A Theory of Model Selection in Reinforcement Learning

by

Nan Jiang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2017

Doctoral Committee:

Professor Satinder Singh Baveja, Chair
Assistant Professor Jacob Abernethy
Professor Michael L. Littman, Brown University
Professor Susan A. Murphy
Assistant Professor Ambuj Tewari

Nan Jiang

nanjiang@umich.edu

ORCID iD: 0000-0002-9526-6148

Acknowledgments

I would like to give warm thanks to Satinder Singh, my advisor, who spent a boundless amount of time training me to become a researcher, for his all-around advice in research methodology, writing and presentation, and career; to Jake Abernethy, Michael Littman, Susan Murphy, and Ambuj Tewari for serving on my thesis committee and providing valuable insights into this document; to Alex Kulesza, who has always been a great collaborator, for his generous help and useful advice on everything—you have literally changed my PhD life; to Lihong Li, who introduced me to people at ICML when I felt lost as a newcomer and calmed my anxiety with encouragement during job search—it is my great fortune to be your intern and friend; to Alekh Agarwal, Akshay Krishnamurthy, John Langford, and Rob Schapire for the great collaboration we had (and will continue) at MSR; to Emma Brunskill, Tim Mann, Marek Petrik, Joelle Pineau, and Philip Thomas for making the RL community feel like a warm family; to Rick Lewis, who advised me on my early projects, for bringing a relaxed atmosphere to the meetings; to Clay Scott, whose seminar course is my favorite ever, for introducing me into statistical learning theory and showing me great teaching practices; to Kimberly Mann and other staff in CSE for making everything as smooth as possible; to Jesh Bratman, Rob Cohn, Michael Shvartsman, Monica Eboli, Xiaoxiao Guo, Sean Newman, and other students in the RL lab, with whom I shared joys and hardships of being a PhD student; to Qi Chen and Xu Zhang for being long-time friends and my matchmakers; to Xin Rong, whose endless passion always inspired me—I could not name the grief when you left us; to my parents, my aunt, and my grandfather for their love and constant support in my academic career; and especially to Iris, my dear sunshine, for bringing happiness and balance to my life—the PhD journey has never been a pain when I know that I can return to the place where you are.

TABLE OF CONTENTS

Acknowledgments . . . . **ii**

List of Figures . . . . **vi**

List of Tables . . . . **vii**

Abstract . . . . **viii**

Chapter

1 Introduction **1**

1.1 Thesis Statement 3
1.2 Contributions 3

2 Background **6**

2.1 Markov Decision Processes 6
2.1.1 Interaction protocol 6
2.1.2 Policy and value 7
2.1.3 Bellman equations for policy evaluation 8
2.1.4 Bellman optimality equations 9
2.1.5 Notes on the MDP setup 10
2.2 Planning in MDPs 12
2.2.1 Policy Iteration 12
2.2.2 Value Iteration 13
2.3 Reinforcement Learning in MDPs 15
2.3.1 Data collection protocols 16
2.3.2 Performance measures 18
2.3.3 Monte-Carlo methods 20
2.3.4 Tabular methods 22
2.3.5 State abstractions 23

3 Dependence of Effective Planning Horizon on Data Size **27**

3.1 Introduction 27
3.2 Preliminaries 29
3.3 Planning Horizon and A Complexity Measure 30
3.3.1 A counting complexity measure 31
3.3.2 Planning loss bound 35

3.3.3 Handling uncertain rewards 36
3.4 Rademacher Complexity Bound 38
3.4.1 An empirical Rademacher bound 40
3.5 Experimental Results 43
3.5.1 Optimal planning depth in UCT 46
3.5.2 Selecting γ via cross-validation 47
3.6 Related Work and Discussions 48
3.7 Proof of Theorem 3.2 50
3.8 Proof of Theorem 3.4 53
3.9 Proof of Theorem 3.5 54
3.10 Proof of Theorem 3.7 55

4 Doubly Robust Off-policy Evaluation 58

4.1 Introduction 58
4.2 Problem Statement and Existing Solutions 60
4.2.1 Off-policy value evaluation 60
4.2.2 Doubly robust estimator for contextual bandits 63
4.3 DR estimator for the sequential setting 64
4.3.1 The estimator 64
4.3.2 Variance analysis 64
4.3.3 Confidence intervals 65
4.3.4 An extension 66
4.4 Hardness of Off-policy Value Evaluation 66
4.5 Experiments 70
4.5.1 Comparison of Mean Squared Errors 70
4.5.2 Application to safe policy improvement 76
4.6 Related Work and Discussions 77
4.7 Proof of Theorem 4.1 78
4.8 Bias of DR-v2 79
4.9 Cramer-Rao Bound for Discrete DAG MDPs 80

5 Adaptive Selection of State Abstraction 85

5.1 Introduction 85
5.2 Preliminaries 87
5.2.1 Abstractions for model-based RL 87
5.2.2 Problem statement 88
5.3 Bounding the Loss of a Single Abstraction 88
5.4 Proposed Algorithm and Theoretical Analysis 91
5.4.1 Intuition of the algorithm 91
5.4.2 Theoretical analysis 93
5.4.3 Extension to arbitrary-size candidate sets 96
5.5 Related Work and Discussions 98
5.5.1 Hypothesis test based algorithms 98
5.5.2 Reduction to off-policy evaluation 100
5.5.3 The online setting 100

5.6 Proof of Theorem 5.1 101
5.7 Proof of Theorem 5.2 103
5.8 Proof of Theorem 5.8 105

6 Repeated Inverse Reinforcement Learning 108

6.1 Introduction 108
6.2 Problem Setup 110
6.2.1 Notations 110
6.2.2 Repeated Inverse RL framework 111
6.3 The Challenge of Identifying Rewards 112
6.4 Agent Chooses the Tasks 113
6.4.1 Omnipotent identification algorithm 113
6.5 Nature Chooses the Tasks 114
6.5.1 The linear bandit setting 115
6.5.2 Ellipsoid Algorithm for Repeated Inverse RL 117
6.5.3 Lower bound 118
6.5.4 On identification when nature chooses tasks 120
6.6 Working with Trajectories 123
6.7 Related Work and Discussions 125
6.7.1 Inverse RL, AI safety, and value alignment 125
6.7.2 Connections to online learning and bandit literature 127
6.7.3 Alternative formulation using constraints 129
6.8 Proof of Lemma 6.7 130
6.9 Tightness of Theorem 6.9 130
6.10 A Technical Note on Theorem 6.11 131
6.11 Proof of Theorem 6.6 131
6.12 Proof of Theorem 6.11 133

7 Conclusion 135

7.1 Discussions and Future Research Possibilities 136

Bibliography 138

LIST OF FIGURES

3.1 Training and test losses as a function of planning horizon 33
3.2 Empirical illustration of the Rademacher complexity and the guidance discount factor γ. 41
3.3 Planning loss as a function of γ for a single MDP 44
3.4 Optimal guidance discount factor increases with data size 44
3.5 Optimal planning horizon increases with the number of sample trajectories in UCT 45
3.6 Selecting γ by cross-validation 47

4.1 Comparison of off-policy evaluation methods on Mountain Car 72
4.2 Comparison of off-policy evaluation methods on Sailing 73
4.3 Comparison of off-policy evaluation methods on the KDD cup donation dataset 75
4.4 Safe policy improvement in Mountain Car 76

5.1 Illustration of the behavior of Algorithm 1 in different regimes of data size. 90
5.2 Intuition for the abstraction selection algorithm 93

6.1 Illustration of the protocol when human and agent communicate using trajectories in Repeated Inverse RL 124

LIST OF TABLES

3.1 An analogy between empirical risk minimization and certainty-equivalent planning. 31

5.1 Comparison of algorithms that can be applied to abstraction selection . 99

ABSTRACT

Reinforcement Learning (RL) is a machine learning paradigm where an agent learns to accomplish sequential decision-making tasks from experience. Applications of RL are found in robotics and control, dialog systems, medical treatment, etc. Despite the generality of the framework, most empirical successes of RL to-date are restricted to simulated environments, where hyperparameters are tuned by trial and error using large amounts of data. In contrast, collecting data with active intervention in the real world can be costly, time-consuming, and sometimes unsafe. Choosing the hyperparameters and understanding their effects in face of these data limitations, i.e., *model selection*, is an important yet open direction that we need to study to enable such applications of RL, which is the main theme of this thesis.

More concretely, this thesis presents theoretical results that improve our understanding of 3 hyperparameters in RL: planning horizon, state representation (abstraction), and reward function. The 1st part of the thesis focuses on the interplay between planning horizon and limited amount of data, and establishes a formal explanation for how a long planning horizon can cause overfitting. The 2nd part considers the problem of choosing the right state abstraction using limited batch data; I show that cross-validation type methods require importance sampling and suffer from exponential variance, and a novel regularization-based algorithm enjoys an oracle-like property. The 3rd part investigates reward misspecification and tries to resolve it by leveraging expert demonstrations, which is inspired by AI safety concerns and bears close connections to inverse reinforcement learning.

A recurring theme of the thesis is the deployment of formulations and techniques

from other machine learning theory (mostly statistical learning theory): the planning horizon work explains the overfitting phenomenon by making a formal analogy to empirical risk minimization and by proving planning loss bounds that are similar to generalization error bounds; the main result in the abstraction selection work takes the form of an oracle inequality, which is a concept from structural risk minimization for model selection in supervised learning; the inverse RL work provides a mistake-bound type analysis under arbitrarily chosen environments, which can be viewed as a form of no-regret learning. Overall, by borrowing ideas from mature theories of machine learning, we can develop analogies for RL that allow us to better understand the impact of hyperparameters, and develop algorithms that automatically set them in an effective manner.

CHAPTER 1

Introduction

Reinforcement Learning (RL) is a subfield of machine learning that studies how an agent can learn to make sequential decisions in environments with unknown dynamics. It provides a general and unified framework that captures many important applications of Artificial Intelligence (AI), including news recommendation and online advertising, dialog systems, self-driving cars, robots for daily life, adaptive medical treatments, and so on [Singh et al., 2002, Ng et al., 2003, Li et al., 2010, Lei et al., 2012].

Recently, empirical successes have demonstrated that RL methods can conquer video and board game domains with large observation or state spaces, typically by incorporating sophisticated function approximation techniques [Mnih et al., 2015, Silver et al., 2016]. While these successes have drawn wide attention to RL research and inspired the design of new benchmarks [Brockman et al., 2016, Johnson et al., 2016], the algorithmic advances in game environments have not quite translated into successes in applications where simulators of high fidelity are not available ("*non-simulator*" applications). This situation arguably arises from the fact that game environments possess many nice properties that cannot be expected in non-simulator applications: in a simulator, data can be generated indefinitely up to computational limits (e.g., AlphaGo generated 30 million self-play games [Silver et al., 2016]), taking arbitrarily bad actions has no real-world effects (e.g., crashing a car in a driving game is nothing compared to crashing a real car), and there is often a well-defined objective (e.g., winning or achieving high score). Understanding and developing RL algorithms under these limitations is important and challenging, and this thesis presents a set of theoretical efforts towards this goal.

As an example of something that is almost trivially simple in simulators while highly difficult in non-simulator applications, consider the problem of assessing the performance of an RL solution (i.e., estimating the value of a policy). As will be introduced in Chapter 2, one of the most effective methods is Monte-Carlo policy evaluation, which directly deploys the solution in the real system, runs it for a while, and estimates the value from the observed returns. This strategy proves to be useful and successful in simulators, and forms the basis of model selection in empirical RL today: if a practitioner is uncertain about which hyperparameters to use in the algorithm, he/she can try different hyperparameters, obtain the output policies, evaluate each of them by the Monte-Carlo procedure, and choose the one with the best performance. In non-simulator applications, however, this approach is often infeasible for a number of reasons: first, sometimes we have concerns about the policy's negative consequences, and we want to estimate its value before actually deploying it; second, for hyperparameter tuning, we would need a separate set of Monte-Carlo trajectories for each possible hyperparameter setting, which can be unrealistic when data collection is expensive (as is often the case in e.g., medical domains [Murphy et al., 2001]).

Training machine learning algorithms and tuning hyperparameters under data constraints, however, are neither new nor unique problems to RL. In the supervised learning context, even beginners are taught to partition their dataset into training / validation / test sets, train their algorithm on the training set, tune hyperparameters on the validation set (or by cross-validation) to avoid overfitting, and report final performance on the test set. When the set of hyperparameters is large or infinite (e.g., for decision trees), cross-validation-type methods may exceed the statistical capacity of the validation set, and regularization techniques become handy for model selection [Scott, 2004]. Besides practical techniques, statistical learning theory also provides deep mathematical understanding of the behavior of supervised learning algorithms under limited data and explain how overfitting happens and why regularization can work [Vapnik, 1992, 1998].

While we will borrow these experiences and theories from supervised learning into RL, RL also faces some unique challenges that are not exhibited in other machine learning paradigms. For example, cross-validation is much easier in supervised learning than in RL, as counter-factual reasoning is straight-forward in supervised learning (i.e., it is easy to answer what-if questions such as "would my prediction be correct if I were to use a different classifier?") yet nontrivial in RL (i.e., it is very hard to predict "what would this trajectory look like if I were to follow a different policy?"). Furthermore, model selection in supervised learning often focuses on the choice of function class or regularization parameters. In RL, there is a richer space of possibilities that are explored much less, including horizon, state representation, and reward function, some of which are traditionally not viewed as hyperparameters. Regarding this situation, this thesis presents efforts towards a theory for model selection in reinforcement learning.

1.1 Thesis Statement

Using concepts and techniques from statistical learning theory, we can develop analogies for reinforcement learning that allow us to better understand the impact of hyperparameters in RL, including planning horizon, state representation, and reward function, and design algorithms that automatically learn them in a sample-efficient manner.

1.2 Contributions

Here we give an overview of the thesis and summarize the contributions. Chapter 2 introduces preliminaries of reinforcement learning. Chapters 3, 4, 5, and 6 contain new contributions, which are introduced below. Finally, Chapter 7 concludes the thesis and suggests future research directions.

Dependence of Effective Planning Horizon on Data Size (Chapter 3)
In RL, a discount factor specifies how far an agent should look ahead into the future, and is closely related to the notion of planning horizon. Despite its importance, existing literature provided limited understanding of its role in RL algorithms, especially in the realistic setting of insufficient data. In this chapter, we show a perhaps surprising result that with a limited amount of data, an agent can compute a *better* policy by using a discount factor in the algorithm that is smaller than the groundtruth specified in the problem definition. An explanation for this phenomenon is provided based on principles of learning theory: that a large discount factor causes *overfitting*. The statement is established theoretically by making an analogy between supervised learning (where we search over hypotheses) and reinforcement learning (where we search over policies), and showing that a small discount factor can control the effective size of the policy space and hence avoid overfitting. This chapter is based on joint work with Alex Kulesza, Satinder Singh, and Richard Lewis [Jiang et al., 2015b].

Doubly Robust Off-policy Evaluation (Chapter 4)

This chapter focuses on the problem of estimating the value of a policy using data generated using a different one, i.e., the *off-policy* value evaluation problem. Studying this problem is central to understanding model selection in RL: if we have an accurate off-policy evaluation estimator, we can solve the model selection problem by using a cross-validation-like procedure. This chapter develops a new estimator for off-policy evaluation, which is generalized from its bandit version [Dudík et al., 2011] and improves the state of the art. At the same time, we also provide a hardness result, showing that without prior knowledge, any unbiased estimator suffers a worst-case variance that is exponential in the problem's horizon when the size of the state space is not constrained. The result confirms that reducing model selection to cross-validation via off-policy evaluation may not be effective when a model-based approach is not available. This chapter is based on joint work with Lihong Li [Jiang and Li, 2016].

Adaptive Selection of State Abstraction (Chapter 5)

This chapter discusses how to choose a good state abstraction from a candidate set based on a limited batch dataset. This is the situation where reduction to off-policy evaluation is not effective, thus we turn to regularization-typed methods. We consider the setting where candidate abstract state representations are finite aggregations of states and they have a nested structure, and show that a statistical test based algorithm adaptively chooses a good representation based on data, and enjoys a performance guarantee nearly as good as that of an "oracle" with extra access to the discrepancy information of each abstraction. This chapter is based on joint work with Alex Kulesza and Satinder Singh [Jiang et al., 2015a].

Repeated Inverse Reinforcement Learning (Chapter 6)

In the previous chapters, we adopt the standard RL formulation and take it for granted that rewards are well-defined and revealed to the agent as part of the dataset. In some realistic situations, however, it has long been recognized that specifying a detailed and comprehensive reward function that is well aligned with human interest can be difficult, and this has grown into a serious concern on the threat of future AI to humanity [Bostrom, 2003, Russell et al., 2015, Amodei et al., 2016]. In this chapter we tackle this meta-level problem of learning reward functions. We start from the Inverse RL framework proposed by Ng and Russell [2000], which tries to recover reward function of human behavior who are assumed to act in a way that maximizes the true reward function. We propose a novel framework that allows repeated interactions between the agent and the environments, which gives hints towards resolving the fundamental unidentifiability issue of Inverse RL. This chapter is based on joint work with Kareem Amin and Satinder Singh [Amin et al., 2017].

CHAPTER 2

Background

2.1 Markov Decision Processes

In reinforcement learning, the interactions between the agent and the environment are often described by a Markov Decision Process (MDP) [Puterman, 1994], specified by:

- State space $\mathcal{S}$. This thesis only considers finite state spaces.
- Action space $\mathcal{A}$. This thesis only considers finite action spaces.
- Transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is the space of probability distributions over $\mathcal{S}$ (i.e., the probability simplex). $P(s'|s, a)$ is the probability of transitioning into state s' upon taking action a in state s'.
- Reward function $R : \mathcal{S} \times \mathcal{A} \to [0, R_{\max}]$, where $R_{\max} > 0$ is a constant. $R(s, a)$ is the immediate reward associated with taking action a in state s.
- Discount factor $\gamma \in [0, 1)$, which defines a horizon for the problem.

2.1.1 Interaction protocol

In a given MDP $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$, the agent interacts with the environment according to the following protocol: the agent starts at some state s_1; at each time step $t = 1, 2, \ldots$, the agent takes an action $a_t \in \mathcal{A}$, obtains the immediate reward $r_t = R(s_t, a_t)$, and observes the next state s_{t+1} sampled from $P(\,\cdot\,|s_t, a_t)$, or $s_{t+1} \sim P(\,\cdot\,|s_t, a_t)$. The interaction record

$$\tau = (s_1, a_1, r_1, s_2, \ldots, s_{H+1})$$

is called a *trajectory* of length H.

In some situations, it is necessary to specify how the initial state s_1 is generated. In this thesis, we consider s_1 sampled from an initial distribution $\mu \in \Delta(\mathcal{S})$. When μ is of importance to the discussion, we include it as part of the MDP definition, and write $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma, \mu)$.

2.1.2 Policy and value

A (deterministic and stationary) policy $\pi : \mathcal{S} \to \mathcal{A}$ specifies a decision-making strategy in which the agent chooses actions adaptively based on the current state, i.e., $a_t = \pi(s_t)$. More generally, the agent may also choose actions according to a stochastic policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, and with a slight abuse of notation we write $a_t \sim \pi(\,\cdot\,|s_t)$. A deterministic policy is its special case when $\pi(\,\cdot\,|s)$ is a point mass for all $s \in \mathcal{S}$.

The goal of the agent is to choose a policy π to maximize the expected discounted sum of rewards, or *value*:

$$\mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \;\big|\; \pi, s_1\Big]. \tag{2.1}$$

The expectation is with respect to the randomness of the trajectory, that is, the randomness in state transitions and the stochasticity of π. Notice that, since r_t is non-negative and upper bounded by $R_{\max}$, we have

$$0 \le \sum_{t=1}^{\infty} \gamma^{t-1} r_t \le \sum_{t=1}^{\infty} \gamma^{t-1} R_{\max} = \frac{R_{\max}}{1-\gamma}. \tag{2.2}$$

Hence, the discounted sum of rewards (or the discounted **return**) along any actual trajectory is always bounded in range $[0, \frac{R_{\max}}{1-\gamma}]$, and so is its expectation of any form. This fact will be important when we later analyze the error propagation of planning and learning algorithms.

Note that for a fixed policy, its value may differ for different choice of s_1, and we define the value function $V_M^\pi : \mathcal{S} \to \mathbb{R}$ as

$$V_M^\pi(s) = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \;\big|\; \pi, s_1 = s\Big],$$

which is the value obtained by following policy π starting at state s. Similarly we define the action-value (or Q-value) function $Q_M^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as

$$Q_M^\pi(s,a) = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \;\Big|\; \pi, s_1 = s, a_1 = a\Big].$$

Henceforth, the dependence of any notation on M will be made implicit whenever it is clear from context.

2.1.3 Bellman equations for policy evaluation

Based on the principles of dynamic programming, V^π and Q^π can be computed using the following *Bellman equations for policy evaluation*: $\forall s \in \mathcal{S}, a \in \mathcal{A}$,

$$\begin{aligned} V^\pi(s) &= Q^\pi(s, \pi(s)). \\ Q^\pi(s,a) &= R(s,a) + \gamma \mathbb{E}_{s' \sim P(\,\cdot\,|s,a)}\big[V^\pi(s')\big]. \end{aligned} \tag{2.3}$$

In $Q^\pi(s, \pi(s))$ we treat π as a deterministic policy for brevity, and for stochastic policies this shorthand should be interpreted as $\mathbb{E}_{a \sim \pi(\,\cdot\,|s)}[Q^\pi(s,a)]$.

Since $\mathcal{S}$ is assumed to be finite, upon fixing an arbitrary order of states (and actions), we can treat V^π and any distribution over $\mathcal{S}$ as vectors in $\mathbb{R}^{|\mathcal{S}|}$, and R and Q^π as vectors in $\mathbb{R}^{|\mathcal{S} \times \mathcal{A}|}$. This is particularly helpful as we can rewrite Equation 2.3 in an matrix-vector form and derive an analytical solution for V^π using linear algebra as below.

Define P^π as the transition matrix for policy π with dimension $|\mathcal{S}| \times |\mathcal{S}|$, whose (s, s')-th entry is

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

In fact, this matrix describes a Markov chain induced by MDP M and policy π. Its s-th row is the distribution over next-states upon taking actions according to π at state s, which we also write as $[P(\,\cdot\,|s,\pi)]^\top$.

Similarly define R^π as the reward vector for policy π with dimension $|\mathcal{S}| \times 1$, whose s-th entry is

$$[R^\pi]_s = \mathbb{E}_{a \sim \pi(\,\cdot\,|s)}[R(s,a)].$$

Then from Equation 2.3 we have

$$\begin{aligned}[V^\pi]_s = Q^\pi(s,\pi(s)) &= [R^\pi]_s + \gamma \mathbb{E}_{a\sim\pi(\,\cdot\,|s)}\mathbb{E}_{s'\sim P(\,\cdot\,|s,a)}\left[V^\pi(s')\right] \\ &= [R^\pi]_s + \gamma \mathbb{E}_{s'\sim P(\,\cdot\,|s,\pi)}\left[V^\pi(s')\right] \\ &= [R^\pi]_s + \gamma \langle P(\,\cdot\,|s,\pi), V^\pi\rangle,\end{aligned}$$

where $\langle\,\cdot\,,\,\cdot\,\rangle$ is dot product. Since this equation holds for every $s \in \mathcal{S}$, we have

$$V^\pi = R^\pi + \gamma P^\pi V^\pi \quad \Rightarrow \quad (\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi)V^\pi = R^\pi,$$

where $\mathbf{I}_{|\mathcal{S}|}$ is the identity matrix. Now we notice that matrix $(\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi)$ is always invertible. In fact, for any non-zero vector $x \in \mathbb{R}^{|\mathcal{S}|}$,

$$\begin{aligned}&\left\|(\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi)x\right\|_\infty = \|x - \gamma P^\pi x\|_\infty \\ &\geq \|x\|_\infty - \gamma\|P^\pi x\|_\infty && \text{(triangular inequality for norms)} \\ &\geq \|x\|_\infty - \gamma\|x\|_\infty && \text{(each element of } P^\pi x \text{ is a convex average of } x) \\ &= (1-\gamma)\|x\|_\infty > 0 && (\gamma < 1, x \neq \mathbf{0})\end{aligned}$$

So we can conclude that

$$V^\pi = (\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi)^{-1} R^\pi. \tag{2.4}$$

State occupancy

When the reward function only depends on the current state, i.e., $R(s,a) = R(s)$, R^π is independent of π, and Equation 2.4 exhibits an interesting structure: implies that the value of a policy is *linear* in rewards, and the rows of the matrix $(\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi)^{-1}$ give the linear coefficients that depend on the initial state. Such coefficients, often represented as a vector, are called *discounted state occupancy* (or state occupancy for short). It can be interpreted as the expected number of times that each state is visited along a trajectory, where later visits are discounted more heavily.[1]

2.1.4 Bellman optimality equations

There always exists a stationary and deterministic policy that simultaneously maximizes $V^\pi(s)$ for all $s \in \mathcal{S}$ and maximizes $Q^\pi(s,a)$ for all $s \in \mathcal{S}, a \in \mathcal{A}$ [Puterman,

[1] When rewards depend on actions, we can define discounted state-action occupancy in a similar way and recover the fact that value is linear in reward.

1994], and we denote this *optimal policy* as $\pi^\star_M$ (or $\pi^\star$). We use $V^\star$ as a shorthand for $V^{\pi^\star}$, and $Q^\star$ similarly.

$V^\star$ and $Q^\star$ satisfy the following set of *Bellman optimality equations* [Bellman, 1956]: $\forall s \in \mathcal{S}, a \in \mathcal{A}$,

$$\begin{aligned} V^\star(s) &= \max_{a \in \mathcal{A}} Q^\star(s,a). \\ Q^\star(s,a) &= R(s,a) + \gamma \mathbb{E}_{s' \sim P(\,\cdot\,|s,a)}\big[V^\star(s')\big]. \end{aligned} \tag{2.5}$$

Once we have $Q^\star$, we can obtain $\pi^\star$ by choosing actions greedily (with arbitrary tie-breaking mechanisms):

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} Q^\star(s,a), \ \forall s \in \mathcal{S}.$$

We use shorthand π_Q to denote the procedure of turning a Q-value function into its greedy policy, and the above equation can be written as

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

To facilitate future discussions, define the *Bellman optimality operator* $B_M : \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$ (or simply B) as follows: when applied to some vector $Q \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$,

$$(BQ)(s,a) = R(s,a) + \gamma \langle P(\,\cdot\,|s,a), \max_{a \in \mathcal{A}} Q(\cdot, a) \rangle. \tag{2.6}$$

This allows us to rewrite Equation 2.5 in the following concise form, which implies that $Q^\star$ is the fixed point of the operator B:

$$Q^\star = BQ^\star.$$

2.1.5 Notes on the MDP setup

Before moving on, we make notes on our setup of MDP and discuss alternative setups considered in the literature.

Finite horizon and episodic setting

Our definition of value (Equation 2.1) corresponds to the infinite-horizon discounted setting of MDPs. Popular alternative choices include the finite-horizon undiscounted setting (actual return of a trajectory is $\sum_{t=1}^H r_t$ with some finite horizon $H < \infty$) and the infinite-horizon average reward setting (return is

$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} r_t$). The latter case often requires additional conditions on the transition dynamics (such as ergodicity) so that values can be well-defined [Sutton and Barto, 1998], and will not be discussed in this thesis.

The finite-horizon undiscounted (or simply finite-horizon) setting can be emulated using the discounted setting by augmenting the state space. Suppose we have an MDP M with finite horizon H. Define a new MDP $\tilde{M} = (\tilde{\mathcal{S}}, \mathcal{A}, \tilde{P}, \tilde{R}, \gamma)$ such that $\tilde{\mathcal{S}} = \mathcal{S} \times [H] \bigcup \{s_{\text{absorbing}}\}$ ($[H] = \{1, \dots, H\}$). Essentially we make H copies of the state space and organize them in levels, with an additional absorbing state $s_{\text{absorbing}}$ where all actions transition to itself and yield 0 reward. There is only non-zero transition probability from states at level h to states at level $h+1$ with $\tilde{P}((s', h+1) \,\big|\, (s,h), a) = P(s'|s,a)$, and states at the last level (s, H) transition to $s_{\text{absorbing}}$ deterministically. Finally we let $\tilde{R}((s,h),a) = R(s,a)$ and $\gamma = 1$. (In general $\gamma = 1$ may lead to infinite value, but here the agent always loops in the absorbing state after H steps and gets finite total rewards.) The optimal policy for finite-horizon MDPs is generally non-stationary, that is, it depends on both s and the time step h.

The MDP described in the construction above can be viewed as an example of **episodic** tasks: the environment deterministically transitions into an absorbing state after a fixed number of time steps. The absorbing state often corresponds to the notion of termination, and many problems are naturally modeled using an episodic formulation, including board games (a game terminates once the winner is determined) and dialog systems (a session terminates when the conversation is concluded).

Stochastic rewards

Our setup assumes that reward r_t only depends on s_t and a_t deterministically. In general, r_t may also depend on s_{t+1} and contain additional noise that is independent from state transitions as well as reward noise in other time steps. As special cases, in inverse RL literature [Ng and Russell, 2000, Abbeel and Ng, 2004], reward only depends on state, and in contextual bandit literature [Langford and Zhang, 2008], reward depends on the state (or *context* in bandit terminologies) and action but has additional independent noise.

All these setups are equivalent to having a state-action reward with regard to the policy values: define $R(s,a) = \mathbb{E}[r_t | s_t = s, a_t = a]$ where s_{t+1} and the independent noise are marginalized out. The value functions V^π and Q^π for any π remains the same when we substitute in this equivalent reward function. That said, reward randomness may introduce additional noise in the sample trajectories and affect learning efficiency.

Negative rewards

Our setup assumes that $r_t \in [0, R_{\max}]$. This is without loss of generality in the infinite-horizon discounted setting: for any constant $c > 0$, a reward function $R \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$ is equivalent to $R + c\mathbf{1}_{|\mathcal{S}\times\mathcal{A}|}$, as adding c units of reward to each state-action pair simply adds a constant "background" value of $c/(1-\gamma)$ to the value of all policies for all initial states. Therefore, when the rewards may be negative but still have bounded range, e.g., $R(s,a) \in [-a, b]$ with $a, b > 0$, we can add a constant offset $c = a$ to the reward function and define $R_{\max} = a + b$, so that after adding the offset the reward lies in $[0, R_{\max}]$.

The fact that reward function is invariant under constant offset has important implications in inverse reinforcement learning, and will be discussed in detail in Chapter 6.

2.2 Planning in MDPs

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

2.2.1 Policy Iteration

The policy iteration algorithm starts from an arbitrary policy $\pi_0 = \pi$, and repeat the following iterative procedure: for $t = 1, 2, \ldots$

$$\pi_t = \pi_{Q^{\pi_{t-1}}}.$$

Here t is the iteration index and should not be confused with the time step in the MDP. Essentially, in each iteration we compute the Q-value function of π_{t-1} (e.g., using the analytical form given in Equation 2.4), and then compute the greedy policy for the next iteration. The first step is often called policy evaluation, and the second step is often called policy improvement.

The policy value is guaranteed to improve monotonically over all states until $\pi^\star$ is found [Puterman, 1994]. More precisely, $Q^{\pi_t}(s,a) \geq Q^{\pi_{t-1}}(s,a)$ holds for all $t \geq 1$

and $s \in \mathcal{S}, a \in \mathcal{A}$, and in at least one (s, a) pair the improvement is strictly positive. Therefore, the termination criterion for the algorithm is $Q^{\pi_t} = Q^{\pi_{t-1}}$. Since we are only searching over stationary and deterministic policies, and a new policy that is different from all previous ones is found every iteration, the algorithm is guaranteed to terminate in $|\mathcal{A}|^{|\mathcal{S}|}$ iterations.

2.2.2 Value Iteration

Value Iteration computes a series of Q-value functions to directly approximate $Q^\star$, without going back and forth between value functions and policies as in Policy Iteration. Let $Q^{\star,0}$ be the initial value function, often initialized to $\mathbf{0}_{|\mathcal{S}\times\mathcal{A}|}$. The algorithm computes $Q^{\star,t}$ for $t = 1, 2, \ldots, H$ in the following manner:

$$Q^{\star,t} = B\, Q^{\star,t-1}. \tag{2.7}$$

Recall that B is the Bellman optimality operator defined in Equation 2.6.

We provide two different interpretations to understand the behavior of the algorithm, and use this opportunity to introduce some mathematical results that will be repeatedly used in later chapters. Both interpretations will lead to the same bound on $\|Q^{\star,H} - Q^\star\|_\infty$ as a function of H. If H is large enough, we can guarantee that $Q^{\star,H}$ is sufficiently close to $Q^\star$, and the following result bounds the suboptimality (or loss) of acting greedily with respect to an approximate Q-value function:

Lemma 2.1 ([Singh and Yee, 1994]). $\|V^\star - V^{\pi_Q}\|_\infty \leq \dfrac{2\|Q - Q^\star\|_\infty}{1-\gamma}$.

Bounding $\|Q^{\star,H} - Q^\star\|_\infty$: the fixed point interpretation

Value Iteration can be viewed as solving for the fixed point of B, i.e., $Q^\star = BQ^\star$. The convergence of such iterative methods is typically analyzed by examining the *contraction* of the operator. In fact, the Bellman optimality operator is a γ-contraction under ℓ_∞ norm [Puterman, 1994]: for any $Q, Q' \in \mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}$

$$\|BQ - BQ'\|_\infty \leq \gamma \|Q - Q'\|_\infty . \tag{2.8}$$

To verify, we expand the definition of B for each entry of $(BQ - BQ')$:

$$\begin{aligned}\big|[BQ - BQ']_{s,a}\big| &= \big|R(s,a) + \gamma\langle P(\cdot\,|s,a), V_Q\rangle - R(s,a) - \gamma\langle P(\cdot\,|s,a), V_{Q'}\rangle\big| \\ &\leq \gamma\big|\langle P(\cdot\,|s,a), V_Q - V_{Q'}\rangle\big| \leq \|V_Q - V_{Q'}\|_\infty \leq \|Q - Q'\|_\infty .\end{aligned}$$

The last step uses the fact that $\forall s \in \mathcal{S}$, $|V_Q(s) - V_{Q'}(s)| = \max_{a\in\mathcal{A}} |Q(s,a) - Q'(s,a)|$. The easiest way to see this is to assume $V_Q(s) > V_{Q'}(s)$ (the other direction is symmetric), and let a_0 be the greedy action for Q at s. Then

$$|V_Q(s) - V_{Q'}(s)| = Q(s,a_0) - V_{Q'}(s) \le Q(s,a_0) - Q'(s,a_0) \le \max_{a\in\mathcal{A}} |Q(s,a) - Q'(s,a)|.$$

Using the contraction property of B, we can show that as t increases, $Q^\star$ and $Q^{\star,t}$ becomes exponentially closer under ℓ_∞ norm:

$$\|Q^{\star,t} - Q^\star\|_\infty = \|BQ^{\star,t-1} - BQ^\star\|_\infty \le \gamma \|Q^{\star,t-1} - Q^\star\|_\infty.$$

Since $Q^\star$ has bounded range (recall Equation 2.2), for $Q^{\star,0} = \mathbf{0}_{|\mathcal{S}\times\mathcal{A}|}$ (or any function in the same range) we have $\|Q^{\star,0} - Q^\star\|_\infty \le R_{\max}/(1-\gamma)$. After H iterations, the distance shrinks to

$$\|Q^{\star,H} - Q^\star\|_\infty \le \gamma^H R_{\max}/(1-\gamma). \tag{2.9}$$

To guarantee that we compute a value function ϵ-close to $Q^\star$, it is sufficient to set

$$H \ge \frac{\log \frac{R_{\max}}{\epsilon(1-\gamma)}}{1-\gamma}. \tag{2.10}$$

The base of $\log$ is e in this thesis unless specified otherwise. To verify,

$$\gamma^H \frac{R_{\max}}{1-\gamma} = (1-(1-\gamma))^{\frac{1}{1-\gamma}\cdot H(1-\gamma)} \frac{R_{\max}}{1-\gamma} \le \left(\frac{1}{e}\right)^{\log \frac{R_{\max}}{\epsilon(1-\gamma)}} \frac{R_{\max}}{1-\gamma} = \epsilon.$$

Here we used the fact that $(1-1/x)^x \le 1/e$ for $x > 1$.

Equation 2.10 is often referred to as the **effective horizon**. The bound is often simplified as $H = O(\frac{1}{1-\gamma})$, and used as a rule of thumb to translate between the finite-horizon undiscounted and the infinite-horizon discounted settings.[2] In this thesis we will often use the term "horizon" generically, which should be interpreted as $O(\frac{1}{1-\gamma})$ in the discounted setting.

Bounding $\|Q^{\star,H} - Q^\star\|_\infty$: the finite-horizon interpretation

Equation 2.9 can be derived using an alternative argument, which views Value Iteration as optimizing value for a finite horizon. $V^{\star,H}(s)$ is essentially the optimal value

[2] The logarithmic dependence on $1/(1-\gamma)$ is ignored as it is due to the magnitude of the value function.

for the expected value of the finite-horizon return: $\sum_{t=1}^{H} \gamma^{t-1} r_t$. For any stationary policy π, define its H-step truncated value

$$V^{\pi,H}(s) = \mathbb{E}\Big[\sum_{t=1}^{H} \gamma^{t-1} r_t \;\big|\; \pi, s_1 = s\Big]. \tag{2.11}$$

Due to the optimality of $V^{\star,H}$, we can conclude that for any $s \in \mathcal{S}$ and $\pi : \mathcal{S} \to \mathcal{A}$, $V^{\pi,H}(s) \le V^{\star,H}(s)$. In particular,

$$V^{\pi^\star,H}(s) \le V^{\star,H}(s).$$

Note that the LHS and RHS are not to be confused: $\pi^\star$ is the stationary policy that is optimal for infinite horizon, and to achieve the finite-horizon optimal value on the RHS we may need a non-stationary policy (recall the discussion in Section 2.1.5).

The LHS can be lower bounded as $V^{\pi^\star,H}(s) \ge V^\star(s) - \gamma^H R_{\max}/(1-\gamma)$, because $V^{\pi^\star,H}$ does not include the nonnegative rewards from time step $H+1$ on. (In fact the same bound applies to all policies.) The RHS can be upper bounded as $V^{\star,H}(s) \le V^\star(s)$: $V^\star$ should dominate any stationary and non-stationary policies, including the one that first achieves $V^{\star,H}$ within H steps and picks up some non-negative rewards afterwards with any behavior. Combining the lower and the upper bounds, we have $\forall s \in \mathcal{S}$,

$$V^\star(s) - \frac{\gamma^H R_{\max}}{1-\gamma} \le V^{\star,H}(s) \le V^\star(s),$$

which immediately leads to Equation 2.9.

2.3 Reinforcement Learning in MDPs

In the previous section we considered policy evaluation and optimization when the full specification of the MDP is given, and the major challenge is computational. In the *learning* setting, the MDP specification, especially the transition function P and sometimes the reward function R, is not known to the agent. Instead, the agent can take actions in the environment and observe the state transitions, and may find approximate solutions to policy evaluation or optimization after accumulating some amounts of interaction experience, or *data*. While computation remains an important aspect in the learning setting, this thesis will largely focus on *sample efficiency*, that is, the problem of achieving a learning goal using as little data as possible.

There are 3 major challenges in reinforcement learning, which many discussions in this thesis will center around:

– Temporal credit assignment In RL, the value obtained by the agent is the result of decisions made over multiple steps, and a fundamental question is how we should credit the outcome to each of the preceding decisions that lead to it. The challenge is addressed by applying principles of dynamic programming. It is often mentioned to contrast other machine learning paradigms, such as supervised learning, where the agent directly observes supervisory signals and the problem lacks a long-term nature.

– Generalization In many challenging RL problems, the state (and action) space is large and the amount of data available is relatively limited. An agent will not succeed in its learning objective unless it *generalizes* what is learned about one state to other states. This challenge is encountered in other machine learning paradigms as well, and is often addressed by *function approximation* techniques [Bertsekas and Tsitsiklis, 1996].

– Exploration (and exploitation) In RL, the characteristics of the data are largely determined by how the agent chooses actions during data collection. Taking actions to collect a dataset that provides a comprehensive description of the environment, i.e., performing good *exploration*, is a highly non-trivial problem. Sometimes when assessed by some online measures (more on this in Section 2.3.2), the agent needs to balance between pursuing the value obtainable with current knowledge (exploitation) and sacrificing performance temporarily to learn more (exploration). In either case, the challenge is often addressed by the principle of *optimism in face of uncertainty* [Auer et al., 2002].

For the most part in this thesis we will address the temporal credit assignment challenge and the generalization challenge. The exploration challenge is not addressed, and for the purpose of theoretical analyses we will make assumptions that the data is collected in a sufficiently exploratory manner. In the remainder of this section, we first introduce the data collection protocols, and describe different performance measures for RL algorithms. Once the protocols and performance measures are set up properly, we move on and introduce basic algorithms and fundamental solution concepts.

2.3.1 Data collection protocols

In this thesis we consider the following flexible protocol for data collection that subsumes a number of settings considered in the literature. In particular, the dataset D

consists of $|D|$ sample trajectories, each of which has length H_D. In the i-th trajectory, the agent starts from some initial state $s_1^{(i)}$, takes actions according to a certain policy (potentially random and non-stationary), and records the H-step truncated trajectory $(s_1^{(i)}, a_1^{(i)}, r_1^{(i)}, s_2^{(i)}, \ldots, s_{H_D+1}^{(i)})$ in the dataset.

To fully specify the characteristics of the dataset, we need to determine (1) the value of H_D, (2) how $s_1^{(i)}$ is chosen, and (3) how the actions are chosen. Below we discuss a few combinations of interest, and for brevity we will drop the superscript $(\,\cdot\,)^{(i)}$ temporarily.

(a) $H_D = 1$**, round-robin** s **and** a

In this setting, the agent only observes transition tuple (s, a, r, s'). (We drop the time steps in subscripts for brevity as there is only 1 time step.) The state s and action a are chosen cyclically through the state-action space $\mathcal{S} \times \mathcal{A}$, which ensures that every state-action pair receives the same number of samples, often denoted as n. The total number of transitions $|D| = n \cdot |\mathcal{S} \times \mathcal{A}|$. This setting simplifies the data collection procedure and guarantees that all states and actions are uniformly covered in the dataset, which often simplifies the analysis of model-based RL methods [e.g. Mannor et al., 2007, Paduraru et al., 2008]. Chapter 3 of this thesis will use this protocol in the theoretical analysis.

(b) $H_D = 1$**,** s **and** a **sampled from distribution**

When the state (and action) space is very large and the budget of data size N is limited, the previous protocol **(a)** becomes inappropriate as we cannot even set n to 1. A useful variant of **(a)** is to assume that (s, a) is sampled i.i.d. from some distribution $p \in \Delta(\mathcal{S} \times \mathcal{A})$, where p is supported on the full space of $\mathcal{S} \times \mathcal{A}$. If the RL algorithm to be applied incorporates some generalization schemes, we may still hope the algorithm can succeed when there are (s, a) pairs that we have not seen even once in the data. This setting has been adopted in the analyses of approximate dynamic programming methods such as Fitted Value Iteration [Munos, 2007], and will be used in Chapter 5 of this thesis.

(c) $H_D > 1$**,** $s_1^{(i)} \sim \mu$**,** $a_t^{(i)}$ **chosen using a fixed policy**

The previous two protocols assume that the dataset covers all states and actions automatically, which can be unrealistic sometimes. A more realistic protocol is that the initial state $s_1^{(i)}$ is sampled i.i.d. from the initial distribution μ (recall Section 2.1.1). For discounted problems, the trajectory length H_D is often set to the effective horizon (Equation 2.10); for undiscounted finite-horizon problems, H is naturally the horizon as in the problem specification (see Section 2.1.5).

In this thesis, Chapter 4 will use this protocol to study the off-policy evaluation problem, where actions are chosen according to a fixed stochastic policy. One might wonder what happens if some states are not reachable from the support of μ, as this implies that we are not getting any data for those states. However, if the initial states are *always* sampled from μ when the agent is deployed, the unreachable states may be treated as if they did not exist and are not our concern.

(d) $H_D > 1$**,** $s_1^{(i)} \sim \mu$**,** $a_t^{(i)}$ **chosen by the algorithm**

All the previous protocols fix the choice of actions during data collection, and consequently the characteristics of the dataset is out of the agent's control (i.e., they correspond to the *batch* setting of RL). In protocol **(d)**, we give the algorithm full control over these actions. Intuitively (and informally), now the agent needs to take smart actions to ensure that it collects a "good" dataset that provides a comprehensive description of the environment, which is indeed the exploration challenge. While exploration is not the focus of this thesis, we introduce this protocol for completeness.

There are, of course, other protocols that are not in the incomplete list here. For example, the strongest data collection protocol is $H_D = 1$ and s_1, a_1 chosen by the algorithm, which can be used to emulate any of the above protocols and is required by the Sparse Sampling algorithm [Kearns et al., 2002]. Another protocol is that the data is a single long trajectory, which is often combined with some ergodicity-type assumptions to prevent the agent to get stuck in some subset of the state space [Kearns and Singh, 2002, Auer and Ortner, 2007, Jaksch et al., 2010].

2.3.2 Performance measures

Now that we have protocols for data collection, and an algorithm outputs some results based on the collected data, we need to specify the measure that we use to assess the quality of these results.

