.

[154] K.-i. Kawarabayashi and M. Thorup, "Deterministic global minimum cut of a simple graph in near-linear time,"

in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, 2015, pp. 665–674.

[155] C. Chekuri and S. Li, "A note on the hardness of approximating the k-way hypergraph cut problem," 2015.

[156] P. Manurangsi, "Almost-polynomial ratio ETH-hardness of approximating densest k-subgraph," in *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, 2017, pp. 954–961.

[157] K. Chandrasekaran, C. Xu, and X. Yu, "Hypergraph k-cut in randomized polynomial time," in *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2018, pp. 1426–1438.

[158] A. Rubinstein, T. Schramm, and S. M. Weinberg, "Computing exact minimum cuts without knowing the graph," in *Innovations in Theoretical Computer Science Conference*, vol. 94, 2018, pp. 39:1–39:16.

[159] D. R. Karger, "Minimum cuts in near-linear time," *Journal of the ACM*, vol. 47, no. 1, pp. 46–76, 2000.

[160] D. R. Karger and C. Stein, "A new approach to the minimum cut problem," *Journal of the ACM*, vol. 43, no. 4, pp. 601–640, 1996.

[161] E. Mossel, R. O'Donnell, and R. P. Servedio, "Learning juntas," in *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, 2003, pp. 206–212.

[162] E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky, "Testing juntas," *Journal of Computer and System Sciences*, vol. 68, no. 4, pp. 753–787, 2004.

[163] H. Chockler and D. Gutfreund, "A lower bound for testing juntas," *Information Processing Letters*, vol. 90, no. 6, pp. 301–305, 2004.

[164] E. Blais, "Testing juntas nearly optimally," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009, pp. 151–158.

[165] M. Saglam, "Near log-convexity of measured heat in (discrete) time and consequences," in *IEEE Annual Symposium on Foundations of Computer Science*, 2018, pp. 967–978.

[166] X. Chen, R. A. Servedio, L.-Y. Tan, E. Waingarten, and J. Xie, "Settling the query complexity of non-adaptive junta testing," *Journal of the ACM*, vol. 65, no. 6, pp. 40:1–40:18, 2018.

[167] N. H. Bshouty, "Almost optimal distribution-free junta testing," in *34th Computational Complexity Conference*, 2019, pp. 2:1–2:13.

[168] I. Dinur and K. Golubev, "Direct sum testing: The general case," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2019, pp. 40:1–40:11.

[169] R. M. Roth and K. Viswanathan, "On the hardness of decoding the Gale-Berlekamp code," in *2007 IEEE International Symposium on Information Theory*, 2007, pp. 1356–1360.

[170] M. Karpinski and W. Schudy, "Linear time approximation schemes for the Gale-Berlekamp game and related minimization problems," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009, pp. 313–322.

[171] R. Klimmek and F. Wagner, "A simple hypergraph min cut algorithm," 1996, technical Report B.

[172] M. Queyranne, "Minimizing symmetric submodular functions," *Mathematical Programming*, vol. 82, no. 1-2, pp. 3–12, 1998.

[173] H. Saran and V. V. Vazirani, "Finding k cuts within twice the optimal," *SIAM Journal on Computing*, vol. 24, no. 1, pp. 101–108, 1995.

[174] S. Iwata and J. Orlin, "A simple combinatorial algorithm for submodular function minimization," in *Proceedings of*

the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, 01 2009, pp. 1230–1237.

[175] E. Blais, C. L. Canonne, T. Eden, A. Levi, and D. Ron, "Tolerant junta testing and the connection to submodular optimization and function isomorphism," in *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2018, pp. 2113–2132.

[176] Y. T. Lee, A. Sidford, and S. C.-w. Wong, "A faster cutting plane method and its implications for combinatorial and convex optimization," in *IEEE Annual Symposium on Foundations of Computer Science*, 2015, pp. 1049–1065.

[177] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning*, 2015, pp. 1889–1897.

[178] W. Liu, F. Liu, R. Tang, B. Liao, G. Chen, and P. A. Heng, "Balancing between accuracy and fairness for interactive recommendation with reinforcement learning," in *Proceedings of the Pacific-Asia conference on Knowledge Discovery and Data Mining*, 2020.

[179] W. Liu, R. Tang, J. Li, J. Yu, H. Guo, X. He, and S. Zhang, "Field-aware probabilistic embedding neural network for CTR prediction," in *Proceedings of the ACM Conference on Recommender Systems*, 2018, pp. 412–416.

[180] W. Liu, J. Guo, N. Sonboli, R. Burke, and S. Zhang, "Personalized fairness-aware re-ranking for microlending," in *Proceedings of the ACM Conference on Recommender Systems*, 2019, pp. 467–471.

[181] J. Zhang, X. Shi, S. Zhao, and I. King, "STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 4264–4270.

