Spring 2022

Efficient Data-Driven Robust Policies for Reinforcement Learning

Bahram Behzadian
University of New Hampshire, Durham

Efficient Data-Driven Robust Policies

for Reinforcement Learning

BY

Bahram Behzadian

M.Sc. in Computer Science, University of New Hampshire
NH, USA 2019

DISSERTATION

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Doctor of Philosophy
in
Computer Science

May, 2022

This dissertation has been examined and approved in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science by:

Marek Petrik, *Doctoral Advisor*
Assistant Professor of Computer Science,
University of New Hampshire

Mohammad Ghavamzadeh
Senior Staff Research Scientist,
Google Research

Wheeler Ruml
Professor of Computer Science,
University of New Hampshire

Clint Chin Pang Ho
Assistant Professor of Data Science,
City University of Hong Kong

Laura Dietz
Assistant Professor of Computer Science,
University of New Hampshire

On May 12, 2022

Approval signatures are on file with the University of New Hampshire Graduate School.

To my family and friends

ACKNOWLEDGEMENTS

I am thankful to my advisor, Marek Petrik, for coaching and supporting me throughout this work. His wisdom and deep understanding of mathematics have driven me on this voyage. His advice was essential in shaping and refining many of the ideas described in this thesis.

The members of my dissertation committee played a critical role in guiding this dissertation's topic. I want to thank Clint Chin Pang Ho for our discussions that led me to deepen my understanding of Robust MDPs. Also, I am grateful to have Mohammad Ghavamzadeh on my committee. He carefully reviewed my work and asked critical questions. I appreciate the detailed comments and encouragement that Wheeler Ruml and Laura Dietz provided on my research and thesis drafts.

This work was also supported by generous funding from National Science Foundation. Conversations with my lab-mate, Reazul Hasan Russel, made the long hours in the lab much more pleasant. Finally, I want to thank my family. They were supportive throughout the long years of my education.

Contents

List of Tables

List of Figures

ABSTRACT

Efficient Data-Driven Robust Policies
for Reinforcement Learning

by

Bahram Behzadian
University of New Hampshire, May, 2022

Applying the reinforcement learning methodology to domains that involve risky decisions like medicine or robotics requires high confidence in the performance of a policy before its deployment. Markov Decision Processes (MDPs) have served as a well-established model in reinforcement learning (RL). An MDP model assumes that the exact transitional probabilities and rewards are available. However, in most cases, these parameters are unknown and are typically estimated from data, which are inherently prone to errors. Consequently, due to such statistical errors, the resulting computed policy's actual performance is often different from the designer's expectation. In this context, practitioners can either be negligent and ignore parameter uncertainty during decision-making or be pessimistic by planning to be protected against the worst-case scenario. This dissertation focuses on a moderate mindset that strikes a balance between the two contradicting points of view. This objective is also known as the percentile criterion and can be modeled as risk-aversion to epistemic uncertainty. We propose several RL algorithms that efficiently compute reliable policies with limited data that notably improve the policies' performance and alleviate the computational

complexity compared to standard risk-averse RL algorithms. Furthermore, we present a fast and robust feature selection method for linear value function approximation, a standard approach to solving reinforcement learning problems with large state spaces. Our experiments show that our technique is faster and more stable than alternative methods.

CHAPTER 1

Introduction

Reinforcement Learning (RL) involves an automated planning problem under uncertainty. RL's goal is to design AI that can plan for environments where there may be incomplete or incorrect information. In such environments, the actions or decisions may not always have the same results, and there may be trade-offs between possible outcomes. RL is applied in industrial applications, primarily manufacturing, inventory management, power systems, finance, and invasive species management. This work focuses on a specific type of RL in which the solution policies are risk-averse and robust concerning uncertain problem parameters. The goal is to obtain policies that mitigate risk-averse and robust algorithms' conservative performance with a guaranteed expected return.

An agent in RL gathers information about its environment and then learns from the collected data by repeatedly replaying its experiences. In the beginning, the agent knows nothing but the rules, similar to a simulation of a chess game. The agent interacts with its environment in discrete time steps. The agent chooses an available action in each step and sends it to the environment. The environment moves to a new state, and the agent receives a reward associated with its action. The objective is to collect as much reward as possible. The initial strategy is to collect rewards through trial and error. Afterward, the agent investigates the gathered information to recognize particular features or policies to act more intelligently.

Conventional RL methods focus on maximizing some notion of cumulative reward [75]. Such approaches are considered risk-neutral decision-making. However, some decision-makers

are willing to give up some rewards to protect against significant losses or catastrophic outcomes. Such unwanted outcomes are mostly correlated with errors in estimating parameters involved in the decision-making process. Moreover, the use of simulations as environmental models in RL is widespread. However, the simulation and the actual environment difference can lead to unpredictable, often unwanted results. The agent strategy is sensitive to the parameters that describe the environment. In many practical problems, the estimation of these parameters is far from accurate. Hence, estimation errors are limiting factors in applying RL to real-world problems.

Risk-neutral methods are often too risky in mission-critical problems [56, 65, 81]. Reinforcement learning is applied to investigate medical treatment decisions, such as HIV, diabetes, and liver transplants [1, 71]. However, RL solutions are prone to risky treatment due to the unpredictable nature of point estimation techniques used to estimate the model parameters. A vital component of every RL model is how to describe stochastic changes in the system over time. The underlying model's parameters profoundly influence the decisions. One strong assumption in standard approaches is that the model is known with certainty. However, models are generated from population-based observational data for medical treatment decisions. Due to patient diversity, these observations cannot account for the natural variation in the estimated parameters, such as the probability of allergic reaction. Consequently, it is crucial to promote optimization models, such as risk-averse RL, that can also consider this variation.

For example, medical treatment decisions have to consider the pros and cons of each available medicine. The uncertainties in the medication's outcomes and numerous treatment options challenge the decision-making problem. Physicians often make these decisions according to the results of unpredictable trials and observations. Nevertheless, it is unclear how to control medications' effects and avoid the threat of disease-related problems such as death. Figure 1.1 shows a simple illustration of decision-making under uncertainty for one single step. The decision-maker has two options for treatment. The first option will

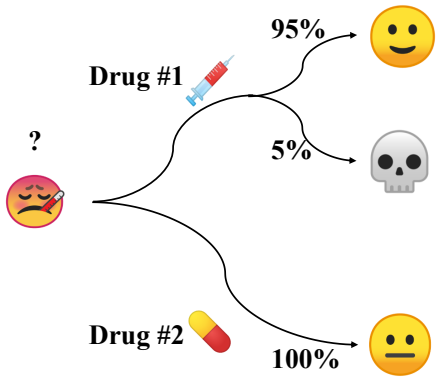

Figure 1.1: A simple illustration of decision-making under uncertainty.

significantly improve the patient's health level with a probability of 95%; however, there is a 5% chance that the patient will die due to an allergic reaction. The second option will not significantly improve, but it will guarantee that death will not happen. A risk-neutral agent chooses option one since, on average, the patients will be at a higher health level than the second option.

Markov decision processes (MDPs) are a helpful framework for modeling sequential decision-making under uncertainty within dynamic environments [67]. An MDP framework models the environment with specific parameters such as transition probabilities and rewards. We will discuss these parameters in detail in the following chapter. The environment models typically are estimated from data or learned from experience, which produces some estimation error. There is a spectrum of solutions to an MDP, from negligent to pessimistic. Negligent methods ignore parameter uncertainty during decision-making; conversely, pessimistic algorithms compute policies protected against worst-case scenarios.

Most of the progress in solving MDPs with parameter uncertainty is focused on studying robust MDPs [34, 57]. We assume that the model's parameters are unknown in a robust MDP setting, but they exist in a predefined region–so-called ambiguity sets. Ambiguity sets represent plausible errors in the model's parameters. For example, the ambiguity set for

the probability of allergic reaction in figure 1.1 can be defined as a range from 2% to 10%. In order to choose a robust policy, we must consider the worst-case scenario. Given the ambiguity set, a robust solution is computed under the worst-case outcome of a decision. However, unfortunately, the robust policies obtain in this way are often overly conservative and too pessimistic. In the last decades, there has been a rising interest in efficient risk-averse (a.k.a, risk-sensitive) decision-making systems. The biggest challenge in such approaches is making models that can identify *risk* accurately.

1.1 Contributions

This dissertation focuses on efficiently computing data-driven policies for reinforcement learning that maximize guaranteed returns and take parametric uncertainties into account in decision-making. Practitioners can either be negligent and ignore parameter uncertainty during decision-making or be pessimistic by planning to be protected from a worst-case scenario. In this work, the idea is to find a moderate mindset that balances the two contradicting points of view. This objective is also known as the percentile criterion [12] and can be modeled as risk-aversion to epistemic uncertainty. Optimizing the percentile criterion is a highly intractable problem. However, the contributions in this dissertation help optimize this criterion approximately by adopting the Robust MDPs (RMDPs) framework. RMDPs mitigate MDPs' sensitivity to estimation errors by computing an optimal policy for the worst plausible realization of the transition probabilities. This set of possible transition probabilities is known as the *ambiguity set*. The ambiguity set determines the quality and robustness of an RMDP solution while considering an underlying rectangularity assumption. The critical question is how to construct the ambiguity sets from state transition samples to optimize the percentile criterion.

1.1.1 Optimizing Percentile Criterion using Robust MDPs

Existing techniques construct ambiguity sets as confidence regions by applying concentration inequalities resulting in overly conservative solutions. We proposed a new approach for optimizing the percentile criterion using RMDPs beyond conventional ambiguity sets. First, we determined error bounds on the performance loss of the RMDP policy concerning the optimal percentile solution. These bounds show that the RMDP solution's sub-optimality depends on the absolute size of the ambiguity set and, most notably, its span along a specific direction. Then, we considered asymmetric ambiguity sets defined in weighted L_1 and L_∞ balls and proposed a linear-time algorithm that minimizes their size and span simultaneously. In addition, we derived new sampling guarantees to facilitate the algorithm for both Bayesian and frequentist settings. Experimental results indicate that the suggested optimized ambiguity sets improve significantly compared to prior construction methods. The work has been published in the *24th International Conference on Artificial Intelligence and Statistics* [6].

1.1.2 Fast Algorithms for L_∞-Constrained S-Rectangular Robust MDPs

In this work, we addressed the problem of the high computational complexity of computing robust policies. RMDPs with S-rectangular ambiguity sets can be solved in polynomial time. However, calculating the worst-case realization of transition probabilities often requires solving a linear program (LP) or another convex optimization problem. Modern solvers are efficient, but as the problem size grows, solving an LP for every state becomes computationally prohibitive. Although recent results show that RMDPs with L_1 sets can be solved efficiently, RMDPs with S-rectangular ambiguity sets defined in the L_∞ ball can currently be computed only by using general-purpose LP solvers, which are tedious and slow. We proposed a fast, exact algorithm for solving RMDPs with L_∞-constrained ambiguity sets. This approach combines a novel homotopy continuation method with a bisection method to solve RMDPs in quasi-linear time, which compares favorably with the cubic time complex-

ity of general interior-point LP algorithms. The experimental results confirm the practical viability of this approach and show that it outperforms a leading commercial optimization package by several orders of magnitude. This work has recently been published in *the 35th Conference on Neural Information Processing Systems*. [5].

1.1.3 Fast Feature Selection for Linear Value Function Approximation

Linear value function approximation is one of the standard approaches to solving reinforcement learning problems with large state spaces. However, since designing good approximation features is difficult, automatic feature selection is still an important and ongoing research topic. Aligned with previous contributions to compute robust solutions for RL agents, we proposed a new feature selection method based on a low-rank factorization of the transition matrix. This approach derives features directly from high-dimensional raw inputs, such as image data. The technique is easy to implement using SVD, and the experiments show that it is faster and more stable than alternative methods. This work appeared in the *29th International Conference on Automated Planning and Scheduling* [4].

Note that we focus on offline (batch) reinforcement learning [41] in this dissertation. In batch RL, all domain samples are provided in advance as a batch, and it is impossible or difficult to gather additional samples. This is common in many practical domains. For example, it is usually too dangerous and expensive to run additional tests in medical applications. Another example is that it may take an entire growing season to obtain a new batch of samples for ecological applications.

1.2 Outline

The dissertation is arranged as follows: Chapter 2 provides the mathematical foundation of robust MDPs and percentile criterion that is required to understand the work presented in the following chapters. Chapter 3 represents the detailed theories of weighted norm-bounded ambiguity sets in the Bayesian setting. Chapter 4 extends the theories and concepts that

are developed in Chapter 3 into the frequentist setting. In Chapter 5, we present RMDPs with S-rectangular ambiguity sets, which can be solved in polynomial time. In Chapter 6, we present a fast feature selection algorithm for reinforcement learning, which can effectively reduce the number of features in batch RL. We conclude this dissertation in Chapter 7 and provide the detailed technical results and formal proofs in Appendix A.

CHAPTER 2

Background and Formulations

Standard Markov Decision Processes (MDPs) are suitable models for sequential decision-making in which the decision's outcomes are uncertain. An MDP model contains decision time-stamps, states, actions, rewards, and transition probabilities. Any action at each state results in a reward and determines the state at the next time-stamp with respect to the transition probability function. Policies are instructions for which action to choose under any circumstances. A rational decision-maker seeks policies that are optimal in some predefined measures [8, 67]. Although the standard MDP frameworks consider uncertainty in every decision outcome using the transition probability function, we might treat uncertainty at a higher level. Generally, the transition probabilities need to be estimated from data. Such estimations are prone to errors and could considerably influence the optimal policy [51]. In this work, we are interested in Robust MDPs (RMDP) and risk-averse MDPs that are a conservative extension of the general MDPs. The following sections describe the general MDP components and expand the formulation to the RMDP and Chance-constrained MDPs.

***Notation*:** We reserve lower case bold characters for vectors and upper case characters for matrices. For example, bold letters, like $\boldsymbol{x}_s$, indicate an s-th vector, while y_s would indicate the s-th element of a vector $\boldsymbol{y}$. The symbol Δ^x denotes the probability simplex in $\mathbb{R}^x_+$ (non-negative vectors that sum to 1). We also use $\mathcal{A}^{\mathcal{B}}$ to denote the set of all functions $\mathcal{A} \to \mathcal{B}$. Finally, we use $\mathbf{I}$, $\mathbf{1}$, $\mathbf{0}$ to denote an identity matrix, a vector of ones, and a vector of zeros, respectively.

2.1 Markov Decision Process

We consider the standard infinite-horizon MDP setting with finite states $\mathcal{S} = \{1, \ldots, S\}$ and actions $\mathcal{A} = \{1, \ldots, A\}$. The agent can take any action $a \in \mathcal{A}$ in every state $s \in \mathcal{S}$ and transitions to the next state s' according to the *true* transition function $P^\star : \mathcal{S} \times \mathcal{A} \to \Delta^S$, where Δ^S is a probability simplex. For any transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta^S$, we use the shorthand $\boldsymbol{p}_{s,a} = P(s,a)$ to denote the vector of transition probabilities from a state $s \in \mathcal{S}$ and an action $a \in \mathcal{A}$. The agent also receives a reward $r_{s,a,s'} \in \mathbb{R}$; we use $\boldsymbol{r}_{s,a} = (r_{s,a,s'})_{s' \in \mathcal{S}} \in \mathbb{R}^S$ to denote the vector of rewards. The goal is to compute a deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the γ-discounted return [67]:

$$\max_{\pi \in \Pi} \ \rho(\pi, P) = \max_{\pi \in \Pi} \ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r_{S_t, \pi(S_t), S_{t+1}}\right] \ ,$$

where $S_0 \sim \boldsymbol{p}_0$, $S_{t+1} \sim P^\star(S_t, \pi(S_t))$, $\boldsymbol{p}_0 \in \Delta^S$ is the initial state probability distribution, and Π is the set of all deterministic policies. The return function ρ is parameterized by P, because we assume them to be uncertain or unknown.

We consider the batch RL setting in which the transition function must be estimated from a fixed dataset $D = (s_t, a_t, s'_t)_{t=1,\ldots,T}$ generated by a behavior policy. We describe the Bayesian setup first and outline the frequentist extension in Section 4.1. Bayesian techniques start with a prior distribution over the transition function $P^\star$ and then derive a posterior distribution f over $P^\star$ [12, 18, 83]. We use the concise notation $\tilde{P} = P^\star \,|\, D$ to represent the posterior over the transition function conditioned on the data D. In other words, $\mathbb{E}[\tilde{P}] = \mathbb{E}[P^\star \,|\, D]$.

2.2 Robust MDPs

This section surveys the basic properties of RMDPs; please see [31, 34, 81] for example for more details. We consider a finite RMDP model with states $\mathcal{S} = \{1, \ldots, S\}$ and actions

$\mathcal{A} = \{1, \ldots, A\}$. The agent takes an action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, it receives a reward $r_{s,a} \in \mathbb{R}$ and transitions to the next state $s' \in \mathcal{S}$ with a probability of $P_{s,a,s'}$. The transition probabilities P are unknown but are restricted to be in an ambiguity set $\mathcal{P} \subseteq (\Delta^S)^{S \times A}$. The initial state is distributed according to $\boldsymbol{p}_0 \in \Delta^S$.

We aim to compute a policy $\pi : \mathcal{S} \to \Delta^A$ from the set of stationary *randomized* policies Π that maximizes the expected γ-discounted return $\rho : \Pi \times \mathcal{P} \to \mathbb{R}$ for the worst-case transition probabilities:

$$\max_{\pi \in \Pi} \min_{P \in \mathcal{P}} \ \rho(\pi, P) \ . \tag{2.1}$$

Here, $\rho(\pi, P)$ is the standard discounted infinite-horizon return for a policy π defined as

$$\rho(\pi, P) \ = \ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r(S_t, A_t) \mid A_t \sim \pi(S_t), S_{t+1} \sim P_{S_t, A_t}, S_0 \sim \boldsymbol{p}_0\right].$$

The optimization problem in (2.1) can be seen as a zero-sum game, where adversarial nature chooses transition probabilities from the ambiguity set in order to minimize the agent's return. Since solving the general optimization problem in (2.1) is NP-hard (e.g., [81]), most research has focused on RMDPs with S-rectangular and SA-rectangular ambiguity sets, which can be solved in polynomial time [34, 44, 81].

SA-rectangular ambiguity sets $\mathcal{P}$ are defined as Cartesian products of sets $\mathcal{P}_{s,a} \subseteq \Delta^S$ for each state s and action a as $\mathcal{P} = \left\{P \in (\Delta^S)^{S \times A} \mid \boldsymbol{p}_{s,a} \in \mathcal{P}_{s,a},\ s \in \mathcal{S},\ a \in \mathcal{A}\right\}$. The intuitive interpretation of SA-rectangularity is that nature can choose the worst transition probabilities from sets $\mathcal{P}_{s,a}$ for each state s and action a *independently*. We focus on ambiguity sets bounded by L_∞-norm distance from nominal transition probabilities $\bar{\boldsymbol{p}}_{s,a} \in \Delta^S$ defined as

$$\mathcal{P}_{s,a} \ = \ \left\{\boldsymbol{p}_{s,a} \in \Delta^S \mid \left\|\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}_{s,a}\right\|_\infty \leq \kappa_{s,a}\right\} \ , \tag{2.2}$$

where $\kappa_{s,a} \geq 0$ is the robustness budget, and the nominal transition probability $\bar{\boldsymbol{p}}_{s,a}$ is typically estimated from samples of state transitions.

To streamline the definition of the robust Bellman operator, we follow the notation of [30] and define a *nature response function* $q : \mathbb{R}_+ \times \mathbb{R}^S \to \mathbb{R}$ that represents nature's response for a particular state s and action a as

$$q_{s,a}(\xi, \boldsymbol{v}) \;=\; \min_{\boldsymbol{p} \in \Delta^S} \; \left\{ r_{s,a} + \gamma \cdot \boldsymbol{p}^\mathsf{T} \boldsymbol{v} \mid \left\| \bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p} \right\|_\infty \leq \xi \right\} \; . \tag{2.3}$$

Then, the SA-rectangular robust Bellman operator $\mathfrak{L} : \mathbb{R}^S \to \mathbb{R}^S$ for a value function $\boldsymbol{v} \in \mathbb{R}^S$ is

$$(\mathfrak{L}\boldsymbol{v})_s \;=\; \max_{a \in \mathcal{A}} \min_{\xi \leq \kappa_{s,a}} q_{s,a}(\xi, \boldsymbol{v}) \; . \tag{2.4}$$

The optimal value function $\boldsymbol{v}^\star \in \mathbb{R}^S$ must satisfy the robust Bellman optimality equation $\boldsymbol{v}^\star = \mathfrak{L}\boldsymbol{v}^\star$ and can be computed either using value iteration, policy iteration, or other methods [24, 31, 34, 37].

S-rectangular ambiguity sets relax the assumptions of SA-rectangular sets and compute less conservative policies but with a higher computational complexity [81]. They are defined as Cartesian products of sets $\mathcal{P}_s \subseteq (\Delta^S)^A$ for each state s as:

$$\mathcal{P} \;=\; \left\{ P \in (\Delta^S)^{S \times A} \;\mid (\boldsymbol{p}_{s,a})_a \in \mathcal{P}_s, \, \forall s \in \mathcal{S} \right\} \; .$$

As with SA-rectangular sets, we also consider marginal ambiguity sets $\mathcal{P}_s$ defined in terms of the L_∞ norm as

$$\mathcal{P}_s \;=\; \left\{ (\boldsymbol{p}_{s,a})_{a \in \mathcal{A}} \in (\Delta^S)^A \mid \sum_{a \in \mathcal{A}} \left\| \bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}_{s,a} \right\|_\infty \leq \kappa_s \right\} \; ,$$

where $\kappa_s \geq 0$ is the robustness budget, and $\bar{\boldsymbol{p}}_{s,a}$ is the nominal transition probability. The important distinction from the SA-rectangular setting is that κ_s depends only on the state and not the action. The S-rectangular Bellman operator is then defined as:

$$(\mathfrak{L}\boldsymbol{v})_s \;=\; \max_{\boldsymbol{d} \in \Delta^A} \min_{\xi \leq \kappa_s} \sum_{a \in \mathcal{A}} d_a \cdot q_{s,a}(\xi, \boldsymbol{v}) \; . \tag{2.5}$$

Notice that the S-rectangular Bellman operator allows for randomizing actions through the probability distribution $\boldsymbol{d}$, improving robustness but introducing additional significant computational complexity [31, 81].

The majority of RMDP methods employ value iteration and policy iteration principles and require computing the robust Bellman operator many times during their run [31, 34, 81]. Therefore, it is crucial to compute it more efficiently than polynomial algorithms. In the following chapters, we develop new quasi-linear time algorithms for computing the robust Bellman operator.

2.3 Percentile Criterion

The Bayesian *percentile criterion* optimization simultaneously optimizes for the policy π and a *high-confidence lower bound* on its performance y:

$$\max_{\pi \in \Pi} \max_{y \in \mathbb{R}} \left\{ y \,|\, \mathbb{P}_{\tilde{P} \sim f} \left[\rho(\pi, \tilde{P}) \geq y \right] \geq 1 - \delta \right\} , \tag{2.6}$$

where f is the probability density function of the random variable $\tilde{P}$. The confidence parameter $\delta \in [0, 1/2)$ bounds the probability that the optimized policy π fails to achieve a return of at least y when deployed. For example, $\delta = 0$ maximizes the worst-case return, and $\delta = 0.5$ maximizes the median return. It is common in practice to choose a small positive value, such as $\delta = 0.05$, in order to achieve meaningful guarantees without being overly conservative. Also, the constraint $\delta < 1/2$ is important as our results (Theorem 3.1.3) do not hold for the risk-seeking setting with $\delta \geq 1/2$.

There are several important practical advantages to optimizing the percentile criterion instead of the average return [12]. First, the output policy is more robust and less likely to fail catastrophically due to model errors. Second, the objective value y in (2.6) provides a high-confidence lower bound on the true return. Having such a guarantee on its return helps avoid an unpleasant surprise when the policy π is deployed. When the confidence over

the lower bound y is insufficiently low, the stakeholder may decide to collect more data or choose a different methodology for guiding their decisions.

We emphasize that we develop algorithms independent of how the posterior distribution f is computed. Bayesian priors can be as simple as independent Dirichlet distributions over $\boldsymbol{p}^{\star}_{s,a}$ for each state s and action a. However, hierarchical Bayesian models are more practical since they generalize among states even when $|D| \ll S$ [12, 63]. Many tools, such as Stan [19] or JAGS, now exist to conveniently and efficiently compute the posterior distribution f using MCMC.

2.4 Linear Value Function Approximation

In this section, we summarize the background on linear value function approximation and feature construction. We consider a reinforcement learning problem formulated as a Markov decision process (MDP) with states $\mathcal{S}$, actions $\mathcal{A}$, transition probabilities $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, and rewards $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ [67]. The value of $P(s, a, s')$ denotes the probability of transitioning to state s' after taking an action a in a state s. The objective is to compute a stationary policy π that maximizes the expected γ-discounted infinite-horizon return. It is well-known that the value function $\boldsymbol{v}^{\pi}$ for a policy π must satisfy the Bellman optimality condition (e.g., Puterman [67]):

$$\boldsymbol{v}^{\pi} = \boldsymbol{r}^{\pi} + \gamma P^{\pi} \boldsymbol{v}^{\pi} \ , \tag{2.7}$$

where P^{π} and $\boldsymbol{r}^{\pi}$ are the matrix of transition probabilities and the vector of rewards, respectively, for the policy π.

Value function approximation becomes necessary in MDPs with large state spaces. The value function $\boldsymbol{v}^{\pi}$ can then be approximated by a linear combination of features $\phi_1, \ldots, \phi_k \in \mathbb{R}^{|\mathcal{S}|}$, which are vectors over states. Using the vector notation, an approximate value function $\tilde{\boldsymbol{v}}^{\pi}$ can be expressed as:

$$\tilde{\boldsymbol{v}}^{\pi} = \Phi \, \boldsymbol{w} \ ,$$

for some vector $\boldsymbol{w} = \{w_1, \ldots, w_k\}$ of scalar weights that quantify the importance of features. Here, Φ is the feature matrix of dimensions $|\mathcal{S}| \times k$; the columns of this matrix are the features ϕ_i.

Numerous algorithms for computing linear value approximation have been proposed [40, 75, 76]. We focus on fixed-point methods that compute the *unique* vector of weights $\boldsymbol{w}_\Phi^\pi$ that satisfy the projected Bellman equation (2.7):

$$\boldsymbol{w}_\Phi^\pi = \Phi^+(\boldsymbol{r}^\pi + \gamma P^\pi \Phi \boldsymbol{w}_\Phi^\pi) \ , \tag{2.8}$$

where Φ^+ is the Moore-Penrose pseudo-inverse of Φ and $\Phi^+ = (\Phi^\mathsf{T}\Phi)^{-1}\Phi^\mathsf{T}$ when columns of Φ are linearly independent (e.g., Golub and Van Loan [22]). This equation follows by applying the orthogonal projection operator $\Phi(\Phi^\mathsf{T}\Phi)^{-1}\Phi^\mathsf{T}$ to both sides of (2.7).

The following insight will be important when describing the FFS method. The fixed-point solution to (2.8) can be interpreted as a value function of an MDP with a *linearly compressed* transition matrix P_Φ^π and a reward vector $\boldsymbol{r}_\Phi^\pi$ [59, 76]:

$$\begin{aligned} P_\Phi^\pi &= (\Phi^\mathsf{T}\Phi)^{-1}\Phi^\mathsf{T} P^\pi \Phi = \Phi^+ P^\pi \Phi, \\ \boldsymbol{r}_\Phi^\pi &= (\Phi^\mathsf{T}\Phi)^{-1}\Phi^\mathsf{T}\boldsymbol{r}^\pi = \Phi^+ \boldsymbol{r}^\pi \ . \end{aligned} \tag{2.9}$$

The weights $\boldsymbol{w}_\Phi^\pi$ in (2.8) are equal to the value function for this compressed MDP. That is, $\boldsymbol{w}_\Phi^\pi$ satisfies the Bellman equation for the compressed MDP:

$$\boldsymbol{w}_\Phi^\pi = \boldsymbol{r}_\Phi^\pi + \gamma P_\Phi^\pi \boldsymbol{w}_\Phi^\pi \ . \tag{2.10}$$

To construct good features, it is essential to determine their quality in terms of whether they can express an excellent approximate value function.

CHAPTER 3

Optimizing Percentile Criterion Using Robust MDPs

In this chapter, we address the problem of computing reliable policies in reinforcement learning problems with limited data. In particular, we compute policies that achieve good returns with high confidence when deployed. This objective, known as the *percentile criterion*, can be optimized using Robust MDPs (RMDPs). RMDPs generalize MDPs to allow for uncertain transition probabilities chosen adversarially from given ambiguity sets. We show that the RMDP solution's sub-optimality depends on the spans of the ambiguity sets along with the value function. We then propose new algorithms that minimize the span of ambiguity sets defined by weighted L_1 and L_∞ norms. In this chapter, our focus is on Bayesian guarantees, however, in the next chapter, we describe how our methods apply to frequentist guarantees and derive new concentration inequalities for weighted L_1 and L_∞ norms.

Applying reinforcement learning to problem domains that involve high-stakes decisions, such as medicine or robotics, demands that we have high confidence in the quality of a policy before deploying it. Markov Decision Processes (MDPs) represent a well-established model in reinforcement learning [67, 75], but their sequential nature makes them particularly sensitive to parameter errors, which can quickly accumulate [52, 79, 83]. Parameter errors are unavoidable when estimating MDPs from data [42]. We focus on computing policies that maximize high-confidence return guarantees in the batch setting. Such guarantees reduce the chance of disappointing the stakeholders after deploying the policy and give them a choice to gather more data or switch to an alternative strategy [62].