(i) Worst-case loss

Consider policy optimization. Suppose the agent computes a policy π based on the collected data. The worst-case loss measures the quality of the policy by

$$\|V^\star - V^\pi\|_\infty \,. \tag{2.12}$$

Note that $V^\star - V^\pi$ is element-wise non-negative, and the infinite norm takes the largest gap. This measure considers the suboptimality of the proposed policy π for the worst start state s_1. It is mostly used with data collection protocols **(a)** and **(b)** where there are some coverage guarantees over the entire state space, which is the case for Chapter 3 and 5.

Since data is random, so is the output policy π and the worst-case loss. To turn the random variable into deterministic quantities, we can either talk about the expected loss, or adopt the Probably Approximately Correct (PAC) framework [Valiant, 1984] and derive upper bound on the loss that is satisfied with high probability.

(ii) Loss under initial distribution

Demanding the algorithm to guarantee a small worst-case loss under protocols **(c)** and **(d)** may be vacuous, especially if there are states that are very hard or impossible to reach from the initial distribution μ. A more mild and natural performance measure is the loss under the initial distribution μ:

$$\mu^\top V^\star - \mu^\top V^\pi. \tag{2.13}$$

This measure is mostly used with protocol **(c)** and **(d)** when the initial distribution μ is a crucial part of the problem definition, which is the case in Chapter 6. Interestingly, when this measure is combined with **(d)** and the PAC framework, the resulting setting is the theoretical framework for studying exploration in finite-horizon reinforcement learning problems [Dann and Brunskill, 2015, Krishnamurthy et al., 2016, Jiang et al., 2016].

(iii) Worst-case error

Consider policy evaluation, where the agent computes a value function V that approximates V^π for a given π based on data. Similar to **(i)**, we can define the worst-case prediction error as

$$\|V^\pi - V\|_\infty. \tag{2.14}$$

(iv) Error under initial distribution

Similarly we can define the analogy of **(ii)** for policy evaluation. Let the prediction error with respect to the initial distribution μ be:

$$\left|\mu^\top V^\pi - \mu^\top V\right|. \tag{2.15}$$

Essentially, the algorithm only needs to output a scalar $\hat{v}$ in the place of $\mu^\top V$ to approximate $v^\pi := \mu^\top V^\pi$, and this is a standard statistical estimation problem. While $\hat{v}$ depends on the data and is random, we can talk about the properties of $\hat{v}$ as an estimator, such as bias, variance, and Mean Squared Error (MSE). Chapter 4 will use this measure in off-policy evaluation with data collection protocol **(c)**.

(v) Online measures

All previous measures implicitly assume a clear separation between data collection and deployment. With online performance measures, such a distinction disappears: the agent is evaluated at the same time as it collects data. As an example, consider the following measure that counts the number of "mistakes" made by the algorithm: the agent determines a policy which it uses for every next trajectory, and an error is counted if the policy is more than ϵ sub-optimal. This measure is sometimes referred to as a version of sample complexity for reinforcement learning [Kakade et al., 2003, Strehl et al., 2006]. Another well-known example is regret, which is also used in online learning literature [Shalev-Shwartz, 2011]; we will not talk about regret in detail as it makes more sense to discuss it in RL when the data is a single long trajectory.

2.3.3 Monte-Carlo methods

In the remainder of Section 2.3 we will survey a number of RL methods that are highly relevant to this thesis. We start with Monte-Carlo methods. In the RL context, the term "Monte-Carlo" refers to developing estimates of values without using bootstrapped targets, i.e., not invoking Bellman equations for policy evaluation or optimization where both sides of the equations contain unknowns. Instead, Monte-Carlo methods use the random return from the trajectory to form direct estimates.

As a basic example, consider policy evaluation. Given policy π, we are interested in computing V^π. Monte-Carlo policy evaluation performs the following simple steps: $\forall s \in \mathcal{S}$,

- Collect multiple trajectories by starting from $s_1 = s$ and following π for H steps.
- Compute the H-step discounted return $\sum_{t=1}^{H} \gamma^{t-1} r_t^{(i)}$ and take the average as an estimate of $V^\pi(s)$.

The expected value of the estimator is $V^{\pi,H}(s)$, and H is often set large enough to ensure that $V^{\pi,H}(s)$ approximates $V^\pi(s)$ well. In fact, using the analysis in Section 2.2, we can easily get $\|V^{\pi,H} - V^\pi\|_\infty \le \gamma^H R_{\max}/(1-\gamma)$, which takes the same form as Equation 2.9. Hence, the expression for effective horizon in Equation 2.10 applies to the policy evaluation setting as well.

The behavior of the algorithm is very straight-forward: for each s as the initial state, we get i.i.d. sample returns with mean $V^{\pi,H}(s)$ and range $[0, R_{\max}/(1-\gamma)]$. By using standard statistical bounds such as Hoeffding's inequality [Hoeffding, 1963], we can obtain an error bound on $|V^{\pi,H}(s) - V^\pi(s)|$ as a function of the sample size that holds with high probability, and apply union bound to guarantee accurate estimation in all states simultaneously, i.e., low worst-case prediction error. The same type of analysis will be carried out throughout the thesis so we omit the details here.

In general, getting low worst-case error requires the number of total trajectories to scale at least linearly with the size of the state space $|\mathcal{S}|$.[3] Sometimes we are only interested in a scalar value that characterizes the value of a policy, $v^\pi := \mu^\top V^\pi$ (recall performance measure **(iv)**). While we could estimate V^π for all states to a good accuracy and compute $\mu^\top V^\pi$ based on it, there is a simpler and much more effective procedure for doing this:

- Collect trajectories by starting from $s \sim \mu$ and following π for H steps.
- Compute $\sum_{t=1}^{H} \gamma^{t-1} r_t^{(i)}$ and take the average as an estimate of $\mu^\top V^\pi$.

A most notable property of this algorithm is that its accuracy guarantee is completely independent of the size of the state space, which is an elegant property that is often exhibited in Monte-Carlo methods [Kearns et al., 2002]. This algorithm is particularly useful when we want to assess the quality of a policy or compare among multiple policies, hence it forms the basis for policy validation and hyperparameter tuning in state-of-the-art empirical research of reinforcement learning.

[3] In some cases, mostly for episodic problems, it is possible to reuse a trajectory multiple times to form estimates of different states encountered along the trajectory; depending on how multiple occurrences of the same state is handled, the variants are called "first-visit" or "last-visit" Monte-Carlo policy evaluation [Sutton and Barto, 1998]. We do not introduce these variants as they grow the effective sample size at most by a multiplicative factor of H, which does not affect our discussion here.

On the other hand, Monte-Carlo policy evaluation requires the data collection protocol **(d)**, and crucially the policy used to collect data should be the same as the policy to be evaluated (i.e., the algorithm is **on-policy**). When such assumptions fail, especially when the data collection policy is different from the evaluated policy, we face an **off-policy** policy evaluation problem. While extension of Monte-Carlo methods still enjoy independence of the state space size [Precup et al., 2000], the dependence on horizon is exponential [Li et al., 2015b, Jiang and Li, 2016].

2.3.4 Tabular methods

In this section we survey methods that can work with a wider range of data collection protocols than Monte-Carlo methods, and incur polynomial dependence on both size of state space and the horizon. These methods do not address the generalization challenge and can only be applied to problems with finite and small state spaces.[4]

Tabular certainty-equivalence
Certainty-equivalence is a **model-based** RL algorithm, that is, it first estimates an MDP model from data, and then performs policy evaluation or optimization in the estimated model as if it were true. To specify the algorithm it suffices to specify the model estimation step.

Given a dataset D collected using any protocol, we first convert it into a bag of $\{(s, a, r, s')\}$ tuples, where each trajectory $(s_1, a_1, r_1, s_2, \ldots, s_{H+1})$ is broken into H tuples: (s_1, a_1, r_1, s_2), (s_2, a_2, r_2, s_3), …, (s_H, a_H, r_H, s_{H+1}). For every $s \in \mathcal{S}, a \in \mathcal{A}$, define $D_{s,a}$ as the subset of tuples where the first element of the tuple is s and the second is a, and we write $(r, s') \in D_{s,a}$ as the first two elements of the tuple does not need specification. The tabular certainty-equivalence model uses the following estimation of the transition function $\widehat{P}$:

$$\widehat{P}(\,\cdot\,|s,a) = \frac{\sum_{(r,s')\in D_{s,a}} \mathbb{I}\left(s' = (\cdot)\right)}{|D_{s,a}|}. \tag{2.16}$$

Here $\mathbb{I}(\,\cdot\,)$ is the indicator function. In words, $\widehat{P}(s'|s,a)$ is simply the empirical frequency of observing s' after taking a in state s. Similarly when reward function also

[4] The term "tabular" refers to the fact that these algorithms often maintain functions of states as intermediate and output variables, which are traditionally represented as tables when the state space is finite and small.

needs to be learned, the estimate is

$$\widehat{R}(s,a) = \frac{\sum_{(r,s') \in D_{s,a}} r}{|D_{s,a}|}. \tag{2.17}$$

$\widehat{P}$ and $\widehat{R}$ are the maximum likelihood estimates of the transition and the reward functions, respectively. Note that for the transition function to be well-defined we need $n(s,a) > 0$ for every $s, a \in \mathcal{S}$, which is guaranteed in protocol **(a)** where $n(s,a) \equiv n$. In Chapter 3 we will give detailed analyses of learning guarantees for the tabular certainty-equivalence model.

Value-based tabular methods

Certainty-equivalence explicitly stores an estimated MDP model, which has $O(|\mathcal{S}|^2|\mathcal{A}|)$ space complexity, and the algorithm has a *batch* nature, i.e., it is invoked after all the data are collected. In contrast, there is another popular family of RL algorithms that (1) only model the Q-value functions hence has $O(|\mathcal{S}||\mathcal{A}|)$ sample complexity, (2) can be applied in an online manner, i.e., the algorithm runs as more and more data are collected. Well-known examples include Q-learning [Watkins, 1989] and Sarsa [Sutton, 1996].

Another very appealing property of these methods is that it is relatively easy to incorporate sophisticated generalization schemes, such as deep neural networks, which has recently led to many empirical successes [Mnih et al., 2015, Wang et al., 2016]. On the other hand, such methods are typically less sample-efficient than model-based methods and will not be discussed in more details in this thesis.[5]

2.3.5 State abstractions

A common aspect of methods surveyed in the previous section is that the size of dataset (or *sample size*) required to yield learning guarantees is polynomial in the size of the state space. When the size of the state space is very large, as will be the case in many challenging problems, the agent needs to generalize what is learned about one state to other states using prior knowledge to reduce the effective size of the state space.

One of the easiest-to-deploy generalization schemes is state abstraction (or aggregation / compression). A state abstraction is a mapping h that maps the original

[5] While techniques such as experience replay can be used the improve the sample efficiency of many online algorithms [Lin, 1992], the boundary between value-based and model-based methods is also blurred in this case [Vanseijen and Sutton, 2015].

(or raw) state space $\mathcal{S}$ to some finite *abstract* state space;[6] for brevity we do not use additional notation for the codomain of h and simply write it as $h(\mathcal{S})$. Intuitively, if $s^{(1)}$ and $s^{(2)}$ are mapped to the same element, that is $h(s^{(1)}) = h(s^{(2)})$, they are treated as the same state.

Given a problem with state space $\mathcal{S}$ and an abstraction h, a typical usage of h is to (virtually) convert every state s in the dataset D into $h(s)$, and run any tabular algorithm over D with the understanding that the state space is $h(\mathcal{S})$. For example, if we collect a dataset that consists of tuples (s, a, r, s'), we can view each tuple now as $(h(s), a, r, h(s'))$, and build a certainty-equivalence model over state space $h(\mathcal{S})$. This is always doable despite the fact that there might not be a well-defined MDP with state space $h(\mathcal{S})$ that is the groundtruth process for the dataset.

An obvious benefit of using state abstraction is the increase of effective sample size. Suppose we collected a dataset with n samples per (s, a) pair (protocol **(a)**), and an abstraction h maps $s^{(1)}$ and $s^{(2)}$ to the same abstract state $\tilde{s}$. Then, after applying the abstraction, we get $2n$ samples for the state-action pairs $(\tilde{s}, a)$. In certainty-equivalence, we essentially double the sample size for estimating the transition and reward functions for a state-action pair and can enjoy lower variance in the estimates, or in other words, a reduced *estimation error*.

This advantage, of course, comes with a caveat, otherwise we could simply map every $s \in \mathcal{S}$ to the same abstract state and maximize the number of samples per state. The caveat is that if we aggregate states that are very different from each other, the learned models / value functions / policies may lose fidelity to the original MDP and yield arbitrarily bad performance. In certainty-equivalence, this may correspond to a high bias in the estimated transition and reward functions, or in other words, a high *approximation error*. The trade-off between approximation error and estimation error (sometimes informally referred to as the bias-variance trade-off) is a constant theme of statistical machine learning [Mohri et al., 2012].

Intuitively, the approximation error is high when we aggregate states that are very different from each other. The question is, how should we define an (approximate) equivalence notion among states? Whether they share the same optimal action? Whether they share the same $Q^\star$ values? Whether they yield the same rewards and next-state distributions? It turns out that, these criteria define a hierarchy of

[6] For general state abstractions it is more common to use the notation h [Li et al., 2006, Ortner et al., 2014]. In this thesis, however, we will mostly view abstractions under the framework of homomorphisms (h is the initial of "homomorphisms"), so we choose this notation for consistency [Ravindran and Barto, 2004]. In Chapter 6 we will also use h to refer to the time step within a trajectory, which should not be confused with abstractions.

state abstractions; as we move up in the hierarchy, we obtain more aggregation opportunities, but at the same time some algorithms are less well-behaved when used with these abstractions.

Definition 2.1 (Abstraction hierarchy [Li et al., 2006]). Given MDP $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ and state abstraction h that operates on $\mathcal{S}$, define the following types of abstractions:

1. h is $\pi^\star$-irrelevant if there exists an optimal policy $\pi^\star$, such that $\forall s^{(1)}, s^{(2)} \in \mathcal{S}$ where $h(s^{(1)}) = h(s^{(2)})$, $\pi^\star(s^{(1)}) = \pi^\star(s^{(2)})$.

2. h is $Q^\star$-irrelevant if $\forall s^{(1)}, s^{(2)}$ where $h(s^{(1)}) = h(s^{(2)})$, $\forall a \in \mathcal{A}$, $Q^\star(s^{(1)}, a) = Q^\star(s^{(2)}, a)$.

3. h is model-irrelevant if $\forall s^{(1)}, s^{(2)}$ where $h(s^{(1)}) = h(s^{(2)})$, $\forall a \in \mathcal{A}, x' \in h(\mathcal{S})$,

$$R(s^{(1)}, a) = R(s^{(2)}, a), \qquad \sum_{s' \in h^{-1}(x')} P(s'|s^{(1)}, a) = \sum_{s' \in h^{-1}(x')} P(s'|s^{(2)}, a). \tag{2.18}$$

The following property of the hierarchy shows that $\pi^\star$-irrelevance is the most lenient and model-irrelevance is the most strict.

Proposition 2.2 (Theorem 2 of Li et al. [2006][7]). *Model-irrelevance implies $Q^\star$-irrelevance, which further implies $\pi^\star$-irrelevance.*

In RL literature, the notion of model-irrelevance was originally introduced by Givan et al. [2003] as *bisimulations*, and was later generalized to *MDP homomorphisms* to handle action aggregation and permutation [Ravindran, 2004].[8] While this notion is most strict in the abstraction hierarchy, it also secures the success of almost any tabular RL algorithm: given model-irrelevant h, it is fundamentally impossible to distinguish between two datasets, one drawn from an abstract MDP M^h that is a perfect compression of the original MDP M (this MDP is implicit from Equation 2.18 and will be defined explicitly in Chapter 5), and the other drawn from the original MDP and converted using h ($(s, a, r, s') \to (h(s), a, r, h(s'))$); for the purpose of analysis we can simply treat the algorithm as if it were run in M^h, and any guarantee for the algorithm automatically extends.

[7] Li et al. [2006] also included two additional types of abstractions as well as the raw representation in the hierarchy theorem, which are omitted here.

[8] In this thesis we do not address action aggregation and permutation, and will use bisimulations and homomorphisms interchangeably.

On the other hand, if h is $Q^\star$-irrelevant, the compression does not preserve rewards or dynamics in general. While some tabular algorithms can still be applied and their guarantees extend (e.g., Q-learning), these extensions are not automatic and need new analyses (see e.g., Section 8.2.3 in [Li, 2009]). When it comes to $\pi^\star$-irrelevance, it is known that value-based and model-based algorithms may break down [Jong and Stone, 2005]; only policy search methods which directly optimize the return over a policy class can retain guarantees due to their robustness to agnosticity [Williams, 1992].

CHAPTER 3

Dependence of Effective Planning Horizon on Data Size

For MDPs with long horizons (i.e., discount factors close to one), it is common in practice to use reduced horizons during planning to speed computation at the possible expense of computing suboptimal plans. However, perhaps surprisingly, when the model available to the agent is estimated from data, the policy found using a shorter planning horizon can actually be *better* than a policy learned with the true horizon. In this chapter we provide a precise explanation for this phenomenon based on principles of learning theory. We show formally that the planning horizon is a complexity control parameter for the class of policies available to the planning algorithm. In particular, it has an intuitive, monotonic relationship with a simple counting measure of complexity, and a similar relationship can be observed empirically with a more general and data-dependent Rademacher complexity measure. Each complexity measure gives rise to a bound on the planning loss predicting that a planning horizon shorter than the true horizon can reduce overfitting and improve test performance, and we confirm these predictions empirically.

3.1 Introduction

When planning with Markov decision processes (MDPs), we distinguish between two different horizons (or, equivalently, discount factors). The *evaluation horizon*, specified by the problem formulation, is part of the definition of the ultimate measure of performance for a policy and cannot be changed. The *planning horizon*, on the other hand, is a parameter supplied to the planning algorithm; it affects the resulting policy but need not match the evaluation horizon. Generally, the deeper or longer the planning horizon, the greater the computational expense of computing a policy [Kearns et al., 2002, Kocsis and Szepesvári, 2006],[1] while in principle the shallower or shorter the planning horizon (relative to the evaluation horizon), the more suboptimal the resulting policy is likely to be [Kearns et al., 2002]. Thus, there is a tradeoff between computation and optimality that is relatively well-understood in cases where the MDP model (henceforth, simply model) used for planning is accurate.

In this chapter, we argue that there is another important reason to use shorter planning horizons in the more realistic case where the model used for planning is estimated from data: avoiding overfitting. Specifically, we show formally that the planning horizon controls the complexity of the policy class—shorter planning horizons define less complex policy classes. As in supervised learning, the optimal complexity (and therefore the optimal planning horizon) depends on the quantity of data used to estimate the model.

We explore two measures of complexity in this chapter. The first is a simple and intuitive counting measure that we show is monotonically related to the planning horizon. The second is a Rademacher complexity measure [Bartlett and Mendelson, 2003], which allows a more general analysis. For each measure we prove a bound on the planning loss given a particular choice of planning horizon. Each bound has two terms that depend in opposite ways on the planning horizon: one prefers the longest possible planning horizon (up to the true horizon), encouraging fidelity to the ultimate evaluation metric, while the other encourages the shortest possible planning horizon, keeping the policy class simple and thereby reducing the possibility of overfitting. In general, the bounds suggest that some intermediate planning horizon will be optimal. We verify these predictions empirically, showing that even in the absence of computational constraints it can be beneficial to use a reduced planning horizon.

Section 3.2 provides background on planning in MDPs. Section 3.3 formalizes the counting complexity measure. Rademacher complexity is discussed in Section 3.4, and Section 3.5 provides experimental validation of our claims.

[1] The computational dependence is linear in the planning horizon for planning algorithms such as value iteration, but those are limited to problems with small state spaces because of their quadratic dependence on size of the state space. For the large or infinite state space problems that motivate our research, Monte-Carlo Tree Search (or MCTS) [Browne et al., 2012] methods such as UCT [Kocsis and Szepesvári, 2006] are used because their complexity is independent of the size of the state space. However, these state-of-the-art methods have an exponential dependence on the planning horizon.

3.2 Preliminaries

We explicitly distinguish between the evaluation horizon and the planning horizon by the following notations in this chapter: we will use $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma_{\text{eval}})$ to specify the decision-making problem of interest, where γ_{eval} is the evaluation discount factor. We denote the optimal policy for M as $\pi^\star_{M,\gamma_{\text{eval}}}$ to make explicit its dependence on γ_{eval}, and define $V^\pi_{M,\gamma_{\text{eval}}}$ similarly. On the other hand, the optimal policy computed using an arbitrary discount factor γ is denoted as $\pi^\star_{M,\gamma}$, which is optimal for $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, and we define $V^\pi_{M,\gamma}$ similarly.

Certainty-equivalence control In practical settings, we rarely know the true parameters of the agent-environment interaction, i.e., we rarely have an exact model.[2] In this chapter, we are interested in the case where the model is estimated from data; scarcity of data then implies that our model will only be approximate. In *certainty-equivalence control* we act according to the policy that is optimal with respect to the inaccurate model used for planning. Hereafter, we will be concerned with the performance of the policy $\pi^\star_{\widehat{M},\gamma}$, where $\widehat{M}$ is the certainty-equivalence model introduced in Section 2.3.4, and $\gamma \in [0, \gamma_{\text{eval}}]$ is the *guidance discount factor* (which might not be equal to γ_{eval}). Note that our use of the certainty-equivalent policy allows us to abstract away all details of specific planning algorithms and focus solely on the influence of the guidance discount factor γ and its interaction with the quality of the model $\widehat{M}$.

Evaluation We emphasize that the certainty-equivalence policy computed using γ in model $\widehat{M}$ will nonetheless be evaluated in M using γ_{eval}. We capture this explicitly in our definition of the planning loss as the largest (over states) absolute difference in the values of the optimal policy $\pi^\star_{M,\gamma_{\text{eval}}}$ and the CE-control policy $\pi^\star_{\widehat{M},\gamma}$ when each is evaluated in the true environment M with the evaluation discount factor γ_{eval}. This definition is an instantiation of performance measure **(i)** in Section 2.3.2, and

[2] Indeed, even if we could know the model parameters exactly, often it is too representationally and computationally challenging to make them available to the planning agent. For example, we might know P in the form of a local generative model, and yet it could be computationally infeasible to compute the probability model. One could use MCTS algorithms for planning with an accurate generative model without converting to a probability model, but even then, for computational reasons, we would have to use a limited search tree to compute the choice of action at each time step. This is equivalent to exact planning with the inaccurate models implicit in the limited search tree, e.g., [Kearns and Singh, 2002].

can be formally written as

$$\textbf{Planning loss:} \quad \left\| V_{M,\gamma_{\text{eval}}}^{\pi^\star_{M,\gamma_{\text{eval}}}} - V_{M,\gamma_{\text{eval}}}^{\pi^\star_{\widehat{M},\gamma}} \right\|_\infty . \tag{3.1}$$

Discount factors and planning horizon When computing a policy with guidance discount factor γ, there is an implicit notion of planning horizon. The larger γ, the longer the planning horizon, because rewards further into the future have an effect on the choice of optimal action in the current state. Indeed, in tree-search based planning algorithms such as UCT [Browne et al., 2012, Kocsis and Szepesvári, 2006], γ is often explicitly translated into a planning horizon of order $O(1/(1-\gamma))$. Hereafter, we use guidance discount factor and planning horizon interchangeably with the understanding that the actual use depends on the nature of the planning algorithm.

Optimal guidance discount factor The decoupling of γ_{eval} and γ is fundamental to our work. The former is specified by the MDP, while the latter is a parameter under the control of the planning algorithm. If $\widehat{M} = M$, the only reason for $\gamma < \gamma_{\text{eval}}$ would be to obtain computational savings (at the expense of acting suboptimally). Our aim is to show that when $\widehat{M} \neq M$ there is another important reason to pick $\gamma < \gamma_{\text{eval}}$.

Given M and $\widehat{M}$, an optimal guidance discount factor can be defined as follows:

$$\gamma^\star = \underset{0 \le \gamma \le \gamma_{\text{eval}}}{\arg\min} \left\| V_{M,\gamma_{\text{eval}}}^{\pi^\star_{M,\gamma_{\text{eval}}}} - V_{M,\gamma_{\text{eval}}}^{\pi^\star_{\widehat{M},\gamma}} \right\|_\infty . \tag{3.2}$$

This is the discount factor the certainty-equivalence planner should use to minimize planning loss. (In general, there will be a range of optimal values for $\gamma^\star$; for computational reasons it is natural to pick the smallest value in that range.)

3.3 Planning Horizon and A Complexity Measure

Equation 3.2 above suggests that $\gamma^\star < \gamma_{\text{eval}}$ might be optimal—and indeed this is often observed in practice—but we do not yet have clear intuitions about when or why that would be true. We offer the following explanation: γ is a complexity control parameter for certainty-equivalent planning.

	Empirical risk minimization	*Certainty-equivalent planning*
Data	A set of input-output pairs	Empirical model $\widehat{M}$ estimated from (s, a, r, s') tuples
Candidates	Hypotheses class	Policy class
Selection rule	Minimizing training error	Maximizing value $V^{\pi}_{\widehat{M},\gamma}$
Complexity Parameter	E.g., # features, margin	Planning discount factor γ
Explanation of Overfitting	More features / less margin $\Downarrow$ (e.g., via VC-dimension) Richer hypotheses class $\Downarrow$ e.g., [Vapnik, 1999], Eq. 20 Higher variance	Larger γ $\Downarrow$ (our Theorem 3.1) Richer policy class $\Downarrow$ (our Theorem 3.2) Higher variance

Table 3.1: An analogy between empirical risk minimization and certainty-equivalent planning.

3.3.1 A counting complexity measure

Specifically, we will show in this section that γ monotonically controls the number of policies that can be optimal given a fixed state space, action space, and reward function. When $\widehat{M}$ is estimated from a limited data set, we can therefore avoid overfitting in policy selection by restricting the number of available policies through γ. (Later, we will relax the assumption that the reward function is known, and in Section 3.4 we will extend this intuition to a more sophisticated Rademacher measure.)

In the traditional empirical risk minimization setting for supervised learning, training data are used to evaluate the models in a given model class, and the model with the lowest training error is selected [Vapnik, 1992]. Overfitting occurs when the model class is too complex compared to the effective size of the dataset, and one way to avoid overfitting is to limit the complexity of the model class.

We draw analogies to four elements in this scenario (see Table 3.1 for a summary): (1) the size of the dataset, (2) the complexity of the model class, (3) empirical risk minimization as a method for selecting a model from the class of models, and (4) some way to control model complexity. In our planning setting, the size of the dataset corresponds to the number of samples used to estimate $\widehat{M}$. We assume that for every state-action pair (s, a), we observe n samples of the successor state drawn from the true transition function. (For now, we assume that the rewards R are known exactly.) The model class in our setting is the set of policies that are optimal for at least one possible $\widehat{M}$; we refer to this as the policy class. The complexity of the model class corresponds to the size of the policy class, i.e., the *number* of policies that are potentially optimal. Empirical risk minimization corresponds to selecting the optimal policy for $\widehat{M}$, as achieved by certainty-equivalence planning. These three correspondences are evident. It remains to show that reducing the guidance discount factor γ corresponds to reducing the size of the policy class being searched over by planning. Theorem 3.1 shows that this is indeed the case.

Theorem 3.1. *For any fixed state space $\mathcal{S}$, action space $\mathcal{A}$, and reward function R, define the policy class*

$$\Pi_{R,\gamma} = \{\pi : \exists\, P \textit{ s.t. } \pi \textit{ is optimal in } (\mathcal{S}, \mathcal{A}, P, R, \gamma)\}\ . \tag{3.3}$$

Then the following claims hold:

1. $\left|\Pi_{R,0}\right| = 1$ *if, for all $s \in \mathcal{S}$, $\arg\max_{a\in\mathcal{A}} R(s,a)$ is unique.*
2. $\Pi_{R,\gamma} \subseteq \Pi_{R,\gamma'} \quad \forall\gamma,\gamma' : 0 \leq \gamma \leq \gamma' < 1$
3. $\exists\gamma < 1, |\Pi_{R,\gamma}| \geq |\mathcal{A}|^{|\mathcal{S}|-2}$ *if $\exists\, s, s' \in S,\ \max_{a\in\mathcal{A}} R(s,a) > \max_{a'\in\mathcal{A}} R(s',a')$.*

The condition for claim 1 ensures that there are no ties in the maximal reward for each state, and the condition for claim 3 requires that one cannot obtain the maximal reward at every state. Note that $\Pi_{R,\gamma}$ counts policies that are optimal as P is allowed to vary arbitrarily, but explicitly depends on the fixed, known reward function R. (If R were allowed to vary with P, then every policy could be optimal at every γ.) In Sections 3.3.3 and 3.4 we will show how this restriction can be lifted.

Taken together, the three claims of Theorem 3.1 show that γ monotonically adjusts the size of the policy class from 1 to at least $|\mathcal{A}|^{|\mathcal{S}|-2}$, which is "almost all" of the $|\mathcal{A}|^{|\mathcal{S}|}$ possible policies. Thus the choice of guidance discount factor tightly controls complexity. Figure 3.1 illustrates this by showing that, as γ varies from 0 to γ_{eval}, we recover the traditional learning curves from supervised learning. Training loss decreases monotonically as γ increases, while test loss is U-shaped, indicating that an overly large γ causes overfitting. (See the caption for details on how these empirical results where produced and how training and testing loss were defined.) We can also see in Figure 3.1 that the location of the minimum of the test loss curve—that is, the optimal γ—shifts to the right as we get more data.

We now prove the three claims in turn. Claim 1 is straightforward; the optimal policy does not depend on T when $\gamma = 0$, thus the policy that picks the action

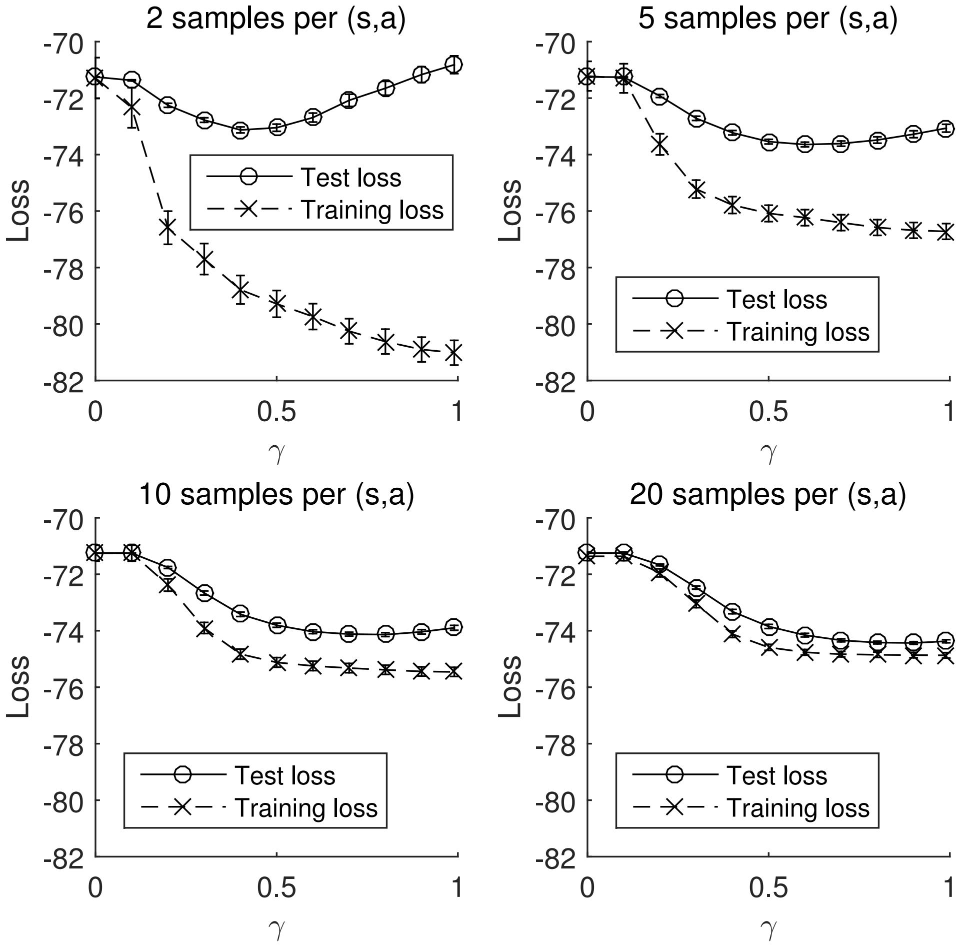

Figure 3.1: Learning curves as a function of γ, the guidance discount factor. For a single MDP M sampled from the RANDOM-MDP distribution specified in Section 3.5 we build $\widehat{M}$ by sampling each state-action pair n = 2, 5, 10, or 20 times; the different subgraphs correspond to different values of n. The reward function is assumed known, and $\gamma_{\text{eval}} = 0.99$. One thousand i.i.d. draws of the datasets lead to a thousand $\widehat{M}$'s for each n. For each $\widehat{M}$, the training loss is the negative value of the certainty-equivalence policy on the estimated model $\widehat{M}$: $-\frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}} V^{\pi^\star_{\widehat{M},\gamma}}_{\widehat{M},\gamma_{\text{eval}}}(s)$, and the test loss is the negative value of that same policy on the actual MDP M: $-\frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}} V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{\text{eval}}}(s)$. The figures show the average training and test loss over the random draws of the datasets with error bars. These learning curves share qualitative properties with learning curves in supervised learning: (1) monotonically decreasing training curves and U-shaped test curves; (2) the smaller the amount of data or the larger the complexity control parameter (for us it is γ), the larger the gap there is between the training curve and the test curve.

with the highest immediate reward is optimal. The assumption guarantees that this policy is unique.

Proof of Theorem 3.1, claim 2. We will prove that for $\gamma \leq \gamma'$, $\pi \in \Pi_{R,\gamma} \Rightarrow \pi \in \Pi_{R,\gamma'}$. Let P be a transition function for which π is optimal in $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. We will construct P' such that the MDP $M' = (\mathcal{S}, \mathcal{A}, P', R, \gamma')$ has the property that for all $\pi' : \mathcal{S} \to \mathcal{A}$,

$$V^{\pi'}_{M',\gamma'} = cV^{\pi'}_{M,\gamma} \ , \tag{3.4}$$

where c is a positive constant that only depends on γ and γ'. Consequently, π is also optimal in M'.

Let $P'(s'|s,a) = (1-\alpha)P(s'|s,a) + \alpha\,\mathbb{I}(s = s')$, where $\mathbb{I}(\,\cdot\,)$ is the indicator function and α is a scalar in the range $[0, 1]$. That is, P' is a transition function where, with probability $1 - \alpha$, transitions behave according to P, but with probability α, a state simply transitions to itself. Recall that

$$V^{\pi'}_{M,\gamma} = (\mathbf{I}_{|\mathcal{S}|} - \gamma P^{\pi'})^{-1}R^{\pi'}, \qquad V^{\pi'}_{M',\gamma'} = (\mathbf{I}_{|\mathcal{S}|} - \gamma' P'^{\pi'})^{-1}R^{\pi'} \ , \tag{3.5}$$

where $P^{\pi'}$ is the $|\mathcal{S}| \times |\mathcal{S}|$ transition matrix for policy π and and $R^{\pi'}$ is the $|\mathcal{S}| \times 1$ reward vector (see Section 2.1.3). We have

$$P'^{\pi'} = (1-\alpha)P^{\pi'} + \alpha\mathbf{I}_{|\mathcal{S}|} \ , \tag{3.6}$$

hence

$$\begin{aligned} V^{\pi'}_{M',\gamma'} &= \Big(\mathbf{I}_{|\mathcal{S}|} - \gamma'\big((1-\alpha)P^{\pi'} + \alpha\mathbf{I}_{|\mathcal{S}|}\big)\Big)^{-1} R^{\pi'} \\ &= \Big((1-\gamma'\alpha)\mathbf{I}_{|\mathcal{S}|} - \gamma'(1-\alpha)P^{\pi'}\Big)^{-1} R^{\pi'} \\ &= \frac{1}{1-\gamma'\alpha}\left(\mathbf{I}_{|\mathcal{S}|} - \frac{\gamma'(1-\alpha)}{1-\gamma'\alpha}P^{\pi'}\right)^{-1} R^{\pi'} \ . \end{aligned}$$

Letting $\frac{\gamma'(1-\alpha)}{1-\gamma'\alpha} = \gamma$, we get $\alpha = \frac{1-\gamma/\gamma'}{1-\gamma}$, which is between 0 and 1 since $0 \leq \gamma \leq \gamma' < 1$, and thus

$$V^{\pi'}_{M',\gamma'} = \frac{1-\gamma}{1-\gamma'} V^{\pi'}_{M,\gamma} \ . \tag{3.7}$$

This completes the proof. □

Proof of Theorem 3.1, claim 3. The proof is by construction. Let $(s^\star, a^\star)$ be a state-action pair that achieves the highest reward among all state-action pairs. Let s' be a state whose maximal reward action a' gives reward strictly less than $R(s^\star, a^\star)$. Such a state always exists under the assumption for this claim in the theorem. Consider an arbitrary policy π, with the only constraints that $\pi(s^\star) = a^\star$ and $\pi(s') = a'$. Then the following transition function makes π optimal for large enough γ:

$$\forall\, s \in \mathcal{S} \quad P(\,\cdot\,|s,a) = \begin{cases} \delta_{s^\star} & \text{if } a = \pi(s), s \neq s' \\ \delta_{s'} & \text{otherwise} \end{cases} \tag{3.8}$$

where $\delta_{(\cdot)}$ denotes the delta distribution. The optimality of π at $s^\star$ and s' is trivial, as both states are absorbing and π chooses the action that maximizes immediate reward. In any other state s, we show that π is optimal by comparing the optimal Q-value of $(s, \pi(s))$ to that of (s, a) for any other action a:

$$Q^\star(s,\pi(s)) = R(s,\pi(s)) + \frac{\gamma}{1-\gamma} R(s^\star, a^\star), \tag{3.9}$$

$$Q^\star(s,a) = R(s,a) + \frac{\gamma}{1-\gamma} R(s', a'). \tag{3.10}$$

We know $R(s^\star, a^\star) - R(s', a') > 0$, and as γ approaches one, $\gamma/(1-\gamma)$ tends to infinity, so for sufficiently large γ we can guarantee that $Q^\star(s,\pi(s)) > Q^\star(s,a)$. Recall that we constrained π in only two states, hence the number of such policies is $|\mathcal{A}|^{|\mathcal{S}|-2}$. □

3.3.2 Planning loss bound

Completing the connection to model class complexity in supervised learning, we show that the loss of the certainty-equivalence policy for $\widehat{M}$ is bounded, with high probability, in terms of the policy class complexity $|\Pi_{R,\gamma}|$. This is analogous to a standard generalization bound [Kearns and Vazirani, 1994], and implies that an intermediate value of γ will generally be optimal; moreover, as the amount of data (n) increases, so does the optimal γ.

Theorem 3.2. *Let M be an MDP with non-negative rewards and evaluation discount factor γ_{eval}. Let $\widehat{M}$ be an MDP comprising the true reward function of M and a transition function estimated from n samples for each state-action pair. Then certainty-equivalence planning with $\widehat{M}$ using guidance discount factor $\gamma \leq \gamma_{eval}$ has planning loss*

$$\left\| V^{\pi^\star_{M,\gamma_{eval}}}_{M,\gamma_{eval}} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{eval}} \right\|_\infty \leq \frac{\gamma_{eval} - \gamma}{(1-\gamma_{eval})(1-\gamma)} R_{\max} + \frac{2\gamma R_{\max}}{(1-\gamma)^2} \sqrt{\frac{1}{2n} \log \frac{2|\mathcal{S}||\mathcal{A}||\Pi_{R,\gamma}|}{\delta}} \tag{3.11}$$

with probability at least $1 - \delta$.

The proof of the theorem is in Section 3.7. The upper bound in Theorem 3.2 has two terms. The first is a bound on the planning loss incurred by using the guidance discount factor γ instead of the evaluation discount factor γ_{eval} in the true M. This term goes to zero as γ increases and approaches γ_{eval}. The second term isolates the planning loss due to the use of $\widehat{M}$ instead of M, but does not depend on γ_{eval}. In contrast to the first term, this term increases with γ, since greater policy class complexity allows performance on M and $\widehat{M}$ to diverge more dramatically. The dependence on the policy complexity $|\Pi_{R,\gamma}|$ is the novelty of our bound, compared to related work bounding loss by model errors or Bellman residuals [Kearns and Singh, 2002, Strehl et al., 2009, Farahmand et al., 2010].

The two terms in the bound of Theorem 3.2 depend in opposite ways on γ, therefore the bound will be optimized at some intermediate value. As the amount of data n increases, the second term will shrink and the bound will prefer larger values of γ. We will observe this behavior empirically in Section 3.5.

3.3.3 Handling uncertain rewards

The above analysis assumes that the reward function is known (i.e., $\widehat{M}$ contains the true reward function R). In this section we extend our analysis to handle cases where the rewards are unknown and must be estimated from data. We first establish a simple generalization of Theorem 3.1, where instead of fixing R we allow it to vary over a set of reward functions; the corresponding policy class is then the union of $\Pi_{R',\gamma}$ over all reward functions R' in the set. Then we prove a generalized version of Theorem 3.2 where rewards are estimated from data, using the new complexity measure.

Theorem 3.3. *For any fixed state space S, action space A, and a space of reward functions $\mathcal{R}$, define the policy class*

$$\Pi_{\mathcal{R},\gamma} = \bigcup_{R' \in \mathcal{R}} \{\pi : \exists\, P \text{ s.t. } \pi \text{ is optimal in } \langle \mathcal{S}, \mathcal{A}, P, R', \gamma \rangle\}. \tag{3.12}$$

Then the following claims hold:

1. $|\Pi_{\mathcal{R},0}| = 1$ *if,* $\forall s \in \mathcal{S}, \exists a \in \mathcal{A}, \min_{R' \in \mathcal{R}} R'(s,a) > \max_{a' \neq a, R' \in \mathcal{R}} R'(s,a')$.

2. $\Pi_{\mathcal{R},\gamma} \subseteq \Pi_{\mathcal{R},\gamma'} \quad \forall \gamma, \gamma' : 0 \leq \gamma \leq \gamma' < 1$

3. $\exists\gamma < 1, |\Pi_{\mathcal{R},\gamma}| \geq |\mathcal{A}|^{|\mathcal{S}|-2}$ *if* $\exists\, R' \in \mathcal{R}, s, s' \in S,\ \max_{a\in\mathcal{A}} R'(s,a) > \max_{a'\in\mathcal{A}} R'(s',a').$

Claims 2 and 3 are direct corollaries of Theorem 3.1: claim 2 follows because a union of supersets is always a superset of the union, and claim 3 follows because the size of a union set is at least the size of any set that contributes to the union. However, claim 1 states that $\Pi_{\mathcal{R},\gamma}$ is a singleton only if, for each state, there exists a dominating action whose most pessimistic reward value (over $\mathcal{R}$) is higher than the most optimistic reward value (over $\mathcal{R}$) of all other actions. Thus, $\Pi_{\mathcal{R},\gamma}$ still grows monotonically with γ, eventually almost covering the whole set of policies, but may not become arbitrarily small as $\gamma \to 0$ unless $\mathcal{R}$ satisfies additional constraints. This is the main price paid in generalizing Theorem 3.1.

We now turn to deriving a planning loss bound that parallels Theorem 3.2 in the setting where rewards are learned from data. The central idea is to first identify a set of reward functions in which the estimated reward function is highly likely to appear, and then use the policy complexity of that set as defined in Equation 3.12.

Theorem 3.4. *Let M be an MDP with non-negative rewards and evaluation discount factor γ_{eval}. Let $\widehat{M}$ be an MDP comprising reward and transition functions estimated from n samples for each state-action pair. Define*

$$\Delta = R_{\max}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}|}{\delta}}, \quad \mathcal{R}_\Delta = \{R' : \forall s \in \mathcal{S}, a \in \mathcal{A}, |R'(s,a) - R(s,a)| \leq \Delta\}.$$

Then certainty-equivalence planning with $\widehat{M}$ using guidance discount factor $\gamma \leq \gamma_{eval}$ has planning loss

$$\left\| V^{\pi^\star_{M,\gamma_{eval}}}_{M,\gamma_{eval}} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{eval}} \right\|_\infty \leq \frac{\gamma_{eval} - \gamma}{(1-\gamma_{eval})(1-\gamma)}R_{\max} + \frac{2R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}||\Pi_{\mathcal{R}_\Delta,\gamma}|}{\delta}} \tag{3.13}$$

with probability at least $1-\delta$.

The proof is deferred to Section 3.8 and is similar to the proof of Theorem 3.2; the major change is that, while the previous complexity measure $|\Pi_{R,\gamma}|$ only depended on the known R and γ, the new complexity measure $|\Pi_{\mathcal{R}_\Delta,\gamma}|$ also depends on the dataset size via Δ. However, the qualitative relationship between the bound and the dataset size is the same: as n increases, Δ decreases, and $\Pi_{\mathcal{R}_\Delta,\gamma}$ shrinks. Thus the first term in Equation 3.13 remains unaffected, and the second term decreases with n, implying that more data will reduce the degree of overfitting, just as in Theorem 3.2.