[182] J. Zhang, X. Shi, I. King, and D.-Y. Yeung, "Dynamic key-value memory networks for knowledge tracing," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 765–774.

[183] C. Gao, W. Zheng, Y. Deng, D. Lo, J. Zeng, M. R. Lyu, and I. King, "Emerging app issue identification from user feedback: Experience on WeChat," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice*, 2019, pp. 279–288.

[184] C. Gao, J. Zeng, M. R. Lyu, and I. King, "Online app review analysis for identifying emerging issues," in *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 48–58.

[185] C. Gao, H. Xu, J. Hu, and Y. Zhou, "Ar-tracker: Track the dynamics of mobile apps via user review mining," in

2015 IEEE Symposium on Service-Oriented System Engineering. IEEE, 2015, pp. 284–290.

[186] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *International Conference on Machine Learning*, 2016, pp. 1329–1338.

[187] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," *arXiv preprint arXiv:1606.01540*, 2016.

[188] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár, "A multipath network for object detection," *arXiv preprint arXiv:1604.02135*, 2016.

[189] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in Neural Information Processing Systems*, 2015, pp. 1693–1701.

[190] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[191] K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with vi-

sual attention." in *International Conference on Machine Learning*, 2015, pp. 77–81.

[192] S. M. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, and G. E. Hinton, "Attend, infer, repeat: Fast scene understanding with generative models," in *Advances in Neural Information Processing Systems*, 2016, pp. 3225–3233.

[193] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, 2015, pp. 802–810.

[194] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "GaAN: Gated attention networks for learning on large and spatiotemporal graphs," *arXiv preprint arXiv:1803.07294*, 2018.

[195] D. Wierstra, A. Foerster, J. Peters, and J. Schmidhuber, "Solving deep memory POMDPs with recurrent policy gradients," in *International Conference on Artificial Neural Networks*, 2007, pp. 697–706.

[196] D. S. Fong, L. Aiello, T. W. Gardner, G. L. King, G. Blankenship, J. D. Cavallerano, F. L. Ferris, and R. Klein, "Retinopathy in diabetes," *Diabetes Care*, vol. 27, no. suppl 1, pp. s84–s87, 2004.

[197] C. Chen, Q. Dou, H. Chen, J. Qin, and P.-A. Heng, "Synergistic image and feature adaptation: Towards cross-modality domain adaptation for medical image segmentation," in *AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 865–872.

[198] Q. Dou, C. Ouyang, C. Chen, H. Chen, and P.-A. Heng, "Unsupervised cross-modality domain adaptation of ConvNets for biomedical image segmentations with adversarial loss," in *International Joint Conference on Artificial Intelligence*, 2018, pp. 691–697.

[199] Y. Zhou, Q. Dou, H. Chen, J. Qin, and P.-A. Heng, "SFCN-OPI: Detection and fine-grained classification of nuclei using sibling FCN with objectness prior interaction," in *AAAI Conference on Artificial Intelligence*, 2018.

[200] H.-T. Cheng, "Algorithms for partially observable Markov decision processes," Ph.D. dissertation, University of British Columbia, 1988.

[201] P. Poupart, "Exploiting structure to efficiently solve large scale partially observable Markov decision processes," Ph.D. dissertation, University of Toronto, 2005.

[202] T. Jaakkola, S. P. Singh, and M. I. Jordan, "Reinforcement learning algorithm for partially observable Markov decision problems," in *Advances in Neural Information Processing Systems*, 1995, pp. 345–352.

[203] F. Doshi-Velez, "The infinite partially observable Markov decision process," in *Advances in Neural Information Processing Systems*, 2009, pp. 477–485.

[204] A. R. Cassandra, M. L. Littman, and N. L. Zhang, "Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes," *arXiv preprint arXiv:1302.1525*, 2013.

[205] A. R. Cassandra, "Exact and approximate algorithms for partially observable Markov decision processes," Ph.D. dissertation, Brown University, 1998.

[206] G. E. Monahan, "State of the art - a survey of partially observable Markov decision processes: Theory, models, and algorithms," *Management Science*, vol. 28, no. 1, pp. 1–16, 1982.

[207] M. Li, P.-J. Wan, and F. F. Yao, "Tighter approximation bounds for minimum CDS in wireless ad hoc networks," in *International Symposium on Algorithms and Computation*, 2009, pp. 699–709.

[208] M. Li, J. Zhang, and Q. Zhang, "Truthful cake cutting mechanisms with externalities: Do not make them care for others too much!" in *International Joint Conference on Artificial Intelligence*, 2015.

[209] E. Anastasiadis, X. Deng, P. Krysta, M. Li, H. Qiao, and J. Zhang, "Network pollution games," *Algorithmica*, vol. 81, no. 1, pp. 124–166, 2019.