We propose a new method for computing reliable policies that achieve, with high con-

fidence, good returns once deployed. This objective is also known as the *percentile criterion* [12] and can be modeled as risk-aversion to epistemic uncertainty [63]. Because optimizing the percentile criterion is NP-hard [12], we use Robust MDPs (RMDPs) [34] to optimize it approximately. We establish new error bounds on the performance loss of the RMDPs' policy compared to the optimal percentile solution. Using these new bounds when constructing the RMDPs leads to policies with significantly better return guarantees than reported in prior work [12, 63].

RMDPs generalize MDPs to allow for uncertain, or unknown, transition probabilities [34, 57, 81]. Transition probabilities are hard to estimate from data, and even small errors significantly impact the returns and policies. RMDPs consider transition probabilities to be chosen adversarially from a so-called *ambiguity set* (or an uncertainty set). The optimal policy is computed by solving a specific zero-sum game in which the agent chooses the best policy, and an adversarial nature chooses the worst transition probabilities from the ambiguity sets. RMDPs are tractable when their ambiguity sets satisfy so-called rectangularity assumptions [23, 50, 81].

Given the goal is to optimize the percentile criterion, the critical question is how to construct the ambiguity sets from state transition samples to optimize the percentile criterion. Prior work constructs ambiguity sets as confidence regions bounded by a distance from a nominal (expected) transition probability [2, 25, 34, 62, 63, 73]. In most cases, the ambiguity sets are represented as L_1-norm (also referred to as total variation) balls around the nominal probability. In comparison with other probability distance measures, like KL-divergence, the polyhedral nature of the L_1-norm allows more efficient computation [30].

The main contribution of this chapter and the following one is a new technique for optimizing the *shape* of ambiguity sets in RMDPs. Prior work simply constructs ambiguity sets with the smallest size, or volume, that are sufficient to provide the desired high-confidence guarantees. Our new bounds show that the *span* of the ambiguity set along a specific direction is much more important than its volume. To minimize their span, we consider

asymmetric ambiguity sets defined in terms of weighted L_1 and L_∞ balls.

In Sections 2.3 and 3.1.1, we provide the core concept of percentile criterion and its relationship with robust MDPs. The remainder of this chapter is organized as follows. First, we explain the general framework in Section 3.1. Section 3.2 describes algorithms that minimize the span of ambiguity sets by optimizing the weights of the norms used in their definition. Then, Section 3.3 describes methods for choosing the size of the weighted-norm ambiguity sets. Finally, the experimental results in Section 3.4 show that minimizing ambiguity sets' span greatly improves the RMDPs' solution quality.

3.1 RMDPs for Percentile Optimization

This section describes the general algorithm for constructing RMDP ambiguity sets for optimizing the percentile criterion. We derive new bounds on the safety and optimality of the RMDP solution and propose a new algorithm that optimizes them. The bounds and algorithms in this section are general and are not restricted to norm-based ambiguity sets.

3.1.1 Percentile Criterion Approximation Using Robust MDPs

Because the optimization in (2.6) is NP-hard [12], we seek new algorithms that can approximate it efficiently. Robust MDPs (RMDPs), which extend regular MDPs, are a convenient and powerful framework that can be used to optimize the percentile criterion. In particular, RMDPs allow for a generic ambiguity set $\hat{\mathcal{P}} \subseteq \left\{P : \mathcal{S} \times \mathcal{A} \to \Delta^S\right\}$ of possible transition functions instead of a single value P. The solution to an RMDP is the best policy for the worst-case plausible transition function:

$$\max_{\pi \in \Pi} \min_{P \in \hat{\mathcal{P}}} \rho(\pi, P) \ . \tag{3.1}$$

The optimization problem in (3.1) is NP-hard [57, 81] but is tractable for rectangular ambiguity sets which are defined independently for each state and action [34, 45]. We,

therefore, restrict our attention to SA-rectangular ambiguity sets defined as p-norm balls around nominal probability distributions for some $w : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^S_{++}$ and $\psi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$:

$$\mathcal{P}(w, \psi) = \{P \in \mathcal{F} \mid P(s,a) \in \mathcal{P}_{s,a}(w(s,a), \psi(s,a))\},$$

where $\mathcal{F} = (\Delta^S)^{\mathcal{S}\times\mathcal{A}}$. In the remainder of this section, we resort to the shorter notation $\boldsymbol{w}_{s,a} = w(s,a)$ and $\psi_{s,a} = \psi(s,a)$ when the meaning is obvious from the context. Note that $\hat{\mathcal{P}}$ refers to a generic ambiguity set, while $\mathcal{P}(w,\psi)$ refers to the specific norm-based one. The ambiguity set $\mathcal{P}_{s,a}(\boldsymbol{w}, \psi)$ for $s \in \mathcal{S}$, $a \in \mathcal{A}$, positive weights $\boldsymbol{w} \in \mathbb{R}^S_{++}$, and budget $\psi \in \mathbb{R}_+$ is defined as:

$$\mathcal{P}_{s,a}(\boldsymbol{w}, \psi) = \left\{\boldsymbol{p} \in \Delta^S \;:\; \left\|\boldsymbol{p} - \bar{\boldsymbol{p}}_{s,a}\right\|_{\boldsymbol{w}} \leq \psi\right\}, \tag{3.2}$$

where $\bar{\boldsymbol{p}}_{s,a} = \mathbb{E}_{\tilde{P}}\left[\tilde{P}(s,a)\right]$ is the mean posterior transition probability. The weighted polynomial norms are defined as $\|\boldsymbol{y}\|_{1,\boldsymbol{w}} = \sum_{i=1}^S w_i \cdot |y_i|$ and $\|\boldsymbol{y}\|_{\infty,\boldsymbol{w}} = \max\{w_i \cdot |y_i| \text{ß} i \in \mathcal{S}\}$. We use the generic notation $\|\cdot\|_{\boldsymbol{w}}$ in statements that hold for both $\|\cdot\|_{1,\boldsymbol{w}}$ and $\|\cdot\|_{\infty,\boldsymbol{w}}$. The weights $\boldsymbol{w}$ in (3.2) determine the shape of the ambiguity set, and the budget ψ determines its size.

Note that the parameter ψ in the definition of $\mathcal{P}_{s,a}(\boldsymbol{w}, \psi)$ is redundant. It can be set to 1 without loss of generality: $\mathcal{P}_{s,a}(\boldsymbol{w}, \psi) = \mathcal{P}_{s,a}(1/\psi \cdot \boldsymbol{w}, 1)$ when $\psi > 0$. In other words, it is possible to change the size of the ambiguity set solely by scaling the weights $\boldsymbol{w}$. To eliminate this redundancy, we assume without loss of generality that the weights of the set are normalized such that $\|\boldsymbol{w}\|_2 = 1$.

In rectangular RMDPs, a unique optimal value function $\hat{\boldsymbol{v}} \in \mathbb{R}^S$ exists and is a fixed point of the robust Bellman operator $\mathfrak{L} : \mathbb{R}^S \to \mathbb{R}^S$ defined for each $s \in \mathcal{S}$ and $\boldsymbol{v} \in \mathbb{R}^S$ as [34]

$$(\mathfrak{L}\boldsymbol{v})_s = \max_{a\in\mathcal{A}} \min_{\boldsymbol{p}\in\hat{\mathcal{P}}_{s,a}} \left(\boldsymbol{r}_{s,a} + \gamma \cdot \boldsymbol{p}^\mathsf{T}\boldsymbol{v}\right). \tag{3.3}$$

The optimal robust value function can be computed using value iteration, policy iteration, and other methods [31, 34, 37]. The optimal robust policy $\hat{\pi} : \mathcal{S} \to \mathcal{A}$ is greedy with respect

to the optimal robust value function $\hat{\boldsymbol{v}}$, and the robust return can be computed from the value function as [31]:

$$\hat{\rho} \;=\; \max_{\pi \in \Pi} \min_{P \in \hat{\mathcal{P}}} \; \rho(\pi, P) \;=\; \boldsymbol{p}_0^{\mathsf{T}} \hat{\boldsymbol{v}} \; .$$

We will find it convenient to use $\hat{\boldsymbol{z}}_{s,a} \in \mathbb{R}^S$, $s \in \mathcal{S}, a \in \mathcal{A}$ to denote the vector of values associated with the transitions from the state s and action a:

$$\hat{\boldsymbol{z}}_{s,a} \;=\; \boldsymbol{r}_{s,a} + \gamma \cdot \hat{\boldsymbol{v}} \; . \tag{3.4}$$

We use $\hat{\mathcal{P}}$ to denote a generic RMDP ambiguity set and use $\mathcal{P}(w, \psi)$ to denote an ambiguity set defined in terms of a weighted norm ball.

An important assumption, which is used throughout this chapter, is that the ambiguity set in the RMDP is constructed to guarantee that it contains the unknown transition probabilities $\tilde{P}$ with a high probability as formalized next.

Assumption 3.1.1. *The RMDP ambiguity set* $\hat{\mathcal{P}} \subseteq \left\{ P : \mathcal{S} \times \mathcal{A} \to \Delta^S \right\}$ *satisfies that:*

$$\mathbb{P}_{\tilde{P}} \big[\tilde{P} \in \hat{\mathcal{P}} \big] \;\geq\; 1 - \delta \, .$$

Assumption 3.1.1 is common when constructing RMDPs for optimizing the percentile criterion [12, 63]. The following theorem shows that Assumption 3.1.1 is a sufficient condition for $\hat{\rho}$ to be a lower bound on the true return of the robust policy $\hat{\pi}$. We state the result in terms of a generic ambiguity set $\hat{\mathcal{P}}$.

Theorem 3.1.2. *If Assumption 3.1.1 holds, then the following inequality is satisfied with probability* $1 - \delta$*:*

$$\hat{\rho} \;\leq\; \rho(\hat{\pi}, \tilde{P}) \; .$$

Please see Appendix A.1.1 for the proof. Theorem 3.1.2 generalizes Theorem 4.2 in [63] by relaxing its assumptions. In particular, Assumption 3.1.1 allows for non-rectangular

ambiguity sets $\hat{\mathcal{P}}$ and does not require the use of a union bound in its construction. We discuss this issue in greater depth in Section 3.3 when we describe algorithms for constructing ambiguity sets that satisfy Assumption 3.1.1.

Next, we bound the performance loss of the RMDP policy $\hat{\pi}$ with respect to the optimal percentile criterion guarantee in (2.6). As we show, the quality of the RMDP policy depends not simply on the absolute size of the ambiguity set ψ, but on its span along a specific direction. The *span* $\beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi)$ of an ambiguity set $\mathcal{P}_{s,a}(\boldsymbol{w}, \psi)$ along a vector $\boldsymbol{z} \in \mathbb{R}^S$ for $s \in \mathcal{S}$ and $a \in \mathcal{A}$ is defined as:

$$\beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi) = \max_{\boldsymbol{p}_1, \boldsymbol{p}_2} \Big\{ (\boldsymbol{p}_1 - \boldsymbol{p}_2)^\mathsf{T} \boldsymbol{z} \mid \boldsymbol{p}_1, \boldsymbol{p}_2 \in \mathcal{P}_{s,a}(\boldsymbol{w}, \psi) \Big\}.$$

The following theorem bounds the performance loss of the RMDP solution when using norm-bounded ambiguity sets. Note that Theorem 3.1.2 implies that, under Assumption 3.1.1, the RMDP return $\hat{\rho}$ bounds the true return with high confidence and therefore must be a lower bound on the optimal $y^\star$ in (2.6).

Theorem 3.1.3. *When Assumption 3.1.1 holds for $\hat{\mathcal{P}} = \mathcal{P}(w, \psi)$, $w : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^S_{++}, \psi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$, then the performance loss with respect to $y^\star$ optimal in* (2.6) *is:*

$$0 \;\leq\; y^\star - \hat{\rho} \;\leq\; \frac{1}{1-\gamma} \cdot \max_{s \in \mathcal{S}} \max_{a \in \mathcal{A}} \; \beta_{\hat{\boldsymbol{z}}_{s,a}}^{s,a}(\boldsymbol{w}, \psi) \;,$$

where $\hat{\rho}$ is a function of w and ψ.

The proof can be found in Appendix A.1.1. The following illustrates how the span along $\hat{\boldsymbol{z}}$ impacts the performance loss of the RMDP policy.

Example 3.1.4. *Consider an MDP with states $\{0, 1, 2, 3\}$ and a single action $\{1\}$. The state 0 is initial, and the states $1, 2, 3$ are terminal with $P(i, 1, i) = 1, i = 1, 2, 3$ with zero rewards. To keep the notation simple, we assume that it is only possible to transition from state 0 to states $1, 2, 3$. The transition* probability $\tilde{\boldsymbol{p}}_{0,1}$ *is uncertain and distributed as $\tilde{\boldsymbol{p}}_{0,1} \sim$*

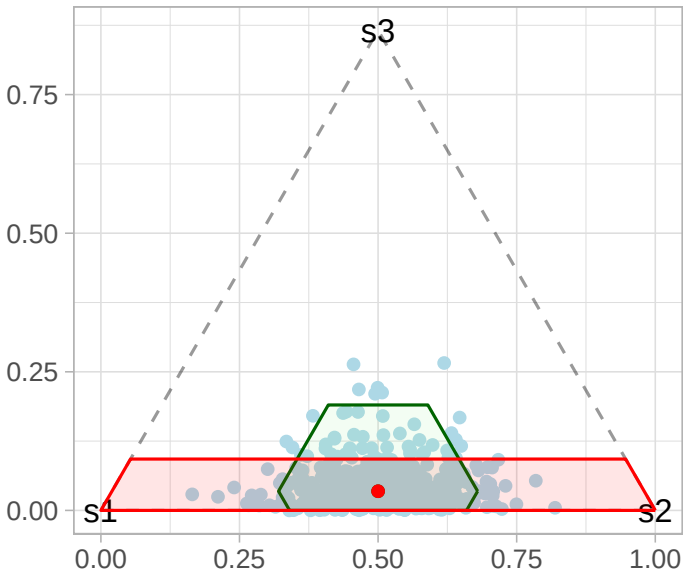

Figure 3.1: Posterior samples of $\tilde{\boldsymbol{p}}$ (blue) and ambiguity sets $\mathcal{P}^{\text{std}}$ (green) and $\mathcal{P}^{\text{opt}}$ (red) from Example 3.1.4.

Dirichlet$(10, 10, 1)$ *with* $\mathbb{E}[\tilde{\boldsymbol{p}}_{0,1}] = [0.48, 0.48, 0.04]$. *The rewards are* $\boldsymbol{r}_{0,1} = [0.25, 0.25, -1]$. *The goal is to maximize the percentile criterion with* $\delta = 0.2$.

Take the MDP from Example 3.1.4 and construct RMDPs with the following two ambiguity sets depicted in Figure 3.1. Let $\mathcal{P}^{\text{std}} = \mathcal{P}_{1,1}(1/\sqrt{3} \cdot \mathbf{1}, 0.1)$ be the standard ambiguity set with uniform weights, and let $\mathcal{P}^{\text{opt}} = \mathcal{P}_{1,1}(1/\sqrt{1.12} \cdot [0.25, 0.25, 1], 0.1)$ be an ambiguity set with optimized weights $\boldsymbol{w} = 1/\sqrt{1.12} \cdot [0.25, 0.25, 1]$. The budgets for both ambiguity sets are minimally sufficient to satisfy Assumption 3.1.1. Intuitively, this means that at least 80% of the posterior samples of $\tilde{\boldsymbol{p}}_{0,1}$ (blue dots in Figure 3.1) must be contained inside of each ambiguity set. Now, with 80% confidence, the RMDP with $\mathcal{P}^{\text{opt}}$ guarantees return $\hat{\rho}^{\text{opt}} = 0.16$, while the RMDP with $\mathcal{P}^{\text{std}}$ guarantees only $\hat{\rho}^{\text{std}} = -0.06$. Although the volumes of $\mathcal{P}^{\text{std}}$ and $\mathcal{P}^{\text{opt}}$ are approximately equal, the span along the dimension $\boldsymbol{z} = [0.25, 0.25, -1]$ of $\mathcal{P}^{\text{opt}}$ is half of the span of $\mathcal{P}^{\text{std}}$.

Armed with the safety and performance loss guarantees in Theorems 3.1.2 and 3.1.3, we propose a new heuristic algorithm in Algorithm 1 which iteratively optimizes the shape of

Algorithm 1: Ambiguity shape optimization scheme.

Input: Confidence $1-\delta$, posterior distribution f over $\tilde{P}$
Output: Ambiguity set $\mathcal{P}(\boldsymbol{w}, \psi)$

1 Compute $\boldsymbol{v}' \in \mathbb{R}^S$ by solving $\max_\pi \ \rho\big(\pi, \mathbb{E}\big[\tilde{P}\big]\big)$ and let $\boldsymbol{z}'_{s,a} \leftarrow \boldsymbol{r}_{s,a} + \gamma \cdot \boldsymbol{v}'$, $s \in \mathcal{S}, a \in \mathcal{A}$;
2 Compute minimal $\psi' : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ such that Assumption 3.1.1 holds for $\mathcal{P}(1/\sqrt{S} \cdot \mathbf{1}, \psi')$; `// Algorithm 3`
3 Compute $\boldsymbol{w}_{s,a} \leftarrow \min_{\boldsymbol{w} \in \mathbb{R}^S_+} \{\beta^{s,a}_{\boldsymbol{z}'}(\boldsymbol{w}, \psi') \mid \|\boldsymbol{w}\|_2 = 1\}$ for each $s \in \mathcal{S}$, $a \in \mathcal{A}$; `// Algorithm 2`
4 Compute minimal $\psi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ such that Assumption 3.1.1 holds for $\mathcal{P}(\boldsymbol{w}, \psi)$; `// Algorithm 3`
5 **return** *Ambiguity set* $\mathcal{P}(\boldsymbol{w}, \psi)$

the ambiguity set in order to improve the guaranteed percentile. It constructs ambiguity sets that minimize the span of the ambiguity set. The algorithm may not construct the optimal ambiguity set because it first uses the nominal value function $\boldsymbol{v}'$. However, the algorithm provides guarantees on the quality of the policy that it computes from Assumption 3.1.1 and Theorems 3.1.2 and 3.1.3.

3.2 Minimizing Ambiguity Spans

This section describes tractable algorithms that optimize the weights $\boldsymbol{w}$ to minimize that span $\beta^{s,a}_{\boldsymbol{z}}$ for some fixed state $s \in \mathcal{S}$, action $a \in \mathcal{A}$, a vector $\boldsymbol{z} \in \mathbb{R}^S$, and a budget $\psi \in \mathbb{R}_+$. We describe an analytical solution and a conic formulation that minimize an upper bound on the span for weighted L_1 and L_∞ sets. The budget ψ is fixed throughout this section; Section 3.3 describes how to optimize it.

The goal of computing the weights $\boldsymbol{w}$ that minimize the span of the ambiguity set for a fixed budget ψ can be formalized as the following optimization problem:

$$\min_{\boldsymbol{w} \in \mathbb{R}^S_+} \Big\{ \beta^{s,a}_{\boldsymbol{z}}(\boldsymbol{w}, \psi) \mid \|\boldsymbol{w}\|_2 = 1 \Big\}. \tag{3.5}$$

The optimization in (3.5) is not obviously convex, but we propose methods that minimize

an *upper* bound on $\beta^{s,a}_{\boldsymbol{z}}(\boldsymbol{w}, \psi)$. Note that minimizing this upper bound also minimizes an upper bound on Theorem 3.1.3.

We first describe two analytical solutions and then describe a more precise but also a more computationally intensive method based on second-order conic approximation. The following lemma provides a bound that enables efficient optimization.

Lemma 3.2.1. *The span $\beta^{s,a}_{\boldsymbol{z}}$ of the ambiguity set $\mathcal{P}_{s,a}(\boldsymbol{w}, \psi)$ is bounded for any $\lambda \in \mathbb{R}$ as:*

$$\beta^{s,a}_{\boldsymbol{z}}(\boldsymbol{w}, \psi) \leq 2 \cdot \psi \cdot \|\boldsymbol{z} - \lambda \cdot \mathbf{1}\|_\star \ , \tag{3.6}$$

where $\|\cdot\|_\star$ is the norm dual to $\|\cdot\|_{\boldsymbol{w}}$.

Recall that the *dual norm* is defined as $\|\boldsymbol{c}\|_\star = \max_{\boldsymbol{x} \in \mathbb{R}^S} \left\{ \boldsymbol{c}^\mathsf{T} \boldsymbol{x} \mid \|\boldsymbol{x}\| \leq 1 \right\}$. In order to use the bound in Lemma 3.2.1, we need to derive the dual norms to the weighted L_1 and weighted L_∞ norms. For unweighted p-norms, it is well known that L_1 and L_∞ norms are dual of each other, but we are not aware of a similar result for their weighted variants. The following lemma establishes that weighted L_1 and L_∞ norms are dual as long as their weights are inverse elementwise.

Lemma 3.2.2. *Suppose that $\boldsymbol{w} \in \mathbb{R}^S$ and $\boldsymbol{w}' \in \mathbb{R}^S$ are positive $w_i > 0, w'_i > 0$ and satisfy that $w'_i = 1/w_i$ for all $i \in \mathcal{S}$. Then:*

$$\|\boldsymbol{z}\|_{\infty, \boldsymbol{w}'} \ = \ \max_{\boldsymbol{x} \in \mathbb{R}^S} \left\{ \boldsymbol{z}^\mathsf{T} \boldsymbol{x} \mid \|\boldsymbol{x}\|_{1, \boldsymbol{w}} = 1 \right\}.$$

Based on the results above, Algorithm 2 summarizes our algorithms for computing weights $\boldsymbol{w}$ that minimize the upper bound on the performance loss in Theorem 3.1.3. The algorithm runs in linear time. Note that the algorithm assumes that a value of λ is given. Although it would be possible to optimize for the best value of λ, our preliminary experimental results suggest that this is not worthwhile because it does not lead to a significant improvement. Instead, we use $\lambda = (\max_i z_i + \min_i z_i)/2$ and $\lambda = \text{median}(\boldsymbol{z})$ for L_∞ and L_1

norms respectively. These are the optimal values (values for which the upper bound is smallest) for the uniform weight version of (3.6). The following proposition states the correctness of this algorithm.

Proposition 3.2.3. *Fix an arbitrary $\lambda \in \mathbb{R}$ and let $\boldsymbol{w}^\star \in \mathbb{R}_+^S$ be the return from Algorithm 2. Then $\boldsymbol{w}^\star$ is an optimal solution to* (3.6) *weighted L_1 and L_∞ norms.*

Please see Appendix A.1.2 for the proof. It is important to recognize that even though Algorithm 2 effectively minimizes the value $\beta_{\boldsymbol{z}}^{s,a}$, it may, in the process, violate Assumption 3.1.1. This is because scaling weights may reduce the probability that $\tilde{P} \in \mathcal{P}$. We are not aware of a tractable algorithm that can optimize the weights $\boldsymbol{w}$ directly while enforcing the constraint of Assumption 3.1.1. Instead, the constraint $\|\boldsymbol{w}\|_2 = \psi$ serves as a proxy to prevent the ambiguity from shrinking. This is why it is necessary to re-optimize the budget ψ in Algorithm 1 after the weights are optimized.

Algorithm 2: Weight optimization.

Input: Norm $q \in \{1, \infty\}$, parameter $\lambda \in \mathbb{R}$
Output: Weights $\boldsymbol{w}^\star \in \mathbb{R}_+^S$ that minimize (3.6)

if $q = 1$ **then**
  $w_i^\star \leftarrow \frac{|z_i - \lambda|^{1/3}}{\sqrt{\sum_{j=1}^S |z_j - \lambda|^{2/3}}}$, $\forall i \in \mathcal{S}$;
else if $q = \infty$ **then**
  $w_i^\star \leftarrow \frac{|z_i - \lambda|}{\sqrt{\sum_{j=1}^S |z_j - \lambda|^2}}$, $\forall i \in \mathcal{S}$;
end
return $\boldsymbol{w}^\star$;

As an alternative to the analytical algorithms in Algorithm 2, we also examine a Second-Order Conic Program (SOCP) formulation. This formulation optimizes a tighter upper bound on $\beta_{\boldsymbol{z}}^{s,a}$ but is more computationally intensive. For any fixed state s and action a, the following SOCP minimizes the bound (3.6) on $\beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi)$ for the L_1 norm:

$$\begin{aligned} \min_{\boldsymbol{g}, c, \lambda} \quad & \psi \cdot c \\ \text{s.t.} \quad & \boldsymbol{g} \geq \max\{\boldsymbol{z} - \lambda \cdot \mathbf{1}, -\boldsymbol{z} + \lambda \cdot \mathbf{1}\} \\ & \boldsymbol{g}^\mathsf{T} \boldsymbol{g} \leq c^2, \quad \boldsymbol{g} \geq \mathbf{0} \, . \end{aligned} \tag{3.7}$$

The SOCP formulation follows from Lemma 3.2.2 and variable substitution $\boldsymbol{g} = \boldsymbol{w} \cdot c$.

Remark 3.2.4 (Unreachable states)**.** *We assume that the prior can specify some transitions as impossible, or unreachable: that is $P(s, a, s') = 0$. This information is used as an additional pre-processing step in optimizing the weights. In particular, if the transition from state s after taking action a to state s' is not possible, then we set $(\boldsymbol{w}_{s,a})_{s'} = \infty$. Or, in other words, each $\boldsymbol{p} \in \mathcal{P}_{s,a}(\boldsymbol{w}, \psi)$ satisfies $p_{s'} = 0$.*

Algorithm 3: Budget optimization.

Input: Posterior samples $P_1, \ldots, P_n$ from $\tilde{P}$, weights $\boldsymbol{w}_{s,a}$, norm $q \in \{1, \infty\}$
Output: Nominal $\bar{\boldsymbol{p}}_{s,a}$ and budget $\psi_{s,a}$

1 Compute nominal $\bar{\boldsymbol{p}}_{s,a} \leftarrow (1/n) \sum_{i=1}^{n} P_i(s, a)$;
2 Compute distance $d_i \leftarrow \left\| P_i(s, a) - \bar{\boldsymbol{p}}_{s,a} \right\|_{q, \boldsymbol{w}_{s,a}}$;
3 Ascending sort: $d_{(j)} \leq d_{(j+1)}$, $j = 1, \ldots, n$;
4 Compute the quantile $\psi_{s,a} \leftarrow d_{(\lceil (1-\delta/(S \cdot A)) \cdot n \rceil)}$;
5 **return** $\bar{\boldsymbol{p}}_{s,a}$ *and* $\psi_{s,a}$

3.3 Minimizing Ambiguity Budgets

This section describes how to determine the size of the ambiguity set in the Bayesian setting in order to minimize the performance loss in Theorem 3.1.3 of the RMDP policy while satisfying Assumption 3.1.1. We assume that the weights $\boldsymbol{w}_{s,a}, s \in \mathcal{S}, a \in \mathcal{A}$ are arbitrary but fixed and aim to construct $\psi_{s,a}, s \in \mathcal{S}, a \in \mathcal{A}$ to minimize the performance loss.

Before describing the algorithm, we state a simple observation that motivates its construction. The following lemma implies that the smaller the ambiguity budget is, the better $\hat{\rho}$ approximates the percentile criterion. Of course, this is only true as long as the budget is sufficiently large for Assumption 3.1.1 to hold. The following proposition follows from the definition of $\beta_{\boldsymbol{z}}^{s,a}$ by algebraic manipulation.

Lemma 3.3.1. *The function $\psi \mapsto \beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}_{s,a}, \psi)$ is non-decreasing.*

We are now ready to describe our method as outlined in Algorithm 3. The algorithm follows the well-known sample average approximation (SAA) approach common in stochastic programming [70]. It constructs ambiguity sets as *credible regions* for the posterior distribution over $\tilde{P}$ similarly to prior work [63]. The next proposition states the correctness of Algorithm 3.

Proposition 3.3.2. *Suppose that $\psi_{s,a}$ are computed by Algorithm 3 for some $\boldsymbol{w}_{s,a}$ for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$. Also let $w : (s,a) \mapsto \boldsymbol{w}_{s,a}$ and $\psi : (s,a) \mapsto \psi_{s,a}$. Then $\mathcal{P}(w,\psi)$ satisfies Assumption 3.1.1 with high probability when a sufficient number of samples from $\tilde{P}$ are used.*

Please see Appendix A.1.3 for the proof. Algorithm 3 constructs credible regions for each state and action separately [54]. A notable limitation of Algorithm 3 is that it constructs the credible regions independently for each state and action. Although this is convenient computationally, it also means that the confidence region needs to rely on the union bound which makes it impractical when the number of states and actions is large. Although, Assumption 3.1.1 allows for construction that avoids union-bound-based construction.

While Proposition 3.3.2 provides asymptotic convergence guarantees, it is possible to obtain finite-sample guarantees by using more careful analysis [46] or by adapting Algorithm 3 as suggested in [32]. We leave this finite-sample analysis for future work.

3.4 Empirical Evaluation

In this section, we evaluate Algorithm 1 empirically using five standard reinforcement domains that have been previously used to evaluate robustness.

Table 3.1 summarize the results for the Bayesian setups. The results compare our algorithms (rows) against baselines (rows) for fixed datasets D for all domains (column). The method names indicate how the weights are computed and which norm is used to define the ambiguity set. Methods denoted as "Uniform" represent $\boldsymbol{w} = \mathbf{1}$ and "Optimized" represent $\boldsymbol{w}$ computed using Algorithms 1 and 2.