3.4 Rademacher Complexity Bound

In the previous section we showed how $|\Pi_{R,\gamma}|$ (and its generalization to unknown rewards) can be used to bound the loss of certainty-equivalence planning. While this simple complexity measure has the advantage of being easy to interpret and allowed us to prove a clean, monotonic relationship with the guidance discount factor, hypothesis-counting measures of complexity are typically weak, whereas modern data-dependent measures can be significantly tighter and more sensitive [Koltchinskii and Panchenko, 2000].

In this section, we present an alternative analysis using a Rademacher complexity measure [Bartlett and Mendelson, 2003]. We provide a loss bound parallel to that in Theorem 3.2 (and 3.4) that is also optimized at an intermediate γ that increases with sample size. Before providing our theoretical results, we first briefly review Rademacher complexity and explain how we apply it to the certainty-equivalent planning setting.

Rademacher complexity Consider a standard binary classification problem in supervised learning, where the data consists of input-output pairs $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \{-1, +1\}$. How can we measure the potential for overfitting when choosing the hypothesis that minimizes the training error from a set of functions $\mathcal{F}$?

One answer is as follows. Construct a new dataset $(x_1, \sigma_1), \ldots, (x_n, \sigma_n)$, where the σ_i are independent, unbiased coin flips. In general, no algorithm can do better than random guessing on this data, since the inputs provide no information about the labels. Thus, if we achieve a low training error by choosing among the functions in $\mathcal{F}$, we must be learning the random patterns in σ_i—i.e., overfitting. We can use this as a proxy for the overfitting that occurs on the original dataset $\{(x_i, y_i)\}_{i=1}^n$.

Slightly more formally, the expected degree of overfitting on the new dataset (over many draws of $\{\sigma_i\}$) is called the *Rademacher complexity* of $\mathcal{F}$ with respect to inputs $\{x_i\}_{i=1}^n$, and we can use it to bound the generalization error when fitting $\{(x_i, y_i)\}_{i=1}^n$ with $\mathcal{F}$.[3] While the motivating example above is Rademacher complex-

[3] The actual analyses often bound generalization error by the Rademacher complexity of the function class that maps (x, y) to the loss of each hypothesis, and then relate this Rademacher complexity

ity's application to classification problems, it also applies to regression problems where $\mathcal{F}$ contains real-valued functions with bounded range. The mathematical definition of Rademacher complexity is given below.

Definition 3.1. Given a function class $\mathcal{F} \subset (\mathcal{X} \to \mathbb{R})$ and X, a collection of n points in $\mathcal{X}$, the empirical Rademacher complexity is defined as

$$\widehat{\mathfrak{R}}_X(\mathcal{F}) = \mathbb{E}_{\substack{\sigma_i \overset{\text{i.i.d.}}{\sim} \text{unif}\{-1,1\} \\ i=1,\ldots,n}} \big[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{x \in X} \sigma_i f(x) \big]. \tag{3.14}$$

Application to certainty-equivalent planning Recall the analogy in Table 3.1: translating training errors to value functions and hypotheses to policies, we can derive a Rademacher complexity measure for the space of value functions corresponding to all possible policies, and therefore bound the degree of overfitting in certainty-equivalent planning. This is formalized in Theorem 3.5, in parallel to Theorems 3.2 and 3.4. Before stating the theorem, we first define a function class induced from an MDP whose Rademacher complexity will be used in the theorem.

Definition 3.2. Let M be an MDP and $\gamma \in [0, 1)$ be any discount factor. Define function class $\mathcal{F}_{M,\gamma} = \{f^\pi_{M,\gamma} : \pi \in \mathcal{S} \to \mathcal{A}\}$, with $f^\pi_{M,\gamma}(r, s') = r + \gamma V^\pi_{M,\gamma}(s')$.

Theorem 3.5. *Let M be an MDP with non-negative rewards and evaluation discount factor γ_{eval}. Let $\widehat{M}$ be an MDP comprising reward and transition functions estimated from n samples for each state-action pair. Then certainty-equivalence planning with $\widehat{M}$ using guidance discount factor $\gamma \le \gamma_{eval}$ has planning loss*

$$\left\| V^{\pi^\star_{M,\gamma_{eval}}}_{M,\gamma_{eval}} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{eval}} \right\|_\infty \le \frac{\gamma_{eval} - \gamma}{(1-\gamma_{eval})(1-\gamma)} R_{\max} + \\ \frac{2}{1-\gamma} \left(2 \max_{\substack{s \in \mathcal{S} \\ a \in \mathcal{A}}} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma}) \; + \; \frac{3R_{\max}}{1-\gamma} \sqrt{\frac{1}{2n} \log \frac{4|\mathcal{S}||\mathcal{A}|}{\delta}} \right)$$

with probability at least $1-\delta$, where $D_{s,a}$ is the set of n pairs of immediate reward and next-state (r, s') sampled from (s, a) in dataset D.

The proof of the theorem is in Section 3.9. The bound has the same decomposition as Theorems 3.2 and 3.4, but replaces the second term (loss due to planning with $\widehat{M}$ under γ) with a bound in terms of the Rademacher complexity of a function class $\mathcal{F}_{M,\gamma}$ in which each function corresponds to a policy in the MDP. For each

to that of the original hypothesis class. We only provide high-level intuition here and do not discuss the technical details.

state-action pair, the empirical model $\widehat{M}$ can be viewed as implicitly learning the expected values of all the functions in $\mathcal{F}_{M,\gamma}$ simultaneously from input samples $D_{s,a}$. The maximal deviation (over all functions) can be bounded by a state-action specific Rademacher complexity, and the worst case complexity (over all state-action pairs) translates to planning loss.

To show that the bound is optimized by an intermediate γ which increases with sample size n, it suffices to show that the second term increases with γ and decreases with n. This would be straightforwardly true if we knew that $\max_{s\in\mathcal{S},a\in\mathcal{A}} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma})$ increased monotonically with γ in the manner of Theorem 3.1. It turns out, however, that there can be cases where the Rademacher complexity is not monotonic in γ (see Figure 3.2, right panel). However, we show empirically that the data-dependent Rademacher complexity is strongly and positively correlated with γ in practice: see the left panel of Figure 3.2, where the relationship appears clearly monotonic. Thus Theorem 3.5 has the same qualitative interpretation as Theorem 3.2 while employing the more sensitive Rademacher measure.

3.4.1 An empirical Rademacher bound

One major advantage of using Rademacher complexity in standard learning theory is that it is computable from data,[4] allowing it to be used as a regularization term. In our Theorem 3.5, however, the function class $F_{M,\gamma}$ depends on the *true* value function $V^{\pi}_{M,\gamma}$, which we do not know during training. In this section, we prove an alternative bound that can be directly computed from data.

An appealing approach is to simply try and replace $\mathcal{F}_{M,\gamma}$ with $\mathcal{F}_{\widehat{M},\gamma}$. However, in the proof of Theorem 3.5 we require that the functions in $\mathcal{F}_{M,\gamma}$ must be *independent* of the dataset $D_{s,a}$, otherwise the Rademacher complexity results (or even simple concentration results) do not apply (for details, see the last step in proof of Lemma 3.13). This independence requirement will be violated if we use $\mathcal{F}_{\widehat{M},\gamma}$ in place of $\mathcal{F}_{M,\gamma}$.

However, for any pair (s,a), $D_{s,a}$ *is* in fact independent of $V^{\pi}_{\widehat{M},\gamma}$ for those π satisfying $\pi(s) \neq a$. This is because $V^{\pi}_{\widehat{M},\gamma}$ can be computed in that case without using the samples in $D_{s,a}$ (from which $\widehat{R}(s,a)$ and $\widehat{P}(\,\cdot\,|s,a)$ are computed).

Generalizing this idea, for any π (where $\pi(s)$ may or may not equal a), we decompose the value function at (s,a) into two parts: the first is the expected sum of

[4] The version of Rademacher complexity we use (notation: $\widehat{\mathfrak{R}}$) is usually referred to as *empirical* Rademacher complexity, and is distinguished from the version where an additional expectation is taken over the input points (notation: $\mathfrak{R}$). The latter gives slightly tighter bounds but requires knowledge about the input distribution, hence cannot be computed from data.

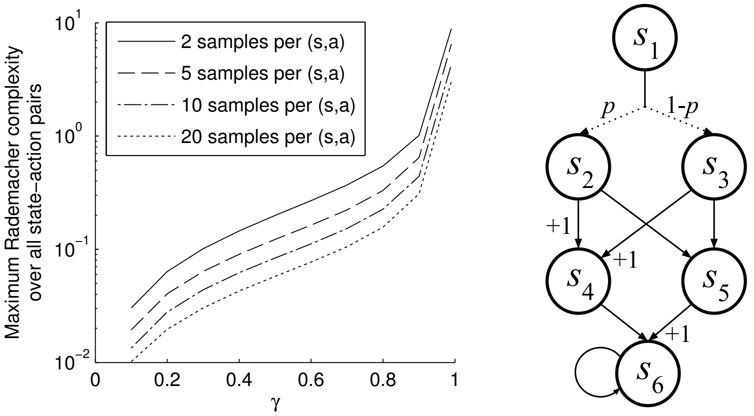

Figure 3.2: **Left:** Empirical illustration of the relationship between the Rademacher complexity measure $(\max_{s,a} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma}))^5$ and the guidance discount factor γ. We sample 10,000 MDPs from the RANDOM-MDP distribution (see Section 3.5). For each MDP, we draw a dataset with n samples for each state-action pair ($n = 2, 5, 10, 20$ respectively), and compute $\max_{s,a} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma})$ as defined in Equation 3.14, for $\gamma = 0.1, 0.2, \ldots, 0.9, 0.99$. We plot the complexity measure (shown in logarithmic scale) averaged over MDPs as a function of γ for each dataset size, and the trend is as expected: the complexity measure monotonically increases with γ, and decreases with dataset size. **Right:** A counter-example where the Rademacher complexity measure is not monotonic in γ. Circles represent states and solid arrows represent actions. A state-action pair gives 0 reward unless marked with $+1$, and all rewards are deterministic. Dotted arrows represent random transitions, which happen after taking the only action in s_1; by definition, this is the only state-action pair with possibly non-zero complexity. There are 4 policies in total (s_2 and s_3 have two actions), which gives the following 4 pairs of $(V^{\pi}_{M,\gamma}(s_2), V^{\pi}_{M,\gamma}(s_3))$: $(1, \gamma)$, $(\gamma, 1)$, $(1, 1)$, (γ, γ). When γ is very close to 1, every policy π gives almost the same values, and $\mathcal{F}_{M,\gamma}$ effectively only contains one element, resulting in a complexity approaching 0 as γ tends to 1 (a single hypothesis can never overfit); for $0 < \gamma < 1$, the complexity is non-zero; for $\gamma = 0$, the complexity is zero again, since every value function $V^{\pi}_{M,\gamma}$ is multiplied by γ in the definition of $f^{\pi}_{M,\gamma}$ and becomes 0. Overall the Rademacher complexity is a non-monotonic function of γ.

discounted rewards obtained before running into (s,a), and the second is the discounted sum of probabilities of running into (s,a) (up to a constant). Both parts are computable from $D \setminus D_{s,a}$ (and hence independent from the samples in $D_{s,a}$) and we can prove concentration results for each term separately. Formally, we have:

Proposition 3.6.

$$V^{\pi}_{\widehat{M},\gamma}(s') = V^{\pi}_{\widehat{M}^-_{s,a},\gamma}(s') \;+\; p^{\pi,s,a}_{\widehat{P},\gamma}(s')\, Q^{\pi}_{\widehat{M},\gamma}(s,a), \tag{3.15}$$

where

$$V^{\pi}_{\widehat{M}^-_{s,a},\gamma}(s') = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} \widehat{R}(s_t,a_t)\cdot \mathbb{I}(\neg A^{s,a}_t) \;\Big|\; s_1 = s' \;;\; \widehat{P}, \pi\Big], \tag{3.16}$$

$$p^{\pi,s,a}_{\widehat{P},\gamma}(s') = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} \mathbb{I}(A^{s,a}_t \wedge \neg A^{s,a}_{t-1}) \;\Big|\; s_1 = s' \;;\; \widehat{P}, \pi\Big], \tag{3.17}$$

$$Q^{\pi}_{\widehat{M},\gamma}(s,a) = \widehat{R}(s,a) + \gamma\langle \widehat{P}(\,\cdot\,|s,a), V^{\pi}_{\widehat{M},\gamma}\rangle. \tag{3.18}$$

Here $\mathbb{E}[\,\cdot\,|\, s_1 = s'; \widehat{P}, \pi]$ is an expectation over trajectories starting in state s', following policy π, and drawing next-states from the transition function $\widehat{P}$, and $A^{s,a}_t$ is the event that (s,a) has been visited before step t, that is, $\exists\, t' \le t, s_{t'} = s, a_{t'} = a$.

This decomposition leads to the following planning loss bound, where all the terms can be computed from the dataset D. The proof appears in Section 3.10.

Theorem 3.7. *Define the function classes:*

$$\mathcal{V}^{s,a}_{\widehat{M},\gamma} = \{f^{\pi}_{\widehat{M}^-_{s,a},\gamma} : \pi \in \mathcal{S} \to \mathcal{A}\}, \quad \mathcal{P}^{s,a}_{\widehat{P},\gamma} = \{p^{\pi,s,a}_{\widehat{P},\gamma} : \pi \in \mathcal{S} \to \mathcal{A}\} \tag{3.19}$$

where $f^{\pi}_{\widehat{M}^-_{s,a},\gamma}(r,s') = r + \gamma V^{\pi}_{\widehat{M}^-_{s,a},\gamma}(s')$. We have w.p. at least $1-\delta$,

$$\left\| V^{\pi^\star_{M,\gamma_{eval}}}_{M,\gamma_{eval}} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{eval}} \right\|_\infty \le \frac{\gamma_{eval} - \gamma}{(1-\gamma_{eval})(1-\gamma)} R_{\max} \;+\; \frac{2}{1-\gamma}\left(2 \max_{s\in\mathcal{S}, a\in\mathcal{A}} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{V}^{s,a}_{\widehat{M},\gamma}) + \frac{2\gamma R_{\max}}{1-\gamma} \max_{s\in\mathcal{S}, a\in\mathcal{A}} \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{P}^{s,a}_{\widehat{P},\gamma}) \;+\; \frac{3(1+\gamma)R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n}\log\frac{8|\mathcal{S}||\mathcal{A}|}{\delta}} \right).$$

[5] We calculate the Rademacher complexity exactly for $|D_{s,a}| = 2, 5, 10$. For $|D_{s,a}| = 20$, we cannot feasibly enumerate all possible values of $\{\sigma_i\}_{i=1}^n$ to compute the expectation in Equation 3.14; instead, we take the standard approach and sample them uniformly to obtain an approximation[El-Yaniv and Pechyony, 2007, Zhu et al., 2009]. We found that 1000 samples was sufficient to give low variance.

3.5 Experimental Results

We now show experimentally that the phenomena predicted by the preceding theoretical discussion do, in fact, appear in practice. In particular, we will see that the optimal choice of guidance discount factor can be smaller than γ_{eval}, and as we increase the amount of data used to estimate the model, a larger γ tends to be preferable.

For these experiments we randomly sampled 1,000 MDPs with 10 states and 2 actions from a distribution we refer to as RANDOM-MDP, defined as follows. For each state-action pair (s, a), the distribution over the next state, $P(\,\cdot\,|s, a)$, is determined by choosing 5 non-zero entries uniformly from all 10 states without replacement, filling these 5 entries with values uniformly drawn from $[0, 1]$, and finally normalizing $P(\,\cdot\,|s, a)$. The mean rewards were likewise sampled uniformly and independently from $[0, 1]$, and the actual reward signals have additive Gaussian noise with standard deviation 0.1. For all MDPs we fixed $\gamma_{\text{eval}} = 0.99$.

For each generated MDP M, and for each value of $n \in \{5, 10, 20, 50\}$, we independently generated 1,000 data sets, each consisting of n trajectories of length 10 starting at uniformly random initial states and choosing uniformly random actions. While our theoretical results assume the data set comprises n samples for each state-action pair, for our experiments we chose to generate trajectories since for most applications they are a more realistic way to collect data. (We also performed the same experiments using samples of state-action pairs and the results were qualitatively similar.)

For each dataset D, we set $\widehat{M}$ to be the maximum-likelihood model as specified in Section 3.2. If some (s, a) has never been seen in a dataset, we set $\widehat{R}(s, a) = 0.5$ and $\widehat{P}(s'|s, a) = 1/|\mathcal{S}|$. For each value of $\gamma \in \{0, 0.1, 0.2, \ldots, 0.9, 0.99\}$, we compute the empirical loss

$$\frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\left(V_{M,\gamma_{\text{eval}}}^{\pi^\star_{M,\gamma_{\text{eval}}}}(s) - V_{M,\gamma_{\text{eval}}}^{\pi^\star_{\widehat{M},\gamma}}(s)\right) , \tag{3.20}$$

and pick the γ that minimizes the loss as an estimate of $\gamma^\star$ (see Equation 3.2), breaking ties randomly.

Figure 3.3 shows the empirical planning loss averaged over datasets as a function of the guidance discount factor γ for a characteristic MDP. Each curve in the figure corresponds to a particular number of trajectories as data. The error bars in this figure and elsewhere show 95% confidence intervals. We can see that the curves exhibit the U-shape predicted by the theory, with minimum planning loss achieved

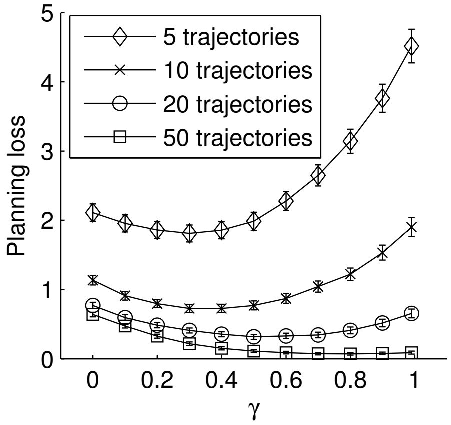

Figure 3.3: Planning loss as a function of γ for a single MDP drawn from the RANDOM-MDP distribution over MDPs defined in the main text. From top to bottom, the curves correspond to increasing dataset sizes and are labeled by the number of trajectories in the dataset. We see that planning loss decreases as the dataset size increases, and the optimal guidance discount factor $\gamma^\star$ (the value that achieves the minimum for each curve) increases with dataset size.

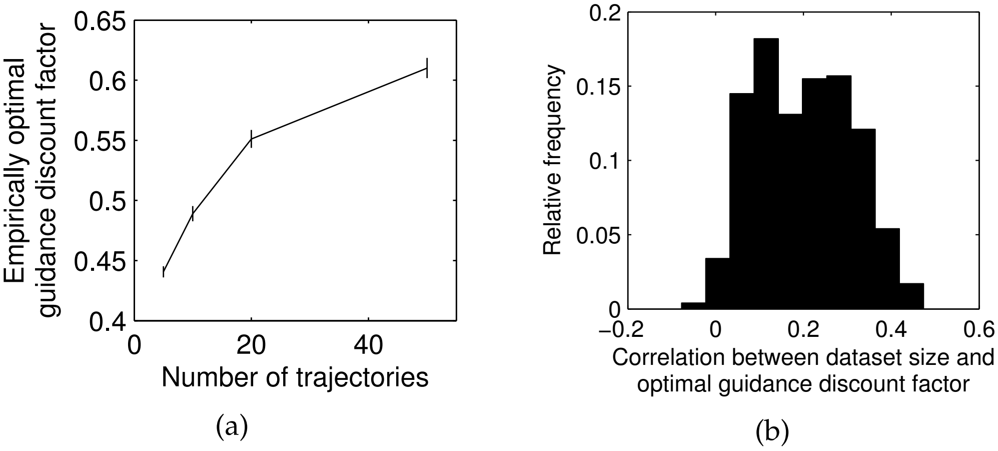

Figure 3.4: **(a)** Optimal guidance discount factor as a function of dataset size, averaged over 1,000 MDPs from RANDOM-MDP and 1,000 datasets for each MDP. Higher values (closer to one) are optimal for minimizing the planning-loss of certainty-equivalence policies as the amount of data increases. **(b)** Histogram of the correlation between dataset size and $\gamma^\star$ over 1,000 randomly generated MDPs from RANDOM-MDP. For almost all the MDPs, there is a positive correlation between dataset size and $\gamma^\star$, indicating that $\gamma^\star$ increasing with dataset size does not only hold in the average sense, but also applies to individual problems.

at some $\gamma^\star$ less than γ_{eval}. As expected, increasing dataset size reduces planning loss in general, and shifts $\gamma^\star$ to the right.

Figure 3.4a explicitly measures this shift by averaging the estimated $\gamma^\star$ across all 1,000 generated MDPs and their datasets. We can see clearly that as the amount of data increases, the optimal guidance discount factor increases as well. In the limit, of course, $\gamma^\star$ should equal γ_{eval}. However, for these values of dataset size the average $\gamma^\star$ is always significantly less than γ_{eval}; this means that using the true evaluation horizon for planning will lead to an increase in loss. While, conventionally, the use of a shorter horizon for planning has been justified based on computational savings, our result shows that in this setting it can decrease loss as well.

To complement the average-case analysis in Figure 3.4a, Figure 3.4b shows the distribution of the correlation between dataset size and $\gamma^\star$ over 1,000 individual MDPs. This correlation is positive with very high probability, implying that in almost all cases (under RANDOM-MDP) the theoretical relationship between dataset size and $\gamma^\star$ is borne out in practice.

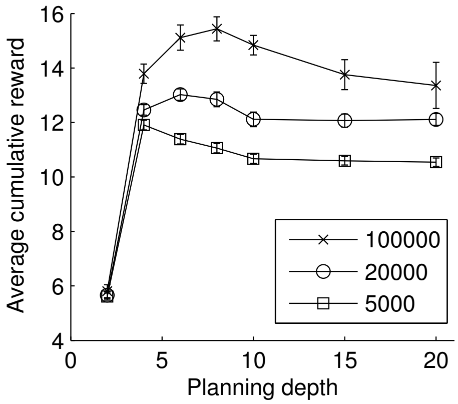

Figure 3.5: Performance of UCT as a function of planning depth. For each curve, the number of UCT trajectories is fixed to 5,000, 20,000, or 100,000. For each point on the graph, the UCB scalar has been separately optimized by sweeping through the values in $10 \cdot \exp\{-2, -1, 0, 1, 2\}$. For the 5,000 and 20,000 trajectory curves, each point is an average of 5,000 independent trials; the 100,000 trajectory curve is an average over 1,000 trials. As the number of trajectories increases, the UCT agent obtains more cumulative reward; on the other hand, the optimal planning depth (analogous to γ in previous experiments) increases as the number of trajectories increases.

3.5.1 Optimal planning depth in UCT

The previous experiments used small-state problems for which we could and did use perfect planning algorithms (value iteration) on the MDPs estimated from data. However, another common planning setting is one where we have an accurate (generative or probability) model, but the state space is so large that exact planning is impossible. Instead, incremental planning algorithms such as UCT are used [Kocsis and Szepesvári, 2006]. These algorithms repeatedly sample a search tree (rooted at the current state) that implicitly defines an inaccurate local model $\widehat{M}$ from which a policy is derived. Here we show that the main intuition obtained above—that planning horizon controls complexity, hence the more inaccurate the model the shorter the planning horizon that should be used—holds for UCT as well (see Jiang et al. [2014] for an alternative approach to controlling complexity in UCT via state abstractions).

In this setting, we do not have "data" in the sense of recorded experiences; instead, the accuracy of the local model is mediated by the number of trajectories sampled at the current state. Similarly, rather than manipulating a continuous discount factor γ we will control complexity via the planning depth, a discrete hyperparameter that sets the maximum length of the sampled trajectories. Our aim is to show that the relationship we have established between dataset size and discount factor for value iteration holds analogously between the number and depth of UCT trajectories.

We used a benchmark POMDP domain *RockSample* [Silver and Veness, 2010] and evaluated UCT's performance with different numbers of trajectories and different maximum depths. A detailed description of this infinite-sized belief-state space domain can be found in [Smith and Simmons, 2004]; we used a map of size 7×8. Since this problem is episodic, we use the average cumulative reward per episode as our evaluation metric in place of planning loss (and so higher is better). Since episodes are usually on the order of hundreds of time steps, setting the planning depth to this level is computationally infeasible. However, Figure 3.5 shows that choosing a small planning depth not only speeds computation but also helps performance when the number of trajectories is limited.

In particular, an intermediate value of planning depth always achieves the highest cumulative reward. Moreover, as the number of trajectories grows from 5000 to 20000 to 100000, that optimal planning depth increases. This is qualitatively the same behavior we have seen before.

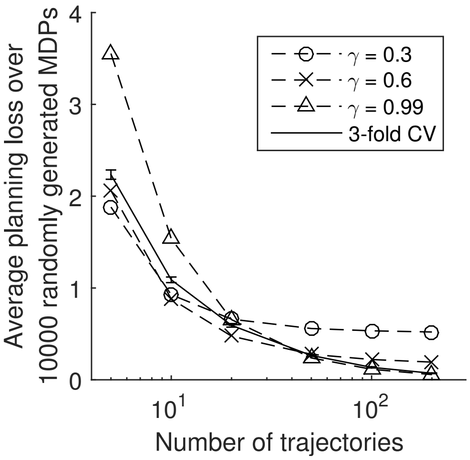

Figure 3.6: 3-fold cross-validation vs. fixed γ. Domain distribution, data generation, and candidate guidance discount factors are the same as in Figure 3.4. We plot average loss as a function of sample size in terms of the number of trajectories. Each dashed curve corresponds to using a particular value of γ for all dataset sizes, and it is clear that a small γ does well for small dataset size but is asymptotically suboptimal with a large dataset, and a large γ does the opposite; the solid curve corresponds to choosing γ via cross-validation, and its performance approximately matches the best γ for each dataset size simultaneously.

3.5.2 Selecting γ via cross-validation

We have seen that choosing $\gamma < \gamma_{\text{eval}}$ often improves performance, but how should we go about selecting the optimal γ in practice? In supervised learning, k-fold cross-validation is one of the most common techniques for selecting hyperparameters to avoid overfitting, and it is easy to apply here as well. (Indeed, we suspect cross-validation is often used in practice for choosing discount factors though we are unaware of any specific reference.)

Specifically, given a dataset D drawn from MDP M, we can split the sample trajectories into (state, action, reward, next-state) tuples, and then divide the tuples randomly into k folds of equal size, $D_1, \ldots, D_k$. For each fold $j = 1, 2, \ldots, k$, the validation model $\widehat{M_j}$ is defined to be the maximum-likelihood model learned from D_j, and the training model $\widehat{M_{-j}}$ is the one learned from $D \setminus D_j$. Then for each candidate γ, the validation value on fold j is given by

$$\text{ValidationValue}_j(\gamma) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} V_{\widehat{M_j}, \gamma_{\text{eval}}}^{\pi^\star_{\widehat{M}_{-j}, \gamma}}(s) \; . \tag{3.21}$$

Cross-validation selects the value of γ that maximizes the validation value averaged over all folds.

However, there is a potential problem. While cross-validation produces unbiased estimates of loss in most supervised settings, in certainty-equivalence planning the use of a finite validation set biases our estimate of a policy's true value. This happens because, although the transition and reward functions in the validation model are themselves unbiased, the validation value of a policy is computed via a nonlinear matrix inverse (see Equation 3.5). Thus, for instance, a myopic policy may perform well in a model estimated from a small validation set due to reduced stochasticity. Under mild assumptions the bias can be shown to decrease much faster than variance when sample size is sufficiently large [Mannor et al., 2007]; however, in practice our data sets are often relatively small.

Despite this caveat, our experiments in this section show that, at least in some instances, cross-validation can still be an effective practical tool for choosing γ. We leave the design and analysis of other cross-validation schemes for MDPs to future work; see also [Paduraru, 2013] for some discussion of this issue.

We validate the cross-validation approach on MDPs drawn from the RANDOM-MDP distribution. The other detailed settings are the same as for Figure 3.4 (see the beginning of Section 3.5), except that for each MDP M we only draw one dataset for each dataset size (in terms of number of trajectories) $n = 5, 10, 20, 50, 100, 200$. Given a dataset, we split the sample tuples (s, a, r, s') randomly into 3 subsets of equal sizes and choose γ using cross-validation (see Equation 3.21), and apply the chosen γ to the model estimated from the full data to compute the certainty-equivalent policy. Figure 3.6 shows the average loss of this 3-fold cross-validation approach compared to the losses obtained using fixed values of γ. We can see that small values of γ incur relatively large loss when there are sufficient samples, and large values of γ incur relatively large loss when there are few samples. In other words, no fixed γ dominates the others over all sample sizes. In contrast, cross-validation is able to achieve loss close to the best fixed γ at each sample size simultaneously by selecting γ adaptively as sample size changes.

3.6 Related Work and Discussions

The loss induced by a finite planning horizon is known as truncation loss (see related bounds given by [Kearns et al., 2002]). Separately, it is also well-understood how planning loss relates to model inaccuracy, which can come from estimation error when the model is constructed from data [Farahmand et al., 2010, Mannor et al., 2007], and/or approximation error when approximations are employed in planning (e.g., state abstractions [Ravindran and Barto, 2004]). It has been noted that such loss can have significant dependence on horizon [Kearns and Singh, 2002, Strehl et al., 2009]. To our knowledge, Petrik and Scherrer [2009] are the first to show how a short horizon can reduce loss when the model is inaccurate due to approximation errors. Our work is the first to explore a similar phenomenon due to estimation errors, and our analysis exploits the structure of these errors as well as established principles in supervised learning to obtain stronger claims about $\gamma^\star$ and dataset size.

Baxter and Bartlett [2001] dealt with the problem of estimating the policy gradient with an infinite horizon ($\gamma_{\text{eval}} \to 1$), and as part of their algorithm they proposed using a reduced discount factor (their β) to trade-off bias and variance in the resulting estimates. However, their gradient estimation setting is simpler than the planning setting we consider, where the model is estimated from batch data and the policy is computed based on the model. It is only in the latter setting that complexity of policy classes plays a role and an explicit connection to statistical learning theory can be made, which is our main contribution.

In the policy search setting, Tewari and Bartlett [2006] studied the complexity of parameterized policy classes and used measures such as VC-dimension to bound the regret given a full specification of the MDP (or POMDP) model. In their setting, the value of a policy is estimated from Monte-carlo trials, and the estimates for different policies use the same sequence of random numbers to generate sample trajectories the setting introduced by Ng and Jordan [2000]. This "reuse of randomness" is crucial to their analysis, and is fundamentally different from standard settings (like ours) where randomness comes from the environment and is not under the agent's control.

In this chapter we focus on providing a theoretical explanation of why small planning horizons can lead to better results in inaccurate models; as a by-product, this suggests an approach to regularizing planning under uncertainty, and we provide some preliminary empirical exploration along this direction (see Section 3.5.2). There exist alternative approaches to handling uncertainty in knowledge of model parameters. One approach is to consider a high probability set of possible MDPs (e.g., with the help of interval estimation if the model is constructed from data [Strehl and Littman, 2005]) and take the worst case performance into consideration when planning; this is known as robust control [Nilim and El Ghaoui, 2005, Bertuccelli et al., 2012]. Another approach is to adopt the Bayesian framework and model the uncertainty in model parameters as a distribution over MDPs and then use Bayes-optimal planning [Strens, 2000]. However, both approaches take the horizon as given without separating the planning and evaluation roles, which is central to our work. For the Bayesian setting, since it is too computationally intensive to obtain the Bayes-optimal policy for most real-world problems, sampling based approximations via MCTS are often used [Ross et al., 2011, Asmuth and Littman, 2011, Guez et al., 2012], and the interaction between planning horizon and degree of approximation still exists. An empirical result for such a setting has been provided in Section 3.5.1.

3.7 Proof of Theorem 3.2

We begin by proving Lemma 3.8 and Lemma 3.9.

Lemma 3.8. *For any MDP M with rewards in $[0, R_{\max}]$, $\forall \pi : \mathcal{S} \to \mathcal{A}$ and $\gamma \leq \gamma_{eval}$,*

$$V^{\pi}_{M,\gamma} \;\leq\; V^{\pi}_{M,\gamma_{eval}} \;\leq\; V^{\pi}_{M,\gamma} + \frac{\gamma_{eval} - \gamma}{(1-\gamma_{eval})(1-\gamma)} R_{\max}\,. \tag{3.22}$$

Proof. The lower bound on $V^{\pi}_{M,\gamma_{\text{eval}}}$ follows directly from the assumption that reward is non-negative and that $\gamma \leq \gamma_{\text{eval}}$. For the upper bound,

$$\begin{aligned}
\left\|V^{\pi}_{M,\gamma_{\text{eval}}} - V^{\pi}_{M,\gamma}\right\|_{\infty} &= \left\|\sum_{t=1}^{\infty}({\gamma_{\text{eval}}}^{t-1} - \gamma^{t-1})(P^{\pi})^{t-1}R^{\pi}\right\|_{\infty} \\
&\leq \sum_{t=1}^{\infty}({\gamma_{\text{eval}}}^{t-1} - \gamma^{t-1})R_{\max} = (\frac{1}{1-\gamma_{\text{eval}}} - \frac{1}{1-\gamma})R_{\max} \\
&= \frac{\gamma_{\text{eval}} - \gamma}{(1-\gamma_{\text{eval}})(1-\gamma)}R_{\max}.
\end{aligned}$$

□

Lemma 3.9. *Given true MDP M, let $\widehat{M}$ be an MDP comprising reward function $\widehat{R} = R$ and transition function $\widehat{P}$ estimated from n samples for each state-action pair, then*

$$\left\|V^{\pi^{\star}_{M,\gamma}}_{M,\gamma} - V^{\pi^{\star}_{\widehat{M},\gamma}}_{M,\gamma}\right\|_{\infty} \leq \frac{2\gamma R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{2|\mathcal{S}||\mathcal{A}||\Pi_{R,\gamma}|}{\delta}}$$

with probability at least $1-\delta$.

We prove Lemma 3.9 with two additional lemmas: Lemma 3.10 translates planning loss to value error, and Lemma 3.11 relates value error to a Bellman-residual-like quantity that has a uniform deviation bound which depends on $|\Pi_{R,\gamma}|$.

Lemma 3.10. *For any $\widehat{M} = \langle S, A, \widehat{P}, \widehat{R}, \gamma \rangle$ with $\widehat{R}$ bounded by $[0, R_{\max}]$,*

$$\left\| V_{M,\gamma}^{\pi^\star_{M,\gamma}} - V_{M,\gamma}^{\pi^\star_{\widehat{M},\gamma}} \right\|_\infty \leq 2 \max_{\pi \in \, \Pi_{\widehat{R},\gamma} \bigcup \{\pi^\star_{M,\gamma}\}} \left\| V^\pi_{M,\gamma} - V^\pi_{\widehat{M},\gamma} \right\|_\infty . \tag{3.23}$$

In particular, if $\widehat{R} = R$, we have

$$\left\| V_{M,\gamma}^{\pi^\star_{M,\gamma}} - V_{M,\gamma}^{\pi^\star_{\widehat{M},\gamma}} \right\|_\infty \leq 2 \max_{\pi \in \Pi_{R,\gamma}} \left\| V^\pi_{M,\gamma} - V^\pi_{\widehat{M},\gamma} \right\|_\infty . \tag{3.24}$$

Proof. $\forall s \in \mathcal{S}$,

$$\begin{aligned}
V_{M,\gamma}^{\pi^\star_{M,\gamma}}(s) - V_{M,\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) = &\left(V_{M,\gamma}^{\pi^\star_{M,\gamma}}(s) - V_{\widehat{M},\gamma}^{\pi^\star_{M,\gamma}}(s) \right) - \left(V_{M,\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) - V_{\widehat{M},\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) \right) + \\
&\left(V_{\widehat{M},\gamma}^{\pi^\star_{M,\gamma}}(s) - V_{\widehat{M},\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) \right) \\
\leq &\left(V_{M,\gamma}^{\pi^\star_{M,\gamma}}(s) - V_{\widehat{M},\gamma}^{\pi^\star_{M,\gamma}}(s) \right) - \left(V_{M,\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) - V_{\widehat{M},\gamma}^{\pi^\star_{\widehat{M},\gamma}}(s) \right) \\
\leq &\, 2 \max_{\pi \in \left\{ \pi^\star_{\widehat{M},\gamma}, \pi^\star_{M,\gamma} \right\}} \left| V^\pi_{M,\gamma}(s) - V^\pi_{\widehat{M},\gamma}(s) \right| .
\end{aligned}$$

(3.23) follows from taking max over all states on both sides of the inequality and the fact that $\pi^\star_{\widehat{M},\gamma} \in \Pi_{\widehat{R},\gamma}$. If $\widehat{R} = R$, $\pi^\star_{M,\gamma}$ is also in $\Pi_{\widehat{R},\gamma} (= \Pi_{R,\gamma})$ and (3.24) follows. □

Lemma 3.11. *For any $\widehat{M} = \langle S, A, \widehat{P}, \widehat{R}, \gamma \rangle$ with $\widehat{R}$ bounded by $[0, R_{\max}]$, $\forall \pi : S \to A$,*

$$\left\| Q^\pi_{M,\gamma} - Q^\pi_{\widehat{M},\gamma} \right\|_\infty \leq \frac{1}{1-\gamma} \max_{s \in \mathcal{S}, a \in \mathcal{A}} \left| \widehat{R}(s,a) + \gamma \langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma} \rangle - Q^\pi_{M,\gamma}(s,a) \right| .$$

Proof. Given any policy π, define state-action value functions $Q_0, Q_1, Q_2, \ldots, Q_m, \ldots$ such that $Q_0 = Q^\pi_{M,\gamma}$, and

$$Q_m(s,a) = \widehat{R}(s,a) + \gamma \langle \widehat{P}(\,\cdot\,|s,a), V_{m-1} \rangle,$$

where $V_{m-1}(s) = Q_{m-1}(s, \pi(s))$. Notice that

$$\begin{aligned}
\|Q_m - Q_{m-1}\|_\infty &= \gamma \max_{s \in \mathcal{S}, a \in \mathcal{A}} \left| \langle \widehat{P}(\,\cdot\,|s,a), (V_{m-1} - V_{m-2}) \rangle \right| \\
&\leq \gamma \max_{s \in \mathcal{S}, a \in \mathcal{A}} \|\widehat{P}(\,\cdot\,|s,a)\|_1 \, \|V_{m-1} - V_{m-2}\|_\infty \\
&= \gamma \, \|V_{m-1} - V_{m-2}\|_\infty \leq \gamma \, \|Q_{m-1} - Q_{m-2}\|_\infty ,
\end{aligned}$$

so

$$\|Q_m - Q_0\|_\infty \leq \sum_{k=0}^{m-1} \|Q_{k+1} - Q_k\|_\infty \leq \|Q_1 - Q_0\|_\infty \sum_{k=1}^{m-1} \gamma^{k-1}.$$

Taking the limit of $m \to \infty$, $Q_m \to Q^\pi_{\widehat{M},\gamma}$, and we have

$$\left\|Q^\pi_{\widehat{M},\gamma} - Q_0\right\|_\infty \leq \frac{1}{1-\gamma} \|Q_1 - Q_0\|_\infty .$$

This completes the proof, noticing that $Q_0 = Q^\pi_{M,\gamma}$, $V_0 = V^\pi_{M,\gamma}$, and $Q_1(s,a) = \widehat{R}(s,a) + \gamma\langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma}\rangle$. □

Proof of Lemma 3.9. From Equation 3.24 in Lemma 3.10 and Lemma 3.11, we have

$$\begin{aligned}\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty &\leq 2 \max_{\pi \in \Pi_{R,\gamma}} \left\|V^\pi_{M,\gamma} - V^\pi_{\widehat{M},\gamma}\right\|_\infty \leq 2 \max_{\pi \in \Pi_{R,\gamma}} \left\|Q^\pi_{M,\gamma} - Q^\pi_{\widehat{M},\gamma}\right\|_\infty \\ &= 2 \max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi\in\Pi_{R,\gamma}}} \left|Q^\pi_{M,\gamma}(s,a) - Q^\pi_{\widehat{M},\gamma}(s,a)\right| \\ &\leq \frac{2}{1-\gamma} \max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi\in\Pi_{R,\gamma}}} \left|R(s,a) + \gamma\langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma}\rangle - Q^\pi_{M,\gamma}(s,a)\right| .\end{aligned}$$

For any particular s, a, π tuple, note that $\langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma}\rangle$ is the average of i.i.d. random variables with bounded support $[0, \gamma R_{\max}/(1-\gamma)]$ and mean $Q^\pi_{M,\gamma}(s,a) - R(s,a)$; according to Hoeffding's inequality, $\forall t > 0$,

$$\mathbb{P}\Big\{ \left|R(s,a) + \gamma\langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma}\rangle - Q^\pi_{M,\gamma}(s,a)\right| > t\Big\} \leq 2\exp\left\{-\frac{2nt^2}{\gamma^2 R^2_{\max}/(1-\gamma)^2}\right\} . \tag{3.25}$$

To obtain a uniform bound over all (s, a, π) tuples, we set the right-hand side of Equation 3.25 to $\delta/|\mathcal{S}||\mathcal{A}||\Pi_{R,\gamma}|$, and solve for t, and the theorem follows. □

Proof of Theorem 3.2. $\forall s \in \mathcal{S}$,

$$\begin{aligned}V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma_{\text{eval}}}(s) - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{\text{eval}}}(s) = &\left(V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma_{\text{eval}}}(s) - V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma}(s)\right) \\ &+ \left(V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma}(s) - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{\text{eval}}}(s)\right) .\end{aligned}$$

By Lemma 3.8, the first term can be bounded by

$$V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma_{\text{eval}}}(s) - V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma}(s) \leq \frac{\gamma_{\text{eval}} - \gamma}{(1-\gamma_{\text{eval}})(1-\gamma)} R_{\max}$$

and by Lemma 3.9, the second term can be bounded as follows w.p. at least $1-\delta$:

$$\begin{aligned} V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma}(s) - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma_{\text{eval}}}(s) &\leq V^{\pi^\star_{M,\gamma_{\text{eval}}}}_{M,\gamma}(s) - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}(s) \\ &\leq V^{\pi^\star_{M,\gamma}}_{M,\gamma}(s) - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}(s) \qquad (\pi^\star_{M,\gamma} \text{ is optimal for } (M,\gamma)) \\ &\leq \frac{2\gamma R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{2|\mathcal{S}||\mathcal{A}||\Pi_{R,\gamma}|}{\delta}}. \end{aligned}$$

□

3.8 Proof of Theorem 3.4

The proof technique is similar to Theorem 3.2. Note that among the lemmas we proved for Theorem 3.2, Lemma 3.8, 3.10, and 3.11 all work for $\widehat{M}$ with an inaccurate reward function and will be reused for proving Theorem 3.4. The only missing piece is a replacement for Lemma 3.9 (which we provide right below), and Theorem 3.4 follows from that directly.

Lemma 3.12. *Given true MDP M, let $\widehat{M}$ be an MDP comprising reward function $\widehat{R}$ and transition function $\widehat{P}$ estimated from n samples for each state-action pair. Let Δ and $\mathcal{R}_\Delta$ be as defined in Theorem 3.4, then*

$$\left\| V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma} \right\|_\infty \leq \frac{2R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}||\Pi_{\mathcal{R}_\Delta,\gamma}|}{\delta}}$$

with probability at least $1-\delta$.

Proof. Similar to the proof of Lemma 3.9, we have

$$\left\| V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma} \right\|_\infty \leq \frac{2}{1-\gamma} \max_{\substack{s\in\mathcal{S},a\in\mathcal{A} \\ \pi\in\,\Pi_{\widehat{R},\gamma}\bigcup\{\pi^\star_{M,\gamma}\}}} \left| \widehat{R}(s,a) + \gamma\langle \widehat{P}(\,\cdot\,|s,a), V^\pi_{M,\gamma}\rangle - Q^\pi_{M,\gamma}(s,a) \right|. \tag{3.26}$$

Applying Hoeffding's inequality and Union Bound to the estimated reward function, we have w.p. at least $1-\delta/2$,

$$\max_{s\in\mathcal{S},a\in\mathcal{A}} \left| \widehat{R}(s,a) - R(s,a) \right| \leq R_{\max}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}|}{\delta}} = \Delta. \tag{3.27}$$

On the other hand, w.p. at least $1-\delta/2$, we have $\forall \pi \in \Pi_{\mathcal{R}_\Delta,\gamma}$ (note that $\mathcal{R}_\Delta$ is deterministic),

$$\left|\widehat{R}(s,a)+\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{M,\gamma}\rangle - Q^{\pi}_{M,\gamma}(s,a)\right| \leq \frac{R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}||\Pi_{\mathcal{R}_\Delta,\gamma}|}{\delta}}. \quad (3.28)$$

By union bound, w.p. at least $1-\delta$, Equation 3.27 and 3.28 will hold simultaneously; the former implies that $\widehat{R} \in \mathcal{R}_\Delta$, which further implies that $\Pi_{\widehat{R},\gamma} \subseteq \Pi_{\mathcal{R}_\Delta,\gamma}$. By definition of $\mathcal{R}_\Delta$, we also know that $\pi^\star_{M,\gamma} \in \mathcal{R}_\Delta$. Combining Equation 3.26 and 3.28, we have

$$\begin{aligned}\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty &\leq \frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi\in\,\Pi_{\widehat{R},\gamma}\bigcup\{\pi^\star_{M,\gamma}\}}}\left|\widehat{R}(s,a)+\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{M,\gamma}\rangle - Q^{\pi}_{M,\gamma}(s,a)\right| \\ &\leq \frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi\in\,\Pi_{\mathcal{R}_\Delta,\gamma}}}\left|\widehat{R}(s,a)+\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{M,\gamma}\rangle - Q^{\pi}_{M,\gamma}(s,a)\right| \\ &\leq \frac{2R_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}||\Pi_{\mathcal{R}_\Delta,\gamma}|}{\delta}}.\end{aligned}$$

□

3.9 Proof of Theorem 3.5

We prove Theorem 3.5 by the following lemma that parallels Lemma 3.9.