[210] B. Li, M. Li, and X. Wu, "Well-behaved online load balancing against strategic jobs," in *International Conference on Autonomous Agents and Multiagent Systems*, 2019, pp. 1243–1251.

[211] W. Sierpiński, "Sur l'équation fonctionnelle f(x+y) = f(x) + f(y)," *Fundamenta Mathematicae*, vol. 1, no. 1, pp. 116–122, 1920.

[212] W. Sierpiński, "Sur les fonctions convexes mesurables," *Fundamenta Mathematicae*, vol. 1, no. 1, pp. 125–128, 1920.

[213] O. Dovgoshey, O. Martio, V. Ryazanov, and M. Vuorinen, "The Cantor function," *Expositiones Mathematicae*, vol. 24, no. 1, pp. 1–37, 2006.

[214] B. B. Mandelbrot, "Self-affine fractals and fractal dimension," *Physica Scripta*, vol. 32, no. 4, p. 257, 1985.

[215] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, 1987.

[216] M. L. Littman, T. L. Dean, and L. P. Kaelbling, "On the complexity of solving Markov decision problems," in *Pro-*

ceedings of the 11th Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., 1995, pp. 394–402.

[217] E. Thelen and L. B. Smith, "Dynamic systems theories," *Handbook of Child Psychology*, 1998.

[218] V. Chockalingam, "The role of interest in prediction and control," The Tea Time Talks, 2019, the plot of the empirical optimal value function of the Mountain Car problem first appears at 6:35. Some follow-up discussions start at 33:20. [Online]. Available: https://youtu.be/aFXdpCDAG2g?t=395

[219] W. S. Lee, N. Rong, and D. Hsu, "What makes some POMDP problems easy to approximate?" in *Advances in Neural Information Processing Systems*, 2008, pp. 689–696.

[220] C. Lusena, J. Goldsmith, and M. Mundhenk, "Nonapproximability results for partially observable Markov decision processes," *Journal of Artificial Intelligence Research*, vol. 14, pp. 83–103, 2001.

[221] Y. V. Tikhonov, "On the rate of approximation of singular functions by step functions," *Mathematical Notes*, vol. 95, no. 3-4, pp. 530–543, 2014.

[222] B. C. Csáji, "Approximation with artificial neural networks," Master's thesis, Etvs Lornd University, 2001.

[223] J. Levesley, C. Salp, and S. L. Velani, "On a problem of K. Mahler: Diophantine approximation and Cantor sets," *Mathematische Annalen*, vol. 338, no. 1, pp. 97–118, 2007.

[224] S. S. Gu, T. Lillicrap, R. E. Turner, Z. Ghahramani, B. Schölkopf, and S. Levine, "Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 3846–3855.

[225] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa, "Learning continuous control policies by stochastic value gradients," in *Advances in Neural Information Processing Systems*, 2015, pp. 2944–2952.

[226] M. Fairbank and E. Alonso, "Value-gradient learning," in *The 2012 International Joint Conference on Neural Networks*. IEEE, 2012, pp. 1–8.

[227] M. Fairbank, "Reinforcement learning by value gradients," *arXiv preprint arXiv:0803.3539*, 2008.

[228] Y. Pan, H. Yao, A.-m. Farahmand, and M. White, "Hill climbing on value estimates for search-control in Dyna," *arXiv preprint arXiv:1906.07791*, 2019.

[229] S. Lim, A. Joseph, L. Le, Y. Pan, and M. White, "Actor-expert: A framework for using action-value methods in continuous action spaces," *arXiv preprint arXiv:1810.09103*, 2018.

[230] L. Baird, "Residual algorithms: Reinforcement learning with function approximation," in *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 30–37.

[231] S. Zhang, "Python implementation of Reinforcement learning: An introduction," 2019. [Online]. Available: https://github.com/ShangtongZhang/reinforcement-learning-an-introduction

[232] M. Pelling, "Formulae for the arc-length of a curve in R^N," *The American Mathematical Monthly*, vol. 84, no. 6, pp. 465–467, 1977.

[233] T. Kamihigashi and C. Le Van, "Necessary and sufficient conditions for a solution of the Bellman equation to be the value function: A general principle," *Documents de travail du Centre dÉconomie de la Sorbonne 2015.07*, 2015, ISSN: 1955-611X.

[234] P. Latham, "Bellman's equation has a unique solution," 2008. [Online]. Available: http://www.gatsby.ucl.ac.uk/~mandana/RLRG/bellman_converges.pdf

[235] M. E. Harmon and L. C. Baird III, "Spurious solutions to the Bellman equation," *Technical Report, Wright-Patterson Air Force Base Ohio: Wright Laboratory, WL-TR-96-'To Be Assigned'*, 1996.

[236] T. J. Jech, *The Axiom of Choice*. Courier Corporation, 2008.