	RS	MR	PG	IM	CP
Uniform L_1	0.60	1.56	5.24	0.97	0.77
Uniform L_∞	0.60	1.56	5.50	0.98	0.76
Optimized L_1	0.25	0.41	1.84	0.90	0.12
Optimized L_∞	0.31	0.39	3.10	0.87	0.19

Table 3.1: Normalized *Bayesian* performance loss $(\bar{\rho} - \hat{\rho})/|\bar{\rho}|$ for $\delta = 0.05$. (Smaller value is better).

As the main metric, we compare the computed return guarantees $\hat{\rho}$ (the return of the RMDP). Because all methods use ambiguity sets that satisfy Assumptions 3.1.1, $\hat{\rho}$ lower bounds $\rho(\hat{\pi}, \tilde{P})$ with probability $1 - \delta$. In order to enable the comparison of the results among different domains, we normalize the guarantee by the maximal nominal return $\bar{\rho} = \max_{\pi \in \Pi} \rho(\pi, \mathbb{E}[\tilde{P}])$. We use $\bar{\rho}$ instead of the unknown $y^\star$.

As a baseline, we compare our results with the standard RMDPs construction [12, 63], which uses uniformly-weighted L_1 and L_∞ norms. We do not compare to policy-gradient-style methods [12] because they cannot be used with general posterior distributions over $\tilde{P}$ in our domains. We note that various modifications to probability norms have been proposed in the RL context (e.g., [49, 77]), but it is unclear how to use them in the context of the percentile criterion.

The results in Table 3.1 show that optimizing the weights in RMDP ambiguity sets decreases the guaranteed performance loss dramatically in Bayesian settings (geometric mean 2.8×). The guarantees improve because the RMDPs with optimized sets simultaneously compute a better policy and a tighter bound on its return. Note that zero losses in the tables may be unachievable ($\bar{\rho} > y^\star$), and losses greater than one are possible (when $\bar{\rho} < 0$).

We now briefly summarize the domains used. *RiverSwim (RS)* is a simple and standard benchmark [74], which is an MDP consisting of six states and two actions (see Figure 3.2). The process follows by sampling synthetic datasets from the true model and then computing the guaranteed robust returns for different methods. The prior is a uniform Dirichlet distribution over reachable states.

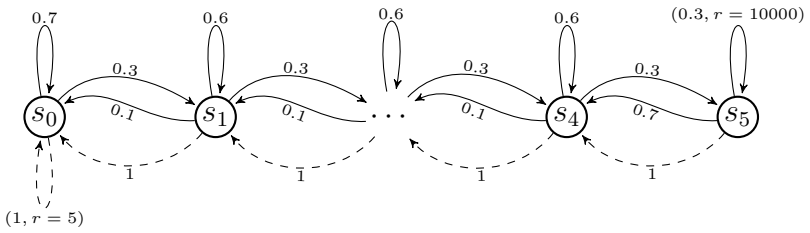

Figure 3.2: RiverSwim problem with six states and two actions (left-dashed arrow, right-solid arrow). The agent starts in either states s_1 or s_2.

Machine Replacement (MR) is a small benchmark MDP problem with $S = 10$ states that models progressive deterioration of a mechanical device [12]. Two repair actions $A = 2$ are available and restore the machine's state. Uses a Dirichlet prior.

Population Growth Model (PG) is an exponential population growth model [38], which constitutes a simple state-space $0, \ldots, S = 50$ with exponential dynamics. At each time step, the land manager has to decide whether to apply a control measure to reduce the species' growth rate. We refer to [79] for more details of the model.

Inventory Management (IM) is a classic inventory management problem [84], with discrete inventory levels $0, \ldots, S = 30$. The purchase cost, sale price, and holding cost are $2.49, 3.99$, and 0.03, respectively. The demand is sampled from a normal distribution with a mean $S/4$ and a standard deviation of $S/6$. It also uses a Dirichlet prior.

Cart-Pole (CP) is the standard RL benchmark problem [10, 75]. We collect samples of 100 episodes from the true dynamics. We fit a linear model with that dataset to generate synthetic samples and aggregate close states to a 200-cell grid ($S = 200$) using the k-nearest neighbor strategy and assume a uniform Dirichlet prior.

This chapter proposed a new approach for optimizing the percentile criterion using RMDPs in the Bayesian setup that goes beyond the conventional ambiguity sets. In the next chapter, we extend this method to the frequentist setting.

CHAPTER 4

Weighted Frequentist Confidence Intervals for Robust MDPs

In Chapter 3, we described new methods for optimizing the shape of ambiguity sets beyond the L_1-norm, in the form of credible intervals, in Bayesian statistics. In this chapter, our focus is on frequentist guarantees. We present two new finite-sample bounds that can be used to construct frequentist ambiguity sets with weighted L_p norms. These bounds are necessary to guarantee high-confidence return guarantees. These results significantly extend the existing bounds, limited to the L_1 deviation [3, 16, 63, 80]. In Section 4.1, we outline the approach in the frequentist setup and present new concentration inequalities for weighted L_1 and L_∞ ambiguity sets. The experimental results in Section 4.2 show significant improvement in the RMDPs' performance.

4.1 Frequentist Guarantees

This section extends the analysis of Bayesian ambiguity sets to outline how our results apply to frequentist guarantees. The advantage of the frequentist setup is that it guarantees even without needing access to a prior distribution. The disadvantage is that, without reasonable priors, frequentist settings may need an excessive amount of data to provide credible guarantees. The main contribution in this section is new sampling bounds for weighted L_1 and L_∞ ambiguity sets.

The frequentist perspective on the percentile criterion [12] represents a viable alternative to the Bayesian perspective when it is challenging to construct an excellent prior distribution. The frequentist view assumes that the true model $P^\star$ is known. The analysis considers the

uncertainty over datasets. To define the criterion, let $\mathcal{D}$ represent the set of all possible datasets D. Then the pair of algorithms $F : \mathcal{D} \to \Pi$, which computes the policy for a dataset, and $G : \mathcal{D} \to \mathbb{R}$, which estimates the return of the policy, solves the percentile criterion if:

$$\mathbb{P}_{D \sim P^\star} \left[\rho(F(D), P^\star) \geq G(D) \right] \;\geq\; 1 - \delta \;. \tag{4.1}$$

A frequentist modeler assumes that $P^\star_{s,a}$ is fixed and the probability statements are qualified over sampled data sets $(s_t, a_t, s'_t)_{t=1,\ldots,T}$ generated from the true transition probabilities $s'_t \sim \boldsymbol{p}^\star_{s_t,a_t}$.

We make very similar assumptions to the Bayesian setting to construct an RMDP that solves the frequentist percentile criterion. The following assumption restates Assumption 3.1.1 in the frequentist setting; note the change in random variables.

Assumption 4.1.1. *The data-dependent ambiguity set $\hat{\mathcal{P}}$ satisfies:*

$$\mathbb{P}_{D \sim P^\star} \left[P^\star \in \hat{\mathcal{P}} \right] \;\geq\; 1 - \delta \,,$$

where $\hat{\mathcal{P}}$ is a function of D.

Recall that Theorem 3.1.2 establishes that an RMDP that satisfies Assumption 3.1.1 computes a high-confidence lower bound on the return. The proof of Theorem 3.1.2 easily extends to the frequentist setup. Therefore, Assumption 4.1.1 implies that $\mathbb{P}_D \left[\hat{\rho} \leq \rho(\hat{\pi}, P) \right] \geq 1 - \delta$ where $\hat{\rho}$ and $\hat{\pi}$ are the return and policy to the RMDP. In other words, the RMDP algorithm (joint policy and return estimate computation) solves the frequentist percentile criterion in (4.1) when Assumption 4.1.1 holds.

Because the optimization methods described in Section 3.2 make no probabilistic assumptions, they can be applied to the frequentist setup with no change. The optimization of ψ described in Section 3.3 assumes that samples from the posterior over transition functions are available and cannot be readily used to satisfy Assumption 4.1.1. Instead, we present two new finite-sample bounds that can be used to construct frequentist ambiguity sets. Since

prior work has been limited to the ambiguity sets defined in terms L_1 ambiguity sets with uniform weights [3, 16, 63, 80], we derive new high-confidence bounds for ambiguity sets defined using *weighted* L_1 and L_∞ norms. To state our new results, let the nominal point $\bar{\boldsymbol{p}}_{s,a} \in \Delta^S$ in (3.2) be the empirical estimate of the transition probability computed from $n_{s,a} \in \mathbb{N}$ transition samples for each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$.

Theorem 4.1.2 (L_∞ norm)**.** *Suppose that $\mathcal{P}(\boldsymbol{w}, \psi)$ is defined in terms of the $\boldsymbol{w}_{s,a}$-weighted L_∞ norm. Then Assumption 4.1.1 is satisfied if $\psi_{s,a} \in \mathbb{R}_+$ for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$ satisfies the following inequality:*

$$\delta \;\leq\; 2 \cdot SA \cdot \sum_{i=1}^{S} \exp\left(-2\frac{\psi_{s,a}^2 \cdot n_{s,a}}{(\boldsymbol{w}_{sa})_i^2}\right) \;. \tag{4.2}$$

Theorem 4.1.3 (L_1 norm)**.** *Suppose that $\mathcal{P}(\boldsymbol{w}, \psi)$ is defined in terms of the $\boldsymbol{w}_{s,a}$-weighted L_1 norm. Then Assumption 4.1.1 is satisfied if $\psi_{s,a} \in \mathbb{R}_+$ for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$ satisfies the following inequality:*

$$\delta \;\leq\; 2 \cdot SA \cdot \sum_{i=1}^{S-1} 2^{S-i} \cdot \exp\left(-\frac{\psi_{s,a}^2 \cdot n_{s,a}}{2 \cdot (\boldsymbol{w}_{sa})_i^2}\right) \;, \tag{4.3}$$

where positive weights $\boldsymbol{w}_{s,a} \in \mathbb{R}^S_{++}, s \in \mathcal{S}, a \in \mathcal{A}$ are assumed to be sorted in a non-increasing order $(\boldsymbol{w}_{s,a})_i \geq (\boldsymbol{w}_{s,a})_{i+1}$ for $i = 1, \ldots, S-1$.

The proofs of the theorems are in Appendix A.2.1. They follow standard techniques combining the Hoeffding and union bounds.

A natural question is how to construct $\psi_{s,a}$ that satisfies Theorems 4.1.2 and 4.1.3. Although the theorems do not provide us with an analytical solution, the value of $\psi_{s,a}$ can be computed efficiently using the standard bisection method [9]. This is because right-hand side functions in (4.2) and (4.3) are monotonically decreasing in $\psi_{s,a}, s \in \mathcal{S}, a \in \mathcal{A}$. Theorem 4.1.4 further tightens the error bounds using Bernstein's inequality.

Theorem 4.1.4 (Weighted L_1 error bound (Bernstein's style))**.** *Suppose that $\bar{\boldsymbol{p}}_{s,a}$ is the empirical estimate of the transition probability obtained from $n_{s,a}$ samples for some $s \in \mathcal{S}$ and $a \in \mathcal{A}$. If the weights $\boldsymbol{w} \in \mathbb{R}^S_{++}$ are sorted in non-increasing order $w_i \geq w_{i+1}$, then the following holds when using Bernstein's inequality:*

$$\mathbb{P}\left[\left\|\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a}\right\|_{1,\boldsymbol{w}} \geq \psi_{s,a}\right] \leq 2\sum_{i=1}^{S-1} 2^{S-i} \exp\left(-\frac{3\psi^2 n}{6w_i^2 + 4\psi w_i}\right)$$

where $\boldsymbol{w} \in \mathbb{R}^S_{++}$ is the vector of weights. The weights are sorted in non-increasing order.

The proof is available in Appendix A.2.2. Theorems 4.1.2 and 4.1.3 also provide new insights into which ambiguity set may be a better fit for a particular problem. Simple algebraic manipulation and (3.6) show that the L_1 norm is preferable to the L_∞ norm when $\|\boldsymbol{v} - \bar{v} \cdot \mathbf{1}\|_1 > \sqrt{S} \cdot \|\boldsymbol{v} - \tilde{v} \cdot \mathbf{1}\|_\infty$. Here, $\boldsymbol{v} \in \mathbb{R}^S$ is the optimal value function, $\bar{v} = \mathbf{1}^\mathsf{T}\boldsymbol{v}/S$ is the mean value, and $\tilde{v}$ is the median value of $\boldsymbol{v}$.

In terms of their tightness, Theorems 4.1.2 and 4.1.3 are similar to the most well-known bounds on the uniformly-weighted norms. Theorem 4.1.3 recovers the equivalent best-known (Hoeffding-based) result for uniformly-weighted norm within a factor of 2. We are unaware of comparable prior results for ambiguity sets defined in terms of L_∞ norms. Unfortunately, frequentist bounds on probability distributions are generally useful only when the number of samples $n_{s,a}$ is quite large. We also investigated Bernstein-based versions of the bounds, but they show little difference in our experimental results.

Finally, it is important to note that Theorems 4.1.2 and 4.1.3 require that the weights $\boldsymbol{w}$ are independent of data. Therefore, the weights $\boldsymbol{w}$ should be optimized using a dataset different from the one used to estimate ψ. However, in our experiment, we found that reusing the same dataset to optimize both $\boldsymbol{w}$ and ψ empirically does not compromise the percentile guarantees.

	RS	MR	PG	IM	CP
Uniform L_1	0.80	5.83	5.66	1.05	0.78
Uniform L_∞	0.76	3.45	5.65	1.05	0.78
Optimized L_1	0.53	1.05	5.55	0.99	0.77
Optimized L_∞	0.43	0.94	5.56	0.96	0.69

Table 4.1: Normalized *frequentist* performance loss $(\bar{\rho} - \hat{\rho})/|\bar{\rho}|$ for $\delta = 0.05$. (Smaller value is better).

4.2 Detailed Experimental Results For Weighted Ambiguity Sets

Table 4.1 summarize the results for the frequentist setup. The results compare our algorithms (rows) against baselines (rows) for fixed datasets D for all domains (column). The method names indicate how the weights are computed and which norm is used to define the ambiguity set. Methods denoted as "Uniform" represent $\boldsymbol{w} = \mathbf{1}$ and "Optimized" represent $\boldsymbol{w}$ computed using Algorithms 1 and 2. The following section provides the complete report of the statistics and methods (including the SOCP formulation).

4.2.1 Experimental Setup

We assess L_1- and L_∞-bounded ambiguity sets, both with weights and without weights. We compare Bayesian credible regions with frequentist Hoeffding. We assume a true underlying model that produces simulated datasets containing 20 samples for each state and action. The frequentist methods construct ambiguity sets directly from the datasets. Bayesian methods combine the data with a prior to compute a posterior distribution and then draw 20 samples from the posterior distribution to construct a Bayesian ambiguity set.

4.2.2 Full Empirical Results

Tables 4.2 to 4.5 report the high-confidence lower bound on the return for the domains that we investigate. The column denotes the confidence $1 - \delta$ and the algorithm used to compute the weights $\boldsymbol{w}$ for the ambiguity set: "Unif.w" corresponds to $\boldsymbol{w} = \mathbf{1}$, "Analyt.w" corresponds

to weights computed by Algorithm 2, and "SOCP.w" corresponds to weights computed by solving (3.7). The rows indicate which norm was used to define the ambiguity set (L_1 or L_∞) and whether Bayesian (B) or frequentist (H) guarantees were used. Note that the SOCP formulation is limited to the L_1 ambiguity sets.

	$\delta = 0.5$			$\delta = 0.05$		
Method	Unif.w	Analyt.w	SOCP.w	Unif.w	Analyt.w	SOCP.w
L_1B	33887	**51470**	48620	25252	**47284**	43504
$L_\infty B$	33887	**48258**	-	25252	**43247**	-
L_1 H	16354	**33116**	30268	12555	**29472**	26398
L_∞ H	20055	**40166**	-	15184	**35955**	-

Table 4.2: The return with performance guarantees for the RiverSwim experiment. The return of the nominal MDP is 63080.

	$\delta = 0.5$			$\delta = 0.05$		
Method	Unif.w	Analyt.w	SOCP.w	Unif.w	Analyt.w	SOCP.w
L_1B	-38.1	**-22.7**	-26.8	-42.0	**-23.7**	-28.4
$L_\infty B$	-38.1	**-22.6**	-	-42.0	**-23.5**	-
L_1 H	-86.8	**-33.2**	-47.9	-115.0	**-34.5**	-53.1
L_∞ H	-62.9	**-29.5**	-	-74.8	**-32.6**	-

Table 4.3: The return with performance guarantees for the Machine Replacement experiment. The return of the nominal MDP is -16.79.

	$\delta = 0.5$			$\delta = 0.05$		
Method	Unif.w	Analyt.w	SOCP.w	Unif.w	Analyt.w	SOCP.w
L_1B	-25706	**-12151**	-12668	-25741	**-12200**	-12704
$L_\infty B$	-26782	**-15468**	-	-26795	**-15623**	-
L_1 H	-27499	**-27034**	-27409	-27501	**-27047**	-27421
L_∞ H	-27465	**-27143**	-	-27473	**-27184**	-

Table 4.4: The return with performance guarantees for the Population experiment. The return of the nominal MDP is -4127.

Method	$\delta = 0.5$			$\delta = 0.05$		
	Unif.w	Analyt.w	SOCP.w	Unif.w	Analyt.w	SOCP.w
$L_1 B$	3.75	**15.7**	10.9	3.64	**15.0**	10.6
$L_\infty B$	3.04	**20.2**	-	2.87	**19.8**	-
L_1 H	-8.91	**1.58**	-6.18	-8.94	**0.89**	-7.74
L_∞ H	-8.37	**5.83**	-	-8.63	**4.90**	-

Table 4.5: The return with performance guarantees for the Inventory Management experiment. The return of the nominal MDP is 163.1.

Method	$\delta = 0.5$			$\delta = 0.05$		
	Unif.w	Analyt.w	SOCP.w	Unif.w	Analyt.w	SOCP.w
$L_1 B$	3.83	**8.28**	4.21	3.82	**8.25**	4.20
$L_\infty B$	3.81	**7.78**	-	3.78	**7.71**	-
L_1 H	2.81	**3.44**	2.87	2.80	**3.42**	2.85
L_∞ H	3.18	**3.94**	-	3.15	**3.92**	-

Table 4.6: The return with performance guarantees for the Cart-Pole experiment. The return of the nominal MDP is 11.11.

In this chapter and the previous one, we proposed a new approach for optimizing the percentile criterion using RMDPs that goes beyond the conventional ambiguity sets. At the heart of our method are new bounds on the performance loss of the RMDPs with respect to the optimal percentile criterion. These bounds show that the quality of the RMDP is driven by the span of its ambiguity sets along a specific direction. We proposed a linear-time algorithm that minimizes the span of the ambiguity sets and also derived new sampling guarantees. Our experimental results show that this simple RMDP improvement can lead to much better return guarantees. Future work needs to focus on scaling the method to a large state-space using value function approximation or other techniques.

CHAPTER 5

Fast Algorithms for L_∞-constrained S-rectangular Robust MDPs

Markov decision processes (MDPs) are a powerful framework for dynamic decision-making problems and reinforcement learning [8, 67, 75]. The MDP model assumes that the exact transition probabilities and rewards are available. However, these transition probabilities are typically unknown and must be estimated from sampled data. Such estimations are error-prone, and the MDP's solution is sensitive to the introduced statistical errors. In particular, the quality of the optimal policy degrades significantly even with minor errors in the transition probabilities [44].

Robust MDPs (RMDPs) mitigate MDPs' sensitivity to estimation errors by computing an optimal policy for the worst plausible realization of the transition probabilities. This set of plausible transition probabilities is known as the *ambiguity set*. In this chapter, we study RMDPs with S-rectangular ambiguity sets, which can be solved in polynomial time [27]. However, computing the worst-case realization of transition probabilities often requires solving a linear program (LP) or another convex optimization problem. Modern solvers are powerful and efficient, but as the problem size grows, solving an LP for every state becomes computationally prohibitive [30].

Most prior work has focused on RMDPs with L_1-constrained ambiguity sets because convenient concentration inequalities [62, 69, 80] and fast algorithms [31, 34, 64] exist for this scenario. The concentration inequalities play an essential role in the data-driven construction of high-confidence RMDPs. However, ambiguity sets defined by the L_∞ norm are more natural and interpretable by human modelers[14, 21], and can significantly outperform L_1-

based ambiguity sets in many circumstances [6]. Unfortunately, RMDPs with S-rectangular ambiguity sets defined in terms of the L_∞ ball can currently be solved only using general-purpose LP solvers, which are complex and slow.

As our main contribution, we propose a new, fast algorithm for solving RMDPs with L_∞-constrained ambiguity sets. Our algorithm combines a new homotopy continuation method with a bisection method to achieve quasi-linear $\mathcal{O}(SA \log S)$ time complexity concerning the number of states S and actions A. This computational complexity compares favorably with the cubic $\mathcal{O}((SA)^{3.5})$ time complexity of general interior-point LP algorithms. To develop our algorithms, we identify new simplifying properties of the robust optimization problem defined over L_∞ balls.

Although bisection and homotopy methods have been used previously in robust MDPs, their use and assumptions differ significantly from this work. A bisection method was used to solve SA-rectangular RMDPs [55], but their approach does not generalize to S-rectangular RMDP that we target. Homotopy and bisection methods have been used to solve L_1-constrained ambiguity sets [30, 31], but these methods are based on sparsity properties of the L_1 norm, which do not hold for the L_∞ norm. We elaborate on this crucial difference after we introduce our algorithm. The existing efficient algorithms developed for the SA-rectangular RMDPs with L_∞ balls [21] do not generalize to S-rectangular RMDPs. Developing fast optimization algorithms for S-rectangular RMDPs is challenging because optimal policies may need to be randomized.

Several fast new methods have been proposed recently for solving RMDPs more efficiently. They suggest replacing the standard value and policy iteration methods with more efficient algorithms, such as forms of modified policy iteration [31, 37] or gradient descent [24]. Most of these accelerated methods can further benefit from the fast Bellman operator algorithms that we propose in this work.

In Section 2.2, we described the basic Robust MDP framework. The remainder of this chapter is organized as follows. Section 5.1 proposes a new homotopy method for solving

SA-rectangular ambiguity sets, which serves as a building block for our main contribution. In Section 5.2, we propose a bisection method that can solve, in combination with the homotopy method, RMDPs with S-rectangular ambiguity sets. Finally, Section 5.3 presents experimental results that show that our method is over $1,000$ times faster than using Gurobi, a leading commercial linear solver when solving RMDPs with hundreds of states.

5.1 Computing SA-Rectangular Bellman Operator in Linear Time

This section develops a new quasi-linear time algorithm for computing the SA-rectangular robust Bellman operator defined by the L_∞ norm. This entails solving the following optimization problem

$$(\mathfrak{L}\boldsymbol{v})_s \;=\; \max_{a\in\mathcal{A}} \min_{\xi\le\kappa_{s,a}} q_{s,a}(\xi,\boldsymbol{v}) \;. \tag{5.1}$$

The algorithm developed in this section also serves as the primary building block of the S-rectangular algorithm described in Section 5.2. The remainder of the section is organized as follows. Section 5.1.1 first analyzed the LP formulation of the function q and, then, Section 5.1.2 uses these properties to develop a new, fast homotopy continuation algorithm.

Computing the SA-rectangular robust Bellman operator for a fixed state s, action a, and a value function $\boldsymbol{v}$ requires one to evaluate the nature response function $q_{s,a}(\xi,\boldsymbol{v})$ in (2.3). Because the symbols $s, a, \boldsymbol{v}$ are fixed throughout this section, we omit them in the notation. For example, we use $q(\xi)$ instead of $q_{s,a}(\xi,\boldsymbol{v})$, and $\bar{\boldsymbol{p}}$ in place of $\bar{\boldsymbol{p}}_{s,a}$. To further eliminate clutter, let $\boldsymbol{z} = r_{s,a}\cdot\mathbf{1} + \gamma\cdot\boldsymbol{v}$. Then, the following optimization problem

$$q_{s,a}(\xi,\boldsymbol{v}) \;=\; \min_{\boldsymbol{p}\in\Delta^S} \left\{ r_{s,a} + \gamma\cdot\boldsymbol{p}^\mathsf{T}\boldsymbol{v} \mid \left\|\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}\right\|_\infty \le \xi \right\} \;. \tag{5.2}$$

can be formulated as the following parametric LP:

$$\begin{aligned} q(\xi) \;&=\; \min_{\boldsymbol{p}\in\Delta^S} \left\{ \boldsymbol{p}^\mathsf{T}\boldsymbol{z} \mid \|\bar{\boldsymbol{p}} - \boldsymbol{p}\|_\infty \le \xi \right\} \\ &=\; \min_{\boldsymbol{p}\in\mathbb{R}^S} \left\{ \boldsymbol{z}^\mathsf{T}\boldsymbol{p} \mid \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1,\, -\xi \le p_i - \bar{p}_i \le \xi,\, p_i \ge 0,\, i = 1,\ldots,S \right\} \;. \end{aligned} \tag{5.3}$$

The remainder of this section develops fast algorithms for solving (5.3) for all values $\xi \geq 0$.

5.1.1 Properties of Nature Response Function q

The LP in (5.3) can be solved using generic solvers, like Gurobi or Mosek, but these are impractically slow for solving RMDPs. The optimization in (5.3) can also be solved in quasi-linear time for any *fixed* $\xi \geq 0$, as we summarize in Appendix A.3.5. The known quasi-linear algorithm is, unfortunately, insufficient for solving the S-rectangular robust Bellman operator in Section 5.2. In this section, we prove results that pave the way for solving (5.3) for *all* $\xi \geq 0$ simultaneously in quasi-linear time, which enables efficient algorithms for both S- and SA-rectangular RMDPs.

It will be convenient to use $\boldsymbol{p}^\star(\xi)$ to refer to an optimal solution in (5.3). To avoid unnecessary technicalities, we assume that all elements of $\boldsymbol{z}$ are distinct, which guarantees that the optimal solution $\boldsymbol{p}^\star(\xi)$ is unique. In practice, one may add an arbitrarily small value to the elements of $\boldsymbol{z}$ to ensure that they are all distinct. To get some intuition into the form of the nature response function $q(\xi)$ and its optimal solution $\boldsymbol{p}^\star(\xi)$, consider the following simple example.

Example 5.1.1. *Consider an* RMDP *with six states, one action,* $\boldsymbol{z} = (-1, 0, 1, 2, 3, 4)^\mathsf{T}$*, and nominal transition probabilities* $\bar{\boldsymbol{p}} = (0.0, 0.1, 0.3, 0.1, 0.2, 0.3)^\mathsf{T}$*. The functions* $q(\xi)$ *and* $\boldsymbol{p}^\star(\xi)$ *are depicted in Figures 5.1 and 5.2, where Figure 5.2 shows the evolution of each* $p_i(\xi)$ *using a different color for each* i*.*

The following property of the function q is indispensable for our analysis and shows that $q(\xi)$ is always of the form depicted in Figure 5.1. It follows from standard LP properties and is proved in Appendix A.3.1.

Lemma 5.1.2. *The function* $q(\xi)$ *is continuous, piecewise linear, non-increasing, and convex in* ξ*.*

To develop an efficient algorithm, we now analyze the structure of the *bases* of the

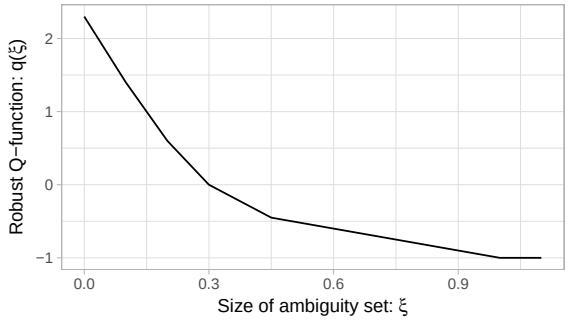

Figure 5.1: Function $q(\xi)$ in Example 5.1.1.

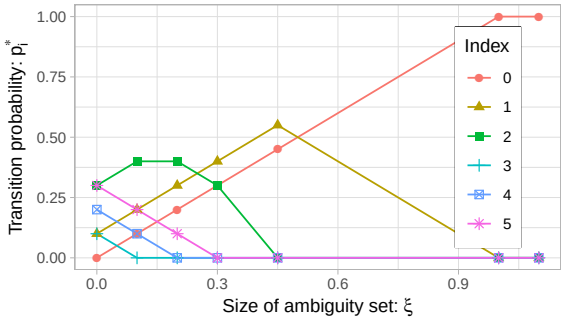

Figure 5.2: Probabilities $\boldsymbol{p}^\star(\xi)$ in Example 5.1.1.

LP (5.3). Recall that a *basis* is a subset of S *linearly independent* constraints in the LP, which must hold with equality. There are S constraints included in each basis because S is the number of optimization variables. Note that constraints may be active (or violated) without being included in the basis.

To represent a basis in (5.3), we use sets $\mathcal{R}_B, \mathcal{D}_B, \mathcal{N}_B, \mathcal{T}_B \subseteq \{1, \ldots, S\}$ to indicate which constraints are included in the basis with their meanings summarized in Table 5.2. If $i \in \mathcal{D}_B$ we call it a *donor*, if $i \in \mathcal{R}_B$, we call it a *receiver*, and if $i \in \mathcal{N}_B$, we call it a *none*. The set $\mathcal{T}_B = \{1, \ldots, S\} \setminus \mathcal{R}_B \setminus \mathcal{D}_B \setminus \mathcal{N}_B$ represents the remaining indexes and $i \in \mathcal{T}_B$ is called a *trader*. Lemma 5.1.4 below justifies the names for these sets.