Lemma 3.13. *Given the true MDP M, let $\widehat{M}$ be an MDP comprising reward function $\widehat{R}$ and transition function $\widehat{P}$ both estimated from n samples for each state-action pair, then*

$$\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty \leq \frac{2}{1-\gamma}\left(2\max_{\substack{s\in\mathcal{S}\\ a\in\mathcal{A}}}\widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma}) + \frac{3R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n}\log\frac{4|\mathcal{S}||\mathcal{A}|}{\delta}}\right), \quad (3.29)$$

with probability at least $1-\delta$.

Proof. From Equation 3.23 in Lemma 3.10 and Lemma 3.11, we have

$$\begin{aligned}\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty &\leq \frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\left|\widehat{R}(s,a)+\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{M,\gamma}\rangle - Q^{\pi}_{M,\gamma}(s,a)\right| \\ &= \frac{2}{1-\gamma}\max_{s\in\mathcal{S},a\in\mathcal{A}}\ \max_{\pi:S\to A}\left|\widehat{R}(s,a)+\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{M,\gamma}\rangle - Q^{\pi}_{M,\gamma}(s,a)\right|.\end{aligned}$$

Recall that in the statement of Theorem 3.5, we defined $f^{\pi}_{M,\gamma}$ to be the mapping

$(r, s') \mapsto r + \gamma V^{\pi}_{M,\gamma}(s')$. So

$$\begin{aligned}
&\max_{\pi: S \to A} \left| \widehat{R}(s,a) + \gamma \langle \widehat{P}(\,\cdot\,|s,a), V^{\pi}_{M,\gamma} \rangle - Q^{\pi}_{M,\gamma}(s,a) \right| \\
=\ &\max_{\pi: S \to A} \left| \frac{1}{n} \sum_{(r,s') \in D_{s,a}} f^{\pi}_{M,\gamma}(r,s') - \mathbb{E}_{(r,s') \sim \mathbb{P}_{s,a}} \left[f^{\pi}_{M,\gamma}(r,s') \right] \right|,
\end{aligned}$$

where $(r, s') \in D_{s,a}$ means that (r, s') is a sample reward & next-state pair from (s, a) in dataset D, and $\mathbb{P}_{s,a}$ is the underlying true distribution. By noticing that $f^{\pi}_{M,\gamma}$ has function value bounded in $[0, R_{\max}/(1-\gamma)]$, we have the following bound from the standard Rademacher complexity literature (e.g., [Bartlett and Mendelson, 2003]; also see [Balcan, 2011]): for each $s \in \mathcal{S}, a \in \mathcal{A}$, w.p. $\geq 1 - \delta/(|\mathcal{S}||\mathcal{A}|)$,

$$\begin{aligned}
&\max_{\pi: S \to A} \left| \frac{1}{n} \sum_{(r,s') \in D_{s,a}} f^{\pi}_{M,\gamma}(r,s') - \mathbb{E}_{(r,s') \sim \mathbb{P}_{s,a}} \left[f^{\pi}_{M,\gamma}(r,s') \right] \right| \\
\leq\ &\frac{2}{1-\gamma} \left(2\, \widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{F}_{M,\gamma}) + \frac{3R_{\max}}{1-\gamma} \sqrt{\frac{1}{2n} \log \frac{4|\mathcal{S}||\mathcal{A}|}{\delta}} \right).
\end{aligned}$$

The theorem follows directly from union bound and taking the maximal empirical Rademacher complexity among all state-action pairs. □

3.10 Proof of Theorem 3.7

We first prove Proposition 3.6.

Proof of Proposition 3.6. We start with the definition of $V^{\pi}_{\widehat{M},\gamma}$.

$$\begin{aligned}
V^{\pi}_{\widehat{M},\gamma}(s') &= \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} \widehat{R}(s_t, a_t) \;\Big|\; s_1 = s' \,;\, \widehat{P}, \pi \Big] \\
&= \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} \widehat{R}(s_t, a_t) \big(\mathbb{I}(A^{s,a}_t) + \mathbb{I}(\neg A^{s,a}_t) \big) \;\Big|\; s_1 = s' \,;\, \widehat{P}, \pi \Big] \\
&= \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} \widehat{R}(s_t, a_t)\, \mathbb{I}(A^{s,a}_t) \;\Big|\; s_1 = s' \,;\, \widehat{P}, \pi \Big] + V^{\pi}_{\widehat{M}^{-}_{s,a}}(s').
\end{aligned}$$

Note that $\forall t_2 > t_1 \geq 1$, $\mathbb{I}(A^{s,a}_{t_1}) = 1 \;\Rightarrow\; \mathbb{I}(A^{s,a}_{t_2}) = 1$, so the first term in the last line above can be written as:

$$\begin{aligned}
&\mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^{t-1}\,\mathbb{I}(A^{s,a}_t \wedge \neg A^{s,a}_{t-1})\left(\sum_{t'=t}^{\infty}\gamma^{t'-t}\widehat{R}(s'_t, a'_t)\right)\;\Big|\; s_1 = s'\;;\;\widehat{P},\pi\;\Big]\\
&= \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^{t-1}\,\mathbb{I}(A^{s,a}_t \wedge \neg A^{s,a}_{t-1})\,Q^{\pi}_{\widehat{M},\gamma}(s,a)\;\Big|\; s_1 = s'\;;\;\widehat{P},\pi\;\Big]\\
&= \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^{t-1}\,\mathbb{I}(A^{s,a}_t \wedge \neg A^{s,a}_{t-1})\;\Big|\; s_1 = s'\;;\;\widehat{P},\pi\;\Big]\,Q^{\pi}_{\widehat{M},\gamma}(s,a)\\
&= p^{\pi,s,a}_{\widehat{P},\gamma}(s')Q^{\pi}_{\widehat{M},\gamma}(s,a).
\end{aligned}$$

□

We then prove Theorem 3.7 by the following lemma; it is a replacement of Lemma 3.13.

Lemma 3.14. *Given the true MDP M, let $\widehat{M}$ be an MDP comprising reward function $\widehat{R}$ and transition function $\widehat{P}$ both estimated from n samples for each state-action pair, then*

$$\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty \leq \Bigg(2\max_{s\in\mathcal{S},a\in\mathcal{A}}\widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{V}^{s,a}_{\widehat{M},\gamma}) + \frac{2\gamma R_{\max}}{1-\gamma}\max_{s\in\mathcal{S},a\in\mathcal{A}}\widehat{\mathfrak{R}}_{D_{s,a}}(\mathcal{P}^{s,a}_{\widehat{P},\gamma}) \;+\; \frac{3(1+\gamma)R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n}\log\frac{8|\mathcal{S}||\mathcal{A}|}{\delta}}\Bigg), \tag{3.30}$$

with probability at least $1-\delta$.

Proof. Applying Lemma 3.11 but swapping the roles of M and $\widehat{M}$, we have

$$\begin{aligned}
\left\|V^{\pi^\star_{M,\gamma}}_{M,\gamma} - V^{\pi^\star_{\widehat{M},\gamma}}_{M,\gamma}\right\|_\infty &\leq \frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\left|R(s,a) + \gamma\langle P(\,\cdot\,|s,a), V^{\pi}_{\widehat{M},\gamma}\rangle - Q^{\pi}_{\widehat{M},\gamma}(s,a)\right|\\
&\leq \frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\left|R(s,a) + \gamma\langle P(\,\cdot\,|s,a), V^{\pi}_{\widehat{M},\gamma}\rangle - \widehat{R}(s,a) - \gamma\langle\widehat{P}(\,\cdot\,|s,a), V^{\pi}_{\widehat{M},\gamma}\rangle\right|.
\end{aligned}\tag{3.31}$$

Note that at this step we cannot straight-forwardly apply the Rademacher complexity results, as $\widehat{P}(\,\cdot\,|s,a)$ are not independent of $V^{\pi}_{\widehat{M},\gamma}$. Thanks to the decomposition given in Proposition 3.6, we have the above equal to

$$\frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\Big|R(s,a)+\gamma\langle P(\,\cdot\,|s,a),V^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}\rangle+\gamma\langle P(\,\cdot\,|s,a),p^{\pi,s,a}_{\widehat{P},\gamma}\rangle\cdot Q^{\pi}_{\widehat{M},\gamma}(s,a)$$
$$-\widehat{R}(s,a)-\gamma\langle\widehat{P}(\,\cdot\,|s,a),V^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}\rangle-\gamma\langle\widehat{P}(\,\cdot\,|s,a),p^{\pi,s,a}_{\widehat{P},\gamma}\rangle\cdot Q^{\pi}_{\widehat{M},\gamma}(s,a)\Big|.$$

Using the definition of $f^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}$, we have the above upper bounded by

$$\frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\Big|\mathbb{E}_{(r,s')\sim\mathbb{P}_{s,a}}\big[f^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}(r,s')\big]-\frac{1}{n}\sum_{(r,s')\in D_{s,a}}f^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}(r,s')\Big|$$
$$+\frac{2}{1-\gamma}\max_{\substack{s\in\mathcal{S},a\in\mathcal{A}\\ \pi:S\to A}}\Big|\gamma Q^{\pi}_{\widehat{M},\gamma}(s,a)\cdot\Big(\mathbb{E}_{(r,s')\sim\mathbb{P}_{s,a}}\big[p^{\pi,s,a}_{\widehat{P},\gamma}(s')\big]-\frac{1}{n}\sum_{(r,s')\in D_{s,a}}p^{\pi,s,a}_{\widehat{P},\gamma}(s')\Big)\Big|.$$

By our decomposition, $p^{\pi,s,a}_{\widehat{P},\gamma}$ and $f^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}$ are both independent of (s,a), and the lemma follows from applying standard Rademacher complexity results to each of the two terms above (noticing that $p^{\pi,s,a}_{\widehat{P},\gamma}\in[0,1]$ and $V^{\pi}_{\widehat{M}^{-}_{s,a},\gamma}\in[0,R_{\max}/(1-\gamma)]$). □

CHAPTER 4

Doubly Robust Off-policy Evaluation

We have seen in Chapter 3 that model-based cross-validation methods can be effective for selecting the right planning horizon in the tabular setting. However, when the size of the state space is large and a compact state representation is unknown (e.g., the setting of Chapter 5), model-based off-policy evaluation becomes infeasible. In this chapter we investigate the off-policy version of Monte-Carlo policy evaluation, where one aims to estimate the value of a new policy based on sample trajectories collected by a different policy. Existing general methods either have uncontrolled bias or suffer high variance. In this work, we extend the *doubly robust* estimator for bandits to sequential decision-making problems, which gets the best of both worlds: it is guaranteed to be unbiased and can have a much lower variance than the popular importance sampling estimators. We demonstrate the estimator's accuracy in several benchmark problems, and illustrate its use as a subroutine in safe policy improvement. We also provide theoretical results on the inherent hardness of the problem, and show that our estimator can match the lower bound in certain scenarios.

4.1 Introduction

In this chapter we study the *off-policy value evaluation* problem, where one aims to estimate the value of a policy with data collected by another policy [Sutton and Barto, 1998]. This problem is critical in many real-world applications of reinforcement learning (RL), whenever it is infeasible to estimate policy value by running the policy because doing so is expensive, risky, or unethical/illegal. In robotics and business/marketing applications, for instance, it is often risky (thus expensive) to run a policy without an estimate of the policy's quality [Li et al., 2011a, Bottou et al., 2013, Thomas et al., 2015a]. In medical and public-policy domains [Murphy et al., 2001, Hirano et al., 2003], it is often hard to run a controlled experiment to estimate the treatment effect, and off-policy value evaluation is a form of counterfactual reasoning that infers the causal effect of a new intervention from historical data [Holland, 1986, Pearl, 2009].

There are roughly two classes of approaches to off-policy value evaluation. The first is to fit an MDP model from data via regression, and evaluate the policy against the model. Such a regression based approach has a relatively low variance and works well when the model can be learned to satisfactory accuracy. However, for complex real-world problems, it is often hard to specify a function class in regression that is efficiently learnable with limited data while at the same time has a small approximation error. Furthermore, it is in general impossible to estimate the approximation error of a function class, resulting in a bias that cannot be easily quantified. The second class of approaches are based on the idea of importance sampling (IS), which corrects the mismatch between the distributions induced by the target policy and by the behavior policy [Precup et al., 2000]. Such approaches have the salient properties of being unbiased and independent of the size of the problem's state space, but its variance can be too large for the method to be useful when the horizon is long [Mandel et al., 2014].

In this work, we propose a new off-policy value evaluation estimator that can achieve the best of regression based approaches (low variance) and importance sampling based approaches (no bias). Our contributions are three-fold:

1. A simple doubly robust (DR) estimator is proposed for RL that extends and subsumes a previous off-policy estimator for contextual bandits.

2. The estimator's statistical properties are analyzed (Theorem 4.1), which suggests its superiority over previous approaches. Furthermore, in certain scenarios, we prove that the estimator's variance matches the Cramer-Rao lower bound for off-policy value evaluation (Theorem 4.3).

3. On benchmark problems, the new estimator is much more accurate than importance sampling baselines, while remaining unbiased in contrast to regression-based approaches. As an application, we show how such a better estimator can benefit *safe* policy iteration with a more effective policy improvement step.

4.2 Problem Statement and Existing Solutions

In this chapter we focus on the estimation of the H-step discounted value in MDP $(\mathcal{S}, \mathcal{A}, P, R, \gamma, \mu)$ of a given policy π, defined as

$$v^{\pi,H} := \mathbb{E}\Big[\sum_{t=1}^{H} \gamma^{t-1} r_t \mid \pi, s_1 \sim \mu\Big]. \tag{4.1}$$

Recall that μ is the initial distribution introduced in Section 2.1.2, and the dependence on the MDP M is made implicit throughout this chapter. Note that here we are not looking at the infinite-horizon discounted value, but instead its H-step truncated version; the latter can always approximates the former to a desired accuracy when H is set to the effective horizon (see Equation 2.10), and in this chapter we ignore this error due to truncation.[1] For the discussions in this chapter, it will also be convenient to recall the H-step value function of policy π, denoted as $V^{\pi,H}(s)$ and $Q^{\pi,H}(s,a)$ (recall Equation 2.11). Finally, to align with the off-policy evaluation literature in bandits, we will consider the more general setting that reward r_t has mean $R(s_t, a_t)$ but contains additional independent noise.

4.2.1 Off-policy value evaluation

For simplicity, we assume that the data (a set of length-H trajectories) is sampled using a fixed stochastic policy[2] π_0, known as the *behavior policy*. Our goal is to estimate $v^{\pi_1,H}$, the value of a given *target policy* π_1 from data trajectories. (Note that this setup is an instantiation of data collection protocol **(c)** and performance measure **(iv)** in Section 2.3.) Below we review two popular families of estimators for off-policy value evaluation.

Notation Since we are only interested in the value of π_1, the dependence of value functions on policy is omitted. In terms like $V^{\pi_1,H-t+1}(s_t)$, we also omit the dependence on horizon and abbreviate as $V(s_t)$, assuming there are $H+1-t$ remaining steps. Also, all (conditional) expectations are taken with respect to the distribution induced by initial distribution μ and policy π_0, unless stated otherwise. Finally, we

[1] Dealing with this error is routine in theoretical RL literature, and an example can be found in Chapter 6 of this thesis.

[2] Analyses in this paper can be easily extended to handle data trajectories that are associated with different behavior policies.

use the shorthand: $\mathbb{E}_t\big[\,\cdot\,\big] := \mathbb{E}\big[\,\cdot\,\big|\, s_1, a_1, \ldots, s_{t-1}, a_{t-1}\big]$ for conditional expectations, and $\mathbb{V}_t\big[\,\cdot\,\big]$ for variances similarly.

4.2.1.1 Regression estimators

If the true parameters of the MDP are known, the value of the target policy can be computed recursively by the Bellman equations: let $V^0(s) \equiv 0$, and for $h = 1, 2, \ldots, H$,

$$Q^h(s,a) := \mathbb{E}_{s' \sim P(\cdot|s,a)}\big[R(s,a) + \gamma V^{h-1}(s')\big]\,, \tag{4.2}$$

$$V^h(s) := \mathbb{E}_{a \sim \pi_1(\cdot|s)}\big[Q^h(s,a)\big]\,. \tag{4.3}$$

This suggests a two-step, *regression based* procedure for off-policy value evaluation: first, fit an MDP model $\widehat{M}$ from data; second, compute the value function from Equation 4.3 using the estimated parameters $\widehat{P}$ and $\widehat{R}$. Evaluating the resulting value function, $\widehat{V}^H(s)$, on a sample of initial states and the average will be an estimate of $v^{\pi_1,H}$. (Alternatively, one could generate artificial trajectories for evaluation without explicitly referring to a model [Fonteneau et al., 2013].) When an exact state representation is used and each state-action pair appears sufficiently often in the data, such regression estimators have provably low variances and negligible biases [Mannor et al., 2007], and often outperform alternatives in practice [Paduraru, 2013]. Furthermore, this estimator requires minimal knowledge about the behavioral policy, which is often necessary for alternative methods (e.g., importance sampling as is introduced next). As a result, the estimator is robust against misrecording of behavior policy.

However, real-world problems usually have a large or even infinite state space, and many state-action pairs will not be observed even once in the data, rendering the necessity of generalization in model fitting. To generalize, one can either apply function approximation to fitting $\widehat{M}$ [Jong and Stone, 2007, Grünewälder et al., 2012], or to fitting the value function directly [Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998, Dann et al., 2014]. While the use of function approximation makes the problem tractable, it can introduce bias to the estimated value when the MDP parameters or the value function cannot be represented in the corresponding function class. Such a bias is in general hard to quantify from data, thus breaks the credibility of estimations given by regression based approaches [Farahmand and Szepesvári, 2011, Marivate, 2015, Jiang et al., 2015a].

4.2.1.2 Importance sampling (IS) estimators

The IS estimator provides an unbiased estimate of π_1's value by averaging the following function of each trajectory $(s_1, a_1, r_1, \ldots, s_{H+1})$ in the data: define the per-step importance ratio as $\rho_t := \pi_1(a_t|s_t)/\pi_0(a_t|s_t)$, and the cumulative importance ratio $\rho_{1:t} := \prod_{t'=1}^{t} \rho_{t'}$; the basic (trajectory-wise) IS estimator, and an improved step-wise version are given as follows:

$$\hat{v}_{\text{IS}} := \rho_{1:H} \cdot \left(\sum_{t=1}^{H} \gamma^{t-1} r_t \right), \tag{4.4}$$

$$\hat{v}_{\text{step-IS}} := \sum_{t=1}^{H} \gamma^{t-1} \rho_{1:t} \, r_t. \tag{4.5}$$

Given a dataset D, the IS estimator is simply the average estimate over the trajectories, namely $\frac{1}{|D|} \sum_{i=1} V_{\text{IS}}^{(i)}$, where $|D|$ is the number of trajectories in D and $V_{\text{IS}}^{(i)}$ is IS applied to the i-th trajectory. (This averaging step will be omitted for the other estimators in the rest of this chapter, and we will only specify the estimate for a single trajectory). Typically, IS, even the step-wise version, suffers from very high variance, which easily grows exponentially in horizon.

A variant of IS, weighted importance sampling (WIS), is a biased but consistent estimator, given as follows together with its step-wise version: define $w_t = \sum_{i=1}^{|D|} \rho_{1:t}^{(i)}/|D|$ as the average cumulative important ratio at horizon t in a dataset D, then from each trajectory in D, the estimates given by trajectory-wise and step-wise WIS are respectively

$$\hat{v}_{\text{WIS}} = \frac{\rho_{1:H}}{w_H} \big(\sum_{t=1}^{H} \gamma^{t-1} r_t \big), \tag{4.6}$$

$$\hat{v}_{\text{step-WIS}} = \sum_{t=1}^{H} \gamma^{t-1} \frac{\rho_{1:t}}{w_t} r_t \,. \tag{4.7}$$

WIS has lower variance than IS, and its step-wise version is considered as the most practical point estimator in the IS family [Precup, 2000, Thomas, 2015]. We will compare to the step-wise IS/WIS baselines in the experiments.

4.2.2 Doubly robust estimator for contextual bandits

Contextual bandits may be considered as MDPs with horizon 1, and the sample trajectories take the form of (s, a, r). Suppose now we are given an estimated reward function $\widehat{R}$, possibly from performing regression over a *separate* dataset, then the doubly robust estimator for contextual bandits [Dudík et al., 2011] is defined as:

$$\hat{v}_{\mathrm{DR}} := \widehat{V}(s) + \rho \left(r - \widehat{R}(s, a) \right), \tag{4.8}$$

where $\rho := \frac{\pi_1(a|s)}{\pi_0(a|s)}$ and $\widehat{V}(s) := \sum_a \pi_1(a|s)\widehat{R}(s, a)$. It is easy to verify that $\widehat{V}(s) = \mathbb{E}_{a \sim \pi_0}\left[\rho \widehat{R}(s, a)\right]$, as long as $\widehat{R}$ and ρ are independent, which implies the unbiasedness of the estimator. Furthermore, if $\widehat{R}(s, a)$ is a good estimate of r, the magnitude of $r - \widehat{R}(s, a)$ can be much smaller than that of r. Consequently, the variance of $\rho(r - \widehat{R}(s, a))$ *tends to* be smaller than that of ρr, implying that DR often has a lower variance than IS [Dudík et al., 2011].

In the case where the importance ratio ρ is unknown, DR estimates both ρ and the reward function from data using some parametric function classes. The name "doubly robust" refers to fact that if *either* function class is properly specified, the DR estimator is asymptotically unbiased, offering two chances to ensure consistency. In this paper, however, we are only interested in DR's variance-reduction benefit.

Requirement of independence In practice, the target policy π_1 is often computed from data, and for DR to stay unbiased, π_1 should not depend on the samples used in Equation 4.8; the same requirement applies to IS. While $\widehat{R}$ should be independent of such samples as well, it is not required that π_1 and $\widehat{R}$ be independent of each other. For example, we can use the same dataset to compute π_1 and $\widehat{R}$, although an independent dataset is still needed to run the DR estimator in Equation 4.8. In other situations where π_1 is given directly, to apply DR we can randomly split the data into two parts, one for fitting $\widehat{R}$ and the other for applying Equation 4.8. The same requirements and procedures apply to the sequential case (discussed below). In Section 4.5, we will empirically validate our extension of DR in both kinds of situations.

4.3 DR estimator for the sequential setting

4.3.1 The estimator

We now extend the DR estimator for bandits to the sequential case. A key observation is that Equation 4.5 can be written in a recursive form. Define $\hat{v}^{0}_{\text{step-IS}} := 0$, and for $t = 1, \ldots, H$,

$$\hat{v}^{H+1-t}_{\text{step-IS}} := \rho_t \left(r_t + \gamma\, \hat{v}^{H-t}_{\text{step-IS}} \right). \tag{4.9}$$

It can be shown that $\hat{v}^{H}_{\text{step-IS}}$ is equivalent to $\hat{v}_{\text{step-IS}}$ given in Equation 4.5. While the rewriting is straight-forward, the recursive form provides a novel and interesting insight that is key to the extension of the DR estimator: that is, we can view the step-wise importance sampling estimator as dealing with a bandit problem at each horizon $t = 1, \ldots, H$, where s_t is the context, a_t is the action taken, and the observed stochastic return is $r_t + \gamma\, \hat{v}^{H-t}_{\text{step-IS}}$, whose expected value is $Q(s_t, a_t)$. Then, if we are supplied with $\widehat{Q}$, an estimate of Q (possibly via regression on a separate dataset), we can apply the bandit DR estimator at each horizon, and obtain the following unbiased estimator: define $\hat{v}^{0}_{\text{DR}} := 0$, and

$$\hat{v}^{H+1-t}_{\text{DR}} := \widehat{V}(s_t) + \rho_t \Big(r_t + \gamma\, \hat{v}^{H-t}_{\text{DR}} - \widehat{Q}(s_t, a_t) \Big). \tag{4.10}$$

The DR estimate of the policy value is then $\hat{v}_{\text{DR}} := \hat{v}^{H}_{\text{DR}}$.

Implementation Note Recall that the dependence of $\widehat{V}$ and $\widehat{Q}$ on the remaining number of steps is omitted (see Section 5.2.1). When computed from an estimated MDP model, the value functions for different number of remaining steps may be obtained by applying Bellman update operator iteratively H times starting from $\widehat{V}^{0}(s) \equiv 0$.

4.3.2 Variance analysis

In this section, we analyze the variance of DR in Theorem 4.1 and show that DR is preferable than step-wise IS when a good value function $\widehat{Q}$ is available. The analysis is given in the form of the variance of the estimate for a *single* trajectory, and the variance of the estimate averaged over a dataset D will be that divided by $|D|$ due to the *i.i.d.* nature of D. The proof of the theorem can be found in Section 4.7.

Theorem 4.1. *$\hat{v}_{DR}$ is an unbiased estimator of $v^{\pi_1,H}$, whose variance is given recursively as follows: $\forall t = 1, \ldots, H$,*

$$\begin{aligned}\mathbb{V}_t\Big[\hat{v}_{DR}^{H+1-t}\Big] = \mathbb{V}_t\Big[V(s_t)\Big] + \mathbb{E}_t\Big[\mathbb{V}_t\Big[\rho_t \Delta(s_t, a_t) \;\Big|\; s_t\Big]\Big] \\ + \mathbb{E}_t\Big[\rho_t^2\, \mathbb{V}_{t+1}\Big[r_t\Big]\Big] + \mathbb{E}_t\Big[\gamma^2 \rho_t^2\, \mathbb{V}_{t+1}\Big[\hat{v}_{DR}^{H-t}\Big]\Big],\end{aligned} \tag{4.11}$$

where $\Delta(s_t, a_t) := \widehat{Q}(s_t, a_t) - Q(s_t, a_t)$ and $\mathbb{V}_{H+1}\Big[\hat{v}_{DR}^{0} \;\Big|\; s_H, a_H\Big] = 0$.

On the RHS of Equation 4.11, the first 3 terms are variances due to different sources of randomness at time step t: state transition randomness, action stochasticity in π_0, and reward randomness, respectively; the 4th term contains the variance from future steps. The key conclusion is that DR's variance depends on $\widehat{Q}$ via the error function $\Delta = \widehat{Q} - Q$ in the 2nd term, hence DR with a good $\widehat{Q}$ will enjoy reduced variance, and in general outperform step-wise IS as the latter is simply DR's special case with a trivial value function $\widehat{Q} \equiv 0$.

4.3.3 Confidence intervals

As mentioned in the introduction, an important motivation for off-policy value evaluation is to guarantee safety before deploying a policy. For this purpose, we have to characterize the uncertainty in our estimates, usually in terms of a confidence interval (CI). The calculation of CIs for DR is straight-forward, since DR is an unbiased estimator applied to *i.i.d.* trajectories and standard concentration bounds apply. For example, Hoeffding's inequality states that for random variables with bounded range b, the deviation of the average from n independent samples from the expected value is at most $b\sqrt{\frac{1}{2n}\log\frac{2}{\delta}}$ with probability at least $1-\delta$. In the case of DR, $n = |D|$ is the number of trajectories, δ the chosen confidence level, and b the range of the estimate, which is a function of the maximal magnitudes of r_t, $\widehat{Q}(s_t, a_t)$, ρ_t and γ. The application of more sophisticated bounds for off-policy value evaluation in RL can be found in Thomas et al. [2015a]. In practice, however, strict CIs are usually too pessimistic, and normal approximations are used instead [Thomas et al., 2015b]. In the experiments, we will see how DR with normally approximated CIs can lead to more effective and reliable policy improvement than IS.

4.3.4 An extension

From Theorem 4.1, it is clear that DR only reduces the variance due to action stochasticity, and may suffer a large variance even with a perfect Q-value function $\widehat{Q} = Q$, as long as the MDP has substantial stochasiticity in rewards and/or state transitions. It is, however, possible to address such a limitation. For example, one modification of DR that further reduces the variance in state transitions is:

$$\hat{v}_{\text{DR-v2}}^{H+1-t} = \widehat{V}(s_t) + \rho_t\Big(r_t + \gamma\hat{v}_{\text{DR-v2}}^{H-t} - \widehat{R}(s_t, a_t) - \gamma\widehat{V}(s_{t+1})\frac{\widehat{P}(s_{t+1}|s_t, a_t)}{P(s_{t+1}|s_t, a_t)}\Big), \qquad (4.12)$$

where $\widehat{P}$ is the transition probability of the MDP model that we use to compute $\widehat{Q}$. While we can show that this estimator is unbiased and reduces the state-transition-induced variance with a good reward & transition functions $\widehat{R}$ and $\widehat{P}$ (we omit proof), it is impractical as the true transition function P is unknown. However, in problems where we are confident that the transition dynamics can be estimated accurately (but the reward function may be poorly estimated), we can assume that $P(\cdot) = \widehat{P}(\cdot)$, and the last term in Equation 4.12 becomes simply $\gamma\widehat{V}(s_{t+1})$. This generally reduces more variance than the original DR at the cost of introducing a small bias. The bias is bounded in Proposition 4.2, whose proof is deferred to Section 4.8. In Section 4.5.1.3 we will demonstrate the use of such an estimator by an experiment.

Proposition 4.2. *Define $\epsilon = \max_{s,a} \|\widehat{P}(\cdot|s, a) - P(\cdot|s, a)\|_1$. Then, the bias of DR-v2, computed by Equation 4.12 with the approximation $\widehat{P}/P \equiv 1$, is bounded by $\epsilon V_{\max} \sum_{t=1}^{H} \gamma^t$, where $V_{\max}$ is a bound on the magnitude of $\widehat{V}$.*

4.4 Hardness of Off-policy Value Evaluation

In Section 4.3.4, we showed the possibility of reducing variance due to state transition stochasticity in a special scenario. A natural question is whether there exists an estimator that can reduce such variance without relying on strong assumptions like $\widehat{P} \approx P$. In this section, we answer this question by providing hardness results on off-policy value evaluation via the *Cramer-Rao lower bound* (or *C-R bound* for short), and comparing the C-R bound to the variance of DR.

Before stating the results, we emphasize that, as in other estimation problems, the C-R bound depends crucially on how the MDP is parameterized, because the parameterization captures our prior knowledge about the problem. In general, the more structural knowledge is encoded in parameterization, the easier it is to recover the true parameters from data, and the lower the C-R bound will be. While strong assumptions (*e.g.*, parametric form of value function) are often made in the *training* phase to make RL problems tractable, one may not want to count them as prior knowledge in *evaluation*, as every assumption made in evaluation decreases the credibility of the value estimate. (This is why regression-based methods are not trustworthy; see Section 4.2.1.1.) Therefore, we first present the result for the hardest case when no assumptions (other than discrete decisions & outcomes) – especially the Markov assumption that the last observation is a state – are made, to ensure the most credible estimate. A relaxed case is discussed afterwards.

Definition 4.1. An MDP is a *discrete tree MDP* if

- State is represented by history: that is, $s_t = h_t$, where $h_t := o_1 a_1 \cdots o_{t-1} a_{t-1} o_t$. The o_t's are called *observations*. We assume discrete observations and actions.
- Initial states take the form of $s = o_1$. Upon taking action a, a state $s = h$ can only transition to a next state in the form of $s' = hao$, with probability $P(o|h,a)$.
- As a simplification, we assume $\gamma = 1$, and non-zero rewards only occur at the end of each trajectory. An additional observation o_{H+1} encodes the reward randomness so that reward function $R(h_{H+1})$ is deterministic. In this case, the MDP is solely parameterized by transition probabilities.

Theorem 4.3. *For discrete tree MDPs, the variance of any unbiased off-policy value estimator is lower bounded by*

$$\sum_{t=1}^{H+1} \mathbb{E}\Big[\rho_{1:(t-1)}^2 \mathbb{V}_t\big[V(s_t)\big]\Big]. \tag{4.13}$$

Observation 4.4. *The variance of DR applied to a discrete tree MDP when $\widehat{Q} = Q$ is equal to Equation 4.13.*

The theorem follows from Cramer-Rao bound (CRB) for the off-policy evaluation problem, and the claim follows directly by unfolding the recursive form of Equation 4.11 and noticing that $\Delta \equiv 0$, $\mathbb{V}_{t+1}\big[r_t\big] \equiv 0$ for $t = 1, \ldots, H-1$, and $\mathbb{V}_{H+1}\big[V(s_{H+1})\big]$ is just a re-writing of $\mathbb{V}_{H+1}\big[r_H\big]$.

Proof of Observation 4.4. The result follows directly by unfolding the recursion in Equation 4.11 and noticing that $\Delta \equiv 0$, $\mathbb{V}_{t+1}\big[r_t\big] \equiv 0$ for $t < H$, and $\mathbb{V}_{H+1}\big[V(s_{H+1})\big] = \mathbb{V}_{H+1}\big[r_H\big]$. □

Implication When minimal prior knowledge is available, the lower bound in Theorem 4.3 equals the variance of DR with a perfect Q-value function, hence the part of variance due to state transition stochasticity (which DR fails to improve even with a good Q-value function) is intrinsic to the problem and cannot be eliminated without extra knowledge. Moreover, the more accurate $\widehat{Q}$ is, the lower the variance DR tends to have. A related hardness result is given by Li et al. [2015a] for MDPs with *known* transition probabilities.

Relaxed Case In Section 4.9, we discuss a relaxed case where the MDP has a Directed Acyclic Graph (DAG) structure, allowing different histories of the same length to be identified as the same state, making the problem easier than the tree case. The two cases share almost identical proofs, and below we give a concise proof of Theorem 4.3; see Section 4.9 for a fully expanded version.

Proof of Theorem 4.3. In the proof, it will be convenient to index rows and columns of a matrix (or vector) by histories, so that $A_{h,h'}$ denotes the (h, h') entry of matrix A. Furthermore, given a real-valued function f, $[f(h, h')]_{h,h'}$ denotes a matrix whose (h, h') entry is given by $f(h, h')$.

We parameterize a discrete tree MDP by $\mu(o)$ and $P(o|h, a)$, for h of length $1, \ldots, H$. For convenience, we treat $\mu(o)$ as $P(o|\emptyset)$, and the model parameters can be encoded as a vector θ with $\theta_{hao} = P(o|h, a)$, where ha contains $|ha| = 0, \ldots, H$ alternating observations & actions.

These parameters are subject to the normalization constraints that have to be taken into consideration in the C-R bound, namely $\forall h, a, \sum_{o \in O} P(o|h, a) = 1$. In matrix form, we have $F\theta = \mathbf{1}$, where F is a block-diagonal matrix with each block being a row vector of 1's; specifically, $F_{ha,h'a'o} = \mathbb{I}(ha = h'a')$. Note that F is the Jacobian of the constraints. Let U be a matrix whose column vectors consist of an orthonormal basis for the null space of F. From Moore Jr [2010, Eqn. (3.3) and Corollary 3.10], we obtain a Constrained Cramer-Rao Bound (CCRB):

$$KU(U^\top IU)^{-1}U^\top K^\top, \tag{4.14}$$

where I is the Fisher Information Matrix (FIM) without taking the constraints into consideration, and K the Jacobian of the quantity $v^{\pi_1,H}$ that we want to estimate. Our calculation of the CCRB consists of four main steps.

1) Calculation of I: By definition, the FIM I is computed as $\mathbb{E}\Big[\Big(\frac{\partial \log P_0(h_{H+1})}{\partial \theta}\Big)\Big(\frac{\partial \log P_0(h_{H+1})}{\partial \theta}\Big)^\top\Big]$, with $P_0(h_{H+1}) := \mu(o_1)\pi_0(a_1|o_1)P(o_2|o_1, a_1)\ldots P(o_{H+1}|h_H, a_H)$ being the probability of observing h_{H+1} under policy π_0.

Define a new notation $g(h_{H+1})$ as a vector of indicator functions, such that $g(h_{H+1})_{hao} = 1$ whenever hao is a prefix of h_{H+1}. Using this notation, we have $\frac{\partial \log P_0(h_{H+1})}{\partial \theta} = \theta^{\circ -1} \circ g(h_{H+1})$, where $\circ$ denotes element-wise power/multiplication. We rewrite the FIM as $I = \mathbb{E}\Big[[\theta_h^{-1}\theta_{h'}^{-1}]_{h,h'} \circ \big(g(h_{H+1})g(h_{H+1})^\top\big)\Big] = [\theta_h^{-1}\theta_{h'}^{-1}]_{h,h'} \circ \mathbb{E}\big[g(h_{H+1})g(h_{H+1})^\top\big]$. Now we compute $\mathbb{E}\big[g(h_{H+1})g(h_{H+1})^\top\big]$. This matrix takes 0 in all the entries indexed by hao and $h'a'o'$ when neither of the two strings is a prefix of the other. For the other entries, without loss of generality, assume $h'a'o'$ is a prefix of hao; the other case is similar as I is symmetric. Since $g(h_{H+1})_{hao}g(h_{H+1})_{h'a'o'} = 1$ if and only if hao is a prefix of h_{H+1}, we have $\mathbb{E}\big[g(h_{H+1})_{(hao)} \cdot g(h_{H+1})_{(h'a'o')}\big] = P_0(hao)$, and consequently $I_{(hao),(h'a'o')} = \frac{P_0(hao)}{P(o|h,a)P(o'|h',a')} = \frac{P_0(ha)}{P(o'|h',a')}$.

2) Calculation of $(U^\top I U)^{-1}$: Since I is quite dense, it is hard to compute the inverse of $U^\top I U$ directly. Note, however, that for any matrix X with matching dimensions, $U^\top I U = U^\top(F^\top X^\top + I + XF)U$, because by definition U is orthogonal to F. Observing this, we design X to make $D = F^\top X^\top + I + XF$ diagonal so that $U^\top D U$ is easy to invert. This is achieved by letting $X_{(h'a'o'),(ha)} = 0$ except when $h'a'o'$ is a prefix of ha, in which case we set $X_{(h'a'o'),(ha)} = -\frac{P_0(ha)}{P(o'|h',a')}$. It is not hard to verify that D is diagonal with $D_{(hao),(hao)} = I_{(hao),(hao)} = \frac{P_0(ha)}{P(o|h,a)}$.

With the above trick, we have $(U^\top I U)^{-1} = (U^\top D U)^{-1}$. Since CCRB is invariant to the choice of U, we choose U to be diag($\{U_{(ha)}\}$), where $U_{(ha)}$ is a diagonal block with columns forming an orthonormal basis of the null space of the none-zero part of $F_{(ha),(\cdot)}$ (an all-1 row vector). It is easy to verify that such U exists and is column orthonormal, with $FU = [0]_{(ha),(ha)}$. We also rewrite $D = \text{diag}(\{D_{(ha)}\})$ where $D_{(ha)}$ is a diagonal matrix with $(D_{(ha)})_{o,o} = \frac{P_0(ha)}{P(o|h,a)}$, and we have $U(U^\top I U)^{-1}U^\top = \text{diag}(\{U_{(ha)}\big(U_{(ha)}^\top D_{(ha)} U_{(ha)}\big)^{-1} U_{(ha)}^\top\})$.

The final step is to notice that each block in the expression above is simply $\frac{1}{P_0(ha)}$ times the CCRB of a multinomial distribution $p = P(\cdot|h, a)$, which is diag$(p) - pp^\top$ [Moore Jr, 2010, Eqn. (3.12)].

3) Calculation of K: Recall that we want to estimate

$$v = v^{\pi_1, H} = \textstyle\sum_{o_1} \mu(o_1) \sum_{a_1} \pi_1(a_1|o_1) \ \cdots \sum_{o_{H+1}} P(o_{H+1}|h_H, a_H) R(h_{H+1})\,.$$

Its Jacobian, $K = \partial v / \partial \theta$, can be computed by $K_{(hao)} = P_1(ha)V(hao)$, where $P_1(o_1 a_1 \cdots o_t a_t) := \mu(o_1)\pi_1(a_1) \cdots P(o_t|h_{t-1}, a_{t-1})\pi_1(a_t|h_t)$ is the probability of observing a sequence under policy π_1.

4) The C-R bound: Putting all the pieces together, Equation 4.14 is equal to

$$\begin{aligned} &\sum_{ha} \frac{P_1(ha)^2}{P_0(ha)} \Big(\sum_o P(o|h,a)V(hao)^2 - \big(\sum_o P(o|h,a)V(hao) \big)^2 \Big) \\ &= \textstyle\sum_{t=0}^{H} \sum_{|ha|=t} P_0(ha) \frac{P_1(ha)^2}{P_0(ha)^2} \mathbb{V}\big[V(hao) \mid h, a\big]. \end{aligned}$$

Noticing that $P_1(ha)/P_0(ha)$ is the cumulative importance ratio, and $\sum_{|ha|=t} P_0(ha)(\cdot)$ is taking expectation over sample trajectories, the lower bound is equal to

$$\sum_{t=0}^{H} \mathbb{E}\Big[\rho_{1:t}^2 \mathbb{V}_{t+1}\big[V(s_{t+1})\big]\Big] = \sum_{t=1}^{H+1} \mathbb{E}\Big[\rho_{1:(t-1)}^2 \mathbb{V}_t\big[V(s_t)\big]\Big]. \qquad \square$$

4.5 Experiments

Throughout this section, we will be concerned with the comparison among the following estimators. For compactness, we drop the prefix "step-wise" from step-wise IS & WIS. Further experiment details can be found in Appendix **??**.

1. (IS) Step-wise IS of Equation 4.5;
2. (WIS) Step-wise WIS of Equation 4.7;
3. (REG) Regression estimator (details to be specified for each domain in the "model fitting" paragraphs);
4. (DR) Doubly robust estimator of Equation 4.10;
5. (DR-bsl) DR with a state-action independent $\widehat{Q}$.

4.5.1 Comparison of Mean Squared Errors

In these experiments, we compare the accuracy of the point estimate given by each estimator. For each domain, a policy π_{train} is computed as the optimal policy of the MDP model estimated from a training dataset D_{train} (generated using π_0), and the target policy π_1 is set to be $(1-\alpha)\pi_{\text{train}} + \alpha\pi_0$ for $\alpha \in \{0, 0.25, 0.5, 0.75\}$. The parameter α controls similarity between π_0 and π_1. A larger α tends to make off-policy evaluation easier, at the cost of yielding a more conservative policy when

π_{train} is potentially of high quality.

We then apply the five estimators on a separate dataset D_{eval} to estimate the value of π_1, compare the estimates to the groundtruth value, and take the average estimation errors across multiple draws of D_{eval}. Note that for the DR estimator, the supplied Q-value function $\widehat{Q}$ should be independent of the data used in Equation 4.10 to ensure unbiasedness. We therefore split D_{eval} further into two subsets D_{model} and D_{test}, estimate $\widehat{Q}$ from D_{model} and apply DR on D_{test}.

In the above procedure, DR does not make full use of data, as the data in D_{model} do not go into the sample average in Equation 4.10. To address this issue, we propose a more data-efficient way of applying DR in the situation when $\widehat{Q}$ has to be estimated from (a subset of) D_{eval}, and we call it *k-fold DR*, inspired by k-fold cross validation in supervised learning: we partition D_{eval} into k subsets, apply Equation 4.8 to each subset with $\widehat{Q}$ estimated from the remaining data, and finally average the estimate over all subsets. Since the estimate from each subset is unbiased, the overall average remains unbiased, and has lower variance since all trajectories go into the sample average. We only show the results of 2-fold DR as model fitting is time-consuming.

4.5.1.1 Mountain Car

Domain description Mountain car is a widely used benchmark problem for RL with a 2-dimensional continuous state space (position and velocity) and deterministic dynamics [Singh and Sutton, 1996]. The state space is $[-1.2, 0.6] \times [-0.07, 0.07]$, and there are 3 discrete actions. The agent receives -1 reward every time step with a discount factor 0.99, and an episode terminates when the first dimension of state reaches the right boundary. The initial state distribution is set to uniformly random, and behavior policy is uniformly random over the 3 actions. The typical horizon for this problem is 400, which can be too large for IS and its variants, therefore we accelerate the dynamics such that given (s, a), the next state s' is obtained by calling the original transition function 4 times holding a fixed, and we set the horizon to 100. A similar modification was taken by Thomas [2015], where every 20 steps are compressed as one step.