Our homotopy algorithm will leverage the specific behavior of the optimal solution $\boldsymbol{p}^\star(\xi)$ as a function of ξ. Because each basis B represents a set of S linearly independent inequalities with S variables, a unique solution $\boldsymbol{p}_B(\xi)$ exists for any value ξ. Note that $\boldsymbol{p}_B(\xi)$ need not be optimal or feasible.

The following lemma establishes the properties of the bases in (5.3) that we need to

Index $i \in$	Constraints in B
$\mathcal{R}_B$ (receiver)	$p_i - \bar{p}_i \leq \xi$ in B
$\mathcal{D}_B$ (donor)	$\bar{p}_i - p_i \leq \xi$ in B
$\mathcal{N}_B$ (none)	$p_i \geq 0$ in B

Table 5.1: Composition of B for $i \in \mathcal{S}$.

consider in our optimization; the proof can be found in Appendix A.3.1.

Lemma 5.1.3. *Suppose that* $\boldsymbol{p}^\star$ *is optimal in* (5.3) *for some* $\xi \geq 0$*. Then, there exists a basis* B *such that* (i) $\boldsymbol{p}^\star = \boldsymbol{p}_B(\xi)$*,* (ii) *sets* $\mathcal{R}_B, \mathcal{D}_B, \mathcal{N}_B, \mathcal{T}_B$ *do not intersect,* (iii) $|\mathcal{R}_B| + |\mathcal{D}_B| + |\mathcal{N}_B| + |\mathcal{T}_B| = S$*,* (iv) $|\mathcal{T}_B| = 1$*, and* (v) $z_i < z_j < z_k$ *for each* $i \in \mathcal{R}_B, j \in \mathcal{T}_B, k \in \mathcal{D}_B \cup \mathcal{N}_B$*.*

Lemma 5.1.3 is important because it limits the bases relevant to the optimization, which is crucial for building fast algorithms. In particular, it shows that the sets $\mathcal{R}, \mathcal{D}, \mathcal{N}, \mathcal{T}$ *partition* the set $\mathcal{S}$, and there is always exactly one trader. The lemma also shows that z coefficients for receivers are smaller than the coefficient for the trader, which is smaller than the coefficients for donors and nones.

The following lemma establishes the rate of change of the linear function $\boldsymbol{p}_B(\xi)$, which is the last necessary component for our homotopy algorithm. The lemma's proof is in Appendix A.3.1.

Lemma 5.1.4. *The derivatives* $\dot{\boldsymbol{p}} = \nabla_\xi \boldsymbol{p}_B(\xi)$ *for any basis* B *that satisfies the properties in Lemma 5.1.3 are equal for each* $i \in \mathcal{S}$ *to*

$$\dot{p}_i = 1 \text{ if } i \in \mathcal{R}_B, \quad \dot{p}_i = -1 \text{ if } i \in \mathcal{D}_B, \quad \dot{p}_i = 0 \text{ if } i \in \mathcal{N}_B, \quad \dot{p}_i = |\mathcal{D}_B| - |\mathcal{R}_B| \text{ if } i \in \mathcal{T}_B \ .$$

Moreover, the slope is $\dot{q} = {}^{d}\!/\!_{d\xi}\, q_B(\xi) = \sum_{i \in \mathcal{R}_B} z_i - \sum_{j \in \mathcal{D}_B} z_j + \sum_{\tau \in \mathcal{T}_B} \dot{p}_\tau z_\tau$ *.*

Note that Lemma 5.1.3 shows that each $i \in \mathcal{S}$ is either a receiver, a donor, a trader, or none. Lemma 5.1.4 then shows that with an increasing ξ, a *donor* donates its probability mass, a *receiver* receives probability mass, a *trader* either donates or receives at a variable rate, and a *none* remains unchanged.

5.1.2 Homotopy Algorithm

We are now ready to describe the proposed homotopy method and prove its correctness and complexity. Algorithm 4 summarizes a conceptual version of the homotopy algorithm. As

discussed below, one needs to avoid computing the full gradient $\nabla_\xi \boldsymbol{p}_B(\xi)$ to achieve quasi-linear time complexity. The complete algorithm with quasi-linear runtime is described in Algorithm 9 in the appendix.

The main idea of Algorithm 4 is simple: it iteratively computes the linear segments of $q(\xi)$ for all $\xi \geq 0$. The algorithm starts with $\xi = 0$, where the optimal solution is $\boldsymbol{p}_0 = \bar{\boldsymbol{p}}$ with objective value $q_0 = \boldsymbol{p}_0^\mathsf{T}\boldsymbol{z}$. Then, the algorithm tracks the optimal bases in $q(\xi)$ as ξ increases. When the $\boldsymbol{p}_{B_t}(\xi)$ becomes infeasible with the increasing ξ, the algorithm finds a new optimal basis B_{t+1} and continues until it arrives at a basis with $d/d\xi\, q(\xi') = 0$; the function q is constant for all $\xi \geq \xi'$. Since $q(\xi)$ is piecewise linear in ξ (see Lemma 5.1.2), we obtain its full description from all optimal bases.

Algorithm 4: Homotopy method to compute $q(\xi)$

Input: Objective $\boldsymbol{z}$, and nominal probabilities $\bar{\boldsymbol{p}}$
Output: Breakpoints $(\xi_t)_{t=0,\ldots,T+1}$ and $(q_t)_{t=0,\ldots,T+1}$ such that $q_t = q(\xi_t)$

Initialize $\xi_0 \leftarrow 0$, $t \leftarrow 0$, $\boldsymbol{p}_0 \leftarrow \bar{\boldsymbol{p}}$ and $q_0 \leftarrow q(\xi_0) = \boldsymbol{p}_0^\mathsf{T}\boldsymbol{z}$, $\tau_0 = \lceil S/2 \rceil$ and basis B_0 such that: ;

$\mathcal{T}_{B_0} = \{\tau_0\},\ \mathcal{R}_{B_0} = \{i \,|\, i < \tau_0\},\ \mathcal{D}_{B_0} = \{j \,|\, j > \tau_0\},\ \mathcal{N}_{B_0} = \{\}$;

while $\dot{q}_t < 0$ **do**

  Compute maximum step size for B_t to remain feasible ($\mathcal{T}_{B_t} = \{\tau_t\}$): ;

  $\Delta\xi_t \leftarrow \max\{\xi \geq 0 \mid \boldsymbol{p}_t + \xi \cdot \nabla_\xi \boldsymbol{p}_{B_t}(\xi_t) \geq \boldsymbol{0},\ |(\boldsymbol{p}_t + \xi \cdot \nabla_\xi \boldsymbol{p}_{B_t}(\xi_t) - \bar{\boldsymbol{p}})_{\tau_t}| \leq \xi_t + \xi\}$;

  Update breakpoints: ;

    $\boldsymbol{p}_{t+1} \leftarrow \boldsymbol{p}_t + \Delta\xi_t \cdot \nabla_\xi \boldsymbol{p}_{B_t}(\xi_t)$;

    $q_{t+1} \leftarrow \boldsymbol{p}_{t+1}^\mathsf{T}\boldsymbol{z}$;

    $\xi_{t+1} \leftarrow \xi_t + \Delta\xi_t$;

  Let $B_{t+1} \leftarrow$ next basis with the steepest slope (see Lemma 5.1.7 and Table 5.2); ;

  Let $t \leftarrow t + 1$;

end

Let $\xi_{T+1} \leftarrow 1$ and $q_{T+1} \leftarrow q_T$;

return $(\xi_t)_{t=0,\ldots,T+1}$, *and* $(q_t)_{t=0,\ldots,T+1}$

The following theorem proves the correctness of Algorithm 4. Informally, the theorem shows that the function q is piecewise linear with *breakpoints* (points of non-linearity) only at $\xi_t, t = 1, \ldots, T+1$. Note that $\xi_{T+1} = 1$ because this is the upper bound on the L_∞ norm

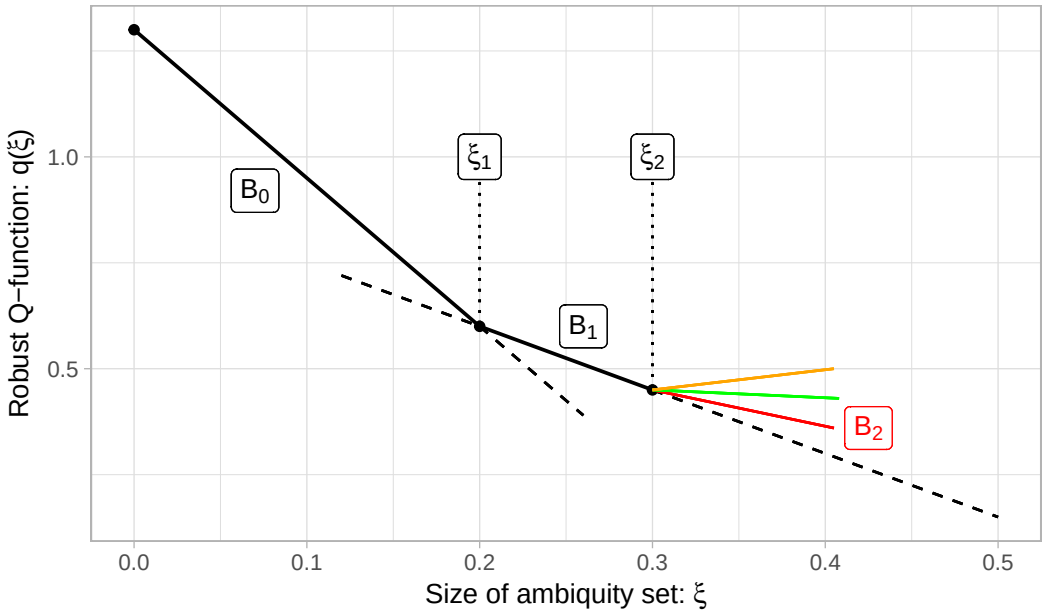

Figure 5.3: An illustration of Algorithm 4.

of a difference of two discrete probability distributions, and, as a result, the function $q(\xi)$ is constant for $\xi > 1$. The proof can be found in Appendix A.3.1.

Theorem 5.1.5. *Suppose that Algorithm 4 returns* $(\xi_t)_{t=0,\dots,T+1}$ *and* $(q_t)_{t=0,\dots,T+1}$. *Then* $q(\alpha \cdot \xi_t + (1-\alpha) \cdot \xi_{t+1}) = \alpha \cdot q(\xi_t) + (1-\alpha) \cdot q(\xi_{t+1})$ *for* $\alpha \in [0,1]$ *and* $t = 0, \dots, T+1$.

We will refer to Figure 5.3 in order to provide the intuition that underlies the construction of Algorithm 4 and its correctness. The figure depicts an example state of Algorithm 4 at $t = 2$ and Line 10. The solid lines show the values q_{B_1} and q_{B_2} when they are feasible and optimal. The dashed lines indicate when the bases are infeasible or suboptimal at each breakpoint ξ_1, ξ_2. The colored lines at ξ_2 indicate the slopes for the possible candidates for B_2. The algorithm chooses a basis with a minimal slope.

The correctness of Algorithm 4 follows from the following three lemmas. The first lemma shows that the algorithm chooses the initial basis with the minimum possible slope.

Lemma 5.1.6. *The basis* B_0 *constructed in Line 4 of Algorithm 4 is feasible at* $\xi = 0$ *and has*

Type	B_{t+1}	$\mathcal{D}_{B_{t+1}}$	$\mathcal{R}_{B_{t+1}}$	$\mathcal{T}_{B_{t+1}}$	$\mathcal{N}_{B_{t+1}}$
1: $\mathcal{D} \to \mathcal{N}$	$\hat{B}^1$	$\mathcal{D}_{B_t} \setminus \{l\}$	$\mathcal{R}_{B_t}$	$\mathcal{T}_{B_t}$	$\mathcal{N}_{B_t} \cup \{l\}$
2: $\mathcal{T} \to \mathcal{N}$	$\hat{B}^2$	$\mathcal{D}_{B_t}$	$\mathcal{R}_{B_t} \setminus \{m\}$	$\{m\}$	$\mathcal{N}_{B_t} \cup \mathcal{T}_{B_t}$
3: $\mathcal{T} \to \mathcal{D}$	$\hat{B}^3$	$\mathcal{D}_{B_t} \cup \mathcal{T}_{B_t}$	$\mathcal{R}_{B_t} \setminus \{m\}$	$\{m\}$	$\mathcal{N}_{B_t}$

Table 5.2: Possible types of basis change at a breakpoint ξ_{t+1} described in Lemma 5.1.7.

the steepest slope among other feasible basis B that satisfies the conditions of Lemma 5.1.3:

$$ {}^{d}\!/_{d\xi}\, q_{B_0}(0) \;\leq\; {}^{d}\!/_{d\xi}\, q_{B}(0) \;. $$

The second lemma shows that the following basis will be selected according to one of the rules in Table 5.2.

Lemma 5.1.7. *Let a basis B_t be optimal for ξ_{t+1} in Algorithm 4, such that $\boldsymbol{p}^\star(\xi_{t+1}) = \boldsymbol{p}_{B_t}(\xi_{t+1})$ and $q(\xi_{t+1}) = q_{B_t}(\xi_{t+1})$. Assume that $\boldsymbol{p}_{B_t}(\xi)$ is infeasible for $\xi > \xi_{t+1}$. If $\mathcal{B}$ are all bases feasible for some $\xi > \xi_t$ then one with the steepest slope can be constructed as*

$$ \operatorname*{argmin}_{B \in \mathcal{B}} \frac{d}{d\xi} q(\xi_{t+1}) \;\ni\; \begin{cases} \hat{B}^1 & \text{if} \quad (\boldsymbol{p}_{B_t}(\xi_{t+1}))_l = 0, \text{ for some } l \in \mathcal{D}_{B_t} \\ \hat{B}^2 & \text{if} \quad (\boldsymbol{p}_{B_t}(\xi_{t+1}))_\tau = 0, \text{ and } \mathcal{T}_{B_t} = \{\tau\} \\ \hat{B}^3 & \text{if} \quad (\bar{\boldsymbol{p}} - \boldsymbol{p}_{B_t}(\xi_{t+1}))_\tau = \xi_{t+1}, \text{ and } \mathcal{T}_{B_t} = \{\tau\} \end{cases} , $$

where $\hat{B}^1, \hat{B}^2, \hat{B}^3$ are defined in Table 5.2 and $m \in \operatorname{argmax}_{i \in \mathcal{R}_{B_t}} z_i$.

Lemma 5.1.7 shows that there are three possible types of basis change; any other possible choice of the basis would contradict the continuity of $q(\xi)$ (Lemma 5.1.2). Recall also that Lemma 5.1.3 shows that there is always exactly one trader. The *first* type of basis change occurs when p_l for a donor $l \in \mathcal{D}$ reaches zero; the donor turns into a none in the new basis. The *second* type of basis change occurs when the trader probability mass becomes zero; the trader then turns into a none and the receiver with the largest z value becomes the new trader. The *third* type of basis change happens when the trader's gradient satisfies

$^{d}/_{d\xi}\, p_\tau(\xi) < -1$ and its probability mass reaches to its lower bound for a given ξ making the basis infeasible for greater values of ξ. The trader then becomes a donor, and, again, the receiver with the largest z value becomes the new trader.

Finally, the third lemma shows that the optimal basis B_t identified at ξ_t remains feasible until ξ_{t+1}. Note that the convexity of $q(\xi)$ implies that the feasible basis remains optimal.

Lemma 5.1.8. *If B_t is feasible and optimal at ξ_t in Algorithm 4, then it is also optimal on the interval $[\xi_t, \xi_t + \Delta\xi_t]$ computed in Line 4 of Algorithm 4.*

We now turn to the computational complexity of Algorithm 9. As the following theorem shows, the number of iterations T in Algorithm 4 is at most $\mathcal{O}(S)$. Unfortunately, keeping track of $\boldsymbol{p}_t$ in each iteration of Algorithm 4 requires also $\mathcal{O}(S)$ time leading to the overall time complexity of $\mathcal{O}(S^2)$. To adapt Algorithm 4 to run in quasi-linear time, Algorithm 9, in the appendix, generates the necessary values ξ_t, q_t without tracking the complete $\boldsymbol{p}_t$ values. Its runtime is quasi-linear because it needs to sort the values of $\boldsymbol{z}$ to perform the optimizaton in Line 10 in constant time.

Theorem 5.1.9. *Algorithm 4 terminates in at most $\mathcal{O}(S)$ iterations and can be adapted to run in $\mathcal{O}(S \log S)$ time (see Algorithm 9 in Appendix A.3.2).*

We conclude by discussing the relationship with the homotopy method proposed for solving RMDPs with the L_1 ambiguity sets [30]. Although our algorithm is also a homotopy method, it is based on analysis that departs significantly from earlier work. The simplifying properties for the L_∞ ambiguity sets differ considerably from the L_1 norm. When the ambiguity sets are defined as L_1 balls, only two components of $\boldsymbol{p}$ change at the time. Figure 5.2 illustrates that when the ambiguity sets are L_∞ balls, all components of $\boldsymbol{p}$ may change with the increasing ξ. The fast algorithm for the L_∞-constrained RMDP relies on the more subtle structure of the optimal bases described in Lemma 5.1.3 which leads to a more complex algorithm.

5.2 Computing S-Rectangular Bellman Operator in Linear Time

In this section, we propose a fast algorithm for compute the robust Bellman operator (2.5) for S-rectangular RMDPs. We assume a fixed state $s \in \mathcal{S}$ and omit the subscripts throughout the section. For instance, the nominal probabilities for state s and action a are denoted by $\bar{\boldsymbol{p}}_a \in \Delta^A$. We also assume a fixed value function $\boldsymbol{v} \in \mathbb{R}^S$ and let $\boldsymbol{z}_a = r_{s,a} \cdot \mathbf{1} + \gamma \cdot \boldsymbol{v}$ for $a \in \mathcal{A}$.

The fast algorithm for computing the S-rectangular robust Bellman operator builds on Algorithm 4. As Theorem 5.1.9 shows, the function q_a defined in (2.3) is piecewise linear with $\mathcal{O}(S)$ linear segments that can be computed efficiently by Algorithm 9. Since q_a is piecewise linear, it is easy to construct its inverse just by swapping ξ_t and q_t to get the following function:

$$q_a^{-1}(u) \;=\; \min_{\boldsymbol{p} \in \Delta^S} \left\{ \|\boldsymbol{p} - \bar{\boldsymbol{p}}_a\|_\infty \mid \boldsymbol{p}^\mathsf{T} \boldsymbol{z}_a \le u \right\}, \quad \forall a \in \mathcal{A}. \tag{5.4}$$

The function q_a^{-1} returns the budget that nature needs to achieve a response u. Using the function q_a^{-1}, we can reformulate the S-rectangular robust Bellman operator as:

$$(\mathfrak{L}\boldsymbol{v})_s = \max_{\boldsymbol{d} \in \Delta^A} \min_{\boldsymbol{\xi} \in \mathbb{R}_+^A} \left\{ \sum_{a \in \mathcal{A}} d_a \cdot q_a(\xi_a) \mid \sum_{a \in \mathcal{A}} \xi_a \le \kappa \right\} = \min_{u \in \mathbb{R}} \left\{ u \mid \sum_{a \in \mathcal{A}} q_a^{-1}(u) \le \kappa \right\} \tag{5.5}$$

The correctness of this formulation follows by standard duality arguments and is proved in Lemma A.3.3 in Appendix A.3.3.

The optimization in (5.5) is remarkable because its objective is one-dimensional function with one constraint. A natural algorithm to use with such an optimization problem is the bisection method outlined in Algorithm 5 (see Algorithm 11 in the appendix for a more detailed algorithm). Algorithm 5 keeps an interval $[u_{\min}, u_{\max}]$ such that the optimal $u^\star$ satisfies that $u^\star \in [u_{\min}, u_{\max}]$. In every time step, the algorithm bisects the interval $[u_{\min}, u_{\max}]$ in half and updates $u_{\min}, u_{\max}$ in order to preserve that $u^\star \in [u_{\min}, u_{\max}]$. One may think of $u_{\min}$ as the maximal known infeasible u in (5.5) and of $u_{\max}$ as the minimal known feasible

u in (5.5).

Algorithm 5: Bisection method for solving (5.4).

Input: Desired precision ϵ, functions $q_a^{-1}, \forall a \in \mathcal{A}$
Output: $\hat{u}$ such that $|u^\star - \hat{u}| \leq \epsilon$, where $u^\star$ is optimal in Equation (5.4)

Initialize bounds: ;
    $u_{\min} \leftarrow \min_{a\in\mathcal{A}, s\in\mathcal{S}} (\boldsymbol{z}_a)_s$;
    $u_{\max} \leftarrow \max_{a\in\mathcal{A}, s\in\mathcal{S}} (\boldsymbol{z}_a)_s$;
while $u_{\max} - u_{\min} > 2\ \epsilon$ **do**
    Let $u \leftarrow (u_{\min} + u_{\max})/2$;
    if $\sum_{a\in\mathcal{A}} q_a^{-1}(u) \leq \kappa$ **then**
        $u_{\max} \leftarrow u$
    else
        $u_{\min} \leftarrow u$
    end
end
return $(u_{\min} + u_{\max})/2$

The time complexity of Algorithm 5 depends on the desired precision ϵ. To remove this dependence on ϵ, it is sufficient to replace the bisection by binary search over the breakpoints; we give the details of this method in Algorithm 11 in the appendix. The following theorem proved in Appendix A.3.3, summarizes the correctness and complexity of the proposed algorithms.

Theorem 5.2.1. *The combined Algorithms 4 and 5 compute the S-rectangular robust Bellman operator for any state $s \in \mathcal{S}$ and can be adapted (see Algorithms 9 and 11) to run in time $\mathcal{O}(SA\log(SA))$.*

5.3 Numerical Results

This section compares the empirical runtime of Algorithms 4 and 5 with the runtime of Gurobi 9.1, a leading LP solver. The results were generated on a computer with an Intel i7-9700 CPU with 32 GB RAM; the algorithms are implemented in C++.

As the main benchmark problem, we use the classic *inventory management (IM)* problem [84]. In this problem, the decision-maker must decide at every time step how much

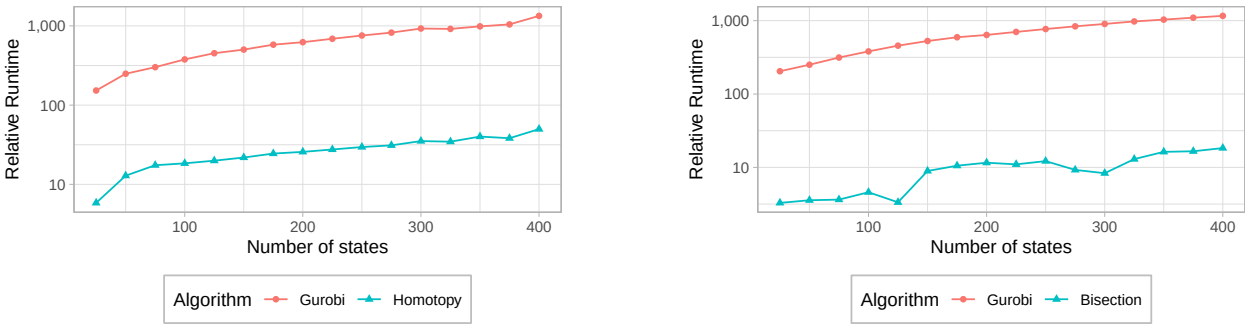

Figure 5.4: Relative computation time (unitless) of our algorithms and an LP solver over nominal MDP in SA-rectangular (left) and S-rectangular (right) inventory management RMDP.

inventory to order. The number of states and actions in this problem corresponds to the holding capacity and order size respectively. This makes it easy to scale the number of states and actions and evaluate how the algorithms scale with problem size. To evaluate the performance of our methods on small problems, we also consider the *RiverSwim (RS)* domain [74], and the *Machine Replacement (MR)* domain [13]. Please see Appendix A.3.4 for the detailed description of these domains.

Figure 5.4 shows the time to compute the robust Bellman operator for a single state in the inventory management domain. The x-axis represents the number of states (maximum holding capacity) in the domain. The number of actions is the same as the number of states. The y-axis represents the time to compute the robust Bellman operator divided by the time to calculate the standard (non-robust) Bellman operator. The results show that even in MDPs with a few hundred states, our proposed algorithms are about 100-times faster than the leading LP solver. Interestingly, our algorithm is an order of magnitude faster, even for minor problems. We use a robustness budget $\kappa = 1.2$, but the computation time is insensitive to the particular choice of κ.

Table 5.3 compares the time to compute the robust Bellman operator on machine replacement (MR), river swim (RS), and inventory management (IM) with 30 state problems. It is worth emphasizing that MR and RS are tiny problems with less than 30 states, yet our algorithms are up to 800 times faster than using an LP solver. This indicates that our

Rect.	Algorithm	MR	RS	IM
SA	Algorithm 4	$< \mathbf{1}$	**3**	**10**
SA	Gurobi LP	2960	2240	9770
S	Algorithm 5	**40**	**52**	**67**
S	Gurobi LP	129	217	2740

Table 5.3: Time (ms) to compute $\mathfrak{L}$ for S- and SA-rectangular RMDPs with L_∞ sets.

Rect.	Algorithm	MR	RS	IM
SA	[Ho]-Alg.1	< 1	2	1
SA	Gurobi LP	92	363	1140
S	[Ho]-Alg.2	1	2	5
S	Gurobi LP	79	317	2260

Table 5.4: Time (ms) to compute $\mathfrak{L}$ for S- and SA-rectangular RMDPs with L_1 sets [31].

methods scale well with the number of states and that the constant overhead is relatively small. For the sake of completeness, we include in Table 5.4 the timing results obtained for the RMDP with L_1 ambiguity sets. These results show that solving the L_∞-constrained RMDP is more complex than the L_1-constrained RMDP, but also that we can achieve similar dramatic speedups in L_∞ constrained RMDPs as [31].

In this chapter, we introduced a new homotopy method for calculating robust Bellman operators for S- and SA-rectangular ambiguity sets constructed with L_∞-norm ball. Theoretically, we show that the worst-case time complexity of our algorithms is quasi-linear: $\mathcal{O}(SA\log(S))$. The algorithms also perform well in practice, outperforming a leading LP solver by several orders of magnitude.

In addition to being faster than a general-purpose LP solver, our algorithms are also much simpler. They make it possible to solve L_∞-constrained RMDPs without the cost and complexity of involving a general LP solver. Although free and open-source LP solvers are available, their performance falls significantly short of commercial ones. The algorithms we propose are also easy to combine with value function approximation methods in RMDPs [78].

It is important to understand whether similar algorithms can be developed for RMDPs

with more complex ambiguity sets, such as ones defined using Wasserstein distance, L_2-norm, or KL-divergence.

CHAPTER 6

Low-rank Feature Selection for Linear Value Function

Reinforcement learning (RL) methods typically use value function approximation to solve problems with large state spaces [75, 76]. The approximation makes it possible to generalize from a small number of samples to the entire state space. Perhaps the most common methods for value function approximation are neural networks and linear methods. Neural networks offer unparalleled expressibility in complex problems, but linear methods remain popular due to their simplicity, interpretability, ease of use, and low sample and computational complexity.

This chapter, aligned with previous chapters, focuses on *batch reinforcement learning* [41]. In batch RL, all domain samples are provided in advance as a batch, and it is impossible or difficult to gather additional samples. This is common in many functional areas. In medical applications, for example, it is usually too risky and expensive to run additional tests. In environmental applications, it may take a whole growing season to obtain a new batch of samples.

Overfitting is a particularly difficult challenge in practical deployments of batch RL. Detecting that the solution overfits the available data can be complex. Using a regular test set does not work in RL because of the difference between the sampling policy and the optimized policy. Also, off-policy policy evaluation remains difficult in large problems [35]. As a result, a solution that overfits the training batch is often discovered only after it has been deployed and real damage has been done.

With linear approximation, overfitting occurs more easily when too many features are

used. In this chapter, we present Fast Feature Selection (FFS), a new method that can effectively reduce the number of features in batch RL. To avoid confusion, we use the term *raw features* to refer to the natural features of a given problem. They could, for example, be the individual pixel values in video games or particular geographic observations in geospatial applications. Raw features are usually numerous, but each feature alone has a low predictive value. FFS constructs (rather than selects) a small set of useful features that are a linear combination of the provided raw features. The constructed features are designed to be used in concert with LSTD, LSPI, and other related batch RL methods.

FFS reduces the number of features by computing a low-rank approximation of the transition matrix after it is compressed using the available raw features. Low-rank matrix approximation and completion gained popularity from their use in collaborative filtering [54], but they have been also applied to reinforcement learning and other machine learning domains [11, 58, 68]. None of this prior work, however, computes a low-rank approximation of the compressed transition matrix.

Several feature selection methods for reducing overfitting in RL have been proposed previously, but none of them explicitly target problems with low-rank (compressed) transition probabilities. L_1 regularization, popularized by the LASSO, has been used successfully in reinforcement learning [39, 43, 66]. L_1 regularization assumes that only a few of the features are sufficient to obtain a good approximation. This is not a reasonable assumption when individual raw features are of a low quality.

Proto-value functions [48] use the spectral decomposition of the transition probability matrix or of a related random walk. Although the spectrum of a matrix is closely related to its rank, eigenvector-based methods provide weak approximation guarantees even when the majority of the eigenvalues are zero [61]. BEBFs and Krylov are other techniques that work well when the characteristic polynomial of the transition probability matrix is of a small degree [60, 61]; this property is unrelated to the matrix rank.

The closest prior method to FFS is LFD [72]. LFD works by computing 1) a linear

encoder that maps the raw features of a state to a small-dimensional space and 2) a linear decoder that maps the small-dimensional representation back to the raw features. While LFD was not introduced as a low-rank approximation technique, we show that similarly to FFS, it introduces no additional error when the matrix of transition probabilities is low-rank. LFD, unfortunately, has several limitations. It involves solving a non-convex optimization problem, is difficult to analyze, and provides no guidance for deciding on the right number of features to use.

As the main contribution, this chapter proposes and analyzes FFS both theoretically and empirically. We derive new bounds that relate the singular values of the transition probability matrix to the approximation error. As a secondary contribution, we provide a new interpretation of LFD as a type of low-rank approximation method. We argue that FFS improves on LFD in terms of providing fast and predictable solutions, similar or better practical performance, and guidance on how many features should be selected.

We summarize the relevant properties of linear value function approximation in Markov decision processes in Chapter 2. The remainder of this chapter is organized as follows. We present the preliminary Bellman error analysis in Section 6.1. Section 6.2 describes FFS and new bounds that relate singular values of the compressed transition probability matrix to the approximation error. Section 6.3 then compares FFS with other feature construction algorithms, and, finally, the empirical evaluation in Section 6.4 indicates that FFS is a promising feature selection method.

6.1 Bellman Error Analysis

The standard bound on the performance loss of a policy computed using, for example, approximate policy iteration can be bounded as a function of the Bellman error (e.g., Williams and Baird [82]). To motivate FFS, we use the following result that shows that the Bellman error can be decomposed into the error in 1) the compressed rewards and in 2) the compressed transition probabilities.

Theorem 6.1.1 ([72]). *Given a policy π and features Φ, the Bellman error of a value function $v = \Phi \boldsymbol{w}_\Phi^\pi$ satisfies:*

$$\mathrm{BE}_\Phi = \underbrace{(\boldsymbol{r}^\pi - \Phi \boldsymbol{r}_\Phi^\pi)}_{\Delta_r^\pi} + \gamma \underbrace{(P^\pi \Phi - \Phi P_\Phi^\pi)}_{\Delta_P^\pi} \boldsymbol{w}_\Phi^\pi \ .$$

We seek to construct a basis that minimizes both $\|\Delta_r^\pi\|_2$ and $\|\Delta_P^\pi\|_2$. These terms can be used to bound the L_2 norm of Bellman error as:

$$\begin{aligned} \| \mathrm{BE}_\Phi \|_2 &\leq \|\Delta_r^\pi\|_2 + \gamma \|\Delta_P^\pi\|_2 \|\boldsymbol{w}_\Phi^\pi\|_2 \leq \\ &\leq \|\Delta_r^\pi\|_2 + \gamma \|\Delta_P^\pi\|_F \|\boldsymbol{w}_\Phi^\pi\|_2 \ , \end{aligned} \tag{6.1}$$

where the second inequality follows from $\|X\|_2 \leq \|X\|_F$.

The Bellman error (BE) decomposition in (6.1) has two main limitations. The first limitation is that it is expressed in the L_2 norm rather than the L_∞ norm, which is needed for standard Bellman residual bounds [82]. This can be addressed, in part, by using the weighted L_2 norm bounds [53]. The second limitation of (6.1) is that it depends on $\|\boldsymbol{w}_\Phi^\pi\|_2$ besides the terms $\|\Delta_r^\pi\|_2, \|\Delta_P^\pi\|_2$ that we focus on. Since $\|\boldsymbol{w}_\Phi^\pi\|_2$ can be problem-dependent, the theoretical analysis of its impact on the approximation error is beyond the scope of this work.

6.2 FFS: A Fast Low-Rank Approximation for Feature Selection

In this section, we describe the proposed method for selecting features from a low-rank approximation of the transition probabilities. To simplify the exposition, we first introduce the method for the tabular case and then extend it to the batch RL setting with many raw features in Section 6.2.1.

The Tabular Fast Feature Selection algorithm is summarized in Algorithm 6. Informally, the algorithm selects the top k left-singular vectors and the reward function for the features. Our error bounds show that including the reward function as one of the features is critical.

Algorithm 6: TFFS: Tabular Fast low-rank Feature Selection

Input: Transition matrix P, rewards $\boldsymbol{r}$, and number of features $k+1$

Compute SVD decomposition of P: $P = U\Sigma V^{\mathsf{T}}$;

Assuming decreasing singular values in Σ, select the first k columns of U: $U_1 \leftarrow [u_1, \ldots, u_k]$;

return *Approximation features:* $\Phi = [U_1, \boldsymbol{r}]$.

It is not surprising that when the matrix P is of a rank at most k then using the first k left-singular vectors will result in no approximation error. However, such low-rank matrices are rare in practice. We now show that it is sufficient that the transition matrix P is close to a low-rank matrix for TFFS to achieve small approximation errors. In order to bound the error, let the SVD decomposition of P be $\mathrm{SVD}(P) = U\Sigma V^{\mathsf{T}}$, where

$$U = \begin{bmatrix} U_1 & U_2 \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}, \qquad V = \begin{bmatrix} V_1 & V_2 \end{bmatrix} .$$

That implies that the transition probability matrix can be expressed as:

$$P = U_1\Sigma_1 V_1^{\mathsf{T}} + U_2\Sigma_2 V_2^{\mathsf{T}} .$$

Let matrix U_1 have k columns and let the singular values be ordered decreasingly. Then, Algorithm 6 generates $\Phi = [U_1, \boldsymbol{r}]$. The following theorem bounds the error regarding the largest singular value for a vector not included in the features.

Theorem 6.2.1. *Assuming k features Φ computed by Algorithm 6, the error terms in Theorem 6.1.1 are upper bounded as:*

$$\|\Delta_P\|_2 \leq \|\Sigma_2\|_2,$$
$$\|\Delta_r\|_2 = 0 .$$

The proof of the theorem is deferred to Appendix A.4.1. Theorem 6.2.1 implies that if we

choose Φ in a way that the singular values in Σ_2 are zero (when the transition matrix is low rank), Δ_P would be zero. That means that for a matrix of rank k there is no approximation error because $\|\Delta_P\|_2 = 0$. More broadly, when the rank of the matrix is greater than k, the error is minimized by choosing the singular vectors with the greatest singular values. That means that TFFS chooses features Φ that minimize the error bound in Theorem 6.2.1.

6.2.1 Using Raw Features

Using TFFS in batch RL is impractical since the transition matrix and reward vector are usually too large and are not available directly. The values must instead be estimated from samples and the raw features.

Algorithm 7: FFS: Fast low-rank Feature Selection from raw features

Input: Sampled raw features A, next state of raw feature A' , rewards $\boldsymbol{r}$, and number of features $k+1$

1 Estimate compressed transition probabilities $P_A = A^+A'$ as in LSTD ;
2 Compute SVD decomposition of P_A: $P_A = U\Sigma V^\mathsf{T}$;
3 Compute compressed reward vector: $\boldsymbol{r}_A = A^+\boldsymbol{r}$;
4 Assuming decreasing singular values in Σ, select the first k columns of U: $U_1 \leftarrow [u_1, \ldots, u_k]$;
5 **return** *Approximation features:* $\widehat{\Phi} = [U_1, \boldsymbol{r}_A]$.

As described in the introduction, we assume that the domain samples include a potentially large number of low-information raw features. We use A to denote the $n \times l$ *matrix of raw features*. As with Φ, each row corresponds to one state, and each column corresponds to one *raw* feature. The compressed transition matrix is denoted as $P_A = A^+PA$ and compressed rewards are denoted as $\boldsymbol{r}_A = A^+\boldsymbol{r}$ and are computed as in (2.9). To emphasize that the matrix P is not available, we use $A' = PA$ to denote the expected value of features after one step. Using this notation, the compressed transition probabilities can be expressed as $P_A = A^+A'$.

Algorithm 7 describes the FFS method that uses raw features. Similar to TFFS, the algorithm computes an SVD of the transition matrix. Note that the features $\widehat{\Phi}$ are linear

combinations of the raw features. To get the actual state features, it is sufficient to compute $A\widehat{\Phi}$ where $\widehat{\Phi}$ is the output of Algorithm 7. The matrix $\widehat{\Phi}$ represents features for P_A and is of a dimension $l \times k$ where l is the number of raw features in A. We omit the details for how the values are estimated from samples, as this is well known, and refer the interested reader to Johns, Petrik, and Mahadevan [36], Lagoudakis and Parr [40].

Using raw features to compress transition probabilities and rewards is simple and practical, but it is also essential to understand the consequences of relying on these raw features. Because FFS computes features that are a linear combination of the raw features, they cannot express more complex value functions. FFS thus introduces additional error—akin to bias—but reduces sampling error—akin to variance. The following theorem shows that the errors due to our approximation and using raw features merely add up with no additional interactions.

Theorem 6.2.2. *Assume that the raw features A for P and computed features $\widehat{\Phi}$ for P_A are normalized, such that $\|A\|_2 = ||\widehat{\Phi}||_2 = 1$. Then:*

$$\|\Delta_P^{A\widehat{\Phi}}\|_2 \leq \|\Delta_P^A\|_2 + \|\Delta_{P_A}^{\widehat{\Phi}}\|_2 \ ,$$
$$\|\Delta_r^{A\widehat{\Phi}}\|_2 \leq \|\Delta_r^A\|_2 + \|\Delta_{r_A}^{\widehat{\Phi}}\|_2 \ ,$$

where the superscript of Δ indicates the feature matrix for which the error is computed; for example $\Delta_{P_A}^{\widehat{\Phi}} = P_A\widehat{\Phi} - \widehat{\Phi}(P_A)_{\widehat{\Phi}}$.

Note that the normalization of features required in Theorem 6.2.2 can be achieved by multiplying all features by an appropriate constant, which is an operation that does not affect the approximate value function. Scaling features does, however, affect the magnitude of $\boldsymbol{w}_\Phi$, which, as we discuss above, is problem-specific and largely independent of the feature selection method used.

Perhaps one of the most attractive attributes of FFS is its simplicity and low computational complexity. Selecting the essential features only requires computing the singular value

decomposition—for which many efficient methods exist—and augmenting the result with the reward function. As we show next, this simple approach is well-motivated by bounds on approximation errors.

We described FFS in terms of singular value decomposition and showed that when the (compressed) transition probability matrix has a low rank, the approximation error is likely to be small. Next, we describe the relationship between FFS and other feature selection methods in more detail.

6.3 Related Feature Selection Methods

In this section, we describe similarities and differences between FFS and related feature construction or selection methods.

Perhaps the best-known method for feature construction is the technique of *proto-value functions* [47, 48]. Proto-value functions are closely related to spectral approximations [61]. This approximation uses the eigenvector decomposition of the transition matrix $P = S\Lambda S^{-1}$, where S is a matrix with eigenvectors as its columns and Λ is a diagonal matrix with eigenvalues that are sorted from the largest to the smallest. The first k columns of S are then used as the approximation features. As with our FFS method, it is beneficial to augment these features with the reward vector. We will refer to this method as EIG+R in the numerical results. Surprisingly, unlike with FFS, which uses top k left-singular vectors, using the top k eigenvectors does not guarantee zero Bellman residual even if the rank of P is less than k.

Using the Krylov subspace is another feature selection approach [61] which has also been referred to as BEBF [59, 60]. The Krylov subspace $\mathcal{K}$ is spanned by the images of $\boldsymbol{r}$ under the first k powers of P (starting from $P^0 = \mathbf{I}$):

$$\mathcal{K}_k(P, \boldsymbol{r}) = \text{span}\{\boldsymbol{r}, P\boldsymbol{r}, \ldots, P^{k-1}\boldsymbol{r}\} \ .$$

Petrik [61] shows that when k is equal to the degree of the minimal polynomial, the approximation error is zero. Krylov methods are more likely to work in different problem settings than either EIG+R or FFS and can be easily combined with them.

Algorithm 8: LFD: Linear Feature Discovery for a fixed policy π (Song et al. 2016).

```
D_0 ← random(k, l);
i ← 1;
while Not Converged do
    E_i ← A^+ A' D_{i-1} ;
    D_i ← (A E_i)^+ A';
    i ← i + 1 ;
end
return E_k // Same role as Φ̂ in FFS.
```

Finally, Linear Feature Discovery (LFD) [72] is a recent feature selection method that is closely related to FFS. Algorithm 8 depicts a simplified version of the LFD algorithm, which does not consider the reward vector and approximates the value function instead of the Q-function for a fixed policy π. Recall that A is the matrix of raw features and $A' = P^\pi$.

LFD is motivated by the theory of *predictive optimal feature encoding*. A low-rank encoder E^π is *predictively optimal* if there exist decoders D_s^π and D_r^π such that:

$$AE^\pi D_s^\pi = P^\pi A \ , \qquad\qquad AE^\pi D_r^\pi = r^\pi \ .$$

When an encoder and decoder are predictively optimal, then the Bellman error is 0 [72]. Unfortunately, it is almost impossible to find problems in practice in which a predictively optimal controller exists. No bounds on the Bellman error are known when a controller is merely close to predictively optimal. This is in contrast with the bounds in Theorems 6.2.1 and 6.2.2 that hold for FFS.

Although LFD appears to be quite different from FFS, our numerical experiments show that it computes solutions that are similar to the solutions of FFS. We argue that LFD can be interpreted as a coordinate descent method for computing the following low-rank

approximation problem:

$$\min_{E \in \mathbb{R}^{l \times k}, D \in \mathbb{R}^{k \times l}} \|AED - A'\|_F^2 \ . \tag{6.2}$$

This is because the iterative updates of E_i and D_i in Algorithm 8 are identical to solving the following optimization problems:

$$E_i \leftarrow \arg\min_{E \in \mathbb{R}^{l \times k}} \|AED_{i-1} - A'\|_F^2$$

$$D_i \leftarrow \arg\min_{D \in \mathbb{R}^{k \times l}} \|AE_iD - A'\|_F^2$$

The equivalence follows directly from the orthogonal projection representation of linear regression. This kind of coordinate descent is a very common heuristic for computing low-rank matrix completions [28]. Unfortunately, the optimization problem in (6.2) is *non-convex* and coordinate descent, like LFD, may only converge to a local optimum, if at all. Simple algebraic manipulation reveals that any set of k singular vectors represents a local minimum of LFD. Finally, we are not aware of any method that can solve (6.2) optimally.

Similarly to LFD, FFS solves the following optimization problem:

$$\min_{E \in \mathbb{R}^{l \times k}, D \in \mathbb{R}^{k \times l}} \|ED - A^+A'\|_F^2 \ . \tag{6.3}$$

This fact follows readily from the SVD decomposition of A^+A' and the fact that the Frobenius norm is equal to the L_2 norm of the singular values [22, 29].

Note that when using tabular features ($A = \mathbf{I}$) the optimization problems (6.2) and (6.3) are identical. For any other raw features, there are two reasons for preferring (6.3) over (6.2). First, FFS is much easier to solve both in theory and in practice. Second, as Theorem 6.2.2 shows, the approximation error of FFS is simply additive to the error inherent to the raw features. No such property is known for LFD. In the next section, we compare the two methods numerically.

6.4 Empirical Evaluation

In this section, we empirically evaluate the quality of features generated by FFS both with and without using raw features. We focus on a comparison with LFD which was empirically shown to outperform radial basis functions (RBFs) [40], random projections [20], and other methods [72].

We first compare the quality of solutions on a range of synthetic randomly generated problems. The goal is to ensure that the methods behave similarly regardless of the number of samples, or the type of raw features that are used. Then, we use an image-based version of the cart-pole benchmark, used previously by Song et al. [72], to evaluate FFS in more complex settings. This problem is used to evaluate both the solution quality and the computational complexity of the methods.

6.4.1 Synthetic Problems

To compare FFS to other common approaches in feature selection, we start with small policy evaluation problems. Since the policy is fixed throughout these experiments, we omit all references to it. The data matrix $A \in \mathbb{R}^{n \times l}$ only contains the states where n denotes the number of states and l the length of each *raw* feature, with $\Phi \in \mathbb{R}^{n \times k}$ using k features.

The synthetic problems that we use throughout this section have 100 states. The rewards $\boldsymbol{r} \in \mathbb{R}^{100}$ are generated uniformly randomly from the interval of $[-500, 500)$. The stochastic transition probabilities $P \in [0, 1)^{100 \times 100}$ are generated from the uniform Dirichlet distribution. To ensure that the rank of P is at most 40, we compute P as a product $P = XY$, where X and Y are small dimensional. The discount factor we use is $\gamma = 0.95$.

We now proceed by evaluating FFS for both tabular and image-based features. For the sake of consistency, we use FFS to refer to both TFFS in a tabular case and FFS when raw features are available. To evaluate the quality of the value function approximation, we compute the Bellman residual of the fixed-point value function, which is a standard metric

used for this purpose. Recall that the Bellman error can be expressed as

$$\mathrm{BE} = \Delta_{\boldsymbol{r}} + \gamma \Delta_P \boldsymbol{w}_\Phi,$$

where $\boldsymbol{w}_\Phi$ is the value-function given in (2.10). All results we report in this section are an average of 100 repetitions of the experiments. All error plots show the L_2 norm of the Bellman error in a logarithmic scale.

Case 1: Tabular raw features. In this case, the true transition probabilities P and the reward function $\boldsymbol{r}$ are known, and the raw features are an identity matrix: $A = \mathbf{I}$. Therefore all computations are made concerning the precise representations of the underlying MDP.

This is the simplest setting, under which SVD simply reduces to a direct low-rank approximation of the transition probabilities. That is, the SVD optimization problem reduces to:

$$\min_{U_1 \in \mathbb{R}^{n \times k}} \min_{\Sigma_1 V_1^\mathsf{T} \in \mathbb{R}^{k \times n}} \|U_1 \Sigma_1 V_1^\mathsf{T} - P\|_F^2 \ .$$

Similarly, the constructed features will be $\Phi = U_1$. In case of FFS, we can simply add the reward vector to feature's set $\Phi = [U_1, \boldsymbol{r}]$. EIG+R and KRY are implemented as described in Parr et al. [59], Petrik [61]. In case of EIG+R approach, we use the eigenvectors of P as basis functions, and then $\boldsymbol{r}$ is included. For Krylov basis we calculate $\Phi = \mathcal{K}_k(P, \boldsymbol{r})$.

Figure 6.1 depicts the Bellman error for the exact solution when the number of features used for the value function varies from 0 to 100. Note that the Bellman error of FFS is zero for $k \geq 40$. This is because the rank of P is 40, and according to Theorem 6.2.1 the first 40 features obtained by FFS are sufficient to get $\|\mathrm{BE}\|_2 = 0$. This experiment shows FFS is robust and generally outperforms other methods. The only exception is the Krylov method which is more effective when few features are used but is not numerically stable with more features. The Krylov method could be combined relatively easily with FFS to get the best of both bases.

Case 2: Image-based raw features. In this case, the raw features A are not tabular but instead simulate an image representation of states. So the Markov dynamics are experienced only via samples and the functions are represented using an approximation scheme. The matrix A is created by randomly allocated zeros and ones similar to the structure of a binary image. We use LSTD to compute the approximate value function, as described in Section 2.4.

The SVD optimization problem now changes as described in Section 6.2.1. The constructed features will be $\Phi = A\widehat{\Phi}$ and for FFS we include the reward predictor vector $[P_A, \boldsymbol{r}_A]$ in the optimization problem. In the case of the EIG+R method, we multiply the eigenvectors of P_A and $\boldsymbol{r}_A$ with the raw features. The Krylov basis is constructed as: $\Phi = A\,\mathcal{K}_k(P_A, \boldsymbol{r}_A)$ where $\mathcal{K}_k$ is the k-th order Krylov operator.

Figure 6.2 compares the Bellman error for the approximate solution. FFS again outperforms other methods. LFD is unstable when the number of features exceeds the rank of P, and sometimes it is not possible to obtain the pseudo-inverse of matrix AE.

It is worth noting that this section deals with very small MDPs with only about 100 states. It is expected to see a more significant gap in Bellman error of these methods when dealing with large MDPs with enormous and high-dimensional state spaces. In the next section, we compare LFD and FFS with the random projection approach using a more significant and more challenging benchmark problem.

6.4.2 Cart-Pole

These experiments evaluate the similarity between the linear feature encoding (LFD) approach and the fast feature selection (FFS) method on a modified version of cart-pole, which is a standard reinforcement learning benchmark problem. We use random projections [20] as the baseline. The controller must learn a good policy by merely observing the *image* of the cart-pole without direct observations of the angle and angular velocity of the pole. This problem is large enough that the computational time plays an important role, so we also

compare the computational complexity of the three methods.

Note that this is a control benchmark, rather than a value approximation for a fixed policy. Since the goal of RL is to optimize a policy, results on policy optimization are often more meaningful than just obtaining a small Bellman residual which is not sufficient to guarantee that a good policy will be computed [36].

To obtain training data, we collect the specified number of trajectories with the starting angle and angular velocity sampled uniformly on $[-0.1, 0.1]$. The cart position and velocity are set to zero at each episode.

The algorithm was given three consecutive, rendered, gray-scale images of the cart-pole. Each image is downsampled to 39×50 pixels, so the raw state is a $39 \times 50 \times 3 = 5850-$dimensional vector. We chose three frames to preserve the Markov property of states without manipulating the cart-pole simulator in OpenAI Gym. We used $k = 50$ features for all methods.

We follow a setup analogous to Song et al. [72] by implementing least-squares policy iteration [40] to obtain the policy. The training data sets are produced by running the cart for $[50, 100, 200, 400, 600]$ episodes with a random policy. We then run policy iteration to iterate up to 50 times or until there is no change in the $A' = P^{\pi}A$ matrix.

The state of the pole in the classic cart-pole problem is described by its angle and angular velocity. However, in the image-based implementation, the agent does not observe this information. Song et al. [72] chose two successive frames to show the state of the pole. To preserve the Markovian property of the state, they had to modify the simulator and force the angular velocity to match the change in angle per time step $\dot{\theta} = (\theta' - \theta)/\delta t$. We, instead, use the standard simulator from OpenAI Gym and choose the last three consecutive frames rather than two. Three consecutive frames are sufficient to infer θ and $\dot{\theta}$ and construct a proper Markov state. Intriguingly, no linear feature construction methods work well in the original problem definition when using only the last two frames.

The performance of the learned policy is reported for 100 repetitions to obtain the average

number of balancing steps. Figure 6.3 displays the average number of steps during which the pole kept its balance using the same training data sets. For each episode, a maximum of 200 steps was allowed to run. This result shows that on the larger training sets the policies obtained from FFS and LFD are quite similar, but with small training sets, FFS shows a better performance. Both methods outperform random projection (RPr) significantly.

Figure 6.4 depicts the average running time of LFD and FFS for obtaining the value function with $k = 50$. The computation time of FFS grow very slowly as the number of training episodes increases; at 600 training episodes, the maximum number of episodes tested, FFS is 10 times faster than LFD. Therefore, LFD would likely be impractical in large problems with many training episodes.

Both FFS and LFD implementations use randomized SVD in all computations including the computation of pseudo-inverses. The result is usually very close to truncated singular value decomposition. Randomized SVD is fast on large matrices on which we need to extract only a small number of singular vectors. It reduces the time to compute k top singular values for an $m \times n$ matrix from $O(mnk)$ to $O(mn \log(k))$ [26].

In comparison to black-box methods such as neural networks, linear value functions are more interpretable: their behavior is more transparent from an analysis standpoint and feature engineering standpoint. It is comparatively simple to gain some insight into the reasons for which a particular choice of features succeeds or fails. When the features are normalized, the magnitude of each parameter is related to the importance of the corresponding feature in the approximation [40].

Figure 6.5 shows the learned coefficients of Q-function for three actions (left, right and no-op) using color codes. The q-values are obtained by the inner product of raw features (3-frames of cart-pole) and these coefficients. They are computed by the FFS method from 400 training episodes with a random policy. In this experiment, the raw images, taken from the cart-pole environment in the OpenAI Gym toolkit, are preprocessed, converted to gray-scale, and normalized. Therefore, the pole in the raw images is in black, and the value of

black pixels is close to zero. Other areas in the raw features are in white, so these pixel values are closer to one. It is interesting to see how the linear value function captures the dynamics of the pole (the cart is stationary). If the pole is imbalanced, the value function is smaller since the blue area in Figure 6.5 represents negative scalars.

This chapter proposed a new feature construction technique that computes a low-rank approximation of the transition probabilities. We believe that this approach is a promising method for feature selection in batch reinforcement learning. A particular strength of our proposed method is that it is easy to judge its effectiveness by singular values of features not included in the approximation. After all, it would be interesting to study the impact of FFS on finite-sample bounds and robustness in RL.

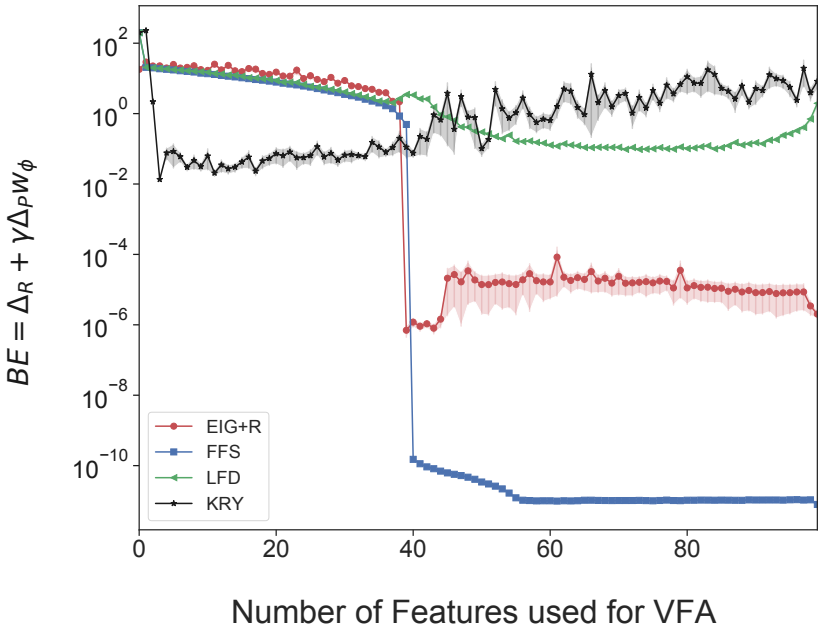

Figure 6.1: Bellman error for the exact solution. The transition matrix is 100×100 and has a low rank with $\text{rank}(P) = 40$. The Input matrix is $A = \mathbf{I}$ an identity matrix.

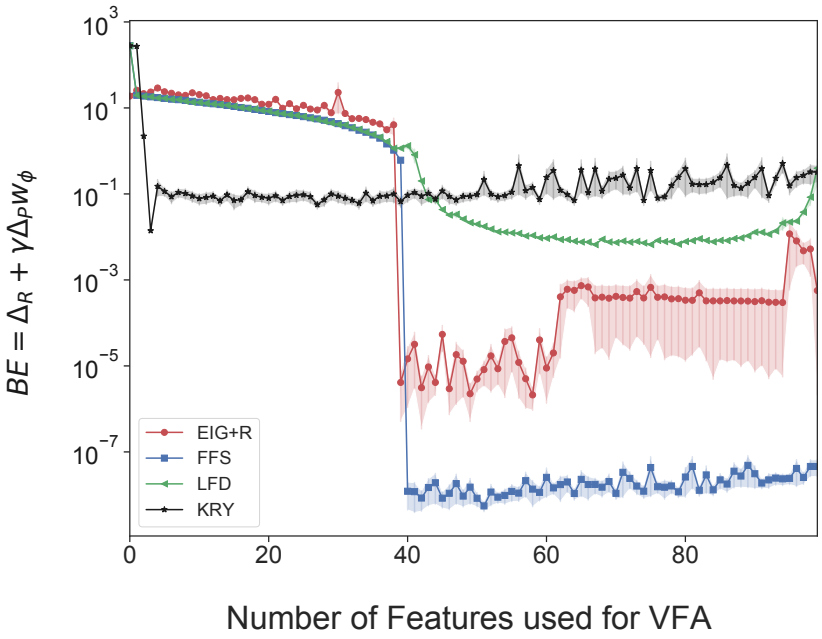

Figure 6.2: Bellman error for the approximate solution. The transition matrix is 100×100 and has a low rank with $\text{rank}(P) = 40$. The Input matrix is $A =$ random binary matrix.

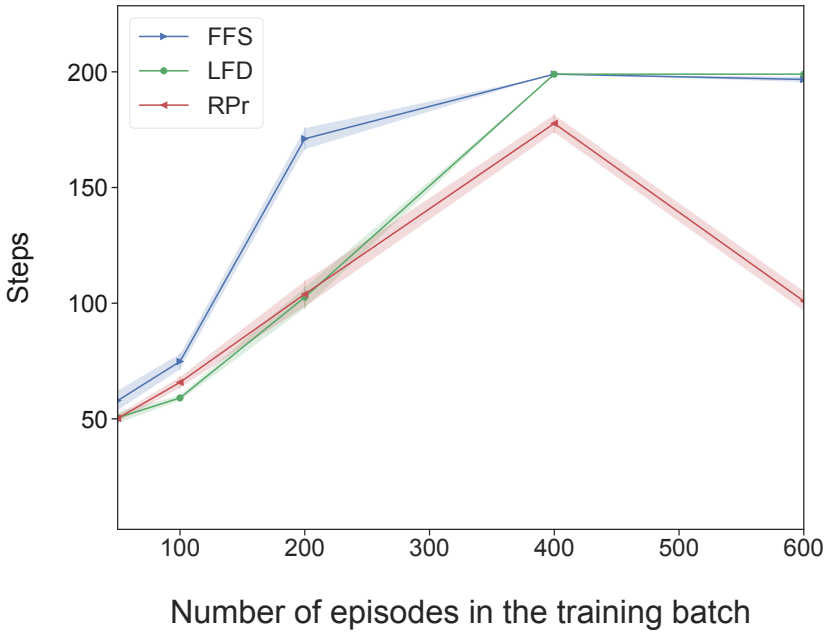

Figure 6.3: The average number of balancing steps with $k = 50$.