Model Construction The model we construct for this domain uses a simple discretization (state aggregation): the two state variables are multiplied by 2^6 and 2^8 respectively and the rounded integers are treated as the abstract state. We then estimate the model parameters from data using a tabular approach. Unseen aggregated

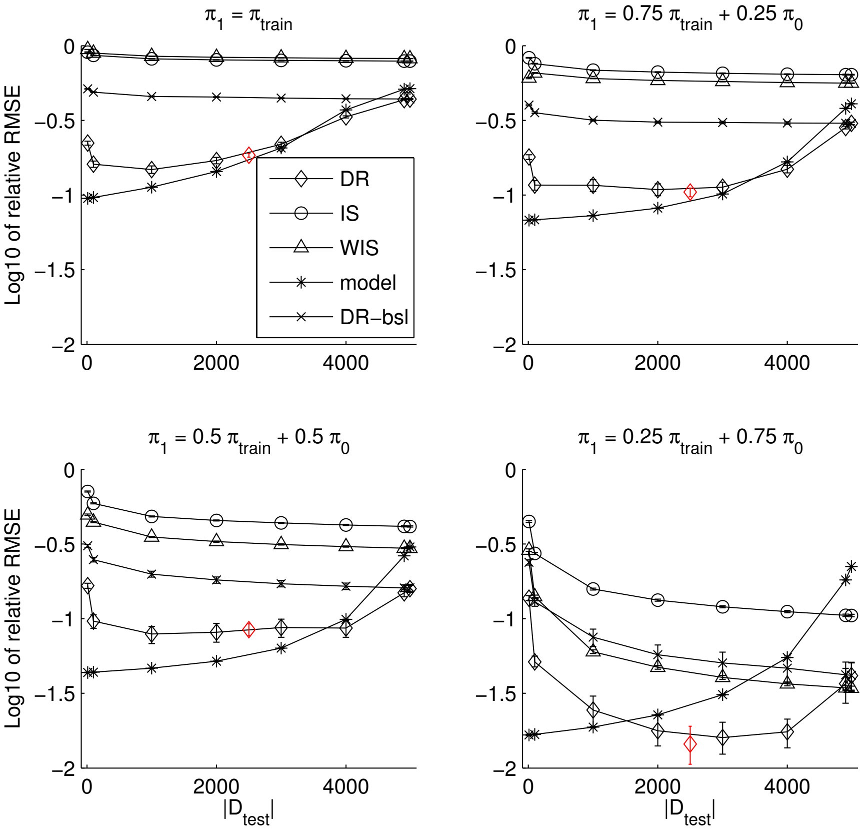

Figure 4.1: Comparison of the methods as point estimators on Mountain Car. 5000 trajectories are generated for off-policy evaluation, and all the results are from over 4000 runs. The subgraphs correspond to the target policies produced by mixing π_{train} and π_0 with different portions. X-axes show the size of D_{test}, the part of the data used for IS/WIS/DR/DR-bsl. The remaining data are used by the regression estimator (REG; DR uses it as $\widehat{Q}$). Y-axes show the RMSE of the estimates divided by the true value in logarithmic scale. We also show the error of 2-fold DR as an isolated point ($\diamond$).

state-action pairs are assumed to have reward $R_{\min} = -1$ and a self-loop transition. Both the models that produces π_{train} and that used for off-policy evaluation are constructed in the same way.

Data sizes and other details The dataset sizes are $|D_{\text{train}}| = 2000$ and $|D_{\text{eval}}| = 5000$. We split D_{eval} such that $D_{\text{test}} \in \{10, 100, 1000, 2000, 3000, 4000, 4900, 4990\}$. DR-bsl uses the step-dependent constant function $\widehat{Q}(s_t, a_t) = \frac{R_{\min}(1-\gamma^{H-t+1})}{1-\gamma}$. Since the estimators in the IS family typically has a highly skewed distribution, the estimates can occasionally go largely out of range, and we crop such outliers in $[V_{\min}, V_{\max}]$ to ensure that we can get statistically significant experiment results within a reasonable

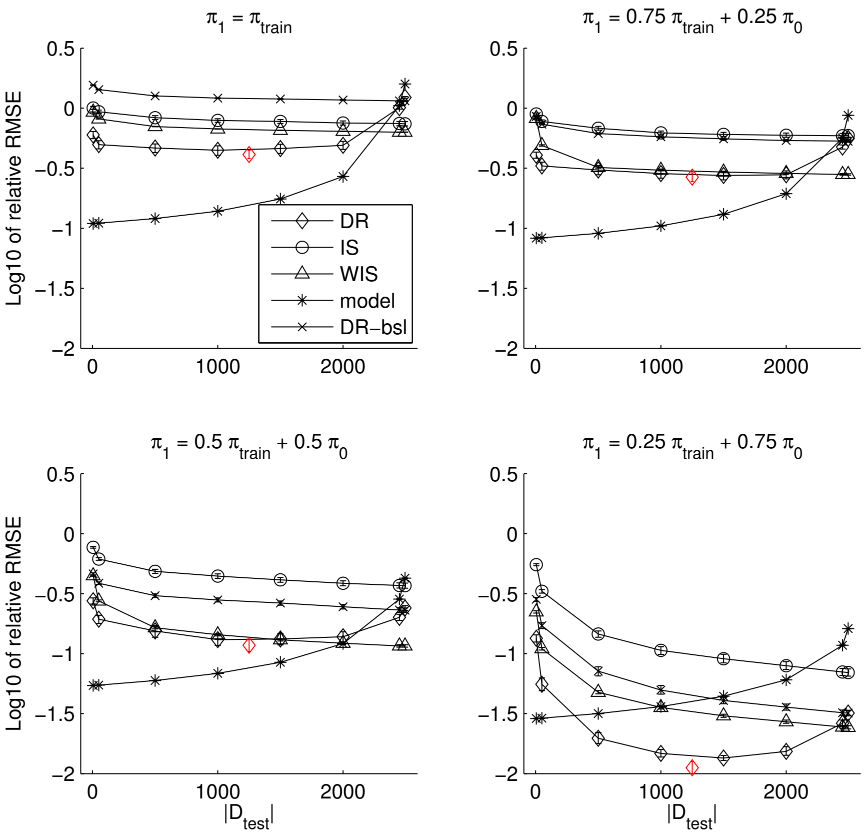

Figure 4.2: Comparison of the methods as point estimators on Sailing (4000 runs). 2500 trajectories are used in off-policy evaluation.

number of simulations. The same treatment is also applied to the experiment on Sailing.

Results See Figure 4.1 for the errors of IS/WIS/DR-bsl/DR on D_{test}, and REG on D_{model}. As $|D_{\text{test}}|$ increases, IS/WIS gets increasingly better, while REG gets worse as D_{model} contains less data. Since DR depends on both halves of the data, it achieves the best error at some intermediate $|D_{\text{test}}|$, and beats using all the data for IS/WIS in all the 4 graphs. DR-bsl shows the accuracy of DR with $\widehat{Q}$ being a constant guess, and it already outperforms IS/WIS most of the time.

4.5.1.2 Sailing

Domain description The sailing domain [Kocsis and Szepesvári, 2006] is a stochastic shortest-path problem, where the agent sails on a grid (in our experiment, a map of size 10×10) with wind blowing in random directions, aiming at the terminal location on the top-right corner. The state is represented by 4 integer variables, representing either location or direction. At each step, the agent chooses to move in one of the 8 directions, (moving against the wind or running off the grid is prohibited), and receives a negative reward that depends on moving direction, wind direction, and other factors, ranging from $R_{\min} = -3 - 4\sqrt{2}$ to $R_{\max} = 0$ (absorbing). The problem is non-discounting, and we use $\gamma = 0.99$ for easy convergence when computing π_{train}.

Model fitting We apply Kernel-based Reinforcement Learning [Ormoneit and Sen, 2002] and supply a smoothing kernel in the joint space of states and actions. The kernel we use takes the form $\exp(-\|\cdot\|/b)$, where $\|\cdot\|$ is the ℓ_2-distance in $S \times A$,[3] and b is the kernel bandwidth, set to 0.25.

Data sizes and other details The data sizes are $|D_{\text{train}}| = 1000$ and $|D_{\text{eval}}| = 2500$, and we split D_{eval} such that $D_{\text{test}} \in \{5, 50, 500, 1000, 1500, 2000, 2450, 2495\}$. DR-bsl uses the step-dependent constant function $\widehat{Q}(s_t, a_t) = \frac{R_{\min}}{2} \frac{1-\gamma^{H-t+1}}{1-\gamma}$, for the reason that in Sail $R_{\min}$ is rarely reached hence too pessimistic as a rough estimate of the magnitude of reward obtained per step.

Results See Figure 4.2. The results are qualitatively similar to Mountain Car results in Figure 4.1, except that: (1) WIS is as good as DR in the 2nd and 3rd graph; (2) in the 4th graph, DR with a 3:2 split outperforms all the other estimators (including the regression estimator) with a significant margin, and a further improvement is achieved by 2-fold DR.

4.5.1.3 KDD Cup 1998 donation dataset

In the last domain, we use the donation dataset from KDD Cup 1998 [Hettich and Bay, 1999], which records the email interactions between an agent and potential donators. A state contains 5 integer features, and there are 12 discrete actions. All trajectories are 22-steps long and there is no discount. The policy π_{train} is generated by training a recurrent neural network on the original data [Li et al., 2015c].

Since no groundtruth values are available for the target policies, we fit a simulator from the true data, and use it as groundtruth for everything henceforward: the true value of a target policy is computed by Monte-Carlo policy evaluation in

[3] The difference of two directions is defined as the angle between them (in degrees) divided by $45°$. For computational efficiency, the kernel function is cropped to 0 whenever two state-action pairs deviate more than 1 in any of the dimensions.

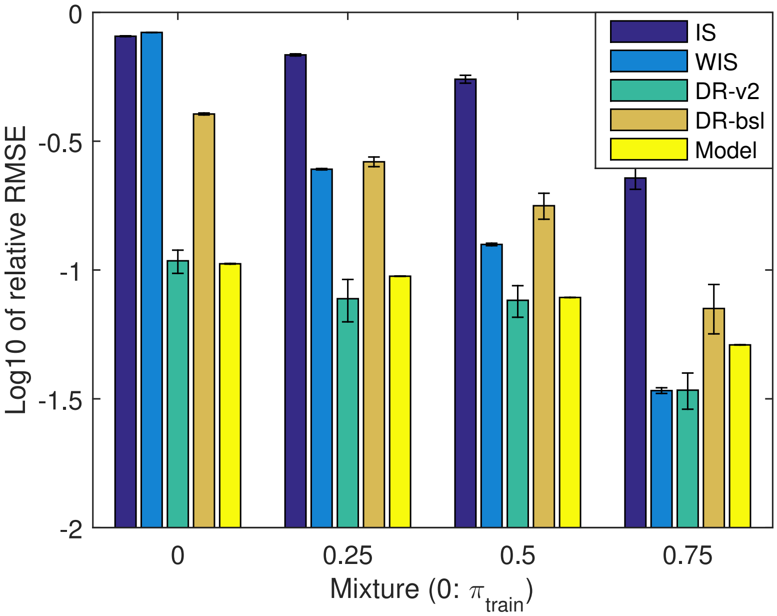

Figure 4.3: Results on the donation dataset, averaged over about 5000 runs. DR-v2 is the estimator in Equation 4.12 with the 2-fold trick. The other estimators are applied to the whole dataset. X-axis shows the portion of which π_0 is mixed into π_{train}.

the simulator, and the off-policy evaluation methods use data generated from the simulator (under a uniformly random policy). The size of dataset generated for off-policy evaluation is equal to that of the true dataset (the one we use to fit the simulator; there are 3754 trajectories in that dataset).

Among the compared estimators, we replace DR with DR-v2 (Section 4.3.4; reason explained below), and use the 2-fold trick.

The MDP model used to compute $\widehat{Q}$ is estimated as follows: each state variable is assumed to evolve independently (a reasonable assumption for this dataset), and the marginal transition probabilities are estimated using a tabular approach, which is exactly how the simulator is fit from real data. Reward function, on the other hand, is fit by linear regression using the first 3 state features (on the contrast, all the 5 features are used when fitting the simulator's reward function). Consequently, we get a model with an almost perfect transition function and a relatively inaccurate reward function, and DR-v2 is supposed to work well in such a situation.[4] See Figure 4.3 for the results; DR-v2 is the best estimator in all situations: it beats WIS

[4] Since there are many possible next-states, for computational efficiency we use a sparse-sample approach when estimating $\widehat{Q}$ using the fitted model $\widehat{M}$: for each (s, a), we randomly sample several next-states from $\widehat{P}(\cdot|s, a)$, and cache them as a particle representation for the next-state distribution. The number of particles is set to 5 which is enough to ensure high accuracy.

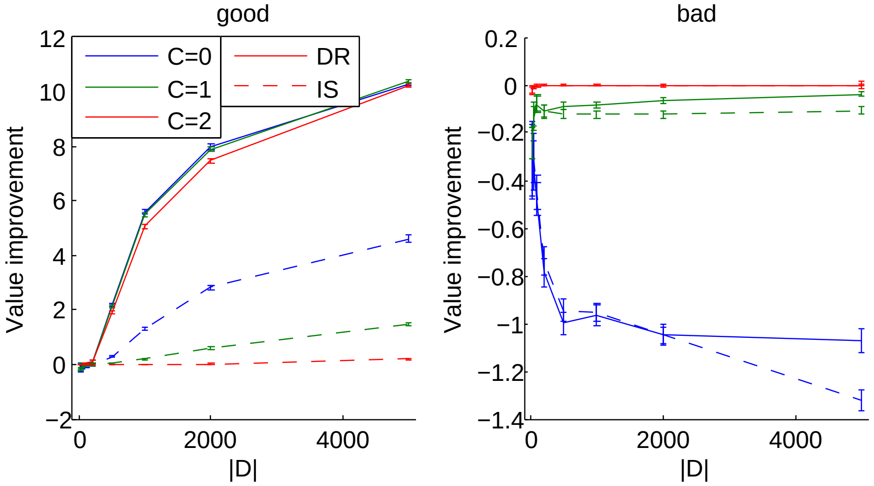

Figure 4.4: Safe policy improvement in Mountain Car. X-axis shows the size of data and y-axis shows the true value of the recommended policy subtracted by the value of the behavior policy.

when π_1 is far from π_0, and beats REG when π_1 and π_0 are close.

4.5.2 Application to safe policy improvement

In this experiment, we apply the off-policy value evaluation methods in safe policy improvement. Given a batch dataset D, the agent uses part of it (D_{train}) to find candidate policies, which may be poor due to data insufficiency and/or inaccurate approximation. The agent then evaluates these candidates on the remaining data (D_{test}) and chooses a policy based on the evaluation. In this common scenario, DR has an extra advantage: D_{train} can be reused to estimate $\widehat{Q}$, and it is not necessary to hold out part of D_{test} for regression.

Due to the high variance of IS and its variants, acting greedily w.r.t. the point estimate is not enough to promote safety. A typical approach is to select the policy that has the highest lower confidence bound [Thomas et al., 2015b], and hold on to the current behavior policy if none of the bounds is better than the behavior policy's value. More specifically, the bound is $V_\dagger - C\sigma_\dagger$, where V is the point estimate, σ is the empirical standard error, and $C \geq 0$ controls confidence level. † is a placeholder for any method that works by averaging a function of sample trajectories; examples considered in this paper are the IS and the DR estimators.

The experiment is conducted in Mountain Car, and most of the setting is the same as Section 4.5.1.1. Since we do not address the exploration-exploitation problem, we keep the behavior policy fixed as uniformly random, and evaluate the recommended policy once in a while as the agent gets more and more data. The candidate policies are generated as follows: we split $|D|$ so that $|D_{\text{train}}|/|D| \in \{0.2, 0.4, 0.6, 0.8\}$; for each split, we compute optimal π_{train} from the model estimated on D_{train}, mix π_{train} and π_0 with rate $\alpha \in \{0, 0.1, \ldots, 0.9\}$, compute their confidence bounds by applying IS/DR on $D \setminus D_{\text{train}}$, and finally pick the policy with the highest score over all splits and α's.

The results are shown on the left panel of Figure 4.4. From the figure, it is clear that DR's value improvement largely outperforms IS, primarily because IS is not able to accept a target policy that is too different from π_0. However, here π_{train} is mostly a good policy (except when $|D|$ is very small), hence the more aggressive an algorithm is, the more value it gets. As evidence, both algorithms achieve the best value with $C = 0$, raising the concern that DR might make unsafe recommendations when π_{train} is poor.

To falsify this hypothesis, we conduct another experiment in parallel, where we have π_{train} minimize the value instead of maximizing it, resulting in policies worse than the behavior policy, and the results are shown on the right panel. Clearly, as C becomes smaller, the algorithms become less safe, and with the same C DR is as safe as IS if not better at $|D| = 5000$. Overall, we conclude that DR can be a drop-in replacement for IS in safe policy improvement.

4.6 Related Work and Discussions

This paper focuses on off-policy value evaluation in finite-horizon problems, which are often a natural way to model real-world problems like dialogue systems. The goal is to estimate the expected return of start states drawn randomly from a distribution. This differs from (and is somewhat easier than) the setting considered in some previous work, often known as off-policy *policy* evaluation, which aims to estimate the whole value function [Precup et al., 2000, 2001, Sutton et al., 2015]. Both settings find important yet different uses in practice, and share the same core difficulty of dealing with distribution mismatch.

The DR technique was first studied in statistics [Rotnitzky and Robins, 1995] to improve the robustness of estimation against model misspecification, and a DR estimator has been developed for dynamic treatment regime [Murphy et al., 2001]. DR was later applied to policy learning in contextual bandits [Dudík et al., 2011], and its finite-time variance is shown to be typically lower than IS. The DR estimator in this work extends the work of Dudík et al. [2011] to sequential decision-making problems. In addition, we show that in certain scenarios the variance of DR matches the statistical lower bound of the estimation problem.

An important application of off-policy value evaluation is to ensure that a new policy to be deployed does not have degenerate performance in policy iteration; example algorithms for this purpose include conservative policy iteration [Kakade and Langford, 2002] and safe policy iteration [Pirotta et al., 2013]. More recently, Thomas et al. [2015a] incorporate lower confidence bounds with IS in approximate policy iteration to ensure that the computed policy meets a minimum value guarantee. Our work compliments their interesting use of confidence intervals by providing DR as a drop-in replacement of IS. We show that after such a replacement, an agent can accept good policies more aggressively hence obtain higher reward, while maintaining the same level of safety against bad policies.

4.7 Proof of Theorem 4.1

Proof. For the base case $t = H + 1$, since $\hat{v}_{\text{DR}}^{0} = V(s_{H+1}) = 0$, it is obvious that at the $(H+1)$-th step the estimator is unbiased with 0 variance, and the theorem holds. For the inductive step, suppose the theorem holds for step $t + 1$. At time step t, we have:

$$
\begin{aligned}
&\mathbb{V}_t\big[\hat v_{\text{DR}}^{H+1-t}\big] \\
&= \mathbb{E}_t\Big[\big(\hat v_{\text{DR}}^{H+1-t}\big)^2\Big] - \Big(\mathbb{E}_t\big[V(s_t)\big]\Big)^2 \\
&= \mathbb{E}_t\Big[\,\Big(\widehat V(s_t) + \rho_t\big(r_t + \gamma \hat v_{\text{DR}}^{H-t} - \widehat Q(s_t,a_t)\big)\Big)^2 - V(s_t)^2\Big] + \mathbb{V}_t\big[V(s_t)\big] \\
&= \mathbb{E}_t\Big[\Big(\rho_t Q(s_t,a_t) - \rho_t \widehat Q(s_t,a_t) + \widehat V(s_t) + \rho_t\big(r_t + \gamma \hat v_{\text{DR}}^{H-t} - Q(s_t,a_t)\big)\Big)^2 - V(s_t)^2\Big] \\
&\qquad + \mathbb{V}_t\big[V(s_t)\big] \\
&= \mathbb{E}_t\Big[\Big(- \rho_t \Delta(s_t,a_t) + \widehat V(s_t) + \rho_t(r_t - R(s_t,a_t)) + \rho_t\gamma\big(\hat v_{\text{DR}}^{H-t} - \mathbb{E}_{t+1}\big[V(s_{t+1})\big]\big)\Big)^2 \\
&\qquad - V(s_t)^2\Big] + \mathbb{V}_t\big[V(s_t)\big] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (4.15) \\
&= \mathbb{E}_t\Big[\mathbb{E}_t\Big[\big(- \rho_t \Delta(s_t,a_t) + \widehat V(s_t)\big)^2 - V(s_t)^2 \;\Big|\; s_t\Big]\Big] + \mathbb{E}_t\Big[\mathbb{E}_{t+1}\big[\rho_t^2 (r_t - R(s_t,a_t))^2\big]\Big] \\
&\qquad + \mathbb{V}_t\big[V(s_t)\big] + \mathbb{E}_t\Big[\mathbb{E}_{t+1}\Big[\Big(\rho_t\gamma\big(\hat v_{\text{DR}}^{H-t} - \mathbb{E}_{t+1}\big[V(s_{t+1})\big]\big)\Big)^2\Big]\Big] \\
&= \mathbb{E}_t\Big[\mathbb{V}_t\big[- \rho_t \Delta(s_t,a_t) + \widehat V(s_t) \mid s_t\big]\Big] + \mathbb{E}_t\Big[\rho_t^2\, \mathbb{V}_{t+1}\big[r_t\big]\Big] \\
&\qquad + \mathbb{E}_t\Big[\rho_t^2\gamma^2\, \mathbb{V}\big[\hat v_{\text{DR}}^{H-t} \mid s_t, a_t\big]\Big] + \mathbb{V}_t\big[V(s_t)\big] \\
&= \mathbb{E}_t\Big[\mathbb{V}_t\big[\rho_t \Delta(s_t,a_t) \mid s_t\big]\Big] + \mathbb{E}_t\Big[\rho_t^2\, \mathbb{V}_{t+1}\big[r_t\big]\Big] + \mathbb{E}_t\Big[\rho_t^2\gamma^2\, \mathbb{V}_{t+1}\big[\hat v_{\text{DR}}^{H-t}\big]\Big] + \mathbb{V}_t\big[V(s_t)\big].
\end{aligned}
$$

This completes the proof. Note that from Equation 4.15 to the next step, we have used the fact that conditioned on s_t and a_t, $r_t - R(s_t,a_t)$ and $\hat v_{\text{DR}}^{H-t} - \mathbb{E}_{t+1}\big[V(s_{t+1})\big]$ are independent and have zero means, and all the other terms are constants. Therefore, the square of the sum equals the sum of squares in expectation. □

4.8 Bias of DR-v2

Proof of Proposition 4.2. Let $\hat v_{\text{DR-v2}'}$ denote Equation 4.12 with approximation $\widehat P = P$. Since $\hat v_{\text{DR-v2}}$ is unbiased, the bias of $\hat v_{\text{DR-v2}'}$ is then the expectation of $\hat v_{\text{DR-v2}'} - \hat v_{\text{DR-v2}}$. Define

$$\beta_t = \mathbb{E}_t\Big[\hat v_{\text{DR-v2}'}^{H+1-t} - \hat v_{\text{DR-v2}}^{H+1-t}\Big].$$

Then, β_1 is the bias we try to quantify, and is a constant. In general, β_t is a random variable that depends on $s_1, a_1, \ldots, s_{t-1}, a_{t-1}$. Now we have

$$\begin{aligned}\beta_t &= \mathbb{E}_t\Big[\rho_t\gamma\left(\hat{v}^{H-t}_{\text{DR-v2}'} - \hat{v}^{H-t}_{\text{DR-v2}}\right) - \rho_t\gamma\widehat{V}(s_{t+1})\left(\frac{\widehat{P}(s_{t+1}|s_t,a_t)}{P(s_{t+1}|s_t,a_t)} - 1\right)\Big] \\ &= \mathbb{E}_t\Big[\rho_t\gamma\beta_{t+1}\Big] - \mathbb{E}_t\Big[\rho_t\gamma\widehat{V}(s_{t+1})\left(\frac{\widehat{P}(s_{t+1}|s_t,a_t)}{P(s_{t+1}|s_t,a_t)} - 1\right)\Big].\end{aligned}$$

In the second term of the last expression, the expectation is taken over the randomness of a_t and s_{t+1}; we keep a_t as a random variable and integrate out s_{t+1}, and get

$$\begin{aligned}&\mathbb{E}_t\Big[\rho_t\gamma\widehat{V}(s_{t+1})\left(\frac{\widehat{P}(s_{t+1}|s_t,a_t)}{P(s_{t+1}|s_t,a_t)} - 1\right)\Big] \\ &= \mathbb{E}_t\Big[\mathbb{E}_{t+1}\Big[\rho_t\gamma\widehat{V}(s_{t+1})\left(\frac{\widehat{P}(s_{t+1}|s_t,a_t)}{P(s_{t+1}|s_t,a_t)} - 1\right)\Big]\Big] \\ &= \mathbb{E}_t\Big[\rho_t\gamma\sum_{s'} P(s'|s_t,a_t)\widehat{V}(s')\left(\frac{\widehat{P}(s'|s_t,a_t)}{P(s'|s_t,a_t)} - 1\right)\Big] \\ &= \mathbb{E}_t\Big[\rho_t\gamma\sum_{s'} \widehat{V}(s')\left(\widehat{P}(s'|s_t,a_t) - P(s'|s_t,a_t)\right)\Big].\end{aligned}$$

Recall that the expectation of the importance ratio is always 1, hence

$$\beta_t \le \mathbb{E}_t\Big[\rho_t\gamma\left(\beta_{t+1} + \epsilon V_{\max}\right)\Big] = \mathbb{E}_t\Big[\rho_t\gamma\beta_{t+1}\Big] + \gamma\epsilon V_{\max}.$$

With an abuse of notation, we reuse β_t as its maximal absolute magnitude over all sample paths $s_1, a_1, \ldots, s_{t-1}, a_{t-1}$. Clearly we have $\beta_{H+1} = 0$, and

$$\beta_t \le \gamma(\beta_{t+1} + \epsilon V_{\max}).$$

Hence, $\beta_1 \le \epsilon V_{\max}\sum_{t=1}^{H}\gamma^t$. □

4.9 Cramer-Rao Bound for Discrete DAG MDPs

Here, we prove a lower bound for the relaxed setting where the MDP is a layered Directed Acyclic Graph instead of a tree. In such MDPs, the regions of the state space reachable in different time steps are disjoint (just as tree MDPs), but trajectories that separate in early steps can reunion at a same state later.

Definition 4.2 (Discrete DAG MDP)**.** An MDP is a *discrete Directed Acyclic Graph (DAG) MDP* if:

- The state space and the action space are finite.
- For any $s \in S$, there exists a unique $t \in \mathbb{N}$ such that, $\max_{\pi:S\to A} P(s_t = s \mid \pi) > 0$. In other words, a state only occurs at a particular time step.
- As a simplification, we assume $\gamma = 1$, and non-zero rewards only occur at the end of each H-step long trajectory. We use an additional state s_{H+1} to encode the reward randomness so that reward function $R(s_{H+1})$ is deterministic and the domain can be solely parameterized by transition probabilities.

Theorem 4.5. *For discrete DAG MDPs, the variance of any unbiased estimator is lower bounded by*

$$\sum_{t=1}^{H+1} \mathbb{E}\Big[\frac{P_1(s_{t-1},a_{t-1})^2}{P_0(s_{t-1},a_{t-1})^2}\mathbb{V}_t\big[V(s_t)\big]\Big],$$

where for trajectory τ, $P_0(\tau) = \mu(s_1)\pi_0(a_1|s_1)P(s_2|s_1,a_1)\dots P(s_{H+1}|s_H,a_H)$, and $P_0(s_t,a_t)$ is its marginal probability; $P_1(\cdot)$ is similarly defined for π_1.

Remark Compared to Theorem 4.3, the cumulative importance ratio $\rho_{1:t-1}$ is replaced by the state-action occupancy ratio $P_1(s_{t-1},a_{t-1})/P_0(s_{t-1},a_{t-1})$ in Theorem 4.5. The two ratios are equal when each state can only be reached by a unique sample path. In general, however, $\mathbb{E}\big[P_1(s_{t-1},a_{t-1})^2/P_0(s_{t-1},a_{t-1})^2\mathbb{V}_t\big[V(s_t)\big]\big] \le \mathbb{E}\big[\rho_{1:t-1}^2\mathbb{V}_t\big[V(s_t)\big]\big]$, hence DAG MDPs are easier than tree MDPs for off-policy value evaluation.

Below we give the proof of Theorem 4.5, which is almost identical to the proof of Theorem 4.3.

Proof of Theorem 4.5. We parameterize the MDP by $\mu(s_1)$ and $P(s_{t+1}|s_t,a_t)$ for $t = 1,\dots,H$. For convenience we will treat $\mu(s_1)$ as $P(s_1|\emptyset)$, so all the parameters can be represented as $P(s_{t+1}|s_t,a_t)$ (for $t=0$ there is a single s_0 and a). These parameters are subject to the normalization constraints that have to be taken into consideration in the Cramer-Rao bound, namely $\forall t, s_t, a_t, \sum_{s_{t+1}} P(s_{t+1}|s_t,a_t) = 1$.

$$\begin{bmatrix} 1\cdots 1 & & & \\ & 1\cdots 1 & & \\ & & \ddots & \\ & & & 1\cdots 1 \end{bmatrix} \theta = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \tag{4.16}$$

where $\theta_{s_t,a_t,s_{t+1}} = P(o|s_t, a_t)$. The matrix on the left is effectively the Jacobian of the constraints, which we denote as F. We index its rows by (s_t, a_t), so $F_{(s_t,a_t),(s_t,a_t,s_{t+1})} = 1$ and other entries are 0. Let U be a matrix whose column vectors consist an orthonormal basis for the null space of F. From Moore Jr [2010, Eqn. (3.3) and Corollary 3.10], we have the Constrained Cramer-Rao Bound (CCRB) being[5] (the dependence on θ in all terms are omitted):

$$KU(U^\top IU)^{-1}U^\top K^\top, \tag{4.17}$$

where I is the Fisher Information Matrix (FIM), and K is the Jacobian of the quantity we want to estimate; they are computed below. We start with I, which is

$$I = \mathbb{E}\Big[\left(\frac{\partial \log P_0(\tau)}{\partial \theta}\right) \left(\frac{\partial \log P_0(\tau)}{\partial \theta}\right)^\top \Big]. \tag{4.18}$$

To calculate I, we define a new notation $g(\tau)$, which is a vector of indicator functions and $g(\tau)_{s_t,a_t,s_{t+1}} = 1$ when (s_t, a_t, s_{t+1}) appears in trajectory τ. Using this notation, we have

$$\frac{\partial \log P_0(\tau)}{\partial \theta} = \theta^{\circ -1} \circ g(\tau), \tag{4.19}$$

where $\circ$ denotes element-wise power/multiplication. Then we can rewrite the FIM as

$$\begin{aligned} I &= \mathbb{E}\Big[[\theta_i^{-1}\theta_j^{-1}]_{ij} \circ (g(\tau)g(\tau)^\top)\Big] \\ &= [\theta_i^{-1}\theta_j^{-1}]_{ij} \circ \mathbb{E}\big[(g(\tau)g(\tau)^\top)\big], \end{aligned} \tag{4.20}$$

where $[\theta_i^{-1}\theta_j^{-1}]_{ij}$ is a matrix expressed by its (i, j)-th element. Now we compute $\mathbb{E}\big[g(\tau)g(\tau)^\top\big]$. On the diagonal, it is $P_0(s_t, a_t, s_{t+1})$, so the diagonal of I is $\frac{P_0(s_t,a_t)}{P(s_{t+1}|s_t,a_t)}$; for non-diagonal entries whose row indexing and column indexing tuples are at the same time step, the value is 0; in other cases, suppose row is (s_t, a_t, s_{t+1}) and column is $s_{t'}, a_{t'}, s_{t'+1}$, and without loss of generality assume $t' < t$, then the entry is $P_0(s_{t'}, a_{t'}, s_{t'+1}, s_t, a_t, s_{t+1})$, with the corresponding entries in I being $\frac{P_0(s_{t'},a_{t'},s_{t'+1},s_t,a_t,s_{t+1})}{P(s_{t'+1}|s_{t'},a_{t'})P(s_{t+1}|s_t,a_t)} = P_0(s_{t'}, a_{t'})P_0(s_t, a_t|s_{t'+1})$.

[5] In fact, existing literature on Contrained Cramer-Rao Bound does not deal with the situation where the unconstrained parameters break the normalization constraints (which we are facing). However, this can be easily tackled by changing the model slightly to $P(o|h, a) = \theta_{hao}/\sum_{o'} \theta_{hao'}$, which resolves the issue and gives the same result.

Then, we calculate $(U^\top IU)^{-1}$. To avoid the difficulty of taking inverse of this non-diagonal matrix, we apply the following trick to diagonalize I: note that for any matrix X with matching dimensions,

$$U^\top IU = U^\top(F^\top X^\top + I + XF)U, \tag{4.21}$$

because by definition U is orthogonal to F. We can design X so that $D = F^\top X^\top + I + XF$ is a diagonal matrix, and $D_{(s_t,a_t,s_{t+1}),(s_t,a_t,s_{t+1})} = I_{(s_t,a_t,s_{t+1}),(s_t,a_t,s_{t+1})} = \frac{P_0(s_t,a_t)}{P(s_{t+1}|s_t,a_t)}$. This is achieved by having XF eliminate all the non-diagonal entries of I in the upper triangle without touching anything on the diagonal or below, and by symmetry $F^\top X^\top$ will deal with the lower triangle. The particular X we take is $X_{(s_{t'},a_{t'},s_{t'+1}),(s_t,a_t)} = -P_0(s_{t'},a_{t'})P_0(s_t,a_t|s_{t'+1})\mathbb{I}(t' < t)$, and it is not hard to verify that this construction diagonalizes I.

With the diagonalization trick, we have $(U^\top IU)^{-1} = (U^\top DU)^{-1}$. Since CCRB is invariant to the choice of U, and we observe that the rows of F are orthogonal, we choose U as follows: let $n_{(s_t,a_t)}$ be the number of 1's in $F_{(s_t,a_t),(\cdot)}$, and $U_{(s_t,a_t)}$ be the $n_{(s_t,a_t)} \times (n_{(s_t,a_t)} - 1)$ matrix with orthonormal columns in the null space of $\begin{bmatrix}1 & \dots & 1\end{bmatrix}$ ($n_{(s_t,a_t)}$ 1's); finally, we choose U to be a block diagonal matrix $U = \text{diag}(\{U_{(s_t,a_t)}\})$, where $U_{(s_t,a_t)}$'s are the diagonal blocks, and it is easy to verify that U is column orthonormal and $FU = 0$. Similarly, we write $D = \text{diag}(\{D_{(s_t,a_t)}\})$ where $D_{(s_t,a_t)}$ is a diagonal matrix with $(D_{(s_t,a_t)})_{s_{t+1},s_{t+1}} = P_0(s_t,a_t)/P(s_{t+1}|s_t,a_t)$, and

$$\begin{aligned}
&U(U^\top IU)^{-1}U^\top = U(U^\top DU)^{-1}U^\top \\
=&\, U(\text{diag}(\{U_{(s_t,a_t)}^\top\})\text{diag}(\{D_{(s_t,a_t)}\})\text{diag}(\{U_{(s_t,a_t)}\}))^{-1}U \\
=&\, U\text{diag}(\{\big(U_{(s_t,a_t)}^\top D_{(s_t,a_t)}U_{(s_t,a_t)}\big)^{-1}\})U \\
=&\, \text{diag}(\{U_{(s_t,a_t)}\big(U_{(s_t,a_t)}^\top D_{(s_t,a_t)}U_{(s_t,a_t)}\big)^{-1}U_{(s_t,a_t)}^\top\}).
\end{aligned} \tag{4.22}$$

Notice that each block in Equation 4.22 is simply $1/P_0(s_t,a_t)$ times the CCRB of a multinomial distribution $P(\cdot|s_t,a_t)$. The CCRB of a multinomial distribution p can be easily computed by an alternative formula [Moore Jr, 2010, Eqn. (3.12)]), which gives $\text{diag}(p) - pp^\top$, so we have,

$$\begin{aligned}
&U_{(s_t,a_t)}\big(U_{(s_t,a_t)}^\top D_{(s_t,a_t)}U_{(s_t,a_t)}\big)^{-1}U_{(s_t,a_t)}^\top \\
=&\, \frac{\text{diag}(P(\cdot|s_t,a_t)) - P(\cdot|s_t,a_t)P(\cdot|s_t,a_t)^\top}{P_0(s_t,a_t)}.
\end{aligned} \tag{4.23}$$

We then calculate K. Recall that we want to estimate

$$v = v^{\pi_1,H} = \sum_{s_1} \mu(s_1) \sum_{a_1} \pi_1(a_1|s_1) \dots \sum_{s_{H+1}} P(s_{H+1}|s_H, a_H) R(s_{H+1})\,,$$

and its Jacobian is $K = (\partial v / \partial \theta_t)^\top$, with $K_{(s_t,a_t,s_{t+1})} = P_1(s_t, a_t) V(s_{t+1})$, where $P_1(\tau) = \mu(s_1)\pi_1(a_1)\dots P(s_{H+1}|s_H, a_H)$ and $P_1(s_t, a_t)$ is the marginal probability.

Finally, putting all the pieces together, we have Equation 4.17 equal to

$$\begin{aligned}
&\sum_{s_t,a} \frac{P_1(s_t,a_t)^2}{P_0(s_t,a_t)} \Big(\sum_{s_{t+1}} P(s_{t+1}|s_t,a_t) V(s_{t+1})^2 - \big(\sum_{s_{t+1}} P(s_{t+1}|s_t,a_t) V(s_{t+1}) \big)^2 \Big) \\
= &\sum_{t=0}^{H} \sum_{s_t} P_0(s_t,a_t) \frac{P_1(s_t,a_t)^2}{P_0(s_t,a_t)^2} \mathbb{V}\big[V(s_{t+1}) \bigm| s_t, a\big] \\
= &\sum_{t=0}^{H} \mathbb{E}\Big[\frac{P_1(s_t,a_t)^2}{P_0(s_t,a_t)^2} \mathbb{V}_{t+1}\big[V(s_{t+1})\big]\Big] = \sum_{t=1}^{H+1} \mathbb{E}\Big[\frac{P_1(s_{t-1},a_{t-1})^2}{P_0(s_{t-1},a_{t-1})^2} \mathbb{V}_t\big[V(s_t)\big]\Big]. \qquad \square
\end{aligned}$$

CHAPTER 5

Adaptive Selection of State Abstraction

State abstractions are often used to reduce the complexity of model-based reinforcement learning when only limited quantities of data are available. However, choosing the appropriate level of abstraction is an important problem in practice. While it is always possible to reduce this problem to off-policy value evaluation (Chapter 4), the exponential lower bound prevents us from developing theoretical guarantees for abstraction selection with polynomial dependence on horizon. Other existing approaches have theoretical guarantees only under strong assumptions on the domain or asymptotically large amounts of data. In this chapter we propose a simple algorithm based on statistical hypothesis testing that comes with a finite-sample guarantee under assumptions on candidate abstractions. Our algorithm trades off the low approximation error of finer abstractions against the low estimation error of coarser abstractions, resulting in a loss bound that depends only on the quality of the *best* available abstraction and is polynomial in planning horizon.

5.1 Introduction

In this chapter, we advance the theoretical understanding of a fundamentally important setting in RL: large state spaces but only limited amounts of data and no pre-existing model. This is, of course, the typical setting for many RL applications, and a number of algorithms that exploit some form of compact function approximation either to learn a model or to directly learn value functions or policies have been applied successfully across domains from control, robotics, resource allocation, and others. Examples of such methods include value function approximation [Bertsekas and Tsitsiklis, 1996], policy-gradient methods [Sutton et al., 1999], kernel RL and related non-parametric dynamic programming algorithms [Ormoneit and Sen, 2002, Lever et al., 2012], and pre-processing with state abstraction/aggregation followed by standard RL algorithms [Li et al., 2006].

However, state-of-the-art theoretical analysis in this area mostly either (1) makes structural assumptions about the domain (e.g., linear dynamics [Parr et al., 2008]) to allow an RL algorithm using a fixed and finite-capacity function approximator to guarantee bounded loss as the size of the dataset grows to infinity, or (2) makes smoothness assumptions about the domain [e.g., Ormoneit and Sen, 2002] but guarantees zero loss only when both the function approximation capacity and the dataset size go to infinity. In contrast, we are interested in analyzing the more realistic case where no assumptions about the domain can be made — other than that it can be described by an MDP — and the dataset is finite.

In particular, we consider a scenario in which a domain expert offers a set of possible state abstractions for a given domain. We assume that these abstractions are finite aggregations of states; for instance, the expert may provide discrete-valued state features, implicitly defining an abstraction that aggregates states with identical feature values. Given a finite amount of data, our task is to discover which abstraction to use for computing a policy from the data. If the dataset is large, we should prefer finer abstractions that are faithful to the domain (those with low *approximation* error), but for smaller datasets, coarse, lossy abstractions may be preferable because they simplify learning (low *estimation* error).

To simplify our analysis, we assume the dataset is fixed in advance. To remove the choice of RL algorithm from our analysis, we again assume certainty equivalence as in Chapter 3, despite that the state representation is determined by the chosen abstraction. When the quality of the abstraction is known, the theory of approximate homomorphisms in MDPs bounds the loss of the certainty equivalence policy [Even-Dar and Mansour, 2003, Ravindran, 2004]. However, here the quality of the abstractions is unknown, and must itself be estimated from data. Existing theoretical results in this setting either have exponential dependence on the effective planning horizon (i.e., reduction to off-policy evaluation; see Chapter 4), or apply to the online setting and depend on the total size of all abstract state spaces under consideration [Ortner et al., 2014]. For our purposes the latter result is no better than simply always choosing the finest abstraction.

Initially, we consider choosing between two abstractions, one of which is a refinement of the other (e.g., the finer abstraction uses a superset of the features of the coarser abstraction). We propose a simple algorithm, and prove a theoretical guarantee that only depends on the *better* abstraction and is *polynomial* in effective planning horizon. Then, we show how to extend our analysis to an arbitrary set of abstractions that are successive refinements.

The algorithm we present and analyze is similar to existing algorithms that aggregate/split states via hypothesis testing with various state aliasing criteria [Jong and Stone, 2005, Dinculescu and Precup, 2010, Talvitie and Singh, 2011, Hallak et al., 2013]. However, our analysis provides the first finite-sample guarantee theoretically justifying this family of methods. Previous theoretical work has assumed that at least one of the candidate abstractions is perfect and will be discovered asymptotically in the limit of data [e.g., Hallak et al., 2013, Section 5]. However, abstractions are usually approximate in practice, and we need abstractions in the first place primarily because the data is *insufficient*. Asymptotic analyses offer little guidance for balancing approximation error and estimation error in this setting. Our analysis shows that a carefully designed hypothesis test can balance this finite-sample trade-off even when none of the abstractions are perfect, and works almost as well as if the abstraction qualities were known in advance.

The rest of the chapter is organized as follows. Section 5.2 introduces preliminaries and defines the abstraction selection problem. Section 5.3 develops a bound on the loss of a single abstraction, setting up the approximation and estimation error trade-off. Section 5.4 proposes and analyzes our algorithm. Section 5.5 reviews other approaches to the abstraction selection problem and compare our algorithm to existing solutions.

5.2 Preliminaries

5.2.1 Abstractions for model-based RL

A state abstraction h is a mapping from the primitive (or raw) state space $\mathcal{S}$ to an abstract state space $h(\mathcal{S})$. We use $h(s) \in h(\mathcal{S})$ to denote the abstract state that contains a particular primitive state s. Following certainty equivalence, we assume that the agent builds a model $\widehat{M}^h$ from a dataset D under abstraction h, and then follows the optimal policy for $\widehat{M}^h$.

Data The dataset D is a set of 4-tuples (s, a, r, s'), collected by repeatedly and independently sampling a state-action pair (s, a) from some fixed distribution p fully supported over $\mathcal{S} \times \mathcal{A}$ (i.e., $p(s, a) > 0 \ \forall s, a$), and then, given (s, a), sampling a reward r from R and a next state s' from P. (Recall that this is the data collection protocol **(b)** in Section 2.3.1.) If some fixed exploration policy is used to collect data, then p will correspond to the state-action occupancy distribution (though the samples will not be strictly independent in this case). For $x \in h(\mathcal{S})$, we denote by $D_{x,a}$ the restriction of D to tuples whose first two elements are $s \in h^{-1}(x)$ and a; that is, $D_{x,a}$ is the portion of the dataset concerning abstract state x and action a.

Model The model estimated from dataset D using abstraction h is $\widehat{M}^h = (h(\mathcal{S}), A, \widehat{P}^h, \widehat{R}^h, \gamma)$, where $\widehat{P}^h$ and $\widehat{R}^h$ are the maximum likelihood estimates (recall Equations 2.16 and 2.17). When referring to the model constructed using the primitive state space, we use the notation $\widehat{M}$, omitting the superscript.

5.2.2 Problem statement

Our goal is to choose an abstraction h from a candidate set $\mathcal{H}$ so as to minimize the loss of the optimal policy for $\widehat{M}^h$:

$$\text{Loss}(h, D) = \left\| V_M^* - V_M^{\pi^*_{\widehat{M}^h}} \right\|_\infty . \tag{5.1}$$

Note that $\pi^*_{\widehat{M}^h}$ is a mapping from $h(\mathcal{S})$ to A, and has to be *lifted* as $[\pi^*_{\widehat{M}^h}]_M : s \mapsto \pi^*_{\widehat{M}^h}(h(s))$ to be evaluated in M. For notational simplicity, we will not distinguish an abstract policy from its lifted version as long as there is no confusion.

For most of the chapter we will be concerned with the following assumption. Later we will discuss how to extend our algorithm and analysis to a more general setting.

Assumption 5.1. $\mathcal{H} = \{h_c, h_f\}$, where finer abstraction h_f is a refinement of coarser abstraction h_c, i.e., $h_f(s) = h_f(s') \Rightarrow h_c(s) = h_c(s'), \forall\, s, s' \in \mathcal{S}$.

5.3 Bounding the Loss of a Single Abstraction

Before proceeding to describe our solution to the abstraction selection problem, we first establish an upper bound on Loss(h, D) for any fixed abstraction h. This will allow us to compare the results of our selection algorithm to the loss bounds we could achieve if the qualities of the abstractions were known in advance. Abstraction quality is characterized by the following definitions, which is an approximate version of Equation 2.18.

Definition 5.2. Let $M^h = \langle h(\mathcal{S}), A, P^h, R^h, \gamma \rangle$, where, for all $x, x' \in h(\mathcal{S})$ and $a \in \mathcal{A}$,

$$P^h(x,a,x') = \frac{\sum_{s \in h^{-1}(x)} p(s,a) \sum_{s' \in h^{-1}(x')} P(s,a,s')}{\sum_{s \in h^{-1}(x)} p(s,a)},$$
$$R^h(x,a) = \frac{\sum_{s \in h^{-1}(x)} p(s,a) R(s,a)}{\sum_{s \in h^{-1}(x)} p(s,a)}.$$

Then M^h is said to be an *approximate homomorphism* of M with *transition error* and *reward error*

$$\epsilon_T^h = \max_{s \in \mathcal{S}, a \in \mathcal{A}} \sum_{x' \in h(\mathcal{S})} \Big| P^h(h(s),a,x') - \sum_{s' \in h^{-1}(x')} P(s,a,s') \Big|,$$
$$\epsilon_R^h = \max_{s \in \mathcal{S}, a \in \mathcal{A}} \left| R^h(h(s),a) - R(s,a) \right|.$$

If $\epsilon_T^h = \epsilon_R^h = 0$, M^h is said to be a (perfect) homomorphism of M, and it is known that $\pi^*_{M^h}$ is an optimal policy for M.[1] As ϵ_T^h and ϵ_R^h increase, $\pi^*_{M^h}$ may incur more loss.

Theorem 5.1 improves upon and tightens existing bounds from the literature on approximate homomorphisms and bisimulation [e.g., Ravindran and Barto, 2004]. (Paduraru et al. [2008] proved a bound tighter than ours by a factor of $1/(1-\gamma)$, but required an asymptotic assumption that $n^h(D)$ is sufficiently large.)

Theorem 5.1 (Loss bound for a single abstraction). *For any abstraction h, $\forall \delta \in (0,1)$, w.p. $\geq 1-\delta$,*

$$\textit{Loss}(h,D) \leq \frac{2}{(1-\gamma)^2}\big(\textit{Appr}(h) + \textit{Estm}(h,D,\delta)\big)$$

where

$$\textit{Appr}(h) = \epsilon_R^h + \frac{\gamma R_{\max} \epsilon_T^h}{2(1-\gamma)},$$
$$\textit{Estm}(h,D,\delta) = \frac{R_{\max}}{1-\gamma} \sqrt{\frac{1}{2n^h(D)} \log \frac{2|h(\mathcal{S})||\mathcal{A}|}{\delta}},$$
$$n^h(D) = \min_{x \in h(\mathcal{S}), a \in \mathcal{A}} |D_{x,a}|.$$

The proof is deferred to Section 5.6. The bound consists of two terms, where

[1] This is exactly Equation 2.18. In general, approximate homomorphisms can incorporate action aggregation/permutation, but in this chapter we only consider aggregation in the state space.

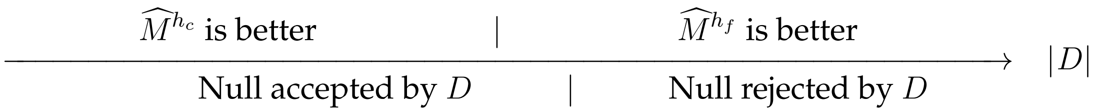

Figure 5.1: **Upper part:** The preferred abstraction changes as dataset size grows beyond some threshold. **Lower part:** Our algorithm uses the dataset to perform a hypothesis test; when dataset size exceeds some threshold, the null hypothesis will be rejected. We show that the two thresholds have bounded difference, regardless of h_c and h_f.

the first increases with the approximation parameters $(\epsilon_T^h, \epsilon_R^h)$ but is independent of the dataset D, and the second has no dependence on ϵ_T^h or ϵ_R^h, but depends on the abstraction via $n^h(D)$, the minimal number of visits to any abstract state-action pair, and $|h(\mathcal{S})|$. The first term is small for accurate abstractions (which have small $(\epsilon_T^h, \epsilon_R^h)$), while the second term is small for compact abstractions (which have small $|h(\mathcal{S})|$ and large $n^h(D)$).

Our goal in this chapter is to select from the candidate set $\mathcal{H}$ an abstraction achieving the lowest loss, and we can use the bound in Theorem 5.1 as a proxy for that loss. (This is a common approach in existing work on abstraction selection as well as machine learning in general; see Section 5.5 for details.) If the size of the dataset is very small, the bound suggests that we should select a coarse abstraction to reduce estimation error. However, as the size of D grows, $n^h(D)$ increases, and the second term goes to zero while the first remains constant, implying that finer and finer abstractions will in general become preferable (see Jiang et al. [2014] for an empirical illustration). Under Assumption 5.1, then, the crucial question is: How much data should we require before selecting h_f over h_c?

If ϵ_T^h and ϵ_R^h were known for both abstractions, we could simply calculate an appropriate boundary from Theorem 5.1. However, in practice, ϵ_T^h and ϵ_R^h are unknown. Nevertheless, we will show that our algorithm can approximately estimate this boundary from data. In particular, we will use D to statistically test whether $Q^*_{M^{h_f}}$ and $Q^*_{M^{h_c}}$ are equal (when lifted); in general, we will reject this hypothesis after we obtain a sufficient amount of data. Perhaps surprisingly, our analysis shows that the point at which this rejection first occurs is almost the same (in the appropriate technical sense) as the point at which h_f becomes preferable to h_c (see Figure 5.1 for an illustration). Thus, we will use this hypothesis test to define a simple algorithm for abstraction selection that is near-optimal with respect to Theorem 5.1.

5.4 Proposed Algorithm and Theoretical Analysis

Before proposing our algorithm, we first define the operators $\widehat{B}^h$ and B^h.

Definition 5.3. Given dataset D and abstraction h, $\widehat{B}^h : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is defined as follows. For any Q-value function $Q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$,

$$(\widehat{B}^h Q)(s,a) = \frac{\sum_{(r,s')\in D_{h(s),a}} (r + \gamma V_Q(s'))}{|D_{h(s),a}|},$$

where $V_Q(s') = \max_{a'\in\mathcal{A}} Q(s',a')$. We define B^h as

$$(B^h Q)(s,a) = \frac{\sum_{s':h(s')=h(s)} p(s',a)\cdot(BQ)(s',a)}{\sum_{s':h(s')=h(s)} p(s',a)},$$

where B is the Bellman optimality operator for M, namely $(BQ)(s,a) = R(s,a) + \gamma \langle P(\cdot\,|s,a), V_Q\rangle$.

The operator $\widehat{B}^h$ is a variation of the Bellman optimality operator for $\widehat{M^h}$, and B^h is the same for M^h. It is not hard to verify that $[Q^*_{\widehat{M^h}}]_M$ and $[Q^*_{M^h}]_M$ are, respectively, fixed points of $\widehat{B}^h$ and B^h (recall that $[\cdot]_M$ is the lifting operation).

With these definitions, we propose Algorithm 1. It computes a particular statistic using D, and then selects h_f if and only if the statistic exceeds a threshold.

Algorithm 1 ComparePair($D, \mathcal{H}, \delta$)

```
  assert $\mathcal{H} = \{h_c, h_f\}$ satisfies Assumption 5.1
  let $Q = [Q^*_{\widehat{M^{h_c}}}]_M$
  if
      $$\left\| \widehat{B}^{h_f} Q - Q \right\|_\infty \geq 2\,\mathrm{Estm}(h_f, D, \delta/3) \qquad (5.2)$$
  then output $h_f$, else output $h_c$
```

5.4.1 Intuition of the algorithm

Before formally analyzing Algorithm 1, we first present an intuitive explanation for its behavior and show that it makes sensible decisions in various scenarios. The central idea is to statistically test whether

$$[Q^*_{M^{h_f}}]_M = [Q^*_{M^{h_c}}]_M, \qquad (5.3)$$

which is equivalent (see Lemma 5.4 and Section 5.8) to

$$\left\| B^{h_f}[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M \right\|_\infty = 0. \tag{5.4}$$

The LHS of Equation (5.4) is effectively the Bellman residual of $Q^*_{M^{h_c}}$ when treating M^{h_f} as the true model. Since the required quantities are not known in advance, we approximate them from data and check whether the measured error exceeds a positive rejection threshold. This gives the selection criterion of Equation (5.2).

Consider two extreme cases. First, when M^{h_c} is a perfect homomorphism of M^{h_f}, Equation (5.3) always holds and we never reject the null hypothesis, thus our algorithm always returns h_c. This makes sense, since the abstractions have equal approximation error but h_c has lower estimation error. On the other hand, when Equation (5.3) does not hold, given enough data our test will reject the null hypothesis and select h_f. Again, this is sensible since h_f has lower approximation error, and in the limit of data the estimation error for both abstractions is zero.

Of course, the usual situation is that Equation (5.3) does not hold but D is finite. Suppose in this case that M^{h_f} is a perfect homomorphism of M; then Algorithm 1 can be seen as approximately comparing the bound in Theorem 5.1 for h_f and h_c, as follows. Since Appr$(h_f) = 0$ and the estimation errors are computable from known quantities, the only unknown quantity needed for this comparison is Appr(h_c). In principle, Appr(h_c) is a function of M and M^{h_c}, and could be approximated from data using $\widehat{M}$ and $\widehat{M}^{h_c}$; however, the estimate of $\widehat{M}$ will be poor when $|\mathcal{S}|$ is large (which is why we require abstraction in the first place). Instead, since h_f is exact by assumption, we can compare M^{h_c} directly to M^{h_f}. The LHS of Equation (5.2) provides this estimate of Appr(h_c); see the left panel of Figure 5.2 for a visual illustration.

In the most general scenario, where the dataset is finite and *both* abstractions are approximate, we need a reliable estimate of Appr(h_c) − Appr(h_f) to make the comparison using Theorem 5.1, but we no longer have a statistically efficient way of estimating Appr(h_f) or Appr(h_c). However, our analysis shows that even when M^{h_f} is not homomorphic to M, the three models can be seen as roughly "on the same line", as visualized in the right panel of Figure 5.2. As a result, we can use the dashed line—a measure of distance between M^{h_f} and M^{h_c}—to approximate the desired difference between the solid lines. This idea is the basis for Lemma 5.7, which is a key ingredient in the theoretical guarantee for Algorithm 1.

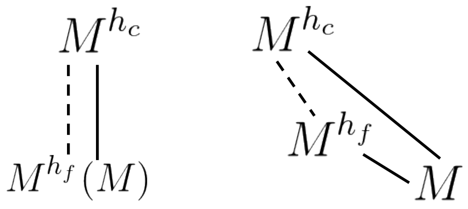

Figure 5.2: **Left panel:** When M^{h_f} is a perfect homomorphism of M, we can obtain the true approximation error of M^{h_c} (solid line) by computing its approximation error w.r.t. M^{h_f} (dashed line). The two notions of approximation are equivalent, but the latter is statistically easier to estimate. **Right panel:** When M^{h_f} is also approximate, our theoretical analysis shows that M, M^{h_f}, and M^{h_c} are always roughly "on the same line", so that the approximation error of M^{h_c} w.r.t. M^{h_f} (dashed line) is a good proxy for the difference between the true approximation errors of M^{h_c} and M^{h_f} (solid lines).

5.4.2 Theoretical analysis

We next state the formal guarantee of our algorithm.

Theorem 5.2. *Given dataset D, if $\mathcal{H}$ satisfies Assumption 5.1, the loss of the abstraction selected by Algorithm 1 is bounded by*

$$\begin{aligned}\frac{2}{(1-\gamma)^2}\min\Big\{&\mathit{Appr}(h_f)+\frac{3-\gamma}{1-\gamma}\mathit{Estm}(h_f,D,\delta/3),\\ &\frac{3-\gamma}{1-\gamma}\mathit{Appr}(h_c)+\frac{1+\gamma}{1-\gamma}\mathit{Estm}(h_c,D,\delta/3)\Big\}\end{aligned} \tag{5.5}$$

with probability at least $1-\delta$.

Equation (5.5) is the minimum of two terms. The first is nearly (up to a factor of $O(1/(1-\gamma))$) the loss bound of h_f using Theorem 5.1, and the second is nearly the loss bound of h_c. Recall that Theorem 5.1 is our proxy for loss; therefore, the loss bound for Algorithm 1 is as good as the loss bound of the *better* abstraction up to a factor linear in $1/(1-\gamma)$. Compared to Theorem 5.1, the estimation error terms in Equation (5.5) have increased from $\mathrm{Estm}(\cdot,\cdot,\delta)$ to $\mathrm{Estm}(\cdot,\cdot,\delta/3)$; however, this has little influence as $\mathrm{Estm}(\cdot,\cdot,\delta)$ depends only square-root logarithmically on $1/\delta$.

Claim 5.3 (Theorem 5.2 is near-optimal w.r.t. Theorem 5.1)**.** *Equation* (5.5) *is at most the minimum of the bound in Theorem 5.1 as applied to h_f and to h_c, up to a factor of $O(\frac{1}{1-\gamma})$.*

We will prove Theorem 5.2 with the help of the following lemmas. Their proofs are deferred to Appendices 5.6 and 5.7.

Lemma 5.4. *For any Bellman optimality operators B_1, B_2 (both operating on $\mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and having contraction rate γ), letting Q_1 and Q_2 be their respective fixed points, we have*

$$\|Q_1 - Q_2\|_\infty \leq \frac{\|B_1 Q_2 - Q_2\|_\infty}{1-\gamma}.$$

Lemma 5.5. *Consider $\widehat{B}^h$ as defined in Definition 5.3. For any $h \in \mathcal{H}$ and deterministic $Q : \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ with bounded range $[0, R_{\max}/(1-\gamma)]$, w.p. $\geq 1-\delta$,*

$$\left\|\widehat{B}^h Q - B^h Q\right\|_\infty \leq \mathit{Estm}(h, D, \delta).$$

Lemma 5.6. *Let B be the Bellman optimality operator of M. For any $Q : \mathbb{R}^{h(\mathcal{S})\times A}$ with bounded range $[0, R_{\max}/(1-\gamma)]$, we have*

$$\left\|B[Q]_M - B^h[Q]_M\right\|_\infty \leq \mathit{Appr}(h).$$

Lemma 5.7. $\forall Q : \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ *with bounded range* $[0, R_{\max}/(1-\gamma)]$,

$$\begin{aligned}\|BQ \,-\, B^{h_c}Q\|_\infty \leq{} & \left\|BQ - B^{h_f}Q\right\|_\infty + \left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty \\ \leq{} & 3\left\|BQ - B^{h_c}Q\right\|_\infty.\end{aligned}$$

We briefly sketch the proof of Theorem 5.2 before proceeding to the details. Recall that our goal is to determine which abstraction has a smaller loss bound according to Theorem 5.1; that is, we want to check whether

$$\text{Appr}(h_c) - \text{Appr}(h_f) \geq \text{Estm}(h_f, D, \delta) - \text{Estm}(h_c, D, \delta),$$

where the LHS is unknown. To approximate it, we first use Lemma 5.7, which implies that

$$\begin{aligned} & \|B[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\|_\infty && (5.6)\\ \approx{} & \left\|B[Q^*_{M^{h_c}}]_M - B^{h_f}[Q^*_{M^{h_c}}]_M\right\|_\infty && (5.7)\\ & + \left\|B^{h_f}[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty. && (5.8)\end{aligned}$$

Expression (5.8) is a quantity closely related to the statistic computed by our algorithm (see Equation (5.4)), so to establish that the statistic is a good proxy for

$\text{Appr}(h_c) - \text{Appr}(h_f)$, we will show that

$$\text{Appr}(h_c) - \text{Appr}(h_f) \approx \text{Expression (5.6)} - \text{Expression (5.7)}.$$

Expression (5.6) is easy to deal with, as the Bellman residual of $[Q^*_{M^{h_c}}]_M$ is a better characterization of the approximation error of h_c than $\text{Appr}(h_c)$.[2] Expression (5.7) is a bit trickier: we know it is not an overestimate, as Lemma 5.6 guarantees that it is upper bounded by $\text{Appr}(h_f)$. However, there exists the risk of underestimation: for instance, if h_c aggregates all primitive states into a single abstract state, then $[Q^*_{M^{h_c}}]_M$ is a constant function and Expression (5.7) only reflects the reward error of h_f, and will not change regardless of the transition error.

To deal with this, we consider two cases separately. First, when h_c is the better abstraction, we have $[Q^*_{M^{h_c}}]_M \approx Q^*_M$, hence

$$\text{Expression (5.7)} \approx \left\|BQ^*_M - B^{h_f} Q^*_M\right\|_\infty . \tag{5.9}$$

According to Lemma 5.4, the RHS of Equation (5.9) is an alternative characterization of the approximation error of h_f, so in this case we will not underestimate too much. On the other hand, when h_f is better, underestimation of its approximation error only biases our selection towards the better abstraction, and is not a concern.

Below we include part of the proof of Theorem 5.2.

Proof of Theorem 5.2. Using Lemma 5.5, w.p. at least $1 - \delta$ we have

$$\left\|\widehat{B}^{h_f}[Q^*_{M^{h_f}}]_M - B^{h_f}[Q^*_{M^{h_f}}]_M\right\|_\infty \le \text{Estm}(h_f, D, \delta/3),$$

and similar concentration bounds hold for $\widehat{B}^{h_c}[Q^*_{M^{h_c}}]_M$ and $\widehat{B}^{h_f}[Q^*_{M^{h_c}}]_M$ simultaneously.

Regardless of which abstraction the algorithm selects, we can always bound its loss using Theorem 5.1, so it suffices to show that we can bound the loss of the selected abstraction in terms of the other. We consider each possibility in turn.

[2] In this discussion we do not strictly distinguish between approximate homomorphism ($\text{Appr}(h)$) and approximate Q^*-irrelevance (the Bellman residual of $Q^*_{M^h}$) in characterizing the approximation error of h. Technical details can be found in proofs and we point the readers to Li et al. [2006] for further reading.

If the algorithm outputs h_c, we can bound the loss of h_c by parameters of h_f:

$$\begin{align}
\mathrm{Loss}(h_c, D) &\leq \frac{2}{(1-\gamma)^2} \left\| B[Q^*_{\widehat{M}^{h_c}}]_M - [Q^*_{\widehat{M}^{h_c}}]_M \right\|_\infty \tag{5.10} \\
&\leq \frac{2}{(1-\gamma)^2} \Bigg(\left\| B[Q^*_{\widehat{M}^{h_c}}]_M - \widehat{B}^{h_f}[Q^*_{\widehat{M}^{h_c}}]_M \right\|_\infty \nonumber \\
&\qquad + \left\| \widehat{B}^{h_f}[Q^*_{\widehat{M}^{h_c}}]_M - [Q^*_{\widehat{M}^{h_c}}]_M \right\|_\infty \Bigg) \tag{5.11} \\
&\leq \frac{2}{(1-\gamma)^2} \Bigg(\left\| B[Q^*_{M^{h_c}}]_M - \widehat{B}^{h_f}[Q^*_{M^{h_c}}]_M \right\|_\infty \nonumber \\
&\qquad + 2\mathrm{Estm}(h_f, D, \delta/3) + 2\gamma \left\| [Q^*_{M^{h_c}}]_M - [Q^*_{\widehat{M}^{h_c}}]_M \right\|_\infty \Bigg) \tag{5.12} \\
&\leq \frac{2}{(1-\gamma)^2} \Bigg(\left\| B[Q^*_{M^{h_c}}]_M - B^{h_f}[Q^*_{M^{h_c}}]_M \right\|_\infty \nonumber \\
&\qquad + 3\mathrm{Estm}(h_f, D, \delta/3) + \frac{2\gamma}{1-\gamma}\mathrm{Estm}(h_c, D, \delta/3) \Bigg) \tag{5.13} \\
&\leq \frac{2}{(1-\gamma)^2} \left(\mathrm{Appr}(h_f) + \frac{3-\gamma}{1-\gamma}\mathrm{Estm}(h_f, D, \delta/3) \right). \nonumber
\end{align}$$

Equation (5.10) is a standard loss bound using the Bellman residual. In Equation (5.11), we use the triangle inequality to introduce the statistic computed by our algorithm. In the first term of Equation (5.12), we replace $[Q^*_{\widehat{M}^{h_c}}]_M$ by $[Q^*_{M^{h_c}}]_M$ using the fact that the Bellman operators have contraction rate γ ($\|BQ - BQ'\|_\infty \leq \gamma \|Q - Q'\|_\infty$), and in the second term we use the fact that the algorithm chose h_c, and thus Equation (5.2) did not hold. Next, we apply the probabilistic guarantees stated at the beginning of the proof to remove the D subscripts on operators and Q-value functions, and finally the $\mathrm{Appr}(h_f)$ term appears thanks to Lemma 5.6.

The rest of the proof is similar and appears in Section 5.7. □

5.4.3 Extension to arbitrary-size candidate sets

We briefly discuss how to extend the above algorithm and analysis to the following setting.

Assumption 5.4. $\mathcal{H} = \{h_1, \ldots, h_{|\mathcal{H}|}\}$, where h_i is a refinement of h_{i-1} for $i = 2, \ldots, |\mathcal{H}|$.

This is the setting considered by Hallak et al. [2013], and $\mathcal{H} = \{h_c, h_f\}$ is the special case where $|\mathcal{H}| = 2$. A natural idea is to use Algorithm 1 as a subroutine, successively comparing the best abstraction seen so far with the remaining elements in $\mathcal{H}$ in some order. The crucial questions are: (1) in what order should we examine the abstractions (e.g., coarse-to-fine, fine-to-coarse, or a random/adaptive order), and (2) can we adapt the analysis in Section 5.4.2 to show that the selected abstraction is still near-optimal w.r.t. Theorem 5.1 for larger $\mathcal{H}$? It turns out that, if we examine abstractions in order from coarse to fine, near-optimality is preserved. Algorithm 2 provides a detailed specification for the process, and Theorem 5.8 gives the resulting guarantee.

Algorithm 2 CompareSequence($D, \mathcal{H}, \delta$)

```
assert $\mathcal{H} = \{h_1, \ldots, h_{|\mathcal{H}|}\}$ satisfies Assumption 5.4
let $\widehat{h} = h_1$    // start with the coarsest abstraction
for i=2 to $|\mathcal{H}|$ do
    $\widehat{h}$ = ComparePair($D, \{\widehat{h}, h_i\}, 2\delta/|\mathcal{H}|^2$)
end for
output $\widehat{h}$
```

Theorem 5.8. *If $\mathcal{H}$ satisfies Assumption 5.4 and has constant size, then Algorithm 2 is near-optimal w.r.t. Theorem 5.1, i.e., the loss of the selected abstraction is upper bounded w.p. at least $1 - \delta$ by*

$$\min_{h \in \mathcal{H}} \left(\mathit{Appr}(h) + \mathit{Estm}\left(h, D, 2\delta/(3|\mathcal{H}|^2)\right)\right)$$

up to a factor polynomial in $1/(1-\gamma)$.

The biggest challenge in generalizing our analysis to the case of $|\mathcal{H}| > 2$ is that the two sides of Equation (5.5) have different semantics—that is, the LHS is loss, while the RHS is approximation/estimation error. This means that successive comparisons cannot (naively) apply the bound transitively. Recall that, in the proof of Theorem 5.2, we considered the selection of h_c and the selection of h_f separately. It turns out that we can modify the analysis to obtain consistent, transitive semantics, but only for the case where h_c is selected. This is enough for near-optimality as long as we order the abstractions from coarse to fine, avoiding the bad case of problematic abstractions. For a more detailed discussion and a proof sketch of Theorem 5.8, see Section 5.8.

5.5 Related Work and Discussions

In this section we review prior theoretical work that is relevant to the abstraction selection problem. The discussion is summarized in Table 5.1.

5.5.1 Hypothesis test based algorithms

Jong and Stone [2005] (row 1) considered the factored MDP setting, where state is determined by a vector of features and an abstraction is a subset of those features. They proposed a selection procedure that statistically tests whether the optimal policy depends on certain features, aiming to aggregate states having the same optimal action and thus create a π^*-*irrelevance* abstraction. However, π^*-irrelevance abstractions can yield sub-optimal policies when applying Q-learning even with infinite data, so this method is not statistically consistent [Li et al., 2006, Theorem 4].

Hallak et al. [2013] (row 2), in the work most closely related to ours, considered the setting of Assumption 5.4 and suggested comparing h_c and h_f by statistically testing whether M^{h_c} is a perfect homomorphism of M^{h_f} using D. They showed theoretically that their procedure will *asymptotically* identify any abstraction that is a *perfect* homomorphism of M. However, if all the candidate abstractions are approximate, or the dataset is finite, their analysis does not apply.

Nevertheless, there are interesting similarities between our Algorithm 1 and the method of Hallak et al. [2013]: both algorithms test *relative* properties of h_c and h_f so as to avoid the large primitive representation, and both choose the coarser abstraction unless a statistical test rejects the null hypothesis that h_c and h_f are (in some sense) equivalent. However, our analysis shows that this type of algorithm can still have provable guarantees even when the data are insufficient and the abstractions are approximate—in fact, it can be near-optimal with respect to a loss bound.

There are several important technical differences between our algorithm and that of Hallak et al. [2013]: (1) We use Q^*-*irrelevance* as the equivalence criterion in our hypothesis test, whereas they use homomorphism; Q^*-irrelevance is a strictly more general relationship than homomorphism [Li et al., 2006, Theorem 2] that avoids the problematic L_1 norm as a characterization of estimation error [Maillard et al., 2014] and enables convenient mathematical tools for finite sample analysis (e.g., the $\widehat{B}^h$ operator). (2) We fully specify the rejection threshold for the hypothesis test (up to the probability guarantee δ) without introducing additional hyperparameters, while in their work the rate of threshold decay as the dataset grows is left to the practitioner. This choice can have a significant impact on the transient behavior of

Table 5.1: Comparison of algorithms that can be applied to the abstraction selection problem. If an entry exhibits a desired property (which we judge by generality and practicality), we mark it as **bold**. In the first row we provide the properties of model-based RL with primitive representation as a baseline to compare against.

	Finite Sample Guarantee		Assumption on Candidate Abstractions	Tuning Hyperparameters	Optimization Objective
	Dependence on Representation	Dependence on Horizon			
No Abstraction	$\lvert\mathcal{S}\rvert$	**Polynomial**	-	-	-
1.Jong and Stone [2005]	[No guarantee]		**Subsets of state features**	**No**[a]	-
2.Hallak et al. [2013]	[Only asymptotic guarantee]		Successive refinements	Statistical test threshold as a function of sample size	Coarseness of perfect homomorphisms
3.Importance Sampling	**No**	Exponential	**No**	**No**	**Loss**
4.Model-based Estimator	$\lvert\mathcal{S}\rvert$	**Polynomial**	**No**	**No**	**Loss**
5.Farahmand and Szepesvári [2011]	Size of regressor abstraction	**Polynomial**	**No**	Choice of regressor abstraction	Bellman residual loss bound
Our method	**Size of best abstraction**	**Polynomial**	Successive refinements	**No**	Approximate homomorphism loss bound

[a] Their algorithm has a single parameter which is the p-value threshold for hypothesis test, and they suggest using 0.05 in practice. In fact, all the methods listed in this table except the first 3 rows require a similar confidence level parameter.

the algorithm.

5.5.2 Reduction to off-policy evaluation

Abstractions can also be selected using a cross-validation procedure: if a second dataset D' is given independently of D, then we can evaluate the policies computed under different abstractions from D (i.e., $\{\pi^*_{M_D^h} : h \in \mathcal{H}\}$) on D'. This reduces the abstraction selection problem to the off-policy evaluation problem, and the loss guarantee depends entirely on the accuracy of the offline evaluation estimator. Recall from Chapter 4 that generally this problem has a lower bound that is exponential in problem horizon, typically incurred by estimators from the Importance Sampling family (row 3), which fails our goal here.

While the exponential dependence can be avoided in model-based estimators, the vanilla version of model-based estimators using the primitive state representation (e.g., in Section 3.5.2) incurs polynomial dependence on $|\mathcal{S}|$, which is unacceptable here.

Alternatively, the validation model can be estimated under an abstraction to avoid the dependence on $|\mathcal{S}|$, but this solution is circular: if we knew a good abstraction for policy evaluation, we could have used it to obtain a good policy in the first place. For instance, Farahmand and Szepesvári [2011] (row 5) proposed an offline policy evaluation procedure that selects value functions (from which policies are computed) based on their estimated Bellman residuals, which are estimated with the help of an additional regressor that learns BQ from data for the candidate Qs. The theoretical guarantee for this method depends on the accuracy of the regressor (see their Theorem 2, especially the dependence on $\bar{b}_k$). For the reason noted above, this is problematic in our setting: the abstractions are themselves regressors (where $\widehat{B}^h Q$ is the function being learned), so if we knew how to select a good abstraction for regression, then the same one could have been used to learn a policy instead.

5.5.3 The online setting

Ortner et al. [2014] proposed a representation (abstraction) selection algorithm in the online exploration and exploitation setting that tests whether a representation faithfully predicts the return of a roll-out trajectory. Their regret bound depends on the *sum* of sizes of the state spaces for all representations under consideration (see their Theorem 3). While the online setting has additional complications (see more detailed discussions in Chapter 7), in our offline setting this bound is loose and can be improved simply by selecting the finest available abstraction. On the other hand, although our algorithm assumes structure in the candidate abstractions (they must be successive refinements), our loss bound depends only on the *best* abstraction.

5.6 Proof of Theorem 5.1

We first prove Lemma 5.4, 5.5 and 5.6.

Proof of Lemma 5.4. $\forall s \in \mathcal{S}, a \in \mathcal{A}$,

$$\left\|Q_1 - Q_2\right\|_\infty = \left\|B_1 Q_1 - B_1 Q_2 + B_1 Q_2 - Q_2\right\|_\infty \le \gamma \left\|Q_1 - Q_2\right\|_\infty + \left\|B_1 Q_2 - Q_2\right\|_\infty .$$

Hence the bound follows. Note that this result subsumes the standard Bellman residual bound, when we let Q_2 be an approximate Q-value function (e.g. $[Q^*_{M^h}]_M$, where $B_2 = B^h$), and Q_1 be the true optimal value function Q^*_M (where $B_1 = B$). Furthermore, thanks to the definition of B^h, we can use this bound in an alternative form, namely bounding $\left\|Q^*_M - [Q^*_{M^h}]_M\right\|_\infty$ by $\left\|Q^*_M - B^h Q^*_M\right\|_\infty$. We will use both forms (and sometimes treating M^h as the true model) throughout the theoretical analysis depending on the context. □

Proof of Lemma 5.5. According to Definition 5.3, $(\widehat{B}^h Q)(s,a)$ is the average of $r + \gamma V_Q(s')$ for $(r, s') \in D_{h(s),a}$, which are independent random variables with bounded range $[0, R_{\max}/(1-\gamma)]$. When $|D_{h(s),a}| > 0$,[3] it is straight-forward to verify that for any deterministic Q, $(B^h Q)(s,a) = \mathbb{E}\Big[D\Big](\widehat{B}^h Q)(s,a) \mid |D_{h(s),a}| > 0$. Hence, Hoeffding's inquality applies, $\forall t > 0$,

$$\mathbb{P}_D\Big\{\Big|(\widehat{B}^h Q)(s,a) - (B^h Q)(s,a)\Big| \ge t\Big\} \le 2 \exp\left(-\frac{2t^2 |D_{x,a}|}{R_{\max}^2/(1-\gamma)^2}\right).$$

Now we find t that makes the inequality hold for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ simultaneously w.p. at least $1 - \delta$ via union bound. Note, however, that $\widehat{B}^h Q$ (and $B^h Q$) takes constant value among states aggregated by h, hence we only have $|h(\mathcal{S})||\mathcal{A}|$ events in the union bound instead of $|\mathcal{S}||\mathcal{A}|$ ones. The t that satisfies our requirement turns

[3] When $|D_{h(s),a}| = 0$, $n^h(D) = 0$ and the RHS of the bound goes to infinity, which promises nothing and is always correct.

out to be

$$t = \frac{R_{\max}}{1-\gamma}\sqrt{\frac{1}{2n^h(D)}\log\frac{2|h(\mathcal{S})||\mathcal{A}|}{\delta}} = \text{Estm}(h, D, \delta). \qquad \square$$

Proof of Lemma 5.6. $\forall s \in \mathcal{S}, a \in \mathcal{A}$,

$$\begin{aligned}
&\left|(B[Q]_M)(s,a) - (B^h[Q]_M)(s,a)\right| \\
=&\left|R(s,a) + \gamma\left\langle P(\,\cdot\,|s,a), [V_Q]_M\right\rangle - R^h(h(s),a) - \gamma\left\langle P^h(\,\cdot\,|h(s),a), V_Q\right\rangle\right| \\
=&\left|R(s,a) + \gamma\left\langle \sum_{s'\in h^{-1}(\cdot)} P(s'|s,a), V_Q - \frac{R_{\max}}{2(1-\gamma)}\right\rangle\right. \\
&\left. - R^h(h(s),a) - \gamma\left\langle P^h(\,\cdot\,|h(s),a), V_Q - \frac{R_{\max}}{2(1-\gamma)}\right\rangle\right| \\
\le\, & \epsilon_R^h + \epsilon_T^h\frac{R_{\max}}{2(1-\gamma)} = \text{Appr}(h). \qquad \square
\end{aligned}$$

Proof of Theorem 5.1. Let $[Q^*_{\widehat{M}^h}]_M$ denote $Q^*_{\widehat{M}^h}$ lifted to M, namely $[Q^*_{\widehat{M}^h}]_M(s) = Q^*_{\widehat{M}^h}(h(s))$. We have,

$$\begin{aligned}
\left\|V^*_M - V_M^{\pi^*_{\widehat{M}^h}}\right\|_\infty &\le \frac{2}{1-\gamma}\left\|Q^*_M - [Q^*_{\widehat{M}^h}]_M\right\|_\infty \qquad \text{([Singh and Yee, 1994])} \\
&\le \frac{2}{1-\gamma}\left(\|Q^*_M - [Q^*_{M^h}]_M\|_\infty + \left\|[Q^*_{M^h}]_M - [Q^*_{\widehat{M}^h}]_M\right\|_\infty\right) \\
&= \frac{2}{1-\gamma}\left(\|Q^*_M - [Q^*_{M^h}]_M\|_\infty + \left\|Q^*_{M^h} - Q^*_{\widehat{M}^h}\right\|_\infty\right).
\end{aligned}$$

According to Lemma 5.6, the first term in the bracket can be bounded as:

$$\|Q^*_M - [Q^*_{M^h}]_M\|_\infty \le \frac{\left\|B[Q^*_{M^h}]_M - B^h[Q^*_{M^h}]_M\right\|_\infty}{1-\gamma} \le \frac{\text{Appr}(h)}{1-\gamma}.$$

For the second term, we use Lemma 5.4 by letting $B_1 = B^{h_f}$ and $B_2 = \widehat{B}^{h_f}$, and then apply Lemma 5.5: w.p. at least $1-\delta$,

$$\left\|Q^*_{M^h} - Q^*_{\widehat{M}^h}\right\|_\infty \le \frac{\left\|B^h[Q^*_{M^h}]_M - \widehat{B}^h[Q^*_{M^h}]_M\right\|_\infty}{1-\gamma} \le \frac{\text{Estm}(h, D, \delta)}{1-\gamma}.$$

Combining the bounds for the two terms and the theorem follows. □

5.7 Proof of Theorem 5.2

We first prove the remaining Lemma.

Proof of Lemma 5.7. The left inequality is trivial from the triangular inequality. To prove the right inequality, we bound $\left\|BQ - B^{h_f}Q\right\|_\infty$ and $\left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty$ by $\left\|BQ - B^{h_c}Q\right\|_\infty$ separately. The key is to notice that, for any $x \in h_f(S)$, $(B^{h_f}Q)(x, a)$ is always a convex average of

$$\{(BQ)(s, a) : s \in h_f^{-1}(x)\}.$$

We first show $\left\|BQ - B^{h_f}Q\right\|_\infty \leq 2\left\|BQ - B^{h_c}Q\right\|_\infty$. Notice that there exist $s, s' \in \mathcal{S}, a \in \mathcal{A}$ s.t. $h_f(s) = h_f(s')$ and

$$|(BQ)(s, a) - (BQ)(s', a)| \geq \left\|BQ - B^{h_f}Q\right\|_\infty.$$

Using the same argument on h_c, it is obvious that

$$\begin{aligned}
\left\|BQ - B^{h_c}Q\right\|_\infty &\geq \max_{\substack{h_c(s)=h_c(s') \\ a\in\mathcal{A}}} |(BQ)(s, a) - (BQ)(s', a)| \;/\; 2 \\
&\geq \max_{\substack{h_f(s)=h_f(s') \\ a\in\mathcal{A}}} |(BQ)(s, a) - (BQ)(s', a)| \;/\; 2 \\
&\geq \left\|BQ - B^{h_f}Q\right\|_\infty \;/\; 2,
\end{aligned}$$

hence the bound follows.

Next we show $\left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty \leq \left\|BQ - B^{h_c}Q\right\|_\infty$. Consider the state-action pair that achieves the max norm of $\left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty$, i.e.

$$\left|(B^{h_f}Q)(s, a) - (B^{h_c}Q)(s, a)\right| = \left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty.$$

Since $(B^{h_f}Q)(s, a)$ is a convex average of $\{(BQ)(s', a) : h_f(s') = h_f(s)\}$, there always exists $s' : h_f(s') = h_f(s)$ such that $(BQ)(s', a) \geq (B^{h_f}Q)(s, a)$, and $s'' : h_f(s'') = h_f(s)$ such that $(BQ)(s'', a) \leq (B^{h_f}Q)(s, a)$. Note that $(B^{h_c}Q)(s, a) = (B^{h_c}Q)(s', a) = (B^{h_c}Q)(s'', a)$, hence either

$$\left|(BQ)(s', a) - (B^{h_c}Q)(s', a)\right|$$

or

$$\left|(BQ)(s'', a) - (B^{h_c}Q)(s'', a)\right|$$

will be no less than

$$\left|(B^{h_f}Q)(s,a) - (B^{h_c}Q)(s,a)\right|,$$

which implies that

$$\left\|B^{h_f}Q - B^{h_c}Q\right\|_\infty \le \left\|BQ - B^{h_c}Q\right\|_\infty . \qquad \square$$

Proof of Theorem 5.2 (continued). Similarly, if the algorithm outputs h_f,

$$\begin{aligned}
&\mathrm{Loss}(h_f, D) \\
&\le \frac{2}{1-\gamma}\left(\left\|Q^*_M - [Q^*_{M^{h_f}}]_M\right\|_\infty + \left\|[Q^*_{M^{h_f}}]_M - [Q^*_{\widehat{M}^{h_f}}]_M\right\|_\infty \right) \\
&\le \frac{2}{(1-\gamma)^2}\left(\left\|B^{h_f}Q^*_M - BQ^*_M\right\|_\infty + \mathrm{Estm}(h_f, D, \delta/3) \right) \qquad (5.14) \\
&\le \frac{2}{(1-\gamma)^2}\Bigg(\left\|B^{h_f}[Q^*_{M^{h_c}}]_M - B[Q^*_{M^{h_c}}]_M\right\|_\infty \\
&\qquad + 2\gamma \left\|Q^*_M - [Q^*_{M^{h_c}}]_M\right\|_\infty + \mathrm{Estm}(h_f, D, \delta/3) \Bigg) \\
&\le \frac{2}{(1-\gamma)^2}\Bigg(3\left\|B^{h_c}[Q^*_{M^{h_c}}]_M - B[Q^*_{M^{h_c}}]_M\right\|_\infty - \left\|B^{h_f}[Q^*_{M^{h_c}}]_M - B^{h_c}[Q^*_{M^{h_c}}]_M\right\|_\infty \\
&\qquad + \frac{2\gamma}{1-\gamma}\left\|B[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty + \mathrm{Estm}(h_f, D, \delta/3) \Bigg) \\
&\le \frac{2}{(1-\gamma)^2}\Bigg(\frac{3-\gamma}{1-\gamma}\left\|B[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty \\
&\qquad - \left\|\widehat{B}^{h_f}[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty + 2\,\mathrm{Estm}(h_f, D, \delta/3) \Bigg) \\
&\le \frac{2}{(1-\gamma)^2}\Bigg(\frac{3-\gamma}{1-\gamma}\left\|B[Q^*_{M^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty - \left\|\widehat{B}^{h_f}[Q^*_{\widehat{M}^{h_c}}]_M - [Q^*_{\widehat{M}^{h_c}}]_M\right\|_\infty \\
&\qquad + 2\,\mathrm{Estm}(h_f, D, \delta/3) + (1+\gamma)\left\|[Q^*_{\widehat{M}^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty \Bigg) \\
&\le \frac{2}{(1-\gamma)^2}\left(\frac{3-\gamma}{1-\gamma}\left\|B[Q^*_{M^{h_c}}]_M - B^{h_c}[Q^*_{M^{h_c}}]_M\right\|_\infty + \frac{1+\gamma}{1-\gamma}\mathrm{Estm}(h_c, D, \delta/3) \right) \qquad (5.15) \\
&= \frac{2}{(1-\gamma)^2}\left(\frac{3-\gamma}{1-\gamma}\mathrm{Appr}(h_c) + \frac{1+\gamma}{1-\gamma}\mathrm{Estm}(h_c, D, \delta/3) \right).
\end{aligned}$$

The derivation is similar to the previous one, with a few small changes. In Equation (5.14), instead of the Bellman residual we use Lemma 5.4 to bound the value difference. We also replace Q^*_M with $[Q^*_{M^{h_c}}]_M$, and use Lemma 5.7 to introduce a term similar to the statistic computed by the algorithm. Then, using the probabilistic guarantees stated at the beginning, we obtain exactly that statistic, and bound it using Equation (5.2). Finally, Appr(h_c) appears from Lemma 5.6 and Estm(h_c) from the probabilistic guarantees. □

5.8 Proof of Theorem 5.8

We first prove a lemma on Bellman residuals.

Lemma 5.9. *For any Q-value function $Q : \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$,*

$$\|BQ - Q\|_\infty \leq (1+\gamma)\,\|Q - Q^*_M\|_\infty \,.$$

Proof.

$$\begin{aligned}\|BQ - Q\|_\infty &= \|BQ - Q^*_M + Q^*_M - Q\|_\infty \\ &\leq \|BQ - BQ^*_M\|_\infty + \|Q^*_M - Q\|_\infty \\ &\leq \gamma\,\|Q^*_M - Q\|_\infty + \|Q^*_M - Q\|_\infty \\ &= (1+\gamma)\,\|Q^*_M - Q\|_\infty \,.\end{aligned}$$

So the lemma follows. □

Theorem 5.8 will a direct corollary of Lemma 5.10, by noticing that the loss of the selected abstraction can be upper bounded by the LHS of Equation (5.16).

Lemma 5.10. *Suppose Assumption 5.4 holds. Let $\widehat{h_i}$ be the best-so-far abstraction among $h_1, \dots, h_i$ found by Algorithm 2, then for $\delta' = 2\delta/(3|\mathcal{H}|^2)$, the following bound holds w.p. $\geq 1-\delta$: $\forall i = 1, 2, \dots, |\mathcal{H}|$,*

$$\begin{aligned}&\left\|[Q^*_{M^{\widehat{h_i}}}]_M - Q^*_M\right\|_\infty + \frac{1}{1-\gamma} \mathit{Estm}(\widehat{h_i}, D, \delta') \\ &\leq \mathit{poly}\Big(\frac{1}{1-\gamma}\Big) \cdot \min_{h\in\{h_1,\dots,h_i\}} (\mathit{Appr}(h) + \mathit{Estm}(h, D, \delta'))\,. \end{aligned} \tag{5.16}$$

Proof. For every pair of possible comparison we require the 3 probabilistic guarantees in the proof of Theorem 5.2 to hold, hence by union bound we can guarantee that each of them occurs w.p. at least $1-\delta'$. Then, we prove the lemma by induction. For the case of $i=1$, it holds obviously from Theorem 5.1, by noticing that the LHS of Lemma 5.10 is an intermediate step of proving Theorem 5.1 (up to $2/(1-\gamma)$), and the RHS is consistent with the final bound.

Suppose the induction assumption holds for i, and consider the comparison between $h_c = \widehat{h_i}$ and $h_f = h_{i+1}$. If h_c is selected, we only need to prove that $\left\|[Q^*_{M^{h_c}}]_M - Q^*_M\right\|_\infty + \frac{1}{1-\gamma}\mathrm{Estm}(h_c, D, \delta')$ can be bounded by $\mathrm{Appr}(h_f)$ and $\mathrm{Estm}(h_f, D, \delta')$, which is possible by slightly adapting the previous analysis. In particular,

$$\begin{aligned}
&\frac{2}{1-\gamma}\left(\|[Q^*_{M^{h_c}}]_M - Q^*_M\|_\infty + \frac{1}{1-\gamma}\mathrm{Estm}(h_c, D, \delta')\right)\\
&\le \frac{2}{(1-\gamma)^2}\Big(\left\|B[Q^*_{M^{h_c}}]_M - B^{h_c}[Q^*_{M^{h_c}}]_M\right\|_\infty + \mathrm{Estm}(h_c, D, \delta')\Big)\\
&\le \frac{2}{(1-\gamma)^2}\Big(\left\|B[Q^*_{M^{h_c}}]_M - \widehat{B}^{h_c}[Q^*_{M^{h_c}}]_M\right\|_\infty + 2\mathrm{Estm}(h_c, D, \delta')\Big)\\
&\le \frac{2}{(1-\gamma)^2}\Big(\left\|B[Q^*_{\widehat{M}^{h_c}}]_M - \widehat{B}^{h_c}[Q^*_{\widehat{M}^{h_c}}]_M\right\|_\infty\\
&\quad + 2\gamma\left\|[Q^*_{\widehat{M}^{h_c}}]_M - [Q^*_{M^{h_c}}]_M\right\|_\infty + 2\mathrm{Estm}(h_c, D, \delta')\Big),
\end{aligned}$$

and now we arrive at Equation (5.10), up to some extra dependence on $\mathrm{Estm}(h_c, D, \delta')$ (which we can always afford), and the difference between δ and δ'. Following the rest part of the previous analysis we will have the desired bound.