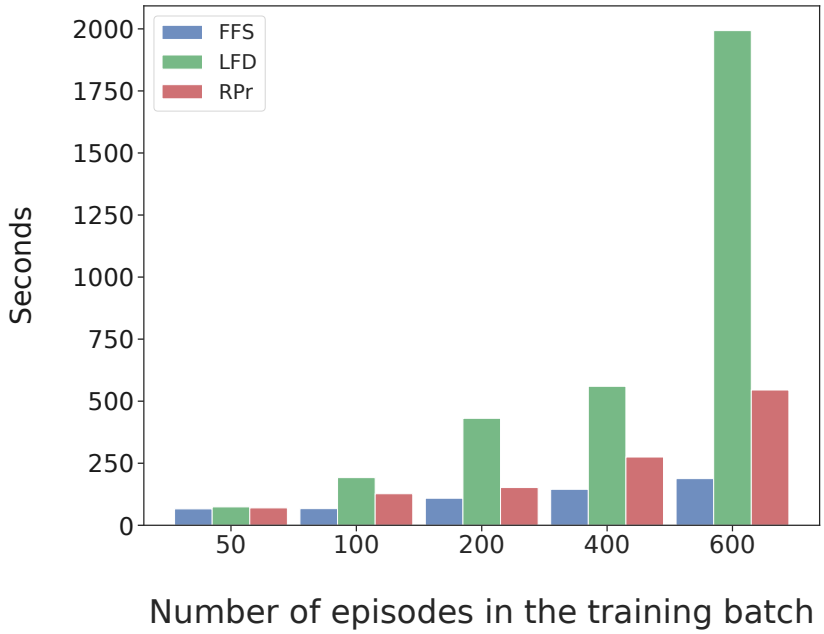

Figure 6.4: Mean running time for estimating the Q-function with $k = 50$.

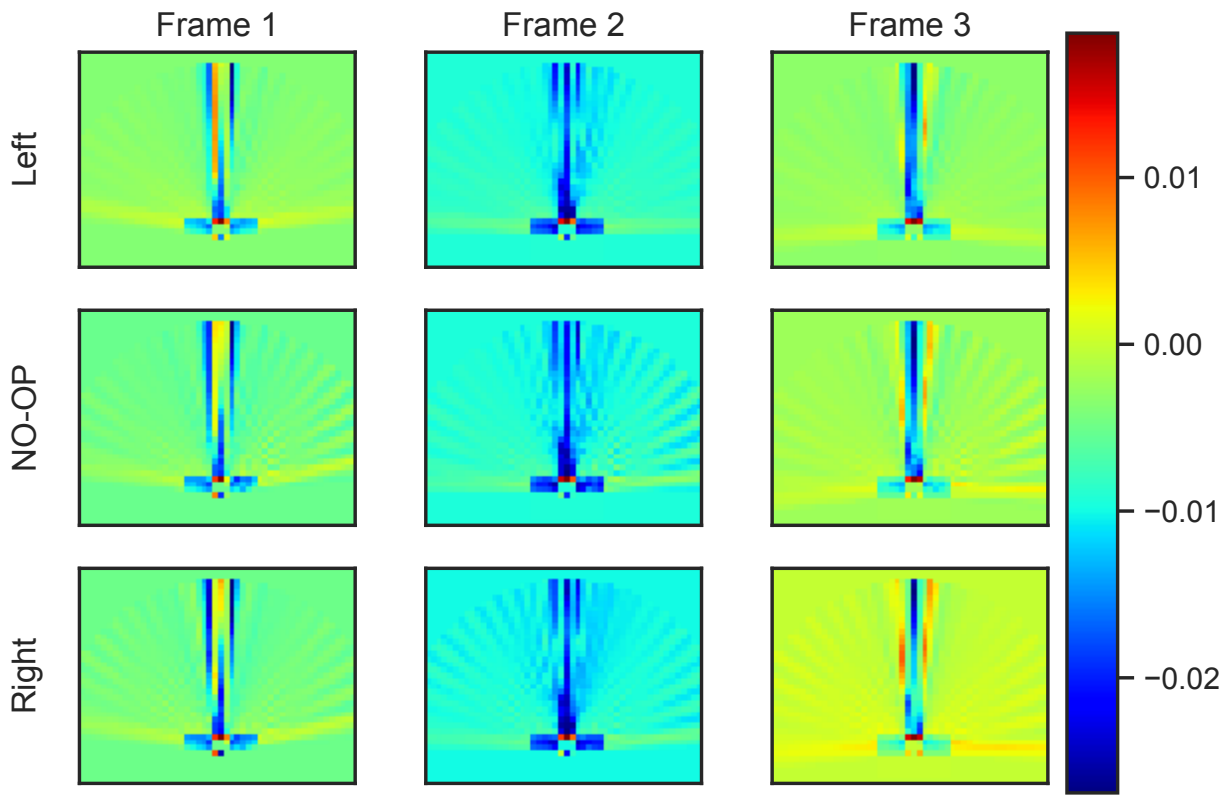

Figure 6.5: Value function in jet color-map.

CHAPTER 7

Conclusion

Robust MDPs (RMDPs) relax the assumption for known transition probabilities by considering a set of transition probabilities, better known as an ambiguity set, containing true values with high confidence. Empirical evidence verifies that RMDP offers benefits over methods that ignore uncertainty in the transition parameters. The combination of tractability and effectiveness has fueled the increasing popularity of RMDP in the RL community. However, RMDPs lack suitable methods of constructing ambiguity sets that lead to robust solutions without being excessively conservative. Many domain-specific techniques have been suggested, but most do not offer finite-sample guarantees or are hard to use.

These realizations give rise to numerous questions. For example, is there any perfect ambiguity set $\mathcal{P}$, and what is the characteristic of a perfect $\mathcal{P}$? If $\mathcal{P}$ is not perfect ambiguity set for tractability, what would be the performance loss? The critical idea to construct $\mathcal{P}$ in the current literature is to identify the smallest ambiguity set that satisfies a Bayesian or frequentist robustness guarantee.

This dissertation aims to answer some of these questions. We illustrate several RL algorithms that efficiently calculate robust policies with limited data that improve the policies' performance and ease the computational complexity compared to standard risk-averse RL algorithms.

First, we proposed a new approach for optimizing the percentile criterion using RMDPs beyond the conventional ambiguity sets. We derived new bounds on the performance loss of the RMDPs concerning the optimal percentile criterion. These bounds show that the

quality of the RMDP is driven by the span of its ambiguity sets along a specific direction. We proposed a linear-time algorithm that minimizes the span of the ambiguity sets and derived new sampling guarantees. Our experimental results show that this simple RMDP improvement can lead to better return guarantees. Future work should focus on scaling the method to a large state-space using value function approximation or other techniques.

Second, we introduced a new homotopy method for calculating robust Bellman operator for S- and SA-rectangular ambiguity sets constructed with L_∞-norm ball. Theoretically, we show that the worst-case time complexity of our algorithms is quasi-linear: $\mathcal{O}(SA\log(S))$. The algorithms also perform well in practice, outperforming a leading LP solver by several orders of magnitude.

In addition to being faster than a general-purpose LP solver, our algorithms are also much more straightforward. They make it possible to solve L_∞-constrained RMDPs without the cost and complexity of involving a general LP solver. Although free and open-source LP solvers are available, their performance falls significantly short of commercial ones. The algorithms we propose are also easy to combine with value function approximation methods in RMDPs.

Furthermore, we present a fast and robust feature selection method, FFS, for linear value function approximation, a common approach to solving reinforcement learning problems with large state spaces. We show that our technique is faster and more stable than alternative methods. FFS computes a low-rank approximation of the transition probabilities. We believe that FFS is a promising method for feature selection in batch reinforcement learning. It is effortless to implement, fast to run, and relatively easy to analyze. A particular strength of FFS is that it is easy to judge its effectiveness by singular values of features not included in the approximation.

There remain some interesting open questions. Because most existing RL algorithms are based on the dynamic programming principle, it is not easy to use them in percentile criterion settings. It is worth studying how to compute safe policies without relying on

common standard RL algorithms. A further question is whether rectangularity assumptions can be relaxed by designing a robust dynamic program that calculates the safe returns directly without constructing an ambiguity set. We will study these questions in future work.

Bibliography

[1] Alagoz, O.; Maillart, L. M.; Schaefer, A. J.; and Roberts, M. S. 2007. Choosing among living-donor and cadaveric livers. *Management Science*, 53(11): 1702–1715.

[2] Auer, P.; Jaksch, T.; and Ortner, R. 2009. Near-optimal Regret Bounds for Reinforcement Learning. *Advances in Neural Information Processing Systems*.

[3] Auer, P.; Jaksch, T.; and Ortner, R. 2010. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(1): 1563–1600.

[4] Behzadian, B.; Gharatappeh, S.; and Petrik, M. 2019. Fast Feature Selection for Linear Value Function Approximation. *The International Conference on Automated Planning and Scheduling (ICAPS)*.

[5] Behzadian, B.; Petrik, M.; and Ho, C. P. 2021. Fast Algorithms for L_∞-constrained S-rectangular Robust MDPs. *Advances in Neural Information Processing Systems*, 34.

[6] Behzadian, B.; Russel, R. H.; Petrik, M.; and Ho, C. P. 2021. Optimizing Percentile Criterion using Robust MDPs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 1009–1017.

[7] Bertsekas, D. P. 2003. *Nonlinear programming*. Athena Scientific.

[8] Bertsekas, D. P.; and Tsitsiklis, J. N. 1996. *Neuro-Dynamic Programming*. Athena Scientific, 1st edition. ISBN 1886529108.

[9] Boyd, S.; and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge: Cambridge University Press. ISBN 9780511804441.

[10] Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

[11] Cheng, B.; Asamov, T.; and Powell, W. B. 2017. Low-Rank Value Function Approximation for Co-optimization of Battery Storage. *IEEE Transactions on Smart Grid*, 3053.

[12] Delage, E.; and Mannor, S. 2010. Percentile Optimization for Markov Decision Processes with Parameter Uncertainty. *Operations Research*, 58(1): 203–213.

[13] Delage, E.; and Ye, Y. 2010. Distributionally robust optimization under moment uncertainty with application to data driven problems. *Operations Research*, 58(3): 595–612.

[14] Delgado, K. V.; De Barros, L. N.; Dias, D. B.; and Sanner, S. 2016. Real-time dynamic programming for Markov decision processes with imprecise probabilities. *Artificial Intelligence*, 230: 192–223.

[15] Devroye, L.; Györfi, L.; and Lugosi, G. 2013. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media.

[16] Dietterich, T.; Taleghan, M.; and Crowley, M. 2013. PAC optimal planning for invasive species management: Improved exploration for reinforcement learning from simulator-defined MDPs. *National Conference on Artificial Intelligence (AAAI)*.

[17] Faísca, N. P.; Dua, V.; and Pistikopoulos, E. N. 2007. *Multiparametric Linear and Quadratic Programming*, chapter 1, 1–23. John Wiley & Sons, Ltd.

[18] Gelman, A.; Carlin, J. B.; Stern, H. S.; and Rubin, D. B. 2014. *Bayesian Data Analysis*. Chapman and Hall/CRC, 3rd edition.

[19] Gelman, A.; Lee, D.; and Guo, J. 2015. Stan: A probabilistic programming language for Bayesian inference and optimization. *Journal of Educational and Behavioral Statistics*, 40(5): 530–543.

[20] Ghavamzadeh, M.; Lazaric, A.; Maillard, O.; and Munos, R. 2010. LSTD with Random Projections. In *Advances in Neural Information Processing Systems (NIPS)*, 721–729.

[21] Givan, R.; Leach, S.; and Dean, T. 2000. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1): 71–109.

[22] Golub, G. H.; and Van Loan, C. F. 2013. *Matrix computations*. JHU press.

[23] Goyal, V.; and Grand-Clement, J. 2018. Robust Markov Decision Process: Beyond Rectangularity. arXiv:1811.00215.

[24] Grand-Clément, J.; and Kroer, C. 2021. First-Order Methods for Wasserstein Distributionally Robust MDP. *arXiv:2009.06790*.

[25] Gupta, V. 2019. Near-Optimal Bayesian Ambiguity Sets for Distributionally Robust Optimization. *Management Science*, 65(9).

[26] Halko, N.; Martinsson, P.-G.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2): 217–288.

[27] Hansen, T. D.; Miltersen, P. B.; and Zwick, U. 2013. Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor. *Journal of the ACM (JACM)*, 60(1): 1–16.

[28] Hastie, T.; Mazumder, R.; Lee, J.; and Zadeh, R. 2015. Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares. *Journal of Machine Learning Research*, 16: 3367–3402.

[29] Hastie, T.; Tibshirani, R.; Friedman, J. H.; and Friedman, J. H. 2009. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

[30] Ho, C. P.; Petrik, M.; and Wiesemann, W. 2018. Fast Bellman Updates for Robust MDPs. In *International Conference on Machine Learning (ICML)*, volume 80, 1979–1988.

[31] Ho, C. P.; Petrik, M.; and Wiesemann, W. 2020. Partial Policy Iteration for L1-Robust Markov Decision Processes. *arXiv:2006.09484*.

[32] Hong, L. J.; Huang, Z.; and Lam, H. 2020. Learning-Based Robust Optimization: Procedures and Statistical Guarantees. *Management Science*.

[33] Ibaraki, T.; and Katoh, N. 1988. *Resource Allocation Problems: Algorithmic Approaches*. The MIT Press.

[34] Iyengar, G. N. 2005. Robust dynamic programming. *Mathematics of Operations Research*, 30(2): 257–280.

[35] Jiang, N.; and Li, L. 2015. Doubly robust Off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning (ICML)*. ISBN 9781510829008.

[36] Johns, J.; Petrik, M.; and Mahadevan, S. 2009. Hybrid least-squares algorithms for approximate policy evaluation. *Machine Learning*, 76(2): 243–256.

[37] Kaufman, D. L.; and Schaefer, A. J. 2013. Robust modified policy iteration. *INFORMS Journal on Computing*, 25(3): 396–410.

[38] Kéry, M.; and Schaub, M. 2011. *Bayesian population analysis using WinBUGS: a hierarchical perspective*. Academic Press.

[39] Kolter, J. Z.; and Ng, A. Y. 2009. Regularization and feature selection in least-squares temporal difference learning. In *International Conference on Machine Learning (ICML)*, 521–528. ACM.

[40] Lagoudakis, M. G.; and Parr, R. 2003. Least-squares policy iteration. *Journal of Machine Learning Research*, 4: 1107–1149.

[41] Lange, S.; Gabel, T.; and Riedmiller, M. 2012. Batch reinforcement learning. In *Reinforcement learning*, 45–73. Springer.

[42] Laroche, R.; Trichelair, P.; des Combes, R. T.; and Tachet, R. 2019. Safe Policy Improvement with Baseline Bootstrapping. In *International Conference of Machine Learning (ICML)*.

[43] Le, L.; Kumaraswamy, R.; and White, M. 2017. Learning sparse representations in reinforcement learning with sparse coding. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2067–2073.

[44] Le Tallec, Y. 2007. *Robust, Risk-Sensitive, and Data-driven Control of Markov Decision Processes*. Ph.D. thesis, MIT.

[45] Le Tallec, Y. 2007. *Robust, risk-sensitive, and data-driven control of Markov decision processes*. Ph.D. thesis, Massachusetts Institute of Technology.

[46] Luedtke, J.; and Ahmed, S. 2008. A Sample Approximation Approach for Optimization with Probabilistic Constraints. *SIAM Journal on Optimization*, 19(2): 674–699.

[47] Mahadevan, S.; and Maggioni, M. 2006. Value function approximation with diffusion wavelets and Laplacian eigenfunctions. In *Advances in Neural Information Processing Systems (NIPS)*, 843–850.

[48] Mahadevan, S.; and Maggioni, M. 2007. Proto-value functions: A Laplacian framework for learning representation and control in Markov decision processes. *Journal of Machine Learning Research*, 8: 2169–2231.

[49] Maillard, O. A.; Mann, T. A.; and Mannor, S. 2014. "How hard is my MDP?" The distribution-norm to the rescue. In *Advances in Neural Information Processing Systems*, 1835–1843.

[50] Mannor, S.; Mebel, O.; and Xu, H. 2016. Robust MDPs with k-rectangular uncertainty. *Mathematics of Operations Research*, 41(4): 1484–1509.

[51] Mannor, S.; Simester, D.; Sun, P.; and Tsitsiklis, J. N. 2007. Bias and variance approximation in value function estimates. *Management Science*, 53(2): 308–322.

[52] Mannor, S.; Simester, D.; Sun, P.; and Tsitsiklis, J. N. 2007. Bias and Variance Approximation in Value Function Estimates. *Management Science*, 53(2): 308–322.

[53] Munos, R. 2007. Performance Bounds in Lp-norm for Approximate Value Iteration. *SIAM journal on control and optimization*, 46(2): 541–561.

[54] Murphy, K. P. 2012. *Machine learning: a probabilistic perspective*. MIT press.

[55] Nilim, A.; and El Ghaoui, L. 2005. Robust Control of Markov Decision Processes with Uncertain Transition Matrices. *Operations Research*, 53(5): 780–798.

[56] Nilim, A.; and Ghaoui, L. E. 2004. Robust solutions to Markov decision problems with uncertain transition matrices. *Operations Research*, 53(5): 780.

[57] Nilim, A.; and Ghaoui, L. E. 2005. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5): 780–798.

[58] Ong, H. Y. 2015. Value Function Approximation via Low Rank Models. *arXiv:1509.00061v1*.

[59] Parr, R.; Li, L.; Taylor, G.; Painter-Wakefield, C.; and Littman, M. L. 2008. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 752–759.

[60] Parr, R.; Painter-Wakefield, C.; Li, L.; and Littman, M. 2007. Analyzing Feature Generation for Value-function Approximation. In *International Conference on Machine Learning (ICML)*, 737–744. ISBN 978-1-59593-793-3.

[61] Petrik, M. 2007. An analysis of Laplacian methods for value function approximation in MDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 35, 2574–2579.

[62] Petrik, M.; Mohammad Ghavamzadeh; and Chow, Y. 2016. Safe Policy Improvement by Minimizing Robust Baseline Regret. In *Advances in Neural Information Processing Systems (NIPS)*.

[63] Petrik, M.; and Russel, R. H. 2019. Beyond Confidence Regions: Tight Bayesian Ambiguity Sets for Robust MDPs. *Advances in Neural Information Processing Systems*.

[64] Petrik, M.; and Subramanian, D. 2014. RAAM : The benefits of robustness in approximating aggregated MDPs in reinforcement learning. In *Neural Information Processing Systems (NIPS)*.

[65] Petrik, M.; and Subramanian, D. 2014. RAAM: The benefits of robustness in approximating aggregated MDPs in reinforcement learning. In *Advances in Neural Information Processing Systems*, 1979–1987.

[66] Petrik, M.; Taylor, G.; Parr, R.; and Zilberstein, S. 2010. Feature selection using regularization in approximate linear programs for Markov decision processes. In *International Conference on Machine Learning (ICML)*.

[67] Puterman, M. L. 2005. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc.

[68] Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *International Conference on World Wide Web (WWW)*, 811–820. ISBN 9781605587998.

[69] Russel, R. H.; Gu, T.; and Petrik, M. 2019. Robust Exploration with Tight Bayesian Plausibility Sets. *Multi-disciplinary conference on Reinforcement Learning and Decision Making (RLDM)*.

[70] Shapiro, A.; Dentcheva, D.; and Ruszczynski, A. 2021. *Lectures on stochastic programming: modeling and theory*. SIAM.

[71] Shechter, S. M.; Bailey, M. D.; Schaefer, A. J.; and Roberts, M. S. 2008. The optimal time to initiate HIV therapy under ordered health states. *Operations Research*, 56(1): 20–33.

[72] Song, Z.; Parr, R. E.; Liao, X.; and Carin, L. 2016. Linear Feature Encoding for Reinforcement Learning. In *Advances in Neural Information Processing Systems (NIPS)*, 4224–4232.

[73] Strehl, A. L.; and Littman, M. L. 2004. An empirical evaluation of interval estimation for markov decision processes. In *16th IEEE International Conference on Tools with Artificial Intelligence*, 128–135. IEEE.

[74] Strehl, A. L.; and Littman, M. L. 2008. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8): 1309–1331.

[75] Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

[76] Szepesvári, C. 2010. *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers.

[77] Taleghan, M. A.; Dietterich, T. G.; Crowley, M.; Hall, K.; and Albers, H. J. 2015. Pac optimal MDP planning with application to invasive species management. *Journal of Machine Learning Research*, 16.

[78] Tamar, A.; Mannor, S.; and Xu, H. 2014. Scaling Up Robust MDPs using Function Approximation. In *International Conference of Machine Learning (ICML)*.

[79] Tirinzoni, A.; Petrik, M.; Chen, X.; and Ziebart, B. 2018. Policy-Conditioned Uncertainty Sets for Robust Markov Decision Processes. In *Advances in Neural Information Processing Systems*, 8939–8949. Curran Associates, Inc.

[80] Weissman, T.; Ordentlich, E.; Seroussi, G.; Verdu, S.; and Weinberger, M. J. 2003. Inequalities for the L_1 deviation of the empirical distribution. *Hewlett-Packard Labs, Tech. Rep.*

[81] Wiesemann, W.; Kuhn, D.; and Rustem, B. 2013. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1): 153–183.

[82] Williams, R. J.; and Baird, L. C. 1993. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, College of Computer Science, Northeastern University.

[83] Xu, H.; and Mannor, S. 2009. Parametric regret in uncertain Markov decision processes. *Proceedings of the IEEE Conference on Decision and Control*, 3606–3613.

[84] Zipkin, P. H. 2000. *Foundations of inventory management*. McGraw-Hill Companies.

APPENDIX A

Technical Results and Proofs

A.1 Optimizing Percentile Criterion Using Robust MDPs

A.1.1 Proofs of Results in Section 3.1

Proof of Theorem 3.1.2. The result can be derived as:

$$\begin{aligned}\mathbb{P}_{\tilde{P}\sim f}\left[\hat{\rho} \le \rho(\hat{\pi}, \tilde{P})\right] &\overset{\text{(a)}}{=} \mathbb{P}_{\tilde{P}\sim f}\left[\rho(\hat{\pi}, \tilde{P}) \ge \max_{\pi\in\Pi} \min_{P\in\hat{\mathcal{P}}} \rho(\pi, P)\right] \\ &\overset{\text{(b)}}{=} \mathbb{P}_{\tilde{P}\sim f}\left[\rho(\hat{\pi}, \tilde{P}) \ge \min_{P\in\hat{\mathcal{P}}} \rho(\hat{\pi}, P)\right] \\ &\overset{\text{(c)}}{\ge} \mathbb{P}_{\tilde{P}\sim f}\left[\tilde{P} \in \hat{\mathcal{P}}\right] \overset{\text{(d)}}{\ge} 1-\delta\ .\end{aligned}$$

The equality (a) follows from the definition of $\hat{\rho}$, the inequality (b) follows from $\hat{\pi} \in \Pi$ and is optimal, (c) follows because $\rho(\hat{\pi}, \tilde{P}) \ge \min_{P\in\hat{\mathcal{P}}} \rho(\hat{\pi}, P)$ whenever $\tilde{P} \in \hat{\mathcal{P}}$, and (d) follows from the theorem's hypothesis. □

Proof of Theorem 3.1.3. Let $\hat{\mathcal{P}} = \mathcal{P}(\boldsymbol{w}, \psi)$ and let $\hat{\rho}$ and $\hat{\pi}$ be the optimal return and policy for $\hat{\mathcal{P}}$ respectively. We start by establishing the following bound:

$$\hat{\rho} \ge \max_{\pi\in\Pi} \rho(\pi, \tilde{P}) - \frac{\beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi)}{1-\gamma}\ ,$$

where

$$\beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) = \max_{s\in\mathcal{S}} \max_{a\in\mathcal{A}} \beta_{\hat{\boldsymbol{z}}}^{s,a}(\boldsymbol{w}, \psi)\ .$$

Let $\hat{\boldsymbol{v}} \in \mathbb{R}^S$ be the optimal robust value function that satisfied $\hat{\boldsymbol{v}} = \mathfrak{L}\hat{\boldsymbol{v}}$ for the ambiguity

set $\hat{\mathcal{P}} = \mathcal{P}(\boldsymbol{w}, \psi)$. We use $\hat{\mathcal{P}}$ as a shorthand for $\mathcal{P}(\boldsymbol{w}, \psi)$ throughout the proof. Recall that $\hat{\rho} = \boldsymbol{p}_0^\mathsf{T} \hat{\boldsymbol{v}}$. We also use $\mathfrak{T}_\pi^P$ to represent the Bellman evaluation operator for a policy $\pi \in \Pi$ and a transition function P defined for each $s \in \mathcal{S}$ as:

$$(\mathfrak{T}_\pi^P v)_s = P(s, \pi(s))^\mathsf{T} (\boldsymbol{r}_{s,a} + \gamma \cdot \boldsymbol{v}) \ .$$

It is well known that $\mathfrak{T}_\pi^P v$ is a contraction, is monotone, and has a unique fixed point. Let $\tilde{v}$ be the unique fixed point of $\mathfrak{T}_{\tilde{\pi}}^{\tilde{P}}$:

$$\tilde{\boldsymbol{v}} = \mathfrak{T}_{\tilde{\pi}}^{\tilde{P}} \tilde{\boldsymbol{v}} \ ,$$

where $\tilde{\pi} \in \arg\max_{\pi \in \Pi} \rho(\pi, \tilde{P})$. Note that it is well known that:

$$\boldsymbol{p}_0^\mathsf{T} \tilde{\boldsymbol{v}} = \rho(\tilde{\pi}, \tilde{P}) \ .$$

Now suppose that $\tilde{P} \in \hat{\mathcal{P}}$, which holds with probability $1 - \delta$ according to Assumption 3.1.1. Then it is easy to see that:

$$\boldsymbol{p}_0^\mathsf{T} \hat{\boldsymbol{v}} = \min_{P \in \hat{\mathcal{P}}} \rho(\pi, P) \leq \rho(\pi, \tilde{P}) \leq \boldsymbol{p}_0^\mathsf{T} \tilde{\boldsymbol{v}} \ .$$

Therefore:

$$0 \leq \boldsymbol{p}_0^\mathsf{T} \tilde{\boldsymbol{v}} - \boldsymbol{p}_0^\mathsf{T} \hat{\boldsymbol{v}} \leq \left\| \tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}} \right\|_\infty \ .$$

We are now ready to establish the probabilistic bound which is based on bounding the Bellman residual as follows:

$$\begin{aligned}
(\mathfrak{T}_{\tilde{\pi}}^{\tilde{P}} \hat{\boldsymbol{v}} - \hat{\boldsymbol{v}})_s &\overset{\text{(a)}}{=} (\mathfrak{T}_{\tilde{\pi}}^{\tilde{P}} \hat{\boldsymbol{v}} - \mathfrak{L} \hat{\boldsymbol{v}})_s \overset{\text{(def)}}{=} \tilde{P}(s, \tilde{\pi}(a))^\mathsf{T} \hat{\boldsymbol{z}}_{s,\tilde{\pi}(s)} - \min_{P \in \hat{\mathcal{P}}} P(s, \hat{\pi}(a))^\mathsf{T} \hat{\boldsymbol{z}}_{s,\hat{\pi}(a)} \\
&\overset{\text{(b)}}{\leq} \tilde{P}(s, \tilde{\pi}(a))^\mathsf{T} \hat{\boldsymbol{z}}_{s,\tilde{\pi}(s)} - \min_{P \in \hat{\mathcal{P}}} P(s, \tilde{\pi}(a))^\mathsf{T} \hat{\boldsymbol{z}}_{s,\tilde{\pi}(a)} \\
&\leq \max_{a \in \mathcal{A}} \left(\tilde{P}(s, a)^\mathsf{T} \hat{\boldsymbol{z}}_{s,a} - \min_{P \in \hat{\mathcal{P}}} P(s, a)^\mathsf{T} \hat{\boldsymbol{z}}_{s,a} \right)
\end{aligned}$$

$$
\begin{aligned}
&\overset{\text{(c)}}{\leq} \max_{a \in \mathcal{A}} \left(\max_{P \in \hat{\mathcal{P}}} P(s,a)^{\mathsf{T}} \hat{\boldsymbol{z}}_{s,a} - \min_{P \in \hat{\mathcal{P}}} P(s,a)^{\mathsf{T}} \hat{\boldsymbol{z}}_{s,a} \right) \\
&\leq \max_{a \in \mathcal{A}} \beta^{s,a}_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \ .
\end{aligned}
$$

(a) follows from $\hat{\boldsymbol{v}}$ being the fixed point of $\mathfrak{L}$, (b) follows from the optimality of $\hat{\pi}$: $\hat{\pi}(s) \in \arg\max_{a \in \mathcal{A}} \min_{\boldsymbol{p} \in \hat{\mathcal{P}}_{s,a}} \boldsymbol{p}^{\mathsf{T}} \boldsymbol{z}_{s,a}$, and (c) follows from $\tilde{P} \in \hat{\mathcal{P}}$. The rest follows by algebraic manipulation. Applying the inequality above to all states, we get:

$$
\mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} - \hat{\boldsymbol{v}} \leq \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \cdot \mathbf{1} \ . \tag{A.1}
$$

We can now use the standard dynamic programming bounding technique to bound $\|\tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}}\|_\infty$ as follows:

$$
\mathbf{0} \overset{\text{(a)}}{\leq} \tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}} \overset{\text{(b)}}{=} \tilde{\boldsymbol{v}} - \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} + \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} - \hat{\boldsymbol{v}} \overset{\text{(A.1)}}{\leq} \tilde{\boldsymbol{v}} - \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} + \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \cdot \mathbf{1} \overset{\text{(c)}}{\leq} \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \tilde{\boldsymbol{v}} - \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} + \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \cdot \mathbf{1} \ .
$$

We have (a) because $\hat{\boldsymbol{v}} \leq \tilde{\boldsymbol{v}}$ because $\mathfrak{L}\tilde{\boldsymbol{v}} \leq \tilde{\boldsymbol{v}}$ and thus $\tilde{\boldsymbol{v}} \geq \mathfrak{L}\mathfrak{L}\tilde{\boldsymbol{v}} \geq \ldots \geq \mathfrak{L} \ldots \mathfrak{L}\tilde{\boldsymbol{v}} \geq \hat{\boldsymbol{v}}$ because $\hat{\boldsymbol{v}}$ is the fixed point of $\mathfrak{L}$ and $\mathfrak{L}$ is monotone. (b) we add $\mathbf{0}$, (c) $\tilde{\boldsymbol{v}}$ is the fixed point of $\mathfrak{T}^{\tilde{P}}_{\tilde{\pi}}$.