If h_f is selected, the beginning part of the previous analysis can be adapted much more easily:

$$\begin{aligned}
&\frac{2}{1-\gamma}\left(\left\|[[Q^*_{M^{h_f}}]_M - Q^*_M\right\|_\infty + \frac{1}{1-\gamma}\mathrm{Estm}(h_f, D, \delta')\right)\\
&\le \frac{2}{(1-\gamma)^2}\Big(\left\|BQ^*_M - B^{h_f}Q^*_M\right\|_\infty + \mathrm{Estm}(h_f, D, \delta')\Big),
\end{aligned}$$

and now we are at Equation (5.14). This time, however, we cannot follow the previous analysis all the way to the end, as our induction assumption promises nothing for $\mathrm{Appr}(h_c)$ and $\mathrm{Estm}(h_c)$. Instead, we can departure from Equation (5.15):

$$(5.15) \le \frac{2}{(1-\gamma)^2}\left(\frac{(3-\gamma)(1+\gamma)}{1-\gamma}\|[Q^*_{M^{h_c}}]_M - Q^*_M\|_\infty + \frac{1+\gamma}{1-\gamma}\mathrm{Estm}(h_c, D, \delta')\right),$$

which follows from Lemma 5.9. Now we can apply our induction assumption, and this shows that the induction assumption holds for $i + 1$, so the lemma follows. □

CHAPTER 6

Repeated Inverse Reinforcement Learning

In the previous chapters, we adopt the standard RL formulation and take it for granted that rewards are well-defined and revealed to the agent as part of the dataset. In real-life situations, however, it has long been recognized that specifying a detailed and comprehensive reward function that is well aligned with human interest can be difficult, and this has grown into a serious concern on the safety of future AI systems [Bostrom, 2003, Russell et al., 2015, Amodei et al., 2016].

One approach to addressing this issue is for the agent to infer human goals by observing human behavior, a problem studied under the Inverse Reinforcement Learning (IRL) framework. However, IRL is generally ill-posed for there are typically many reward functions that rationalize the same observed behavior. While the use of heuristics to select from among the set of feasible reward functions has led to successful applications of IRL to learning from demonstration, such heuristics do not address AI safety. In this chapter we introduce a novel *repeated* IRL problem: the agent has to act on behalf of a human in a sequence of tasks and wishes to minimize the number of tasks that it surprises the human. Each time the human is surprised the agent is provided a demonstration of the desired behavior by the human. We formalize this problem, including how the sequence of tasks is chosen, in a few different ways and provide some foundational results.

6.1 Introduction

One challenge in building AI agents that learn from experience is how to set their goals or rewards. In the Reinforcement Learning (RL) setting, one interesting answer to this question is inverse RL (or IRL) in which the agent infers the rewards of a human by observing the human's policy in a task [Ng and Russell, 2000]. Unfortunately, the IRL problem is ill-posed for there are typically many reward functions for which the observed behavior is optimal in a single task [Abbeel and Ng, 2004]. While the use of heuristics to select from among the set of feasible reward functions has led to successful applications of IRL to the problem of learning from demonstration [e.g., Abbeel et al., 2007], not identifying the reward function poses fundamental challenges to the question of how well and how safely the agent will perform when using the learned reward function in other tasks. This is particularly relevant because IRL is a possible approach to the concern about aligning the agent's values/goals with those of humans for AI safety as society deploys more capable learning agents that impact more people in more ways [Russell et al., 2015, Amodei et al., 2016].

In this chapter, we formalize multiple variations of a *new repeated IRL* problem in which the agent and the human are placed in multiple tasks. We separate the reward function into two components, one which is invariant across tasks and can be viewed as intrinsic to the human, and a second that is task specific. As a motivating example, consider a human doing tasks throughout a work day, e.g., getting coffee, driving to work, interacting with co-workers, and so on. Each of these tasks has a task-specific goal but the human brings to each task intrinsic goals that correspond to maintaining health, financial well-being, not violating moral and legal principles, etc. In our repeated IRL setting, the agent presents a policy for each new task that it thinks the human would do. If the agent's policy "surprises" the human by being sub-optimal, the human presents the agent with the optimal policy. The objective of the agent is to minimize the number of surprises to the human, i.e., to generalize the human's behavior to new tasks.

Quite apart from the connection to AI safety, the repeated IRL problem we introduce and our results are of independent interest in resolving the question of unidentifiability of rewards from observations in standard IRL. Our contributions include: (1) an efficient identification algorithm when the agent can choose the tasks in which it observes human behavior; (2) an upper bound on the number of total surprises when no assumptions are made on the tasks, along with a corresponding lower bound and extension to the setting where interactions carry out in the form of sample trajectories; (3) identification guarantees when the agent can only choose the task rewards but is given a fixed task environment.

6.2 Problem Setup

6.2.1 Notations

We introduce a few special notations and conventions for making the discussions in this chapter convenient.

First, in this chapter we denote an MDP by $M = (\mathcal{S}, \mathcal{A}, P, Y, \gamma, \mu)$, where Y is the reward function; the symbol R is reserved for *task-specific* reward to be introduced later.

Second, the reward function $Y : \mathcal{S} \to \mathbb{R}$ operates on the state space, which is common in IRL literature (e.g., [Ng and Russell, 2000]). We differ slightly, however, in that we assume rewards to occur after transition (i.e., $R(s, a, s') = R(s')$); this change from the usual setting ($R(s, a, s') = R(s)$) is without loss of generality (the only difference being that the uncontrolled reward obtained at the initial state is ignored), and allows us to state many theoretical results much more elegantly.

Third, in this chapter we will always normalize a value function so that it takes the same magnitude as rewards, which has been adopted in Kakade et al. [2003]. For example, the state-value function is defined as

$$V^\pi(s) = (1-\gamma)\,\mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} Y(s_{t+1}) \,\middle|\, s_1 = s; \pi\right].$$

We will use the notation $V^\pi_{P,Y}$ to avoid ambiguity in the transition dynamics (or *environment*) and the reward function used in computing V^π. The matrix-vector equation for policy evaluation is highly important in this chapter; using the concepts introduced in Section 2.1.3, we define the normalized state occupancy vector (with respect to initial distribution μ) as

$$\eta^\pi_{\mu,P} = (1-\gamma)\left(\mu^\top P^\pi \left(\mathbf{I}_{|\mathcal{S}|} - \gamma P^\pi\right)^{-1}\right)^\top .^1$$

The vector is normalized in the sense that it lies in the probability simplex, i.e., it is element-wise non-negative and $\|\eta_{\mu,P}\|_1 = 1$.

In this chapter, the ultimate goodness of a policy will be evaluated by $\mathbb{E}_{s\sim\mu}[V^\pi(s)]$ (i.e., performance measure **(ii)**) in Section 2.3.2), which is the dot product between

[1] The difference from Equation 2.4 is due to the assumption that reward occurs after transition.

the occupancy vector and the reward vector:

$$\mathbb{E}_{s\sim\mu}[V^{\pi}(s)] = Y^{\top}\eta^{\pi}_{\mu,P}. \tag{6.1}$$

Regarding this measure, the loss of a suboptimal policy is naturally defined as

$$\text{loss} = \mathbb{E}_{s\sim\mu}[V^{\star}(s)] - \mathbb{E}_{s\sim\mu}[V^{\pi}(s)]. \tag{6.2}$$

6.2.2 Repeated Inverse RL framework

Here we define the Repeated IRL problem. The human's reward function $\theta_\star \in \mathbb{R}^{|\mathcal{S}|}$ captures his/her safety concerns and intrinsic/general preferences. $\theta_\star$ is unknown to the agent and is the object of interest herein, i.e., if $\theta_\star$ were known to the agent, the concerns addressed in this paper would be solved. We assume that the human cannot directly communicate $\theta_\star$ to the agent but can evaluate the agent's behavior in a task as well as demonstrate optimal behavior.

Formally, a task is defined by a pair (E, R), where $E = (\mathcal{S}, \mathcal{A}, P, \gamma, \mu)$ is the task environment (i.e., an MDP without a reward function), and R is the task-specific reward function (*task reward*). We assume that all tasks share the same $\mathcal{S}, \mathcal{A}, \gamma$, with $|\mathcal{A}| \geq 2$, but **may** differ in the initial distribution μ, dynamics P, and task reward R; all of the task-specifying quantities are known to the agent. In any task, the human's optimal behavior is always with respect to the reward function $Y := \theta_\star + R$. We emphasize again that $\theta_\star$ is intrinsic to the human and remains the same across all tasks. Our use of task specific reward functions R allows for greater generality than the usual IRL setting, but we note that our results apply equally to the case where the task reward is always zero.

While $\theta_\star$ is private to the human, the agent has some prior knowledge on $\theta_\star$, represented as a set of possible parameters $\Theta_0 \subset \mathbb{R}^{|\mathcal{S}|}$ that contains $\theta_\star$. Throughout, we assume that the human's reward has bounded and normalized magnitude, that is, $\|\theta_\star\|_\infty \leq 1$.

A demonstration in (E, R) means revealing $\pi^\star$ to the agent, which optimizes for $Y := \theta_\star + R$ under environment E. A common assumption in the IRL literature is that the full mapping is revealed, which can be unrealistic if some states are unreachable from the initial distribution. We address the issue by requiring only the state occupancy vector $\eta^{\pi^*}_{\mu,P}$. In Section 6.6 we show that this also allows an easy extension to the setting where the human only demonstrates trajectories instead of providing a policy.

Under the above framework for repeated IRL, we consider two settings that differ in how the sequence of tasks are chosen. In both cases, we will want to minimize the number of demonstrations needed.

1. (Section 6.4) *Agent chooses the tasks*, observes the human's behavior in each of them, and infers the reward function. In this setting where the agent is powerful enough to choose tasks arbitrarily, we will show that the agent will be able to *identify* the human's reward function which of course implies the ability to generalize to new tasks.

2. (Section 6.5) *Nature chooses the tasks*, and the agent proposes a policy in each task. The human demonstrates a policy only if the agent's policy is a mistake (a negative surprise), i.e., significantly suboptimal. In this setting we will derive upper and lower bounds on the number of mistakes our agent will make.

6.3 The Challenge of Identifying Rewards

Note that it is impossible to identify $\theta_\star$ from watching human behavior in a single task. This is because any $\theta_\star$ is fundamentally indistinguishable from an infinite set of reward functions that yield exactly the policy observed in the task. We introduce the idea of *behavioral equivalence* below to tease apart two separate issues wrapped up in the challenge of identifying rewards.

Definition 6.1. Two reward functions $\theta, \theta' \in \mathbb{R}^{|\mathcal{S}|}$ are *behaviorally equivalent in MDP tasks*, if for any (E, R), the set of optimal policies for $(R + \theta)$ and $(R + \theta')$ are the same.

We argue that the task of identifying the reward function should amount only to identifying the equivalence class to which $\theta_\star$ belongs. In particular, identifying the equivalence class is sufficient to get perfect generalization to new tasks. Any remaining unidentifiability is merely representational and of no real consequence. Next we present a constraint that captures the reward functions that belong to the same equivalence class.

Proposition 6.1. *θ and θ' are behaviorally equivalent in MDP tasks if and only if $\theta - \theta' = c \cdot \mathbf{1}_{|\mathcal{S}|}$ for some $c \in \mathbb{R}$, where $\mathbf{1}_{|\mathcal{S}|}$ is an all-1 vector of length $|\mathcal{S}|$.*

Proof. To show that $\theta - \theta' = c \cdot \mathbf{1}_{|\mathcal{S}|}$ implies behavioral equivalence, we notice that an occupancy vector $\eta^\pi_{\mu,P}$ always satisfies $\mathbf{1}_{|\mathcal{S}|}^\top \eta^\pi_{\mu,P} = 1$, so the value of any policy differs by a universal constant c under θ and θ', and the set of optimal policies is the same.

To show the other direction, we prove that if $\theta - \theta' \notin \mathrm{span}(\{\mathbf{1}_{|\mathcal{S}|}\})$, then there exists (E, R) such that the sets of optimal policies differ. In particular, we choose $R = -\theta'$, so that all policies are optimal under $R + \theta'$. Since $\theta - \theta' \notin \mathrm{span}(\{\mathbf{1}_{|\mathcal{S}|}\})$, there exists state i and j such that $\theta(i) + R(i) \neq \theta(j) + R(j)$. Suppose i is the one with smaller sum of rewards, then we can make j an absorbing state, and wire the two actions in i to i and j respectively. Under $R + \theta$, the self-loop in state i is suboptimal, and this completes the proof. □

For any class of θ's that are equivalent to each other, we can choose a canonical element to represent this class. For example, we can fix an arbitrary reference state $s_{\text{ref}} \in \mathcal{S}$, and fix the reward of this state to 0 for $\theta_\star$ and all candidate θ. In the rest of the paper, we will always assume such canonicalization in the MDP setting, hence $\theta_\star \in \Theta_0 \subseteq \{\theta \in [-1, 1]^{|\mathcal{S}|} : \theta(s_{\text{ref}}) = 0\}$.

6.4 Agent Chooses the Tasks

In this section, we consider the protocol that the agent chooses a sequence of tasks $\{(E_t, R_t)\}$. For each task (E_t, R_t), the human reveals $\pi_t^\star$, which is optimal for environment E_t and reward function $\theta_\star + R_t$. Our goal is to design an algorithm which chooses $\{(E_t, R_t)\}$ and identifies $\theta_\star$ to a certain accuracy using as few tasks as possible.

6.4.1 Omnipotent identification algorithm

Theorem 6.2 shows that a simple algorithm can identify $\theta_\star$ after only $O(\log(1/\epsilon))$ tasks, if *any* tasks may be chosen. Roughly speaking, the algorithm amounts to a binary search on each component of $\theta_\star$ by manipulating the task reward R_t.[2] See the proof for the algorithm specification.

Theorem 6.2. *If $\theta_\star \in \Theta_0 \subseteq \{\theta \in [-1, 1]^{|\mathcal{S}|} : \theta(s_{ref}) = 0\}$, there exists an algorithm that outputs $\theta \in \mathbb{R}^{|\mathcal{S}|}$ that satisfies $\|\theta - \theta_\star\|_\infty \leq \epsilon$ after $O(\log(1/\epsilon))$ demonstrations.*

Proof. The algorithm chooses the following fixed environment in all tasks: for each $s \in \mathcal{S} \setminus \{s_{\text{ref}}\}$, let one action be a self-loop, and the other action transitions to s_{ref}. In s_{ref}, all actions cause self-loops. The initial distribution over states is uniformly at random over $\mathcal{S} \setminus \{s_{\text{ref}}\}$.

[2] While we present a proof that manipulates R_t, an only slightly more complex proof applies to the setting where all the R_t are exactly zero and the manipulation is limited to the environment.

Each task only differs in the task reward R_t (where $R_t(s_{\text{ref}}) \equiv 0$ always). After observing the state occupancy of the optimal policy, for each s we check if the occupancy is equal to 0. If so, it means that the demonstrated optimal policy chooses to go to s_{ref} from s in the first time step, and $\theta_\star(s) + R_t(s) \leq \theta_\star(s_{\text{ref}}) + R_t(s_{\text{ref}}) = 0$; if not, we have $\theta_\star(s) + R_t(s) \geq 0$. Consequently, after each task we learn the relationship between $\theta_\star(s)$ and $-R_t(s)$ on each $s \in \mathcal{S} \setminus \{s_{\text{ref}}\}$, so conducting a binary search by manipulating $R_t(s)$ will identify $\theta_\star$ to ϵ-accuracy after $O(\log(1/\epsilon))$ tasks. □

As noted before, once the agent has identified $\theta_\star$ within an appropriate tolerance, it can compute a sufficiently-near-optimal policy for all tasks, thus completing the generalization objective through the far stronger identification objective in this setting. This intuition is formalized below.

Proposition 6.3. *The loss of acting greedily with respect to θ in any task (E, R) is bounded by $2\|\theta - \theta_\star\|_\infty$.*

Proof. Let $\pi^\star$ be the policy that maximizes $\theta_\star + R$ and π be the policy that maximizes $\theta + R$.

$$\begin{aligned}
\text{loss} &= (\theta_\star + R)^\top(\eta^{\pi^\star}_{\mu,P} - \eta^{\pi}_{\mu,P}) \\
&= (\theta + R)^\top(\eta^{\pi^\star}_{\mu,P} - \eta^{\pi}_{\mu,P}) + (\theta_\star - \theta)^\top(\eta^{\pi^\star}_{\mu,P} - \eta^{\pi}_{\mu,P}) \\
&\leq 0 + \|\theta_\star - \theta\|_\infty \|\eta^{\pi^\star}_{\mu,P} - \eta^{\pi}_{\mu,P}\|_1 \qquad (\pi \text{ is optimal for } \theta + R) \\
&\leq \|\theta_\star - \theta\|_\infty \left(\|\eta^{\pi^\star}_{\mu,P}\|_1 + \|\eta^{\pi}_{\mu,P}\|_1\right) = 2\|\theta_\star - \theta\|_\infty.
\end{aligned}$$

□

6.5 Nature Chooses the Tasks

While Theorem 6.2 yields a strong identification guarantee, it also relies on a strong assumption, that $\{(E_t, R_t)\}$ may be chosen by the agent in an arbitrary manner. In this section, we let *nature*, who is allowed to be adversarial for the purpose of the analysis, choose $\{(E_t, R_t)\}$.

Generally speaking, we cannot obtain identification guarantees in such an adversarial setup. As an example, if $R_t \equiv 0$ and E_t remains the same over time, we are essentially back to the classical IRL setting and suffer from the degeneracy issue. However, generalization to future tasks, which is our ultimate goal, is easy in this special case: after the initial demonstration, the agent can mimic it to behave optimally in all subsequent tasks without requiring further demonstrations.

More generally, if nature repeats similar tasks, then the agent obtains little new information, but presumably it knows how to behave in most cases; if nature chooses a task unfamiliar to the agent, then the agent is likely to err, but it may learn about $\theta_\star$ from the mistake.

To formalize this intuition, we consider the following protocol: the nature chooses a sequence of tasks $\{(E_t, R_t)\}$ in an arbitrary manner. For every task (E_t, R_t), the agent proposes a policy π_t. The human examines the policy's value under μ_t, and if the loss (recall Equation 6.2)

$$l_t = \mathbb{E}_{s\sim\mu}\left[V^{\pi_t^\star}_{E_t,\,\theta_\star+R_t}(s)\right] - \mathbb{E}_{s\sim\mu}\left[V^{\pi_t}_{E_t,\,\theta_\star+R_t}(s)\right] \tag{6.3}$$

is less than some ϵ then the human is satisfied and no demonstration is needed; otherwise a mistake is counted and $\eta^{\pi_t^\star}_{\mu_t,P_t}$ is revealed to the agent (note that $\eta^{\pi_t^\star}_{\mu_t,P_t}$ can be computed by the agent if needed from π_t^* and its knowledge of the task, so the reader can consider the case of the human presenting the policy w.l.o.g.). The main goal of this section is to design an algorithm that has a provable guarantee on the total number of mistakes.

Before describing and analyzing our algorithm, we first notice that the Equation 6.3 can be rewritten as

$$l_t = (\theta_\star + R)^\top(\eta^{\pi_t^\star}_{\mu_t,P_t} - \eta^{\pi_t}_{\mu_t,P_t}), \tag{6.4}$$

using Equation 6.1. So effectively we are given a set of state occupancy vectors $\{\eta^{\pi}_{\mu_t,P_t} : \pi \in (\mathcal{S} \to \mathcal{A})\}$ each round, and we want to choose the vector that has the largest dot product with $\theta_\star + R$. The exponential size of the set will not be a concern because our main result (Theorem 6.4) has no dependence on the number of vectors, and only depends on the dimension of those vectors. The result is enabled by studying the *linear bandit* version of the problem, which subsumes the MDP setting for our purpose and is also a model of independent interest.

6.5.1 The linear bandit setting

In the linear bandit setting, $\mathcal{D}$ is a finite action space with size $|\mathcal{D}| = K$. Each task is denoted as a pair (X, R). $X = [x^1 \ \cdots \ x^K]$ is a $d \times K$ feature matrix, where x^i is the feature vector for the i-th action, and $\|x^i\|_1 \leq 1$. When we reduce MDPs to linear bandits, each element of $\mathcal{D}$ corresponds to an MDP policy, and the feature vector is the state occupancy of that policy.

$R, \theta_\star \in \mathbb{R}^d$ are the task reward and the background reward, respectively. The initial uncertainty set for $\theta_\star$ is $\Theta_0 \subseteq [-1, 1]^d$. The value of the i-th action is calculated as $(\theta_\star + R)^\top x^i$, and $a^\star$ is the action that maximizes this value. Every round the agent proposes an action $a \in \mathcal{D}$, whose loss is defined as

$$l_t = (\theta_\star + R)^\top (x^{a^\star} - x^a).$$

We now show how to embed the previous MDP setting in linear bandits.

Example 1. Given an MDP problem with variables $\mathcal{S}, \mathcal{A}, \gamma, \theta_\star, s_{\text{ref}}, \Theta_0, \{(E_t, R_t)\}$, we can convert it into a linear bandit problem as follows. All variables with prime belong to the linear bandit problem, and we use $v^{\backslash i}$ to denote the vector v with the i-th coordinate removed.

- $\mathcal{D} = \{\pi : \mathcal{S} \to \mathcal{A}\}$, $d = |\mathcal{S}| - 1$.
- $\theta'_\star = \theta_\star^{\backslash s_{\text{ref}}}, \Theta'_0 = \{\theta^{\backslash s_{\text{ref}}} : \theta \in \Theta_0\}$.
- $x_t^\pi = (\eta^\pi_{\mu_t, P_t})^{\backslash s_{\text{ref}}}$. $R'_t = R_t^{\backslash s_{\text{ref}}} - R_t(s_{\text{ref}}) \cdot \mathbf{1}_d$.

Then for any sequence of policies chosen in the MDP problem, the corresponding sequence of actions in the linear bandit problem suffer exactly the same sequence of losses.

Note that there is a more straightforward conversion by letting $d = |\mathcal{S}|, \theta'_\star = \theta_\star, \Theta'_0 = \Theta_0, x_t^\pi = \eta^\pi_{\mu_t, P_t}, R'_t = R_t$, which also preserves losses. We perform a more succinct conversion in Example 1 by canonicalizing both $\theta_\star$ (already assumed) and R_t (explicitly done here) and dropping the coordinate for s_{ref} in all relevant vectors.

MDPs with linear rewards In IRL literature, a generalization of the MDP setting is often considered, that reward is linear in state features $\phi(s) \in \mathbb{R}^d$ [Ng and Russell, 2000, Abbeel and Ng, 2004]. In this new setting, $\theta_\star$ and R are reward parameters, and the actual reward is the dot product between the reward parameter and $\phi(s)$. This new setting can also be reduced to linear bandits similarly to Example 1, except that the state occupancy is replaced by the discounted sum of expected feature values. Our main result, Theorem 6.4, will still apply automatically, but now the guarantee will only depend on the dimension of the feature space and has no dependence on $|\mathcal{S}|$. We include the conversion below but do not further discuss this setting in the rest of the paper.

Example 2. Consider an MDP problem with state features, defined by $\mathcal{S}, \mathcal{A}, \gamma, d \in \mathbb{Z}^+, \theta_\star \in \mathbb{R}^d, \Theta_0 \subseteq [-1, 1]^d, \{(E_t, \phi_t \in \mathbb{R}^d, R_t \in \mathbb{R}^d)\}$, where task reward and background reward in state s are $\theta_\star^\top \phi_t(s)$ and $R^\top \phi_t(s)$ respectively, and $\theta_\star \in \Theta_0$. Suppose

Algorithm 3 Ellipsoid Algorithm for Repeated Inverse Reinforcement Learning

```
Input: $\Theta_0$.
$\Theta_1 := \mathrm{MVEE}(\Theta_0)$.
for $t = 1, 2, \ldots$ do
    Nature reveals $(X_t, R_t)$.
    Learner plays $a_t = \arg\max_{a \in \mathcal{D}} c_t^\top x_t^a$, where $c_t$ is the center of $\Theta_t$.
    if $l_t > \epsilon$ then
        Human reveals $a_t^\star$.
        $\Theta_{t+1} = \mathrm{MVEE}(\{\theta \in \Theta_t : (\theta - c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) \geq 0\})$.
    else
        $\Theta_{t+1} = \Theta_t$.
    end if
end for
```

$\|\phi_t(s)\|_\infty \leq 1$ always holds, then we can convert it into a linear bandit problem as follows:

- $\mathcal{D} = \{\pi : \mathcal{S} \to \mathcal{A}\}$; d, $\theta_\star$, and R_t remain the same.
- $x_t^\pi = (1-\gamma)\sum_{h=1}^\infty \gamma^{h-1}\mathbb{E}[\phi(s_h) \mid \mu_t, P_t, \pi]/d$.

Note that the division of d in x_t^π is for normalization purpose, so that $\|x_t^\pi\|_1 \leq \|\phi\|_1/d \leq \|\phi\|_\infty \leq 1$.

6.5.2 Ellipsoid Algorithm for Repeated Inverse RL

We propose Algorithm 3, and provide the mistake bound in the following theorem. Note that the pseudo-code also contains the formal protocol of the process.

Theorem 6.4. *For $\Theta_0 = [-1, 1]^d$, the number of mistakes made by Algorithm 3 is guaranteed to be $O(d^2 \log(d/\epsilon))$.*

To prove Theorem 6.4, we quote a result from linear programming literature in Lemma 6.5, which is found in standard lecture notes (e.g., O'Donnell 2011, Theorem 8.8; see also Grötschel et al. 2012, Lemma 3.1.34).

Lemma 6.5 (Volume reduction in ellipsoid algorithm)**.** *Given any non-degenerate ellipsoid B in $\mathbb{R}^d$ centered at $c \in \mathbb{R}^d$, and any non-zero vector $v \in \mathbb{R}^d$, let B^+ be the minimum-volume enclosing ellipsoid (MVEE) of*

$$\{u \in B : (u-c)^\top v \geq 0\}.$$

We have $\dfrac{vol(B^+)}{vol(B)} \leq e^{-\frac{1}{2(d+1)}}$.

Proof of Theorem 6.4. Whenever a mistake is made and the optimal action $a_t^\star$ is revealed, we can induce the constraint $(R_t + \theta_\star)^\top (x_t^{a_t^\star} - x_t^{a_t}) > \epsilon$. Meanwhile, since a_t is greedy w.r.t. c_t, we have $(R_t + c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) \le 0$, where c_t is the center of Θ_t as in Line 5. Taking the difference of the two inequalities, we obtain

$$(\theta_\star - c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) > \epsilon. \tag{6.5}$$

Therefore, the update rule on Line 8 preserves $\theta_\star$ in Θ_{t+1}. Since the update makes a central cut through the ellipsoid, Lemma 6.5 applies and the volume shrinks by a multiplicative constant $e^{-\frac{1}{2(d+1)}}$ every time a mistake is made.

To prove the theorem, it remains to upper bound the initial volume and lower bound the terminal volume of Θ_t. We first show that an update never eliminates $B_\infty(\theta_\star, \epsilon/2)$, the ℓ_∞ ball centered at $\theta_\star$ with radius $\epsilon/2$. This is because, any eliminated θ satisfies $(\theta + c_t)^\top (x_t^{a_t^\star} - x_t^{a_t}) < 0$. Combining this with Equation 6.5, we have

$$\epsilon < (\theta^\star - \theta)^\top (x_t^{a_t^\star} - x_t^{a_t}) \le \|\theta_\star - \theta\|_\infty \|x_t^{a_t^\star} - x_t^{a_t}\|_1 \le 2\|\theta_\star - \theta\|_\infty.$$

(This is very similar to the proof of Proposition 6.3.) We conclude that any eliminated θ should be $\epsilon/2$ far away from $\theta_\star$ in ℓ_∞ distance. Therefore, we can lower bound the volume of Θ_t for any t by that of $\Theta_0 \bigcap B_\infty(\theta_\star, \epsilon/2)$, which contains an infinite-norm ball with radius $\epsilon/4$ in the worst case (when $\theta_\star$ is one of Θ_0's vertices). To simplify calculation, we will further relax this ℓ_∞ ball to its inscribed ℓ_2 ball.

Finally we put everything together: let M_T be the number of mistakes made from round 1 to T, and C_d be the volume of the unit sphere in $\mathbb{R}^d$, we have

$$\begin{aligned} \frac{M_T}{2(d+1)} &\le \log(\text{vol}(\Theta_1)) - \log(\text{vol}(\Theta_{T+1})) \\ &\le \log(C_d(\sqrt{d})^d) - \log(C_d(\epsilon/4)^d) = d \log \frac{4\sqrt{d}}{\epsilon}. \end{aligned}$$

So $M_T \le 2d(d+1)\log\frac{4\sqrt{d}}{\epsilon} = O(d^2 \log \frac{d}{\epsilon})$. □

6.5.3 Lower bound

In Section 6.4, we get an $O(\log(1/\epsilon))$ upper bound on the number of demonstrations, which has no dependence on $|\mathcal{S}|$ (which corresponds to $d+1$ in linear bandits). Comparing Theorem 6.4 to 6.2, one may wonder whether the polynomial dependence on

d is an artifact of the inefficiency of Algorithm 3. We clarify this issue by proving a lower bound, showing that $\Omega(d \log(1/\epsilon))$ mistakes are inevitable in the worst case when nature chooses the tasks. We provide a proof sketch below, and the complete proof is deferred to Section 6.11.

Theorem 6.6. *For any randomized algorithm*[3] *in the linear bandit setting, there always exists $\theta_\star \in [-1,1]^d$ and $\{(X_t, R_t)\}$ which are fixed before the execution of the algorithm,*[4] *such that the expected number of mistakes made by the algorithm under $\theta_\star$ and $\{(X_t, R_t)\}$ is $\Omega(d \log(1/\epsilon))$.*

Proof Sketch. We randomize $\theta_\star$ by sampling each element i.i.d. from Unif($[-1,1]$). We will prove that there exists a strategy of choosing (X_t, R_t) such that any algorithm's expected number of mistakes is $\Omega(d \log(1/\epsilon)$, which proves the theorem as max is no less than average.

In our construction, $X_t = [\mathbf{0}_d,\ \mathbf{e}_{j_t}]$, where j_t is some index to be specified. Hence, every round the agent is essentially asked to decided whether $\theta(j_t) \geq -R_t(j_t)$. The adversary's strategy goes in phases, and R_t remains the same during each phase. Every phase has d rounds where j_t is enumerated over $\{1, \ldots, d\}$.

The adversary will use R_t to shift the posterior on $\theta(j_t) + R_t(j_t)$ so that it is (approximately) centered around the origin; in this way, the agent has about $1/2$ probability to make an error (regardless of the algorithm), and the posterior interval will be halved. Overall, the agent makes $d/2$ mistakes in each phase, and there will be about $\log(1/\epsilon)$ phases in total, which gives the lower bound. □

Applying the lower bound to MDPs The above lower bound is stated for linear bandits. In principle, we need to prove lower bound for MDPs separately, because linear bandits are more general than MDPs for our purpose, and the hard instances in linear bandits may not have corresponding MDP instances. In Lemma 6.7 below, we show that a certain type of linear bandit instances can always be emulated by MDPs with the same number of actions, and the hard instances constructed in Theorem 6.6 indeed satisfy the conditions for such a type; in particular, we require the feature vectors to be non-negative and have ℓ_1 norm bounded by 1. As a corollary, an $\Omega(|\mathcal{S}| \log(1/\epsilon))$ lower bound for the MDP setting (even with a small action space $|\mathcal{A}| = 2$) follows directly from Theorem 6.6.

[3] While our Algorithm 3 is deterministic, randomization is often crucial for online learning in general [Shalev-Shwartz, 2011].

[4] This means that the lower bound can be realized by an *oblivious adversary*, who cannot adapt the tasks to the realization of the random variables drawn by the algorithm.

Lemma 6.7 (Linear bandit to MDP conversion). *Let (X, R) be a linear bandit task, and K be the number of actions. If every x^a is non-negative and $\|x^a\|_1 \leq 1$, then there exists an MDP task (E, R') with $d+1$ states and K actions, such that under some choice of s_{ref}, converting (E, R') as in Example 1 recovers the original problem.*

The proof of this lemma is deferred to Section 6.8.

6.5.4 On identification when nature chooses tasks

While Theorem 6.4 successfully controls the number of total mistakes, it completely avoids the identification problem and does not guarantee to recover $\theta_\star$. Despite that our ultimate goal is to generalize to new tasks, obtaining identification guarantee is still meaningful in the setup of Section 6.5, because the protocol requires human to examine every proposed policy in addition to providing demonstrations upon observing mistakes, and upper bounds on mistakes do not take this supervision burden into consideration. On the other hand, once we have identified $\theta_\star$, generalization to new tasks is guaranteed and no further supervision is ever needed.

In this section we explore further conditions under which we can obtain identification guarantees when nature chooses the tasks. The first condition, stated in Proposition 6.8, implies that if we have made all the possible mistakes, then we have indeed identified the $\theta_\star$, where the identification accuracy is determined by the tolerance parameter ϵ that defines what is counted as a mistake.

Proposition 6.8. *Consider the linear bandit setting. If there exists T_0 such that for any round $t \geq T_0$, no more mistakes can be ever made by the algorithm for any choice of (E_t, R_t) and any tie-braking mechanism, then we have $\theta_\star \in B_\infty(c_{T_0}, \epsilon)$.*

Proof. Assume towards contradiction that $\|c_{T_0} - \theta_\star\|_\infty > \epsilon$. We will choose (R_t, x_t^1, x_t^2) to make the algorithm err. In particular, let $R_t = -c_{T_0}$, so that the algorithm acts greedily with respect to $\mathbf{0}_d$. Since $\mathbf{0}_d^\top x_t^a \equiv 0$, any action would be a valid choice for the algorithm.

On the other hand, $\|c_{T_0} - \theta_\star\|_\infty > \epsilon$ implies that there exists a coordinate j such that $|\mathbf{e}_j^\top(\theta_\star - c_{T_0})| > \epsilon$, where $\mathbf{e}_j$ is a basis vector. Let $x_t^1 = \mathbf{0}_d$ and $x_t^2 = \mathbf{e}_j$. So the value of action 1 is always 0 under any reward function (including $\theta_\star + R_t$), and the value of action 2 is $(\theta_\star + R_t)^\top x_t^2 = (\theta_\star - c_{T_0})^\top \mathbf{e}_j$, whose absolute value is greater than ϵ. At least one of the 2 actions is more than ϵ suboptimal, and the algorithm may take any of them, so the algorithm can err again. □

While Proposition 6.8 shows that identification is guaranteed if the agent exhausts the mistakes, the agent has no ability to actively fulfill this condition when Nature chooses tasks. For a stronger identification guarantee, we may need to grant the agent some freedom in choosing the tasks.

Identification with fixed environment

Here we consider a setting that fits in between Section 6.4 (completely active) and Section 6.5.1 (completely passive), where the environment E (hence the induced feature vectors $\{x^1, x^2, \ldots, x^K\}$) is given and fixed, and the agent can arbitrarily choose the task reward R_t. The goal is to obtain an identification guarantee in this new intermediate setting.

Unfortunately, a degenerate case can be easily constructed that prevents the revelation of any information about $\theta_\star$. In particular, if $x^1 = x^2 = \ldots = x^K$, i.e., the environment is completely uncontrolled, then all actions are equally optimal and nothing can be learned.

More generally, if for some non-zero vector v we have $v^\top x^1 = v^\top x^2 = \ldots = v^\top x^K$, then we may never recover $\theta_\star$ along the direction of v. In fact, Proposition 6.1 can be viewed as an instance of this result where $v = \mathbf{1}_{|\mathcal{S}|}$ (recall that the entries of the state occupancy vector always sum up to 1), and that is why we have to remove such redundancy in Example 1 in order to discuss identification in MDPs. Therefore, to guarantee identification in a fixed environment, the feature vectors must be substantially different in all directions, and we capture this intuition by defining a diversity score spread(X) (Definition 6.2) and showing that the identification accuracy depends inversely on the score (Theorem 6.9).

Definition 6.2. Given the feature matrix $X = \begin{bmatrix} x^1 & x^2 & \cdots & x^K \end{bmatrix}$ whose size is $d \times K$, define spread(X) as the d-th largest singular value of $\widetilde{X} := X(\mathbf{I}_K - \frac{1}{K}\mathbf{1}_K\mathbf{1}_K^\top)$.

Theorem 6.9. *For a fixed feature matrix X, if spread$(X) > 0$, then there exists a sequence $R_1, R_2, \ldots, R_T$ with $T = O(d^2 \log(d/\epsilon))$ and a sequence of tie-break choices of the algorithm, such that after round T we have* $\|c_T - \theta_\star\|_\infty \le \dfrac{\epsilon\sqrt{(K-1)/2}}{\textit{spread}(X)}$.

Proof. It suffices to show that in any round t, if $\|c_t - \theta_\star\|_\infty > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$, then $l_t > \epsilon$. The bound on T follows directly from Theorem 6.4. Similar to the proof of Proposition 6.8, our choice of the task reward is $R_t = -c_t$, so that any $a \in A$ would be a valid choice of a_t, and we will choose the worst action. Note that $\forall a, a' \in \mathcal{D}$,

$$l_t = (\theta_\star + R_t)^\top (x^{a_t^\star} - x^{a_t}) \geq (\theta_\star - c_t)^\top (x^a - x^{a'}).$$

So it suffices to show that there exists $a, a' \in \mathcal{D}$, such that $(\theta_\star - c_t)^\top (x^a - x^{a'}) > \epsilon$. Let $y_t = \theta_\star - c_t$, and the precondition implies that $\|y_t\|_2 \geq \|y_t\|_\infty > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$.

Define a matrix of size $K \times (K(K-1))$

$$D = \begin{bmatrix} 1 & 1 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & -1 \\ 0 & 0 & \cdots & 1 \end{bmatrix}. \tag{6.6}$$

Every column of this matrix contains exactly one 1 and one -1, and the columns enumerate all possible positions of them. With the help of this matrix, we can rewrite the desired result ($\exists\, a, a' \in A$, s.t. $(\theta_\star - c_t)^\top (x^a - x^{a'}) > \epsilon$) as $\|y_t^\top X D\|_\infty \geq \epsilon$. We relax the LHS as $\|y_t^\top X D\|_\infty \geq \|y_t^\top X D\|_2 / \sqrt{K(K-1)}$, and will provide a lower bound on $\|y_t^\top X D\|_2$. Note that

$$y_t^\top X D = y_t^\top (\widetilde{X} + (X - \widetilde{X})) D = y_t^\top \widetilde{X} D,$$

because every row of $(X - \widetilde{X})$ is some multiple of $\mathbf{1}_K^\top$ (recall Definition 6.2), and every column of D is orthogonal to $\mathbf{1}_K$. Let $\widehat{(\cdot)}$ be the vector normalized to unit length,

$$\|y_t^\top \widetilde{X} D\|_2 = \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X} D\|_2 = \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X}\|_2 D\|_2 = \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X}\|_2 \|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2.$$

We lower bound each of the 3 terms. For the first term, we have the precondition $\|y_t\|_2 > \frac{\epsilon\sqrt{(K-1)/2}}{\text{spread}(X)}$. The second term is $\widetilde{X}$ left multiplied by a unit vector, so its ℓ_2 norm can be lower bounded by the smallest non-zero singular value of $\widetilde{X}$ (recall that $\widetilde{X}$ is full-rank), which is spread(X).

To lower bound the last term, note that $DD^\top = 2K\mathbf{I}_K - 2\mathbf{1}_K\mathbf{1}_K^\top$, and rows of $\widetilde{X}$

are orthogonal to $\mathbf{1}_K^\top$ and so is $y_t^\top \widetilde{X}$, so

$$\begin{aligned}\|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2^2 &\geq \inf_{\|z\|_2=1,\, z\perp\mathbf{1}_K} z^\top D D^\top z = \inf_{\|z\|_2=1,\, z\perp\mathbf{1}_K} z^\top (2K\mathbf{I}_K - 2\mathbf{1}_K\mathbf{1}_K^\top) z \\ &= \inf_{\|z\|_2=1,\, z\perp\mathbf{1}_K} 2K z^\top z = 2K.\end{aligned}$$

Putting all the pieces together, we have

$$\begin{aligned}&\|y_t^\top \widetilde{X} D\|_\infty \geq \|y_t\|_2 \|\hat{y}_t^\top \widetilde{X}\|_2 \|\widehat{\hat{y}_t^\top \widetilde{X}}\, D\|_2 / \sqrt{d} \\ &> \frac{\epsilon\sqrt{(K-1)/2}}{\mathrm{spread}(X)} \cdot \mathrm{spread}(X) \cdot \frac{\sqrt{2K}}{\sqrt{K(K-1)}} = \epsilon.\end{aligned}$$ □

The $\sqrt{K}$ dependence in Theorem 6.9 may be of concern as K can be exponentially large. However, Theorem 6.9 also holds if we replace X by any matrix that consists of X's columns, so we may choose a small yet most diverse set of columns as to optimize the bound. We also show in Section 6.9 that Theorem 6.9 is tight in the worst case.

6.6 Working with Trajectories

In previous sections, we have assumed that the human evaluates the agent's performance based on the state occupancy of the agent's policy, and demonstrates the optimal policy in terms of state occupancy as well. In practice, we would like to instead assume that for each task, the agent rolls out a trajectory, and the human shows an optimal trajectory if he/she finds the agent's trajectory unsatisfying. We are still concerned about upper bounding the number of total mistakes, and aim to provide a parallel version of Theorem 6.4.

Unlike in traditional IRL, in our setting the agent is also acting, which gives rise to many subtleties. First, the total reward on the agent's single trajectory is a random variable, and may deviate from the expected value of its policy. Therefore, it is generally impossible to decide if the agent's policy is near-optimal, and instead we assume that the human can check if each action that the agent takes in the trajectory is near-optimal: when the agent takes a at state s, an error is counted if and only if $Q^\star(s,a) < V^\star(s) - \epsilon$.

While this resolves the issue on the agent's side, how should the human provide his/her optimal trajectory?. The most straightforward protocol is that the human

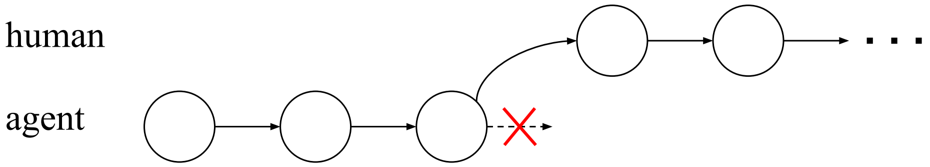

Figure 6.1: Illustration of the protocol in Section 6.6. Circles represent states and arrows represent actions. The agent rolls out a trajectory, and is stopped when taking a suboptimal action. The human continues the trajectory from the problematic state using an optimal policy.

rolls out a trajectory from the specified μ_t. We argue that this is not a reasonable protocol for two reasons: (1) in expectation, the reward collected by the human may be less than that by the agent, which is due to us conditioning on the event that an error is spotted; (2) the human may not encounter the problematic state in his/her own trajectory, hence the information provided in the trajectory may be irrelevant.

To resolve this issue, we consider a different protocol where the human rolls out a trajectory using optimal policy from the very state where the human errs. See Figure 6.1 for an illustration.

Now we discuss how we can prove a parallel of Theorem 6.4 under this new protocol. First, let's assume that the demonstration were still given in state occupancy induced by the optimal policy from the problematic state. In this case, we can treat the problematic state as the initial state, thanks to our assumption-free setup about (E_t, R_t) (hence μ_t). To reduce to our previous solution in Section 6.5, it remains to show that the notion of error in this section (a suboptimal action) implies the notion of error in Section 6.5 (a suboptimal policy): let s be the problematic state and π be the agent's policy, we have

$$V^{\pi}(s) = Q^{\pi}(s, \pi(s)) \leq Q^{\star}(s, \pi(s)) < V^{\star}(s) - \epsilon.$$

So whenever a suboptimal *action* is spotted in state s, it indeed implies that the agent's *policy* is suboptimal for s as the initial state. Hence, we can run Algorithm 3 and Theorem 6.4 immediately applies.

To tackle the remaining issue that the demonstration is in terms of a single trajectory, we will not update Θ_t after each mistake as in Algorithm 3, but only make an update after every mini-batch of mistakes, and aggregate them to form accurate update rules. See Algorithm 4. The choice of batch size n depends on the accuracy we need, and will be determined by the following concentration inequality.

Lemma 6.10 (Azuma's inequality for martingales)**.** *Suppose* $\{S_0, S_1, \ldots, S_n\}$ *is a mar-*

Algorithm 4 Trajectory version of Algorithm 3 for MDPs

```
**Input:** $\Theta_0, H, n$.
// variables with $'$ are converted as in Example 1.
$\Theta_1 := \text{MVEE}(\Theta_0')$, $i \leftarrow 0$, $\bar{Z} \leftarrow 0$, $\bar{Z}^\star \leftarrow 0$.
**for** $t = 1, 2, \ldots$ **do**
    Nature reveals $(E_t, R_t)$.
    Agent rolls-out a trajectory using $\pi_t$ greedily w.r.t. $c_t + R_t'$, where $c_t$ is the center of $\Theta_t$.
    **if** agent takes $a$ in $s$ with $Q^\star(s, a) < V^\star(s) - \epsilon$ **then**
        Human produces an $H$-step trajectory from $s$, whose empirical state occupancy vector (excluding the $s_{\text{ref}}$ coordinate) is denoted as $\hat{z}_i^{\star,H}$.
        $i \leftarrow i + 1$, $\bar{Z}^\star \leftarrow \bar{Z}^\star + \hat{z}_i^{\star,H}$.
        Let $z_i$ be the state occupancy of $\pi_t$ from initial state $s$, and $\bar{Z} \leftarrow \bar{Z} + z_i$.
        **if** $i = n$ **then**
            $\Theta_{t+1} := \text{MVEE}(\{\theta \in \Theta_t : (\theta - c_t)^\top (\bar{Z}^\star - \bar{Z}) \geq 0\})$.
            $i \leftarrow 0$, $\bar{Z} \leftarrow 0$, $\bar{Z}^\star \leftarrow 0$.
        **else**
            $\Theta_{t+1} = \Theta_t$.
        **end if**
    **else**
        $\Theta_{t+1} = \Theta_t$.
    **end if**
**end for**
```

tingale and $|S_i - S_{i-1}| \leq b$ *almost surely. Then with probability at least* $1 - \delta$ *we have* $|S_n - S_0| \leq b\sqrt{2n \log(2/\delta)}$.

Theorem 6.11. $\forall \delta \in (0, 1)$, *with probability at least* $1 - \delta$, *the number of mistakes made by Algorithm 4 with parameters* $\Theta_0 = \{\theta \in [-1, 1]^d : \theta(s_{ref}) = 0\}$, $H = \left\lceil \frac{\log(12/\epsilon)}{1-\gamma} \right\rceil$, *and* $n = \left\lceil \frac{\log(\frac{4d(d+1) \log \frac{6\sqrt{d}}{\epsilon}}{\delta})}{32\epsilon^2} \right\rceil$ *where* $d = |\mathcal{S}| - 1$, *is at most* $\tilde{O}(\frac{d^2}{\epsilon^2} \log(\frac{d}{\delta\epsilon}))$.[5]

The proof of Theorem 6.11 is deferred to Section 6.12.

6.7 Related Work and Discussions

6.7.1 Inverse RL, AI safety, and value alignment

Most existing work in IRL focused on inferring the reward function using data acquired from a fixed environment [Ng and Russell, 2000, Abbeel and Ng, 2004, Coates

[5] A $\log\log(1/\epsilon)$ term is suppressed in $\tilde{O}(\cdot)$.

et al., 2008, Ziebart et al., 2008, Ramachandran and Amir, 2007, Syed and Schapire, 2007, Regan and Boutilier, 2010]. There is prior work on using data collected from multiple — but exogenously fixed — environments to predict agent behavior [Ratliff et al., 2006]. There are also applications where methods for single-environment MDPs have been adapted to multiple environments [Ziebart et al., 2008]. Nevertheless, all these works consider the objective of mimicking an optimal behavior in the presented environment(s), and do not aim at generalization to new tasks.

Walsh et al. [2010] considered a setting where neither the transition dynamics nor the reward functions is known, and the learner rolls out its own trajectories and also receives demonstration trajectories from a teacher. They developed algorithms for choosing the agent's policies such that the number of rounds where the agent is significantly worse than the teacher can be bounded polynomially in the relevant parameters. Their interaction protocol and the form of their theoretical guarantees are very similar to Section 6.6 of this thesis. However, they only considered a single environment and allow the learner to observe the reward r_t in the trajectories. As a result, the demonstration trajectories are helpful but not necessary since the learner could accomplish learning on its own without the help of a teacher. In our setting (and in standard IRL literature), r_t is not observed and human demonstration is indispensable to the learning process.

In the economics literature, the problem of inferring an agent's utility from behavior has long been studied under the heading of utility or preference elicitation [Chajewska et al., 2000, Von Neumann and Morgenstern, 2007, Regan and Boutilier, 2009, 2011, Rothkopf and Dimitrakakis, 2011]. When these models analyze Markovian environments, they assume a fixed environment where the learner can ask certain types of queries, such as bound queries eliciting whether the reward in a state (and action) is above a threshold. While our result in Section 6.4 uses similar techniques to elicit the reward function, we do so purely by observing the human's behavior without external source of information (e.g., query responses).

The issue of reward misspecification is often mentioned in AI safety articles [e.g., Bostrom, 2003, Russell et al., 2015, Amodei et al., 2016]. These articles mostly discuss the ethical concerns and possible research directions, while our paper develops mathematical formulations and algorithmic solutions. Recently, Hadfield-Menell et al. [2016] proposed cooperative inverse reinforcement learning, where the human and the agent act in the same environment, allowing the human to actively resolve the agent's uncertainty on the reward function. However, they only consider a single environment (or task), and the unidentifiability issue of IRL still exists. Combining their interesting framework with our resolution to unidentifiability (by multiple tasks) can be an interesting future direction.

6.7.2 Connections to online learning and bandit literature

In online learning literature, there is a subfield called bandit linear optimization, which considers the regret minimization problem where the payoff function takes a linear form [Dani et al., 2007, Abernethy et al., 2008, Bartlett et al., 2008]. While out setup in Section 6.5.1 bears significant similarities to this line of research, they are also very different. The biggest difference is that the loss of the chosen action is always observed in bandits, while we only observe a weaker signal $\mathbb{I}(l_t \leq \epsilon)$ and an optimal demonstration. Furthermore, in bandit linear optimization, there is a constant set of feature vectors, and the learner competes against the best fixed vector; in our setting, the optimal feature vector is generally different from task to task, and competing against a fixed vector is vacuous. One could then view each candidate $\theta \in \Theta_0$ as an expert, and use algorithms for adversarial multi-armed bandits with expert advice [Auer et al., 1995], as our goal is indeed to compete against the best expert $\theta_\star$.[6] However, even if the loss were observed, the regret of a standard algorithm such as EXP4 is polynomial in K, which would be problematic for us as K can be exponentially large.

Another relevant setting in online learning is called *sleeping bandits*, where in addition to the standard adversarial bandit setting, every round only a subset of the actions is available and the availability may change from round to round. In our setting the whole action space is $\{x \in \mathbb{R}^d : \|x\|_1 \leq 1\}$, and the availability at round t is the columns of X_t. The works along this line of research often differ by the baselines they compete against [Freund et al., 1997, Blum and Mansour, 2007, Kleinberg et al., 2010], and the choice made by Kanade et al. [2009] matches our need most: they compete against the best fixed rank over actions, meaning that the baseline chooses the action with the highest rank in the available set; for us this rank over x is naturally given by the order of $\theta_\star^\top x$ (we treat $R_t \equiv \mathbf{0}_d$ to simplify the discussion here). Despite the relevance, as far as I know, most results in sleeping bandits incur polynomial dependence on the size of the whole action space; Neu and Valko [2014] have looked at combinatorial action space with stochastic availability and developed regret guarantees that are polynomial in the dimension of the action

[6] In fact this expert is *perfect*, that is, it incurs 0 loss. This corresponds to the *realizable* setting in online learning, which explains why we can talk about mistake bounds (instead of regret) and why we do not need a randomized algorithm for the upper bound.

space, but we need geometric action space and adversarial availability.

All the above literature does not assume $\theta_\star$ and compete against either the best fixed θ or the best mapping from availability set to θ in the hindsight. Stochastic linear bandit [Auer, 2002, Dani et al., 2008, Abbasi-Yadkori et al., 2012] is yet another setting that is highly relevant to our work, where $\theta_\star$ is well defined and the value of $\theta_\star^\top x_t^{a_t}$ is corrupted by independent zero-mean noise before it is revealed to the learner. While no-regret algorithms have been proposed and analyzed in this noisy setting, we observe that there is a simple algorithm for the noiseless setting that makes at most d mistakes, which has an interesting connection to the KWIK learning framework [Li et al., 2011b]. We present the concrete setting in the proposition below and specify the algorithm in the proof.

Proposition 6.12. *Consider the linear bandit setting in Section 6.5.1, and let $R_t \equiv \mathbf{0}_d$. Suppose we change the protocol to the following: the value of the action a_t chosen by the agent, that is, $\theta_\star^\top x_t^{a_t}$, is always revealed to the agent, and no demonstration is provided. Under this protocol, there exists an algorithm that makes at most d mistakes, where any suboptimal choice of action is counted as a mistake.*

Proof. The algorithm goes as follows. Before making the first mistake, the agent always chooses a non-zero vector in X; if this is impossible, it means that all vectors are $\mathbf{0}_d$ and they are equally good ($\theta_\star^\top \mathbf{0}_d \equiv 0$).

After making the first mistake at round t_1, the value of $\theta_\star^\top x_{t_1}^{a_{t_1}}$ is revealed to the agent, where $x_{t_1}^{a_{t_1}} \neq \mathbf{0}_d$. Starting from the next round, the agent always chooses a vector that does not lie in $\text{span}(\{x_{t_1}^{a_{t_1}}\})$. If this is impossible, any x available for choice can be written as a multiple of $x_{t_1}^{a_{t_1}}$, say, $x = c \cdot x_{t_1}^{a_{t_1}}$; then we compute the value for each x as $\theta_\star^\top x = c \cdot (\theta_\star^\top x_{t_1}^{a_{t_1}})$ and choose the optimal action without any uncertainty.

Generally, we maintain a sequence of feature vectors, $\{x_{t_i}^{a_{t_i}}\}_{i=1}^k$, where t_i's are the rounds where we make mistakes and k is the total number of mistakes made so far. If all available features in the next round lie in $\text{span}(\{x_{t_i}^{a_{t_i}}\}_{i=1}^k)$ (the "learned" subspace), we can predict the value of each action accurately and suffer no loss; otherwise we add a new vector that is linearly independent of the previous ones, and the dimension of the learned subspace increases by 1. After d mistakes, we will identify $\theta_\star$ exactly, and no further mistakes will be made. □

Note in the proof that, whenever all x in a particular round lie in the learned subspace, the algorithm chooses an x and knows that it is optimal for sure. Algorithms with such a property fit in the KWIK ("know what it knows") framework, and the algorithm used in Proposition 6.12 can be viewed as an adaptation of the deterministic linear regression algorithm in KWIK learning [Li et al., 2011b, Problem 3].

6.7.3 Alternative formulation using constraints

An important motivation for the work in this chapter is AI safety. While we model human's safety concerns and general preferences as a background reward function $\theta_\star$, an alternative formulation is to model safety concerns as constraints, that is, the agent should pursue the task-specific reward under the constraint that certain *unsafe* states should be avoided. We denote the set of unsafe states as $\mathcal{S}_{\text{bad}} \subset \mathcal{S}$, which remains the same from task to task and is the object of interest to the learning agent, just as $\theta_\star$ in the original formulation. Under this formulation, the optimal policy and its value in any given task (E, R) is specified by the following program:

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_{s\sim\mu}\big[V^{\pi}_{E,R}(s)\big] \\ \text{s.t.} \quad & \eta^{\pi}_{\mu,P}(s) = 0, \ \forall s \in \mathcal{S}_{\text{bad}}. \end{aligned} \tag{6.7}$$

We assume that there is always a policy that satisfies the constraint. Whenever the agent violates the safety constraints or achieves suboptimal value, a mistake is counted and the optimal policy described above is demonstrated.

Proposition 6.13. *For the setting described above, there exists an algorithm that makes at most $|\mathcal{S}|$ mistakes.*

Proof. The algorithm maintains a set of safe states $\mathcal{S}_{\text{safe}}$, initialized as the empty set. Whenever an optimal policy is demonstrated, we add the states visited by the demonstrated policy to $\mathcal{S}_{\text{safe}}$, so we always have that $\mathcal{S}_{\text{safe}} \subseteq \mathcal{S} \setminus \mathcal{S}_{\text{bad}}$. In any given task (E, R), the algorithm chooses a policy by solving the following program:

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_{s\sim\mu}\big[V^{\pi}_{E,R}(s)\big] \\ \text{s.t.} \quad & \eta^{\pi}_{\mu,P}(s) = 0, \ \forall s \in \mathcal{S} \setminus \mathcal{S}_{\text{safe}}. \end{aligned} \tag{6.8}$$

If no policy satisfies the constraint, it implies that no policy puts all its occupancy on states in $\mathcal{S}_{\text{safe}}$; therefore, any safe policy, including the one demonstrated to the agent, puts some occupancy on states outside $\mathcal{S}_{\text{safe}}$, and the agent can grow $\mathcal{S}_{\text{safe}}$ by at least 1 element. Using a similar argument, if the policy found by the agent is suboptimal, the true optimal policy must put occupancy on states outside $\mathcal{S}_{\text{safe}}$, and again $\mathcal{S}_{\text{safe}}$ grows by at least 1 element. Since $|\mathcal{S}_{\text{safe}}| \le |\mathcal{S}|$, the algorithm makes at most $|\mathcal{S}|$ mistakes. $\square$

6.8 Proof of Lemma 6.7

The construction is as follows. Choose s_{ref} as the initial state, and make all other states absorbing. Let $R'(s_{\text{ref}}) = 0$ and R' restricted on $\mathcal{S} \setminus \{s_{\text{ref}}\}$ coincide with R. The remaining work is to design the transition distribution of each action in s_{ref} so that the induced state occupancy matches exactly one column of X.

Fixing any action a, and let x be the feature that we want to associate a with. The next-state distribution of (s_{ref}, a) is as follows: with probability $p = \frac{1-\|x\|_1}{1-\gamma\|x\|_1}$ the next-state is s_{ref} itself, and the probability of transitioning to the j-th state in $\mathcal{S} \setminus \{s_{\text{ref}}\}$ is $\frac{1-\gamma}{1-\gamma\|x\|_1} x(j)$. Given $\|x\|_1 \le 1$ and $x \ge 0$, it is easy to verify that this is a valid distribution.

Now we calculate the occupancy of policy $\pi(s_{\text{ref}}) = a$. The normalized occupancy on s_{ref} is

$$(1-\gamma)(p + \gamma p^2 + \gamma^2 p^3 + \cdots) = \frac{p(1-\gamma)}{1-\gamma p} = 1 - \|x\|_1.$$

The remaining occupancy, with a total ℓ_1 mass of $\|x\|_1$, is split among $\mathcal{S} \setminus \{s_{\text{ref}}\}$ proportional to x. Therefore, when we convert the MDP problem as in Example 1, the corresponding feature vector is exactly x, so we recover the original linear bandit problem. $\square$

6.9 Tightness of Theorem 6.9

We show that the theorem is tight up to a constant factor in the worst case. Let $X = \begin{bmatrix} U & -U \end{bmatrix}$ where $U_{d\times d}$ is any orthonormal matrix, so $K = 2d$. This is a valid choice of X because its column vector x satisfies $\|x\|_1 \le \|x\|_2 = 1$. All d singular values of X are $\sqrt{2}$, and $\tilde{X} = X$, so $\text{spread}(X) = \sqrt{2}$, and the bound is $\epsilon\sqrt{2d-1}/2 = O(\epsilon\sqrt{d})$.

Since U is arbitrary, we choose its first row to be $\mathbf{1}_d^\top/\sqrt{d}$. Then we choose an ellipsoid center c and $\theta_\star$ that are $\epsilon\sqrt{d}/2$ different from each other in ℓ_∞ distance, and show that a mistake is impossible. In particular, let c be equal to $\theta_\star$ except on the first coordinate where they differ by $\epsilon\sqrt{d}/2$. Let a be the action taken by the algorithm and $a^\star$ be an optimal action, and R be any task reward, we have

$$\begin{aligned}\text{loss} &= (\theta_\star + R)(x^{a^\star} - x^a)\\ &\leq (\theta_\star + R)(x^{a^\star} - x^a) - (c + R)(x^{a^\star} - x^a)\\ &= (\theta_\star - c)(x^{a^\star} - x^a)\\ &= |\theta_\star(1) - c(1)||x^{a^\star}(1) - x^a(1)|\\ &\leq \epsilon\sqrt{d}/2 \cdot (2/\sqrt{d}) = \epsilon.\end{aligned}$$

In addition, note that the same construction also works if we rescale X with any multiplicative constant $C \in (0,1)$, hence the bound is tight in the worst case not only for $\text{spread}(X) = \sqrt{2}$, but for a range $\text{spread}(X) \in (0, \sqrt{2}]$.

6.10 A Technical Note on Theorem 6.11

Bounding the ℓ_∞ distance between $\theta_\star$ and the ellipsoid center To prove Theorem 6.11, we need an upper bound on $\|\theta_\star - c\|_\infty$ for quantifying the error due to H-step truncation and sampling effects, where c is the ellipsoid center. As far as we know there is no standard result on this issue. However, a simple workaround, described below, allows us to assume $\|\theta_\star - c\|_\infty \leq 2$ without loss of generality.

Whenever $\|c\|_\infty > 1$, there exists coordinate j such that $|c_j| > 1$. We can make a central cut $\mathbf{e}_j^\top(\theta - c) < 0$ (or > 0 depending on the sign of c_j), and replace the original ellipsoid with the MVEE of the remaining shape. This operation never excludes any point in Θ_0, hence it allows the proofs of Theorem 6.4 and 6.11 to work. We keep making such cuts and update the ellipsoid accordingly, until the new center satisfies $\|c\|_\infty \leq 1$. Since central cuts reduce volume substantially (Lemma 6.5) and there is a lower bound on the volume, the process must stop after finite number of operations. After the process stops, we have $\|\theta_\star - c\|_\infty \leq \|\theta_\star\|_\infty + \|c\|_\infty \leq 2$.

6.11 Proof of Theorem 6.6

As a standard trick, we randomize $\theta_\star$ by sampling each element i.i.d. from $\text{Unif}([-1,1])$. We will prove that there exists a strategy of choosing (X_t, R_t) such that any algorithm's expected number of mistakes is $\Omega(d\log(1/\epsilon)$, where the expectation is with respect to the randomness of $\theta_\star$ and the internal randomness of the algorithm. This immediately implies a worst-case result as max is no less than average (regarding the sampling of $\theta_\star$).

In our construction, $X_t = [\mathbf{0}_d,\ \mathbf{e}_{j_t}]$, where j_t is some index to be specified. Hence, every round the agent is essentially asked to decided whether $\theta(j_t) \geq -R_t(j_t)$. The adversary's strategy goes in phases, and R_t remains the same during each phase. Every phase has d rounds where j_t is enumerated over $\{1, \dots, d\}$. To fully specify the nature's strategy, it remains to specify R_t for each phase.

In the 1st phase, $R_t \equiv 0$. For each coordinate j, the information revealed to the agent is one of the following: $\theta_\star(j) > \epsilon$, $\theta_\star(j) \geq -\epsilon$, $\theta_\star(j) < -\epsilon$, $\theta_\star(j) \leq \epsilon$. For clarity we first make an simplification, that the revealed information is either $\theta_\star(j) > 0$ or $\theta_\star(j) \leq 0$; we will deal with the subtleties related to ϵ at the end of the proof.

In the 2nd phase, we fix R_t as

$$R_t(j) = \begin{cases} -1/2 & \text{if } \theta_\star(j) \geq 0, \\ 1/2 & \text{if } \theta_\star(j) < 0. \end{cases}$$

Since $\theta_\star$ is randomized i.i.d. for each coordinate, the posterior of $\theta_\star + R_t$ conditioned on the revealed information is Unif$[-1/2, 1/2]$, for any algorithm and any interaction history. Therefore the 2nd phase is almost identical to the 1st phase except that the intervals have shrunk by a factor of 2. Similarly in the 3rd phase we use R_t to offset the posterior of $\theta_\star + R_t$ to Unif$([-1/4, 1/4])$, and so on.

In phase m, the half-length of the interval is 2^{-m+1}, and the probability that a mistake occurs is at least $1/2 - \epsilon/2^{-m+2}$ for any algorithm. The whole process continues as long as this probability is greater than 0. By linearity of expectation, we can lower bound the total mistakes by the sum of expected mistakes in each phase, which gives

$$\sum_{2^{-m+1} \geq \epsilon} d(1/2 - \epsilon/2^{-m+2}) \geq \sum_{2^{-m+1} \geq 2\epsilon} d \cdot 1/4 \geq \lfloor \log_2(1/\epsilon) \rfloor d/4.$$

The above analysis made a simplification that the posterior of $\theta_\star + R_t$ in phase m is $[-2^{-m+1}, 2^{-m+1}]$. We now remove the simplification. Note, however, that the actual posterior cannot be too different from this simplified version, and their end points can differ by at most ϵ. So the error probability is at least $1/2 - 2\epsilon/(2^{-m+2} - 2\epsilon)$. The rest of the analysis is similar: we count the number of mistakes until the error probability drops below $1/4$, and in each of these phases we get at least $d/4$ mistakes in expectation. The number of such phases is given by

$$1/2 - 2\epsilon/(2^{-m+2} - 2\epsilon) \geq 1/4,$$

which is satisfied if $2^{-m+2} \geq 6\epsilon$, so $m \geq \lfloor \log_2 \frac{2}{3\epsilon} \rfloor$. This completes the proof. □

6.12 Proof of Theorem 6.11

Since the update rule is still in the format of a central cut through the ellipsoid, Lemma 6.5 applies. It remains to show that the update rule preserves $\theta_\star$ and a certain volume around it, and then we can follow the same argument as for Theorem 6.4.

Fixing a mini-batch, let t_0 be the round on which the last update occurs, and $\Theta = \Theta_{t_0}, c = c_{t_0}$. Note that $\Theta_t = \Theta$ during the collection of the current mini-batch and does not change, and $c_t = c$ similarly.

For each $i = 1, 2, \ldots, n$, define $z_i^{\star,H}$ as the expected value of $\hat{z}_i^{\star,H}$, where expectation is with respect to the randomness of the trajectory produced by the human, and let $z_i^\star$ be the infinite-step expected state occupancy. Note that $\hat{z}_i^{\star,H}, z_i^{\star,H}, z_i^\star \in \mathbb{R}^{|\mathcal{S}|-1}$ because the occupancy on s_{ref} is not included.

As before, we have $\theta_\star^\top(z_i^\star - z_i) > \epsilon$ and $c^\top(z_i^\star - z_i) \leq 0$, so $(\theta_\star - c)^\top(z_i^\star - z_i) > \epsilon$. Taking average over i, we get $(\theta_\star - c)^\top(\frac{1}{n}\sum_{i=1}^n z_i^\star - \frac{1}{n}\sum_{i=1}^n z_i) > \epsilon$.

What we will show next is that $(\theta_\star - c)^\top(\frac{\bar{Z}^\star}{n} - \frac{\bar{Z}}{n}) > \epsilon/3$ for $\bar{Z}^\star$ and $\bar{Z}$ on Line 12, which implies that the update rule is valid and has enough slackness for lower bounding the volume of Θ_t as before. Note that

$$\begin{aligned}
&(\theta_\star - c)^\top(\tfrac{\bar{Z}^\star}{n} - \tfrac{\bar{Z}}{n}) = (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^\star - \tfrac{1}{n}\sum_{i=1}^n z_i) \\
&\quad - (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^\star - \tfrac{1}{n}\sum_{i=1}^n z_i^{\star,H}) \\
&\quad - (\theta_\star - c)^\top(\tfrac{1}{n}\textstyle\sum_{i=1}^n z_i^{\star,H} - \tfrac{1}{n}\sum_{i=1}^n \hat{z}_i^{\star,H}).
\end{aligned}$$

Here we decompose the expression of interest into 3 terms. The 1st term is lower bounded by ϵ as shown above, and we will upper bound each of the remaining 2 terms by $\epsilon/3$. For the 2nd term, since $\|z_i^{\star,H} - z_i^\star\|_1 \leq \gamma^H$, the ℓ_1 norm of the average follows the same inequality due to convexity, and we can bound the term using Hölder's inequality given $\|\theta_\star - c\|_\infty \leq 2$ (see details of this result in Section **??**). To verify that the choice of H in the theorem statement is appropriate, we can upper bound the 2nd term as

$$2\gamma^H = 2((1-(1-\gamma))^{\frac{1}{1-\gamma}})^{\log(6/\epsilon)} \leq 2e^{-\log(6/\epsilon)} = \tfrac{\epsilon}{3}.$$

For the 3rd term, fixing $\theta_\star$ and c, the partial sum $\sum_{j=1}^{i}(\theta_\star - c)^\top(z_i^{\star,H} - \hat{z}_i^{\star,H})$ is a martingale. Since $\|z_i^{\star,H}\|_1 \leq 1$, $\|\hat{z}_i^{\star,H}\|_1 \leq 1$, and $\|\theta_\star - c\|_\infty \leq 2$, we can initiate Lemma 6.10 by letting $b = 4$, and setting n to sufficiently large to guarantee that the 3rd term is upper bounded by $\epsilon/3$ with high probability.

Given $(\theta_\star - c)^\top(\frac{\bar{Z}^\star}{n} - \frac{\bar{Z}}{n}) > \epsilon/3$, we can follow exactly the same analysis as for Theorem 6.4 to show that $B_\infty(\theta_\star, \epsilon/6)$ is never eliminated, and the number of updates can be bounded by $2d(d+1)\log\frac{12\sqrt{d}}{\epsilon}$. The number of total mistakes is the number of updates multiplied by n, the size of the mini-batches. Via Lemma 6.10, we can verify that the choice of n in the theorem statement satisfies $|\sum_{j=1}^{i}(\theta_\star - c)^\top(z_i^{\star,H} - \hat{z}_i^{\star,H})| \leq n\epsilon/3$ with probability at least $1 - \delta/\left(2d(d+1)\log\frac{12\sqrt{d}}{\epsilon}\right)$. Union bounding over all updates and the total failure probability can be bounded by δ. □

CHAPTER 7

Conclusion

In the beginning of this thesis we motivated the model selection problem in reinforcement learning by referring to the situation of supervised learning: any supervised learning algorithm would only work well under appropriate choice of hyperparameters, which is why systematic procedures for tuning these hyperparameters play a central role in the practice and the theory of supervised learning. This thesis attempts to establish parallel results in the reinforcement learning setting. In particular, we have looked at 3 types of hyperparameters in RL:

1. Chapter 3 investigated the overfitting phenomenon caused by a long planning horizon (or a large guidance discount factor), and showed that a model-based cross-validation procedure can effective select a good discount factor in the tabular setting.
2. Chapter 5 analyzed the finite-sample performance of state abstractions, and proposed a regularization algorithm that can select a good state abstraction with adaptivity guarantees.
3. Chapter 6 looked at a meta-level problem of learning reward function, and proposed a novel repeated formulation of Inverse RL. We showed that an algorithm can learn the correct reward function while requesting a small number of human demonstrations.

Besides, in Chapter 4 we also examined the off-policy value evaluation problem which plays an important role in batch model selection. We extended the bandit doubly robust estimator and developed an unbiased estimator for the sequential setting with state-of-the-art variance, and also proved lower bound for the problem.

7.1 Discussions and Future Research Possibilities

We briefly discuss some limitations of this thesis and future research possibilities.

Model selection beyond nested abstractions
In Chapter 5 we proposed a new algorithm that can select a good state abstraction among a set of nested ones. A natural question to ask is whether we can relax the assumption of nested abstractions and obtain similar adaptivity guarantees with respect to arbitrary candidate sets. Either proving the statement (i.e., designing effective algorithms) or disproving it (i.e., showing hardness results) will provide a more complete understanding of abstraction selection in the batch setting. Moreover, if the answer is positive, we can further ask whether it is possible to select among an arbitrary set of value-function classes, which subsume state abstractions as special cases (an abstraction corresponds to a space of piece-wise constant functions).

Model selection with exploration in the online setting
This thesis did not address the exploration challenge and assumed batch setting for most chapters. The model selection problem is also important for the online setting where the agent controls action selection and performs exploration. For state abstractions, Ortner et al. [2014] did some valuable investigations but the results were unsatisfying in the agnostic setting. For the most general case where we select among base algorithms as black boxes, even the bandit case has not been looked at until recently [Agarwal et al., 2016]. While the online setting has the unique challenge of exploration, the fact that the agent has control over actions is very powerful, and leveraging this power may be key to lifting some of the limitations in the batch setting.

Off-policy evaluation: beyond exponential lower bound
The exponential lower bound for off-policy evaluation in Chapter 4 is disappointing (though not surprising). A next step is to identify domain structures and exploit them to enable effective off-policy evaluation. Another interesting direction is the setting where we have very limited access to some online data where we get to choose the actions (e.g., testing a new policy at a very small scale). Presumably the amount of data is not sufficient to support Monte-Carlo policy evaluation; how could we combine the offline and online data for optimal value estimation? In any

case, rigorous and effective off-policy evaluation would require identification of realistic yet tractable scenarios in real-life applications.

Interleaved learning of dynamics and reward function

Chapter 6 adopts the common assumption in Inverse RL literature that dynamics are known to the agent. In practice, of course, the dynamics of the environment are typically unknown and also need to be learned. How to incorporate dynamics learning into the Repeated Inverse RL framework is an important question towards practice and may require careful formulation. For example, if we take the current trajectory setting in Section 6.6 and simply remove the knowledge of dynamics from the agent, the problem is hopeless as the agent only obtains one trajectory from the environment and the dynamics are chosen by the adversary every time. In general there might be a multi-task RL / transfer learning component in the problem that needs to be carefully characterized.

BIBLIOGRAPHY

Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Online-to-confidence-set conversions and application to sparse stochastic bandits. In *AISTATS*, volume 22, pages 1–9, 2012.

Pieter Abbeel and Andrew Y Ng. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine learning*, page 1. ACM, 2004.

Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y Ng. An application of reinforcement learning to aerobatic helicopter flight. *Advances in Neural Information Processing Systems*, 19:1, 2007.

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Conference on Learning Theory*, pages 263–274, 2008.

Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E Schapire. Corralling a band of bandit algorithms. *arXiv preprint arXiv:1612.06246*, 2016.

Kareem Amin, Nan Jiang, and Satinder Singh. Repeated Inverse Reinforcement Learning for AI Safety. *arXiv preprint arXiv:1705.05427*, 2017.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

John Asmuth and Michael L Littman. Approaching Bayes-optimalilty using Monte-Carlo tree search. In *Proceedings of the 21st International Conference on Automatated Planning and Scheduling*, 2011.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Peter Auer and Ronald Ortner. Logarithmic online regret bounds for undiscounted reinforcement learning. *Advances in Neural Information Processing Systems*, 19:49, 2007.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations* *of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 322–331. IEEE, 1995.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

Maria-Florina Balcan. *CS 8803 - Machine Learning Theory: Lecture Notes*. Georgia Institute of Technology, 2011. http://www.cs.cmu.edu/~ninamf/ML11/lect1115.pdf.

Peter L Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3: 463–482, 2003.

Peter L Bartlett, Varsha Dani, Thomas Hayes, Sham Kakade, Alexander Rakhlin, and Ambuj Tewari. High-probability regret bounds for bandit online linear optimization. pages 335–342, 2008.

Jonathan Baxter and Peter L Bartlett. Infinite-Horizon Policy-Gradient Estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

Richard Bellman. Dynamic programming and Lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3)*. Athena Scientific, 1996.

Luca F Bertuccelli, Albert Wu, and Jonathan P How. Robust adaptive markov decision processes: Planning with model uncertainty. *Control Systems, IEEE*, 32(5): 96–109, 2012.

Avrim Blum and Yishay Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8(Jun):1307–1324, 2007.

Nick Bostrom. Ethical issues in advanced artificial intelligence. *Science Fiction and Philosophy: From Time Travel to Superintelligence*, pages 277–284, 2003.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis Xavier Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *AAAI/IAAI*, pages 363–369, 2000.

Adam Coates, Pieter Abbeel, and Andrew Y Ng. Learning for control from multiple demonstrations. In *Proceedings of the 25th international conference on Machine learning*, pages 144–151. ACM, 2008.

Varsha Dani, Sham M Kakade, and Thomas P Hayes. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems*, pages 345–352, 2007.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *COLT*, pages 355–366, 2008.

Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

Christoph Dann, Gerhard Neumann, and Jan Peters. Policy evaluation with temporal differences: A survey and comparison. *The Journal of Machine Learning Research*, 15(1):809–883, 2014.

Monica Dinculescu and Doina Precup. Approximate predictive representations of partially observable systems. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 895–902, 2010.

Miroslav Dudík, John Langford, and Lihong Li. Doubly Robust Policy Evaluation and Learning. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1097–1104, 2011.

Ran El-Yaniv and Dmitry Pechyony. Transductive rademacher complexity and its applications. In *Learning Theory*, pages 157–171. Springer, 2007.

Eyal Even-Dar and Yishay Mansour. Approximate equivalence of Markov decision processes. In *Learning Theory and Kernel Machines*, pages 581–594. Springer, 2003.

Amir-massoud Farahmand and Csaba Szepesvári. Model selection in reinforcement learning. *Machine learning*, 85(3):299–332, 2011.

Amir-massoud Farahmand, Csaba Szepesvári, and Rémi Munos. Error Propagation for Approximate Policy and Value Iteration. In *Advances in Neural Information Processing Systems*, pages 568–576, 2010.

Raphael Fonteneau, Susan A. Murphy, Louis Wehenkel, and Damien Ernst. Batch mode reinforcement learning based on the synthesis of artificial trajectories. *Annals of Operations Research*, 208(1):383–416, 2013.

Yoav Freund, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual ACM Symposium on Theory of computing*, pages 334–343. ACM, 1997.

Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 147(1):163–223, 2003.

Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

S Grünewälder, G Lever, L Baldassarre, M Pontil, and A Gretton. Modelling transition dynamics in MDPs with RKHS embeddings. In *Proceedings of the 29th International Conference on Machine Learning*, pages 535–542, 2012.

Arthur Guez, David Silver, and Peter Dayan. Efficient Bayes-Adaptive Reinforcement Learning using Sample-Based Search. In *Advances in Neural Information Processing Systems*, pages 1034–1042, 2012.

Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3909–3917, 2016.

Assaf Hallak, Dotan Di-Castro, and Shie Mannor. Model selection in markovian processes. In *Proceedings of the 19th ACM SIGKDD Conference on Knowledge Discovery and Data mining*, pages 374–382, 2013.

S Hettich and S D Bay. The UCI KDD Archive. `http://kdd.ics.uci.edu`, 1999.

Keisuke Hirano, Guido W. Imbens, and Geert Ridder. Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 71(4):1161–1189, 2003.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

Paul W. Holland. Statistics and causal inference. *Journal of the American Statistical Association*, 81(6):945–960, 1986.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Nan Jiang and Lihong Li. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 652–661, 2016.

Nan Jiang, Satinder Singh, and Richard Lewis. Improving UCT planning via approximate homomorphisms. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1289–1296, 2014.

Nan Jiang, Alex Kulesza, and Satinder Singh. Abstraction Selection in Model-based Reinforcement Learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 179–188, 2015a.

Nan Jiang, Alex Kulesza, Satinder Singh, and Richard Lewis. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1181–1189, 2015b.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual Decision Processes with low Bellman rank are PAC-learnable. *arXiv preprint arXiv:1610.09512*, 2016.

Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. The malmo platform for artificial intelligence experimentation. In *International joint conference on artificial intelligence (IJCAI)*, page 4246, 2016.

Nicholas K Jong and Peter Stone. State abstraction discovery from irrelevant state variables. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 752–757, 2005.

Nicholas K Jong and Peter Stone. Model-based function approximation in reinforcement learning. In *Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems*, page 95, 2007.

Sham Kakade and John Langford. Approximately Optimal Approximate Reinforcement Learning. In *Proceedings of the 19th International Conference on Machine Learning*, volume 2, pages 267–274, 2002.

Sham Machandranath Kakade et al. *On the sample complexity of reinforcement learning*. PhD thesis, University of London London, England, 2003.

Varun Kanade, H Brendan McMahan, and Brent Bryan. Sleeping experts and bandits with stochastic action availability and adversarial rewards. In *AISTATS*, pages 272–279, 2009.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning*, 49(2-3):193–208, 2002.

Michael J Kearns and Umesh V Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994. ISBN 9780262111935. URL http://books.google.com/books?id=vCA01wY6iywC.

Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma. Regret bounds for sleeping experts and bandits. *Machine learning*, 80(2-3):245–272, 2010.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer Berlin Heidelberg, 2006.

Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In *High Dimensional Probability II*, pages 443–457. Springer, 2000.

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pages 1840–1848, 2016.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems*, pages 817–824, 2008.

H Lei, I Nahum-Shani, K Lynch, D Oslin, and SA Murphy. A" smart" design for building individualized treatment sequences. *Annual review of clinical psychology*, 8:21–48, 2012.

Guy Lever, Luca Baldassarre, Arthur Gretton, Massimiliano Pontil, and Steffen Grünewälder. Modelling transition dynamics in MDPs with RKHS embeddings. In *Proceedings of the 29th International Conference on Machine Learning*, pages 535–542, 2012.

Lihong Li. *A unifying framework for computational reinforcement learning theory*. PhD thesis, Rutgers, The State University of New Jersey, 2009.

Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for MDPs. In *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics*, pages 531–539, 2006.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms. In *Proceedings of the 4th International Conference on Web Search and Data Mining*, pages 297–306, 2011a.

Lihong Li, Michael L Littman, Thomas J Walsh, and Alexander L Strehl. Knows what it knows: a framework for self-aware learning. *Machine learning*, 82(3):399–443, 2011b.

Lihong Li, Remi Munos, and Csaba Szepesvári. Toward minimax off-policy value estimation. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS-15)*, pages 608–616, 2015a.

Lihong Li, Rémi Munos, and Csaba Szepesvári. Toward minimax off-policy value estimation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015b.

Xiujun Li, Lihong Li, Jianfeng Gao, Xiaodong He, Jianshu Chen, Li Deng, and Ji He. Recurrent reinforcement learning: A hybrid approach, 2015c. arXiv:1509.03044.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

Odalric-Ambrym Maillard, Timothy A Mann, and Shie Mannor. "How hard is my MDP?" The distribution-norm to the rescue. In *Advances in Neural Information Processing Systems*, pages 1835–1843, 2014.

Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1077–1084, 2014.

Shie Mannor, Duncan Simester, Peng Sun, and John N Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, 2007.

Vukosi N. Marivate. *Improved Empirical Methods in Reinforcement-Learning Evaluation*. PhD thesis, Rutgers University, New Brunswick, NJ, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

Terrence Joseph Moore Jr. *A theory of Cramer-Rao bounds for constrained parametric models*. PhD thesis, University of Maryland, College Park, 2010.

Rémi Munos. Performance bounds in l_p-norm for approximate value iteration. *SIAM journal on control and optimization*, 46(2):541–561, 2007.

Susan A. Murphy, Mark van der Laan, and James M. Robins. Marginal Mean Models for Dynamic Regimes. *Journal of the American Statistical Association*, 96(456):1410–1423, 2001.

Gergely Neu and Michal Valko. Online combinatorial optimization with stochastic decision sets and adversarial losses. In *Advances in Neural Information Processing Systems*, pages 2780–2788, 2014.

Andrew Y Ng and Michael Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Proceedings of the16th Conference on Uncertainty in Artificial Intelligence*, pages 406–415, 2000.

Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670, 2000.

Andrew Y Ng, H Jin Kim, Michael I Jordan, Shankar Sastry, and Shiv Ballianda. Autonomous helicopter flight via reinforcement learning. In *NIPS*, volume 16, 2003.

Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

Ryan O'Donnell. *15-859(E) – linear and semidefinite programming: lecture notes*. Carnegie Mellon University, 2011. https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15859-f11/www/notes/lecture08.pdf.

Dirk Ormoneit and Śaunak Sen. Kernel-based reinforcement learning. *Machine learning*, 49(2-3):161–178, 2002.

Ronald Ortner, Odalric-Ambrym Maillard, and Daniil Ryabko. Selecting Near-Optimal Approximate State Representations in Reinforcement Learning. *arXiv preprint arXiv:1405.2652*, 2014.

Cosmin Paduraru. *Off-policy Evaluation in Markov Decision Processes*. PhD thesis, McGill University, 2013.

Cosmin Paduraru, Robert Kaplow, Doina Precup, and Joelle Pineau. Model-based reinforcement learning with state aggregation. In *8th European Workshop on Reinforcement Learning*, 2008.

Ronald Parr, Lihong Li, Gavin Taylor, Christopher Painter-Wakefield, and Michael L Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 752–759, 2008.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009. ISBN 052189560X.

Marek Petrik and Bruno Scherrer. Biasing approximate dynamic programming with a lower discount factor. In *Advances in Neural Information Processing Systems*, pages 1265–1272, 2009.

Matteo Pirotta, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello. Safe policy iteration. In *Proceedings of the 30th International Conference on Machine Learning*, number 3, pages 307–317, 2013.

Doina Precup. *Temporal abstraction in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 2000.

Doina Precup, Richard S Sutton, and Satinder P Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 759–766, 2000.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-Policy Temporal-Difference Learning with Funtion Approximation. In *Proceedings of the 18th International Conference on Machine Learning*, pages 417–424, 2001.

ML Puterman. Markov decision processes. *Jhon Wiley & Sons, New Jersey*, 1994.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. *Urbana*, 51:61801, 2007.

Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 729–736. ACM, 2006.

Balaraman Ravindran. *An algebraic approach to abstraction in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 2004.

Balaraman Ravindran and A Barto. Approximate homomorphisms: A framework for nonexact minimization in Markov decision processes. In *5th International Conference on Knowledge-Based Computer Systems*, 2004.

Kevin Regan and Craig Boutilier. Regret-based reward elicitation for markov decision processes. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 444–451. AUAI Press, 2009.

Kevin Regan and Craig Boutilier. Robust policy computation in reward-uncertain mdps using nondominated policies. In *AAAI*, 2010.

Kevin Regan and Craig Boutilier. Eliciting additive reward functions for markov decision processes. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 2159, 2011.

Stéphane Ross, Joelle Pineau, Brahim Chaib-draa, and Pierre Kreitmann. A Bayesian approach for learning and planning in partially observable Markov decision processes. *The Journal of Machine Learning Research*, 12:1729–1770, 2011.

Constantin A Rothkopf and Christos Dimitrakakis. Preference elicitation and inverse reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 34–48. Springer, 2011.

Andrea Rotnitzky and James M. Robins. Semiparametric regression estimation in the presence of dependent censoring. *Biometrika*, 82(4):805–820, 1995.

Stuart Russell, Daniel Dewey, and Max Tegmark. Research priorities for robust and beneficial artificial intelligence. *AI Magazine*, 36(4):105–114, 2015.

Clayton D Scott. *Dyadic decision trees*. PhD thesis, University of Wisconsin at Madison, 2004.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. *Advances in Neural Information Processing Systems*, 23:2164–2172, 2010.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Satinder Singh and Richard Yee. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16(3):227–233, 1994.

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research*, 16:105–133, 2002.

Satinder P Singh and Richard S Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158, 1996.

Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 520–527. AUAI Press, 2004.

Alexander L Strehl and Michael L Littman. A theoretical analysis of model-based interval estimation. In *Proceedings of the 22nd International Conference on Machine learning*, pages 856–863. ACM, 2005.

Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. PAC model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888. ACM, 2006.

Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite MDPs: PAC analysis. *The Journal of Machine Learning Research*, 10:2413–2444, 2009.

Malcolm JA Strens. A Bayesian Framework for Reinforcement Learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 943–950, 2000.

Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in Neural Information Processing Systems*, pages 1038–1044, 1996.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, March 1998. ISBN 0-262-19398-1.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 99, pages 1057–1063, 1999.

Richard S Sutton, Ashique Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. CoRR abs/1503.04269, 2015.

Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems*, pages 1449–1456, 2007.

Erik Talvitie and Satinder Singh. Learning to make predictions in partially observable environments without a generative model. *Journal of Artificial Intelligence Research (JAIR)*, 42:353–392, 2011.

Ambuj Tewari and Peter L Bartlett. Sample complexity of policy search with known dynamics. In *Advances in Neural Information Processing Systems*, volume 19, pages 97–104, 2006.

Philip Thomas. *Safe Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2015.

Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High Confidence Off-Policy Evaluation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015a.

Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High Confidence Policy Improvement. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2380–2388, 2015b.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11): 1134–1142, 1984.

Harm Vanseijen and Rich Sutton. A deeper look at planning as learning from replay. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2314–2322, 2015.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems*, pages 831–838, 1992.

Vladimir Vapnik. *Statistical learning theory*, volume 2. Wiley New York, 1998.

Vladimir N Vapnik. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, 10(5):988–999, 1999.

John Von Neumann and Oskar Morgenstern. *Theory of games and economic behavior (60th Anniversary Commemorative Edition)*. Princeton university press, 2007.

Thomas J Walsh, Kaushik Subramanian, Michael L Littman, and Carlos Diuk. Generalizing apprenticeship learning across hypothesis classes. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1119–1126, 2010.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1995–2003, 2016.

Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, University of Cambridge England, 1989.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Xiaojin Zhu, Bryan R Gibson, and Timothy T Rogers. Human Rademacher complexity. In *Advances in Neural Information Processing Systems*, pages 2322–2330, 2009.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008.