Next, apply L_∞ norm to all sides, which is possible because the values are non-negative:

$$
\begin{aligned}
\|\tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}}\|_\infty &\leq \left\| \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \tilde{\boldsymbol{v}} - \mathfrak{T}^{\tilde{P}}_{\tilde{\pi}} \hat{\boldsymbol{v}} + \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \cdot \mathbf{1} \right\|_\infty \\
\|\tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}}\|_\infty &\leq \gamma \cdot \|\tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}}\|_\infty + \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi) \\
\|\tilde{\boldsymbol{v}} - \hat{\boldsymbol{v}}\|_\infty &\leq \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi)/(1-\gamma) \ .
\end{aligned}
$$

The first step follows by triangle inequality, and the second step follows from $\mathfrak{T}^{\tilde{P}}_{\tilde{\pi}}$ being a γ contraction in the L_∞ norm.

To prove the bound on $y^\star$ and $\hat{v}$, we show that $y^\star \leq \zeta$ where $\zeta = \hat{\rho} + \beta_{\hat{\boldsymbol{z}}}(\boldsymbol{w}, \psi)/(1-\gamma)$.

Suppose to the contrary that $y^\star > \zeta$. Realize that $y^\star$ optimal in (2.6) must satisfy:

$$\mathbb{P}_{\tilde{P}\sim f}\left[\max_{\pi\in\Pi}\rho(\pi,\tilde{P}) \geq y^\star\right] \geq 1-\delta \ , \tag{A.2}$$

because $\max_{\pi\in\Pi}\rho(\pi,\tilde{P}) \geq \rho(\pi^\star,\tilde{P})$ for $\pi^\star$ optimal in (2.6). Recall also that from the first part of the theorem:

$$\mathbb{P}_{\tilde{P}\sim f}\left[\max_{\pi\in\Pi}\rho(\pi,\tilde{P}) \geq \zeta\right] \leq \delta \ . \tag{A.3}$$

We now derive a contradiction as follows:

$$\delta \overset{\text{(A.3)}}{\geq} \mathbb{P}_{\tilde{P}\sim f}\left[\max_{\pi\in\Pi}\rho(\pi,\tilde{P}) \geq \zeta\right] \overset{\text{(a)}}{\geq} \mathbb{P}_{\tilde{P}\sim f}\left[\max_{\pi\in\Pi}\rho(\pi,\tilde{P}) \geq y^\star\right] \overset{\text{(A.2)}}{\geq} 1-\delta \ .$$

Here (a) follows from the assumption $y^\star > \zeta$. Then $\delta \geq 1-\delta$ is a contradiction with $\delta < 0.5$. Finally, $0 \leq y^\star - \hat{\rho}$ follows directly from the optimality of $y^\star$ and Theorem 3.1.2, which proves the theorem. □

A.1.2 Proof of Results in Section 3.2

Proof of Lemma 3.2.1. We omit the s,a subscripts to simplify the notation. By relaxing the non-negativity constraints on $\boldsymbol{p}$ and using substitution $\boldsymbol{q}_1 = \boldsymbol{p}_1 - \bar{\boldsymbol{p}}$ and $\boldsymbol{q}_2 = \boldsymbol{p}_2 - \bar{\boldsymbol{p}}$, we get the following upper bound:

$$\begin{aligned}
\beta^{s,a}_{\boldsymbol{z}}(\boldsymbol{w},\psi) &= \max_{\boldsymbol{p}_1,\boldsymbol{p}_2}\left\{(\boldsymbol{p}_1-\boldsymbol{p}_2)^\mathsf{T}\boldsymbol{z} \mid \boldsymbol{p}_1,\boldsymbol{p}_2 \in \mathcal{P}_{s,a}(\boldsymbol{w},\psi)\right\} \\
&= \max_{\boldsymbol{p}_1,\boldsymbol{p}_2}\left\{(\boldsymbol{p}_1-\boldsymbol{p}_2)^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{p}_1-\bar{\boldsymbol{p}}\|_{\boldsymbol{w}} \leq \psi,\ \|\boldsymbol{p}_2-\bar{\boldsymbol{p}}\|_{\boldsymbol{w}} \leq \psi,\ \boldsymbol{p}_1 \in \Delta^S,\ \boldsymbol{p}_2 \in \Delta^S\right\} \\
&\leq \max_{\boldsymbol{p}_1,\boldsymbol{p}_2\in\mathbb{R}^S}\left\{(\boldsymbol{p}_1-\boldsymbol{p}_2)^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{p}_1-\bar{\boldsymbol{p}}\|_{\boldsymbol{w}} \leq \psi,\ \|\boldsymbol{p}_2-\bar{\boldsymbol{p}}\|_{\boldsymbol{w}} \leq \psi,\ \mathbf{1}^\mathsf{T}\boldsymbol{p}_1 = 1,\ \mathbf{1}^\mathsf{T}\boldsymbol{p}_2 = 1\right\} \\
&= \max_{\boldsymbol{q}_1,\boldsymbol{q}_2\in\mathbb{R}^S}\left\{(\boldsymbol{q}_1-\boldsymbol{q}_2)^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{q}_1\|_{\boldsymbol{w}} \leq \psi,\ \|\boldsymbol{q}_2\|_{\boldsymbol{w}} \leq \psi,\ \mathbf{1}^\mathsf{T}\boldsymbol{q}_1 = 0,\ \mathbf{1}^\mathsf{T}\boldsymbol{q}_2 = 0\right\} \\
&= \max_{\boldsymbol{q}_1\in\mathbb{R}^S}\left\{\boldsymbol{q}_1^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{q}_1\|_{\boldsymbol{w}} \leq \psi,\ \mathbf{1}^\mathsf{T}\boldsymbol{q}_1 = 0\right\} + \max_{\boldsymbol{q}_2\in\mathbb{R}^S}\left\{\boldsymbol{q}_2^\mathsf{T}(-\boldsymbol{z}) \mid \|\boldsymbol{q}_2\|_{\boldsymbol{w}} \leq \psi,\ \mathbf{1}^\mathsf{T}\boldsymbol{q}_2 = 0\right\} \ .
\end{aligned}$$

The last equality follows because the the optimization problems over $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are independent. From the absolute homogeneity of the $\|\cdot\|_{\boldsymbol{w}}$ we have that:

$$\max_{\boldsymbol{q}_2 \in \mathbb{R}^S} \left\{ \boldsymbol{q}_2^\mathsf{T}(-\boldsymbol{z}) \mid \|\boldsymbol{q}_2\|_{\boldsymbol{w}} \leq \psi,\, \mathbf{1}^\mathsf{T}\boldsymbol{q}_2 = 0 \right\} \;=\; \max_{\boldsymbol{q}_2 \in \mathbb{R}^S} \left\{ \boldsymbol{q}_2^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{q}_2\|_{\boldsymbol{w}} \leq \psi,\, \mathbf{1}^\mathsf{T}\boldsymbol{q}_2 = 0 \right\} ,$$

and therefore:

$$\beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi) \leq 2 \cdot \max_{\boldsymbol{q} \in \mathbb{R}^S} \left\{ \boldsymbol{q}^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{q}\|_{\boldsymbol{w}} \leq \psi,\, \mathbf{1}^\mathsf{T}\boldsymbol{q} = 0 \right\} .$$

Substituting $\boldsymbol{q} = \boldsymbol{p} - \bar{\boldsymbol{p}}$ we get:

$$\beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi) \leq 2 \cdot \max_{\boldsymbol{p} \in \mathbb{R}^S} \left\{ \boldsymbol{p}^\mathsf{T}\boldsymbol{z} \mid \|\boldsymbol{p} - \bar{\boldsymbol{p}}\|_{\boldsymbol{w}} \leq \psi,\, \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1 \right\} - 2 \cdot \boldsymbol{z}^\mathsf{T}\bar{\boldsymbol{p}} \; . \tag{A.4}$$

We can reformulate the optimization problem on the right-hand side of (A.4), again using variable substitution $\boldsymbol{q} = \boldsymbol{p} - \bar{\boldsymbol{p}}$:

$$\begin{aligned} \max_{\boldsymbol{q} \in \mathbb{R}^S} \quad & 2 \cdot (\boldsymbol{q} + \bar{\boldsymbol{p}})^\mathsf{T}\boldsymbol{z} - 2 \cdot \boldsymbol{z}^\mathsf{T}\bar{\boldsymbol{p}} \\ \text{s.t.} \quad & \|\boldsymbol{q}\|_{\boldsymbol{w}} \leq \psi \\ & \mathbf{1}^\mathsf{T}(\boldsymbol{q} + \bar{\boldsymbol{p}}) = 1 \implies \mathbf{1}^\mathsf{T}\boldsymbol{q} = 0 \; . \end{aligned}$$

Canceling out $\bar{\boldsymbol{p}}^\mathsf{T}\boldsymbol{z}$, we continue with:

$$\begin{aligned} 2 \cdot \max_{\boldsymbol{q} \in \mathbb{R}^S} \quad & \boldsymbol{q}^\mathsf{T}\boldsymbol{z} \\ \text{s.t.} \quad & \|\boldsymbol{q}\|_{\boldsymbol{w}} \leq \psi \\ & \mathbf{1}^\mathsf{T}\boldsymbol{q} = 0 \; . \end{aligned}$$

By applying the method of Lagrange multipliers, we obtain:

$$\begin{aligned} \min_{\lambda \in \mathbb{R}} \max_{\boldsymbol{q} \in \mathbb{R}^S} \quad & \boldsymbol{q}^\mathsf{T}\boldsymbol{z} - \lambda \cdot (\boldsymbol{q}^\mathsf{T}\mathbf{1}) = \boldsymbol{q}^\mathsf{T}(\boldsymbol{z} - \lambda \cdot \mathbf{1}) \\ \text{s.t.} \quad & \|\boldsymbol{q}\|_{\boldsymbol{w}} \leq \psi \; . \end{aligned}$$

Letting $\boldsymbol{x} = \frac{\boldsymbol{q}}{\psi}$, we get:

$$\begin{aligned} \min_{\lambda \in \mathbb{R}} \max_{\boldsymbol{x} \in \mathbb{R}^S} \quad & \psi \cdot \boldsymbol{x}^\mathsf{T}(\boldsymbol{z} - \lambda \cdot \mathbf{1}) \\ \text{s.t.} \quad & \|\boldsymbol{x}\|_{\boldsymbol{w}} \leq 1 \ . \end{aligned}$$

Given the definition of the *dual norm*, $\|\boldsymbol{z}\|_\star = \sup\{\boldsymbol{z}^\intercal \boldsymbol{x} \mid \|\boldsymbol{x}\| \leq 1\}$, we have:

$$\begin{aligned} \beta_{\boldsymbol{z}}^{s,a}(\boldsymbol{w}, \psi) &\leq 2 \cdot \min_{\lambda \in \mathbb{R}} \ \psi \cdot \|\boldsymbol{z} - \lambda \cdot \mathbf{1}\|_\star \\ &\leq 2 \cdot \psi \cdot \|\boldsymbol{z} - \lambda \cdot \mathbf{1}\|_\star \ . \end{aligned}$$

□

Proof of Lemma 3.2.2. Assume we are given a set of positive weights $\boldsymbol{w} \in \mathbb{R}^n_{++}$ for the following weighted L_1 optimization problem:

$$\begin{aligned} \max_{\boldsymbol{x} \in \mathbb{R}^S} \quad & \boldsymbol{z}^\mathsf{T} \boldsymbol{x} \\ \text{s.t.} \quad & \|\boldsymbol{x}\|_{1,\boldsymbol{w}} \leq 1 \ . \end{aligned} \tag{A.5}$$

We have:

$$\begin{aligned} \boldsymbol{x}^\mathsf{T} \boldsymbol{z} = \sum_{i=1}^n x_i \cdot z_i &\leq \sum_{i=1}^n |x_i \cdot z_i| \\ &\overset{(a)}{\leq} \sum_{i=1}^n |x_i| \cdot |z_i| = \sum_{i=1}^n w_i \cdot |x_i| \cdot \frac{1}{w_i} \cdot |z_i| \\ &\leq \max_{i=1,\ldots,n} \left\{ \frac{1}{w_i} \cdot |z_i| \right\} \cdot \sum_{i=1}^n w_i |x_i| = \max_{i=1,\ldots,n} \left\{ \frac{1}{w_i} \cdot |z_i| \right\} \cdot \|\boldsymbol{x}\|_{1,\boldsymbol{w}} \\ &\overset{(b)}{\leq} \max_{i=1,\ldots,n} \left\{ \frac{1}{w_i} |z_i| \right\} = \|\boldsymbol{z}\|_{\infty,\frac{1}{\boldsymbol{w}}} \ . \end{aligned}$$

Here, (a) follows from the Cauchy-Schwarz inequality, and (b) follows from the constraint $\|\boldsymbol{x}\|_{1,\boldsymbol{w}} \leq 1$ of (A.5). □

Proof of Proposition 3.2.3. We use the notation $1/\boldsymbol{w}$ to denote an elementwise inverse of $\boldsymbol{w}$ such that $(1/\boldsymbol{w})_i = 1/w_i, i \in \mathcal{S}$. Note that for weighted L_1-constrained sets $q = \infty$, and for

the L_∞-constrained sets $q = 1$. The value $\bar{\lambda}$ in (3.6) is fixed ahead of time and does not change with $\boldsymbol{w}$. Recall that the constraint $\sum_{i=1}^{S} w_i^2 = 1$ serves to normalize $\boldsymbol{w}$ in order to preserve the desired robustness guarantees with *the same* ψ. This is because scaling both $\boldsymbol{w}$ and ψ simultaneously by an identical factor leaves the ambiguity set unchanged. We adopt the constraint from an approximation of the guarantee by linearization of the upper bound using Jensen's inequality. Next, omitting terms that are constant with respect to $\boldsymbol{w}$ simplifies the optimization to:

$$\boldsymbol{w}^\star \in \underset{\boldsymbol{w}\in\mathbb{R}^S_{++}}{\operatorname{argmin}} \left\{ \left\| \boldsymbol{z} - \bar{\lambda}\mathbf{1} \right\|_{q,\frac{1}{\boldsymbol{w}}} \;:\; \sum_{i=1}^{S} w_i^2 = 1 \right\} . \tag{A.6}$$

For $q = \infty$, the nonlinear optimization problem in (A.6) is convex and can be solved *analytically*. Let $b_i = \left| z_i - \bar{\lambda} \right|$ for $i = 1, \ldots, S$, then (A.6) turns to:

$$\min_{t,\boldsymbol{w}\in\mathbb{R}^S_{++}} \left\{ t \;:\; t \geq b_i/w_i, \sum_{i=1}^{S} w_i^2 = 1 \right\} . \tag{A.7}$$

The constraints $\boldsymbol{w} > \mathbf{0}$ cannot be active since otherwise $1/w_i$ results in undefined division by zero and can be safely ignored. Then, the convex optimization problem in Equation (A.7) has a linear objective, $S+1$ variables ($\boldsymbol{w}$'s and t), and $S+1$ constraints. All constraints are active, therefore, in the optimal solution $\boldsymbol{w}^\star$ [7] which must satisfy:

$$w_i^\star = {}^{b_i}\!/\sqrt{\textstyle\sum_{j=1}^{S} b_j^2} . \tag{A.8}$$

Since $\sum_i w_i^2 = 1$ implies $\sum_i b_i^2/t^2 = 1$, we conclude that $t = \sqrt{\sum_i b_i^2}$. For $q = 1$, the equivalent optimization of (A.7) becomes:

$$\min_{\boldsymbol{w}>\mathbf{0}} \left\{ \sum_{i=1}^{S} b_i/w_i \;:\; \sum_{i=1}^{S} w_i^2 = 1 \right\} . \tag{A.9}$$

Again, the inequality constraints on weights $\boldsymbol{w} > 0$ can be relaxed. Using the necessary

optimality conditions (and a Lagrange multiplier), one solution for the optimal weights $\boldsymbol{w}$ are:

$$w_i^\star = {b_i^{1/3}}/{\sqrt{\sum_{j=1}^S b_j^{2/3}}} \ . \tag{A.10}$$

□

A.1.3 Proof of Results in Section 3.3

Proof of Proposition 3.3.2. The algorithm is an instance of the Sample Average Approximation (SAA) scheme. The result, therefore, is a direct consequence of Theorem 4.2 in [63] and Theorem 5.3 in [70]. □

A.2 Weighted Frequentist Confidence Intervals for Robust MDPs

A.2.1 Proof of Results in Section 4.1

We need several auxiliary results before proving the results.

Theorem A.2.1 (Weighted L_∞ error bound (Hoeffding))**.** *Suppose that $\bar{\boldsymbol{p}}_{s,a}$ is the empirical estimate of the transition probability obtained from $n_{s,a}$ samples for some $s \in \mathcal{S}$ and $a \in \mathcal{A}$. Then:*

$$\mathbb{P}_{\bar{\boldsymbol{p}}_{s,a}}\left[\left\|\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a}\right\|_{\infty,\boldsymbol{w}} \geq \psi_{s,a}\right] \leq 2\sum_{i=1}^S \exp\left(-2\frac{\psi_{s,a}^2 n_{s,a}}{w_i^2}\right) \ . \tag{A.11}$$

Proof. First, we will express the weighted L_∞ distance between two distributions $\bar{\boldsymbol{p}}$ and $\boldsymbol{p}^\star$ in terms of an optimization problem. Let $\mathbf{1}_i \in \mathbb{R}^{\mathcal{S}}$ be the indicator vector for an index $i \in \mathcal{S}$:

$$\begin{aligned}\left\|\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a}\right\|_{\infty,\boldsymbol{w}} &= \max_{\boldsymbol{z}} \left\{\boldsymbol{z}^\mathsf{T} W(\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a}) : \|\boldsymbol{z}\|_1 \leq 1\right\} \\ &= \max_{i\in\mathcal{S}} \Big\{\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a}), -\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a} - \boldsymbol{p}^\star_{s,a})\Big\} \ .\end{aligned}$$

Here, weights are on the diagonal entries of W. Using the expression above, we can bound

the probability in the lemma as follows:

$$\begin{aligned}\mathbb{P}\left[\left\|\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a}\right\|_{\infty,\boldsymbol{w}}\geq\psi\right] &= \mathbb{P}\left[\max_{i\in\mathcal{S}}\left\{\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a}),-\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a})\right\}\geq\psi_{s,a}\right]\\ &\overset{(a)}{\leq} S\max_{i\in\mathcal{S}}\mathbb{P}\left[\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a})\geq\psi_{s,a}\right]+S\max_{i\in\mathcal{S}}\mathbb{P}\left[-\mathbf{1}_i W(\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a})\geq\psi_{s,a}\right]\\ &\overset{(b)}{\leq} 2\sum_{i=1}^{S}\exp\left(-2\frac{\psi^2_{s,a}n}{w_i^2}\right)\ .\end{aligned}$$

Here, (a) follows from union bound, and (b) follows from Hoeffding's inequality since $\mathbf{1}_i^{\mathsf{T}}\bar{\boldsymbol{p}}\in[0,1]$ for any $i\in\mathcal{S}$ and its mean is $\mathbf{1}_i^{\mathsf{T}}\boldsymbol{p}^{\star}$. □

Now we describe a proof of error bound in (A.12) on the weighted L_1 distance between the estimated transition probabilities $\bar{\boldsymbol{p}}$ and the true one $\boldsymbol{p}^{\star}$ over each state $s\in\mathcal{S}=\{1,\ldots,S\}$ and action $a\in\mathcal{A}=\{1,\ldots,A\}$. The proof is an extension to Lemma C.1 (L_1 error bound) in [63].

Theorem A.2.2 (Weighted L_1 error bound (Hoeffding))**.** *Suppose that $\bar{\boldsymbol{p}}_{s,a}$ is the empirical estimate of the transition probability obtained from $n_{s,a}$ samples for some $s\in\mathcal{S}$ and $a\in\mathcal{A}$. If the weights $\boldsymbol{w}\in\mathbb{R}^S_{++}$ are sorted in a non-increasing order $w_i\geq w_{i+1}$, then:*

$$\mathbb{P}_{\bar{\boldsymbol{p}}_{s,a}}\left[\left\|\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a}\right\|_{1,\boldsymbol{w}}\geq\psi_{s,a}\right]\leq 2\sum_{i=1}^{S-1}2^{S-i}\exp\left(-\frac{\psi^2_{s,a}n_{s,a}}{2w_i^2}\right)\ . \tag{A.12}$$

Proof. Let $\boldsymbol{q}_{s,a}=\bar{\boldsymbol{p}}_{s,a}-\boldsymbol{p}^{\star}_{s,a}$. To shorten notation in the proof, we omit the s,a indexes when there is no ambiguity. We assume that all weights are non-negative. First, we will express the $L_{1,\boldsymbol{w}}$ norm of $\boldsymbol{q}$ in terms of an optimization problem. It is worth noting that $\mathbf{1}^{\mathsf{T}}\boldsymbol{q}=0$. Let $\mathbf{1}_{\mathcal{Q}_1},\mathbf{1}_{\mathcal{Q}_2}\in\mathbb{R}^{\mathcal{S}}$ be the indicator vectors for some subsets $\mathcal{Q}_1,\mathcal{Q}_2\subset\mathcal{S}$ where $\mathcal{Q}_2=\mathcal{S}\setminus\mathcal{Q}_1$. According to Lemma 3.2.2 we have:

$$\|\boldsymbol{q}\|_{1,w}=\max_{\boldsymbol{z}}\left\{\boldsymbol{z}^{\mathsf{T}}\boldsymbol{q}:\|\boldsymbol{z}\|_{\infty,\frac{1}{w}}\leq 1\right\}$$

$$= \max_{\mathcal{Q}_1, \mathcal{Q}_2 \in 2^{\mathcal{S}}} \left\{ \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \boldsymbol{q} + \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W(-\boldsymbol{q}) : \mathcal{Q}_2 = \mathcal{S} \setminus \mathcal{Q}_1 \right\} \ .$$

Here weights are on the diagonal entries of W. Using the expression above, we can bound the probability as follows:

$$\begin{aligned}
\mathbb{P}\left[\max_{\mathcal{Q}_1, \mathcal{Q}_2 \in 2^{\mathcal{S}}} \left\{ \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \boldsymbol{q} + \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W(-\boldsymbol{q}) \right\} \geq \psi \right] &\overset{(a)}{\leq} \mathbb{P}\left[\max_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \left\{ \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \boldsymbol{q} \right\} \geq \frac{\psi}{2} \right] + \mathbb{P}\left[\max_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \left\{ \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W(-\boldsymbol{q}) \right\} \geq \frac{\psi}{2} \right] \\
&\leq \sum_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \mathbb{P}\left[\mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \boldsymbol{q} \geq \frac{\psi}{2} \right] + \sum_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \mathbb{P}\left[\mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W(-\boldsymbol{q}) \geq \frac{\psi}{2} \right] \\
&= \sum_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \mathbb{P}\left[\mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W(\bar{\boldsymbol{p}} - \boldsymbol{p}^\star) \geq \frac{\psi}{2} \right] + \sum_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \mathbb{P}\left[\mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W(-\bar{\boldsymbol{p}} + \boldsymbol{p}^\star) \geq \frac{\psi}{2} \right] \\
&\overset{(b)}{\leq} \sum_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \exp\left(-\frac{\psi^2 n}{2 \left\| \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \right\|_\infty^2} \right) + \sum_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \exp\left(-\frac{\psi^2 n}{2 \left\| \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W \right\|_\infty^2} \right) \\
&\overset{(c)}{=} 2 \sum_{i=1}^{S-1} 2^{S-i} \exp\left(-\frac{\psi^2 n}{2 w_i^2} \right) \ .
\end{aligned}$$

(a) follows from union bound, and (b) follows from Hoeffding's inequality. (c) follows by $\mathcal{Q}_1^c = \mathcal{Q}_2$ and sorting weights $\boldsymbol{w} = \{w_1, \ldots, w_n\}$ in non-increasing order. □

Proof of Theorem 4.1.2. The result follows from Lemma A.1 in [63] and Theorem A.2.1 by algebraic manipulation. □

Proof of Theorem 4.1.3. The result follows from Lemma A.1 in [63] and Theorem A.2.2 by algebraic manipulation. □

A.2.2 Bernstein Concentration Inequalities

Proof of Theorem 4.1.4. The proof is similar to the proof of Theorem A.2.2 until section b. The proof continues from section (b) as follows:

$$\begin{aligned}
&\overset{(b)}{\leq} \sum_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \exp\left(-\frac{3\psi^2 n}{24\sigma^2 + 4c\psi} \right) + \sum_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \exp\left(-\frac{3\psi^2 n}{24\sigma^2 + 4c\psi} \right) \\
&\overset{(c)}{\leq} \sum_{\mathcal{Q}_1 \in 2^{\mathcal{S}}} \exp\left(-\frac{3\psi^2 n}{6 \left\| \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \right\|_\infty^2 + 4\psi \left\| \mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W \right\|_\infty} \right) + \sum_{\mathcal{Q}_2 \in 2^{\mathcal{S}}} \exp\left(-\frac{3\psi^2 n}{6 \left\| \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W \right\|_\infty^2 + 4\psi \left\| \mathbf{1}_{\mathcal{Q}_2}^{\mathsf{T}} W \right\|_\infty} \right)
\end{aligned}$$

$$\overset{(d)}{=} 2\sum_{i=1}^{S-1} 2^{S-i} \exp\left(-\frac{3\psi^2 n}{6w_i^2 + 4\psi w_i}\right) .$$

Here (b) follows from Bernstein's inequality where σ^2 is the mean of variance of random variables, and c is their upper bound [15]. In the weighted case, with conservative estimate of variance $\sigma^2 = \left\|\mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W\right\|_\infty^2/4$, and $c = \left\|\mathbf{1}_{\mathcal{Q}_1}^{\mathsf{T}} W\right\|_\infty$, because the random variables are drawn from *Bernoulli* distribution with the maximum possible variance of $1/4$. (d) follows by sorting weights $\boldsymbol{w}$ in non-increasing order. □

A.3 Fast Algorithms for L_∞-constrained S-rectangular Robust MDPs

A.3.1 Proofs of Results in Section 5.1

Proof of Lemma 5.1.2. The functions $q(\xi)$ is convex due to the LP formulation of Equation (5.3); see [17]. □

Proof of Lemma 5.1.3. (i) The statement follows from the results in sections (ii)-(v) of this lemma.

(ii) If the intersection of any pair of $\mathcal{R}_B$, $\mathcal{D}_B$, and $\mathcal{N}_B$ is not an empty set, there exist a component i that satisfies two or more constraints in Table 5.2. In such a scenario, the basis B contains linearly dependent constraints that violate the definition of a basis. $\mathcal{T}_B = \{1, \ldots, S\} \setminus \mathcal{R}_B \setminus \mathcal{D}_B \setminus \mathcal{N}_B$ by definition does not intersect with other sets.

(iii) and (iv) By definition, B in $\mathcal{B}$ implies that the constraint $\mathbf{1}^\mathsf{T}\boldsymbol{p} = 1$ is in B; thus, one needs $S-1$ additional constraints selected from Table 5.1 to form a basis. However, for every $i \in [S]$, at most one of the three constraints in Table 5.1 should be selected, otherwise the constraints would not be linearly independent. Therefore, it implies that there exists exactly one $j \in [S]$ such that none of the three constraints in Table 5.1 is selected in B, and so $j \in \mathcal{T}_B$. For every $i \in [S]\backslash\{j\}$, $i \in \mathcal{R}_B \cup \mathcal{D}_B \cup \mathcal{N}_B$.

(v) We prove this results via contradiction with the following cases. Firstly, suppose there exist a basis B', in which $l < \tau \in \mathcal{T}_{B'}$ where $l \in \mathcal{D}_{B'}$, then we construct another basis B, where $\mathcal{R}_B = \mathcal{R}_{B'} \cup \{l\}$, $\mathcal{D}_B = \mathcal{D}_{B'}\backslash\{l\}$, $\mathcal{N}_B = \mathcal{N}_{B'}$, and $\mathcal{T}_B = \mathcal{T}_{B'}$. By Lemma 5.1.4, we have:

$$\begin{aligned}
\dot{q}_{B'} &= \sum_{i\in\mathcal{R}_{B'}} z_i - \sum_{j\in\mathcal{D}_{B'}} z_j + \left(|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}|\right) z_\tau, \\
\dot{q}_{B} &= \sum_{i\in\mathcal{R}_{B'}} z_i - \sum_{j\in\mathcal{D}_{B'}} z_j + 2z_l + \left(|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}| - 2\right) z_\tau
\end{aligned}$$

and thus $\dot{q}_B - \dot{q}_{B'} = 2(z_l - z_\tau) \leq 0$ as $z_l \leq z_\tau$. The above construction of B also ensure that $p_B(\xi)$ is feasible in a neighborhood of ξ, as long as $p_{B'}(\xi)$ is feasible in a neighborhood of ξ.

Furthermore, suppose there exist a basis B', in which $l < \tau \in \mathcal{T}_{B'}$ where $l \in \mathcal{N}_{B'}$, then we construct another basis B, where $\mathcal{R}_B = \mathcal{R}_{B'} \cup \{l\}$, $\mathcal{D}_B = \mathcal{D}_{B'}$, $\mathcal{N}_B = \mathcal{N}_{B'} \backslash \{l\}$, and $\mathcal{T}_B = \mathcal{T}_{B'}$. By Lemma 5.1.4, we have:

$$\begin{aligned} \dot{q}_{B'} &= \sum_{i \in \mathcal{R}_{B'}} z_i - \sum_{j \in \mathcal{D}_{B'}} z_j + (|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}|)\, z_\tau, \\ \dot{q}_B &= \sum_{i \in \mathcal{R}_{B'}} z_i - \sum_{j \in \mathcal{D}_{B'}} z_j + z_l + (|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}| - 1)\, z_\tau \end{aligned}$$

and thus $\dot{q}_B - \dot{q}_{B'} = z_l - z_\tau \leq 0$ as $z_l \leq z_\tau$. The above construction of B also ensure that $p_B(\xi)$ is feasible in a neighborhood of ξ, as long as $p_{B'}(\xi)$ is feasible in a neighborhood of ξ.

Now we prove the second part of this result.

Suppose there exist a basis B', in which $m > \tau \in \mathcal{T}_{B'}$ where $m \in \mathcal{R}_{B'}$, then we construct another basis B, where $\mathcal{R}_B = \mathcal{R}_{B'} \backslash \{m\}$, $\mathcal{D}_B = \mathcal{D}_{B'} \cup \{m\}$, $\mathcal{N}_B = \mathcal{N}_{B'}$, and $\mathcal{T}_B = \mathcal{T}_{B'}$. By Lemma 5.1.4, we have:

$$\begin{aligned} \dot{q}_{B'} &= \sum_{i \in \mathcal{R}_{B'}} z_i - \sum_{j \in \mathcal{D}_{B'}} z_j + (|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}|)\, z_\tau, \\ \dot{q}_B &= \sum_{i \in \mathcal{R}_{B'}} z_i - \sum_{j \in \mathcal{D}_{B'}} z_j - 2z_m + (|\mathcal{D}_{B'}| - |\mathcal{R}_{B'}| + 2)\, z_\tau \end{aligned}$$

and thus $\dot{q}_B - \dot{q}_{B'} = 2(z_\tau - z_m) \leq 0$ as $z_m \geq z_\tau$. The above construction of B also ensure that $p_B(\xi)$ is feasible in a neighborhood of ξ, as long as $p_{B'}(\xi)$ is feasible in a neighborhood of ξ.

□

Proof of Lemma 5.1.4. Note that if $k \in \mathcal{N}_B$ implies $(\boldsymbol{p}_B(\xi))_k = 0$ for every ξ therefore $\dot{p}_k = 0$. For all components $i \in \mathcal{R}_B$ we have $p_i - \bar{p}_i = \xi$. By taking the derivative with respect to ξ we have $\dot{p}_i = 1$. Similarly, for all $j \in \mathcal{D}_B$ we have $\bar{p}_j - p_j = \xi$. Taking the derivative leads to $\dot{p}_j = -1$. We denote by $\boldsymbol{x}_\mathcal{G}$ the subvector of $\boldsymbol{x} \in \mathbb{R}^S$ formed by the elements x_i, $i \in \mathcal{G}$, where indices are contained in the set $\mathcal{G} \subseteq \mathcal{S}$. We consider a fixed basis B and thus drop the subscript B for the rest of this proof.

Table 5.1 implies the following useful equality that any $\boldsymbol{p}$ must satisfy.

$$\begin{aligned} 1 = \mathbf{1}^\mathsf{T}\boldsymbol{p} &= \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{R} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{D} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{N} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{T} \\ &= \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{R} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{D} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{T} \\ &= \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{R} + \mathbf{1}^\mathsf{T}\boldsymbol{p}_\mathcal{D} + p_\tau \end{aligned}$$

where the second identity follows from the fact that $\forall k \in \mathcal{N}$ implies $p_k = 0$. By taking the derivative $\frac{d}{d\xi}$ from both sides we have:

$$\begin{aligned} 0 &= \mathbf{1}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{R} + \mathbf{1}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{D} + \dot{p}_\tau \\ &= |\mathcal{R}| - |\mathcal{D}| + \dot{p}_\tau. \end{aligned}$$

And finally we have:

$$\begin{aligned} \dot{q} &= \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}} \\ &= \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{R} + \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{D} + \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{N} + \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}}_\mathcal{T} \\ &= \sum_{i\in\mathcal{R}} z_i - \sum_{j\in\mathcal{D}} z_j + \dot{p}_\tau z_\tau \ . \end{aligned}$$

□

Proof of Theorem 5.1.5. The statement is true due to linearity of $q(\xi)$ on the interval $[\xi_t, \xi_{t+1}]$ shown in Lemma 5.1.2, as well as the results in Lemma 5.1.6, Lemma 5.1.7, and Lemma 5.1.8.

□

Proof of Lemma 5.1.6. At $\xi = 0$, we can assume the none set is empty $\mathcal{N}_B = \emptyset$ because one can replace all non-negativity constraints $p_i \geq 0$ with $p_i - \bar{p}_i \leq \xi$ or $\bar{p}_i - p_i \leq \xi$. In Lemma 5.1.3, Section (v), we show for every $B \in \mathcal{B}$, $\forall\, i \in \mathcal{R}_B$, $\forall\, j \in \mathcal{D}_B$, and $\tau \in \mathcal{T}_B$ we have $i < \tau < j$. So $\dot{q}_B$ can be written as:

$$
\begin{aligned}
\dot{q}_B &= \sum_{i\in\mathcal{R}_B} z_i - \sum_{j\in\mathcal{D}_B} z_j + (|\mathcal{D}_B| - |\mathcal{R}_B|)\, z_\tau \\
&= \sum_{i=1}^{\tau-1} z_i - \sum_{j=\tau+1}^{S} z_j + ((S-\tau) - (\tau-1))z_\tau \\
&= \sum_{i=1}^{\tau-1} z_i - \sum_{j=\tau+1}^{S} z_j + (S - 2\tau + 1)z_\tau \\
&= \sum_{k=1}^{S} \text{sign}(k-\tau) z_k + (S-2\tau+1)z_\tau
\end{aligned}
\tag{A.13}
$$

Equation (A.13) shows at $\xi = 0$, the trader's rate $\dot{p}_\tau = S - 2\tau + 1$. We can also show that at $\xi = 0$, for all component $i \in \{1, \ldots, S\}$ we have $-1 \leq \dot{p}_i \leq 1$ because the constraints $p_i - \bar{p}_i \leq \xi$ and $\bar{p}_i - p_i \leq \xi$ are both active in equality. Thus we have

$$
\begin{aligned}
\min_{B\in\mathcal{B}} \quad & \frac{d}{d\xi} q_B(\xi_o) = \boldsymbol{z}^\mathsf{T}\dot{\boldsymbol{p}} \\
& \text{s. t.} \quad \mathbf{1}^\mathsf{T}\dot{\boldsymbol{p}} = 0 \ , \\
& \quad -\mathbf{1} \leq \dot{\boldsymbol{p}} \leq \mathbf{1} \ .
\end{aligned}
\tag{A.14}
$$

Since we previously showed the trader's exchange rate follows from $\dot{p}_\tau = |\mathcal{D}_B| - |\mathcal{R}_B|$ we can conclude $\dot{p}_\tau$ is an integer. Given the constraints in (A.14) at $\xi = 0$, we conclude $\dot{p}_\tau \in \{-1, 0, 1\}$. The index of the trader is obtained from one of the following scenarios:

$$
S - 2\tau + 1 = 0 \implies \tau = \frac{S+1}{2}, \tag{A.15}
$$

$$
S - 2\tau + 1 = 1 \implies \tau = \frac{S}{2}, \tag{A.16}
$$

$$
S - 2\tau + 1 = -1 \implies \tau = \frac{S+2}{2}, \tag{A.17}
$$

When S is an odd number, τ can be only $\frac{S+1}{2}$ because S is also an integer and τ cannot be fractional. And when S is an even number, τ can be either $\frac{S}{2}$ or $\frac{S+2}{2}$. Algorithm 9 returns the exact solution in both cases.

Given the index of trader for B_0, the index of all donors and receivers can be achieved

form Lemma 5.1.2 section (v). We initialize the sets: $\mathcal{T}_{B_0} = \{\lceil S/2 \rceil\}$, $\mathcal{R}_{B_0} = \{i \,|\, i < \tau\}$, $\mathcal{D}_{B_0} = \{j \,|\, j > \tau\}$, $\mathcal{N}_{B_0} = \{\}$;

□

Proof of Lemma 5.1.7. Suppose $z_1 \leq z_2 \leq \cdots \leq z_S$. Consider a base B that is feasible in the neighborhood of $\xi_t > 0$, and satisfies $B = \text{argmin}_{B \in \mathcal{B}} \frac{d}{d\xi} q(\xi_t)$. In Lemma 5.1.4, we show $\forall\, i \in \mathcal{R}_B$ and $\forall\, j \in \mathcal{D}_B \cup \mathcal{N}_B$ and $\tau \in \mathcal{T}_B$ we have $i < \tau < j$, and $\dot{q}_B$ can be written as:

$$\frac{d}{d\xi} q(\xi_t) = \dot{q}_B = \sum_{i \in \mathcal{R}_B} z_i - \sum_{k \in \mathcal{D}_B} z_k + (|\mathcal{D}_B| - |\mathcal{R}_B|)\, z_\tau \tag{A.18}$$

The adjacent basis $B' \in \mathcal{B}$ can be chosen from one of the following cases:

$$B' = \begin{cases} 1 & \mathcal{D}_{B'} = \mathcal{D}_B \backslash \{l\}, \quad \mathcal{N}_{B'} = \mathcal{N}_B \cup \{l\}, \quad \mathcal{T}_{B'} = \mathcal{T}_B, \quad \mathcal{R}_{B'} = \mathcal{R}_B \\ 2 & \mathcal{N}_{B'} = \mathcal{N}_B \cup \{\tau\}, \quad \mathcal{R}_{B'} = \mathcal{R}_B \backslash \{m\}, \quad \mathcal{T}_{B'} = \{m\}, \quad \mathcal{D}_{B'} = \mathcal{D}_B \\ 3 & \mathcal{D}_{B'} = \mathcal{D}_B \cup \{\tau\}, \quad \mathcal{R}_{B'} = \mathcal{R}_B \backslash \{n\} \quad \mathcal{T}_{B'} = \{n\}, \quad \mathcal{N}_{B'} = \mathcal{N}_B \\ 4 & \mathcal{R}_{B'} = \mathcal{R}_B \cup \{\tau\}, \quad \mathcal{D}_{B'} = \mathcal{D}_B \backslash \{o\}, \quad \mathcal{T}_{B'} = \{o\}, \quad \mathcal{N}_{B'} = \mathcal{N}_B \\ 5 & \mathcal{R}_{B'} = \mathcal{R}_B \cup \{\tau\}, \quad \mathcal{N}_{B'} = \mathcal{N}_B \backslash \{p\}, \quad \mathcal{T}_{B'} = \{p\}, \quad \mathcal{D}_{B'} = \mathcal{D}_B \\ 6 & \mathcal{N}_{B'} = \mathcal{N}_B \backslash \{q\}, \quad \mathcal{D}_{B'} = \mathcal{D}_B \cup \{q\}, \quad \mathcal{T}_{B'} = \mathcal{T}_B, \quad \mathcal{R}_{B'} = \mathcal{R}_B \end{cases} \tag{A.19}$$

Case 1 occurs when a donor becomes a none by donating all of its probability mass to a receiver. In this basis change, the index of the trader remains unchanged. B' is an adjacent basis for B since we only remove one active constraint ($\bar{p}_l - p_l \leq \xi$), and add another one ($p_l \geq 0$). In case 2, the trader becomes a none by losing all of its probability mass. The trader's index shifts from τ to m, one of the receivers in B. Note that in case 2 also, B' is an adjacent basis to B. We removed one active constraint ($p_m - \bar{p}_m \leq \xi$), and add another one ($p_\tau \geq 0$). Case 3 is similar to case 2, however in this case the trader reaches its lower bound, and as a result the new active constraint in B' is ($\bar{p}_\tau - p_\tau \leq \xi$). Case 4 occurs when a trader becomes a receiver. In this scenario, the trader's index shifts from τ to o, which was a member of $\mathcal{D}_B$. Case 5 and case 4 are similar. However, the trader in B' belongs to

$\mathcal{N}_B$. In the last case, one of the components in $\mathcal{N}_B$ gain probability mass and moves to the donor's set. In the following, we show that cases 4-6 are not a feasible choice for B'.

Any other case violates Lemma 5.1.3, Section (v). The corresponding $\dot{q}_{B'}$ obtain as follows:

$$\dot{q}_{B'} = \begin{cases} 1 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k + z_l + (|\mathcal{D}_B| - |\mathcal{R}_B| - 1)\, z_\tau \\ 2 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k - z_m + (|\mathcal{D}_B| - |\mathcal{R}_B| + 1)\, z_m \\ 3 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k - z_\tau - z_n + (|\mathcal{D}_B| - |\mathcal{R}_B| + 2)\, z_n \\ 4 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k + z_\tau + z_o + (|\mathcal{D}_B| - |\mathcal{R}_B| - 2)\, z_o \\ 5 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k + z_\tau + (|\mathcal{D}_B| - |\mathcal{R}_B| - 1)\, z_p \\ 6 & \sum_{i\in\mathcal{R}_B} z_i - \sum_{k\in\mathcal{D}_B} z_k - z_q + (|\mathcal{D}_B| - |\mathcal{R}_B| + 1)\, z_\tau \end{cases} \tag{A.20}$$

And hence we have:

$$\dot{q}_{B'} = \begin{cases} 1 & \dot{q}_B + (z_l - z_\tau) \\ 2 & \dot{q}_B + (z_m - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B|) \\ 3 & \dot{q}_B + (z_n - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B| + 1) \\ 4 & \dot{q}_B + (z_o - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B| - 1) \\ 5 & \dot{q}_B + (z_p - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B| - 1) \\ 6 & \dot{q}_B - (z_q - z_\tau) \end{cases} \tag{A.21}$$

Given Lemmas A.3.1 and A.3.2, B'_4, B'_5, and B'_6 are not a suitable choice for B' since $\dot{q}_{B'_4} \leq \dot{q}_B$, $\dot{q}_{B'_5} \leq \dot{q}_B$ and $\dot{q}_{B'_6} \leq \dot{q}_B$.

The choice over B'_1, B'_2, and B'_3 depend on the probability mass of the components at each breakpoint.

In order to minimize the decent rate in the case of $B' = B'_2$, we can show that:

$$\dot{q}_{B'} = \min_{m \in \mathcal{R}_B} \dot{q}_B + (z_m - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B|) \tag{A.22}$$

We know $z_m - z_\tau \leq 0$. And $0 \leq (z_m - z_\tau)(|\mathcal{D}_B| - |\mathcal{R}_B|)$ otherwise Lemma A.3.2 will be violated. As a result we conclude in this particular case $(|\mathcal{D}_B| - |\mathcal{R}_B|) \leq 0$.

In order to minimize Equation (A.22) the term $z_m - z_\tau$ should be minimized. Since $z_1 \leq \cdots \leq z_m \leq \cdots \leq z_\tau$, therefore $m^\star = \tau - 1$. With the same reasoning we can show in the case of $B' = B'_3$ we have $n^\star = \tau - 1$.

Our results follows the continuity assumption of the solution $\boldsymbol{p}^\star = \boldsymbol{p}_B(\xi)$ for all $\xi > 0$, in which a receiver can only become a trader, not a donor nor empty, at each breakpoints. Also, a donor cannot become a receiver unless it becomes a trader first. Otherwise, the continuity assumption will be violated.

□

Lemma A.3.1. *For all $B \in \mathcal{B}$ we have $|\mathcal{D}_B| - |\mathcal{R}_B| \leq 1$.*

Proof. Consider the problem with fixed ξ,

$$q(\xi) \;=\; \min_{\boldsymbol{p} \in \Delta^S} \left\{ \boldsymbol{p}^\mathsf{T} \boldsymbol{z} : \|\bar{\boldsymbol{p}} - \boldsymbol{p}\|_\infty \leq \xi \right\}, \tag{A.23}$$

For any fix $B \in \mathcal{B}$, we know:

$$\begin{aligned}
&\text{if} \quad i \in \mathcal{R}_B \implies p_i = \bar{p}_i + \xi, \\
&\text{if} \quad j \in \mathcal{D}_B \implies p_j = \bar{p}_j - \xi, \\
&\text{if} \quad k \in \mathcal{N}_B \implies p_k = 0, \\
&\text{if} \quad \tau \in \mathcal{T}_B, \quad \exists \Delta \in \mathbb{R} \quad \text{that} \quad p_\tau = \bar{p}_\tau + \Delta.
\end{aligned}$$

We also know

$$\begin{aligned}
\mathbf{1}^{\mathsf{T}}\boldsymbol{p} = 1 &\iff \sum_{i\in\mathcal{R}_B}(\bar{p}_i+\xi) + \sum_{j\in\mathcal{D}_B}(\bar{p}_j-\xi) + \bar{p}_\tau + \Delta = 1 \\
&\iff (1-\sum_{k\in\mathcal{N}_B}\bar{p}_k) + (|\mathcal{R}_B|-|\mathcal{D}_B|)\xi + \Delta = 1 \\
&\iff \Delta = \sum_{k\in\mathcal{N}_B}\bar{p}_k + (|\mathcal{D}_B|-|\mathcal{R}_B|)\xi
\end{aligned}$$

We know for feasibility, $\Delta \leq \xi$ so we have:

$$\begin{aligned}
\sum_{k\in\mathcal{N}_B}\bar{p}_k + (|\mathcal{D}_B|-|\mathcal{R}_B|)\xi &\leq \xi \\
\sum_{k\in\mathcal{N}_B}\bar{p}_k &\leq (|\mathcal{R}_B|-|\mathcal{D}_B|+1)\xi
\end{aligned}$$

Since $\sum_{k\in\mathcal{N}_B}\bar{p}_k \geq 0$, and $\xi > 0$, we conclude $(|\mathcal{R}_B|-|\mathcal{D}_B|+1) \geq 0$. As a result:

$$|\mathcal{D}_B| - |\mathcal{R}_B| \leq 1 \ .$$

□

Lemma A.3.2. *let $(\xi_t)_{t=0,\ldots,T+1}$, and $q(\xi)$ is a piecewise-affine convex function with break-points ξ_l. Then, $\dot{q}_0 \leq \dot{q}_1 \leq \ldots \leq \dot{q}_{T+1}$.*

Proof. The result follows convexity property of function $q(\xi) : \mathbb{R}_+ \to \mathbb{R}$. Let $\xi' \in (\xi_l, \xi_{l+1})$ and $\xi'' \in (\xi_{l+1}, \xi_{l+2})$. By Jensen's inequality, we have for ξ', ξ'', and $t \in [0,1]$:

$$q(\xi' + t(\xi''-\xi')) \leq q(\xi') + t(q(\xi'') - q(\xi'))$$

and hence

$$\frac{q(\xi' + t(\xi''-\xi')) - q(\xi')}{t} + q(\xi') \leq q(\xi'')$$

Let the affine function $\dot{q}_l \cdot \xi + b_l$ represent $q(\xi)$ for $\xi \in (\xi_l, \xi_{l+1})$. Take t sufficiently small enough so that $\xi' + t(\xi'' - \xi') \in (\xi_l, \xi_{l+1})$. The above inequality reduces to

$$\frac{\dot{q}_l \cdot (\xi' + t(\xi'' - \xi')) + b_l - \dot{q}_l \cdot \xi' - b_l}{t} + \dot{q}_l \cdot \xi' + b_l \leq \dot{q}_{l+1} \cdot \xi'' + b_{l+1}$$

$$\dot{q}_l \cdot \xi'' + b_l \leq \dot{q}_{l+1} \cdot \xi'' + b_{l+1}$$

□

Proof of Lemma 5.1.8. The optimization problem in (2.3) can be formulated as the following parametric LP:

$$q(\xi) \;=\; \min_{\boldsymbol{p} \in \mathbb{R}^{\mathcal{S}}} \; \left\{ \boldsymbol{z}^{\mathsf{T}} \boldsymbol{p} \mid \mathbf{1}^{\mathsf{T}} \boldsymbol{p} = 1,\, -\xi \leq p_i - \bar{p}_i \leq \xi,\, p_i \geq 0,\, i = 1, \ldots, S \right\} \;. \tag{A.24}$$

At each basis B_t, there are S constraints that are active and satisfied in equity. In order to maintain the feasibility the basis B_t on the interval $[\xi_t, \xi_t + \Delta\xi_t]$, one needs to keep track of constrains that will be violated first by increasing $\xi \in [\xi_t, \xi_t + \Delta\xi_t]$, and relax all other constraint. Since the donation rate is equal among all donors $\dot{p}_i = -1 \; \forall i \in \mathcal{D}_{B_t}$, the non-negativity constraints could be watched by following the donors with minimal probability mass $\Delta\xi_t \leftarrow \max\{\xi \geq 0 \mid \boldsymbol{p}_t + \xi \cdot \nabla_\xi \boldsymbol{p}_{B_t}(\xi_t) \geq \mathbf{0}\}$. The rate of exchange for the trader varies at each basis, as a result, the trader could violate its lower and upper bound $-\xi \leq p_\tau - \bar{p}_\tau \leq \xi$. The algorithm trace the trader's rate so one can check the constrain via $\Delta\xi_t \leftarrow \max\{\xi \geq 0 \mid |(\boldsymbol{p}_t + \xi \cdot \nabla_\xi \boldsymbol{p}_{B_t}(\xi_t) - \bar{\boldsymbol{p}})_{\tau_t}| \leq \xi_t + \xi\}$. Line 4 of Algorithm 4 combines these constraints and relaxes others.

□

Proof of Theorem 5.1.9. A naive implementation of the homotopy method in Algorithm 4 has a computational complexity of $\mathcal{O}(S^2)$. The algorithm obtains the $\boldsymbol{p}^\star$ at each breakpoint. The number of iteration depends on the number of breakpoints in $q(\xi)$, which is at most $\frac{3}{2}S$.

We observed numerically that the naive implementation performs on par with LP solvers and sometimes even slower. In Algorithm 9, we take advantage of the structural property of the slope of the q-function presented in Lemma 5.1.4, and only trace the optimal probability mass of the *trader* to speed up the method dramatically. Algorithm 9 compute q-function for each state-action pair in $\mathcal{O}(S \log S)$ for sorting the values of $\boldsymbol{z}$.

□

A.3.2 Detailed Homotopy Algorithm

This section provides the detailed procedure of our homotopy algorithm for computing the exact solution for robust Bellman operator with L_∞ constrained ambiguity sets. The algorithm starts with the initialization of the doner, receiver, and trader sets according to Lemma 5.1.6, and then iterates through all breakpoints. Each breakpoint has been obtained concerning the conditions that are described in Lemma 5.1.7. The type of each basis is change is indicated according to Table 5.2. We use a priority queue to keep track of the donor with the smallest probability mass. The algorithm follows the value of q-function at each iteration, however ignores the probability mass values for all components except the trader. The iteration stops as soon as ξ exceeds the budget κ, which is given as an input.

A.3.3 Proofs of Results in Section 5.2

Proof of Theorem 5.2.1. The result follows from the complexity analysis of the bisection algorithm with quasi-linear time complexity in [31], appendix B. □

Lemma A.3.3. *The optimal objective values of Equations* (5.4) *and* (5.5) *are equivalent.*

Proof of Lemma A.3.3. Since the functions q_a, for all $a \in \mathcal{A}$ in Equation (5.5) are convex due to the LP formulation of Equation (5.3). We can exchange the maximization and minimization operators in Equation (5.5) to obtain

$$\min_{\boldsymbol{\xi} \in \mathbb{R}_+^A} \left\{ \max_{\boldsymbol{\pi} \in \Delta^A} \left(\sum_{a \in \mathcal{A}} \pi_a \cdot q_a(\xi_a) \right) \mid \sum_{a \in \mathcal{A}} \xi_a \leq \kappa \right\} , \tag{A.25}$$

Algorithm 9: Homotopy method for $q(\kappa)$ with L_∞ constrained ambiguity set.

Input: LP parameters $\boldsymbol{z}$, κ and $\bar{\boldsymbol{p}}$

Output: Breakpoints $(\xi_t)_{t=0,\dots,T+1}$ and values $(q_t)_{t=0,\dots,T+1}$

Initialize $\xi_0 \leftarrow 0$, $t \leftarrow 0$, $\boldsymbol{p}_0 \leftarrow \bar{\boldsymbol{p}}$ and $q_0 \leftarrow q(\xi_0) = \boldsymbol{p}_0^\mathsf{T}\boldsymbol{z}$;

Sort $\boldsymbol{z}$ in ascending order and rearrange $\bar{\boldsymbol{p}}$ accordingly ;

Initialize the sets: ;

$\mathcal{T} = \{\lceil S/2 \rceil\},\ \mathcal{R} = \{i \,|\, i < \tau\},\ \mathcal{D} = \{j \,|\, j > \tau\},\ \mathcal{N} = \{\}$;

$z_\mathcal{R} = \sum_{i\in\mathcal{R}} z_i; \quad z_\mathcal{D} = \sum_{j\in\mathcal{D}} z_j$;

Push all elements of $\mathcal{D}$ into a min-heap $\mathcal{H}$ according to their probability mass ;

$\xi \leftarrow \xi_0$;

while $\xi < \kappa$ **do**

  $\dot{p}_\tau \leftarrow |\mathcal{D}| - |\mathcal{R}|$; # The trader's rate of exchange ;

  $j \leftarrow \mathcal{H}.top$;

  $\Delta\xi_D \leftarrow p_j - \xi$;

  $\Delta\xi_\tau \leftarrow$ Calculate largest feasible Δp_τ given $\dot{p}_\tau$;

  Find basis change type (Algorithm 10) ;

  $\Delta\xi \leftarrow \max\{\Delta\xi, \kappa - \xi\}$;

  $p_\tau \leftarrow p_\tau + \dot{p}_\tau \cdot \Delta\xi$;

  $q_t = q_{t-1} + (z_\mathcal{R} - z_\mathcal{D} + \dot{p}_\tau z_\tau) \cdot \Delta\xi$;

  $\xi \leftarrow \xi + \Delta\xi;\ \xi_t \leftarrow \xi;\ t \leftarrow t + 1$;

  if *Basis Change is* $\mathcal{D}$ *to* $\mathcal{N}$ **then**

    $z_\mathcal{D} \leftarrow z_\mathcal{D} - z_j$;

    $\mathcal{D} = \mathcal{D}\backslash\{j\}$;

    $\mathcal{N} = \mathcal{N} \cup \{j\}$;

    $\mathcal{H}.pop$;

  else

    if *Basis Change is* $\mathcal{T}$ *to* $\mathcal{D}$ **then**

      $\mathcal{H}.push(\tau)$ # $p = p_\tau + \xi$;

      $\mathcal{D} = \mathcal{D} \cup \{\tau\}$;

      $z_\mathcal{D} \leftarrow z_\mathcal{D} + z_\tau$;

    else if *Basis Change is* $\mathcal{T}$ *to* $\mathcal{N}$ **then**

      $\mathcal{N} = \mathcal{N} \cup \{\tau\}$;

    $\tau \leftarrow \tau - 1$;

    $\mathcal{T} = \{\tau\}$;

    $\mathcal{R} = \mathcal{R}\backslash\{\tau\}$;

    $p_\tau \leftarrow \bar{p}_\tau + \xi$;

    $z_\mathcal{R} \leftarrow z_\mathcal{R} - z_\tau$;

  end

end

The remainder of the function $q(\xi)$ will be constant: $q_{T+1} \leftarrow q_t$;

$\xi_{T+1} \leftarrow \infty$;

return $(\xi_t)_{t=0,\dots,T+1}$, *and* $(q_t)_{t=0,\dots,T+1}$

Algorithm 10: Identifies the type of basis change at each breakpoints.

Input: $\Delta\xi_\tau, \Delta\xi_D$
Output: Type of basis change, and $\Delta\xi$

if $\Delta\xi_\tau > \Delta\xi_D$ **then**
  Basis Change $\leftarrow \mathcal{D}$ *to* $\mathcal{N}$;
  $\Delta\xi \leftarrow \Delta\xi_D$;
else
  $\Delta\xi \leftarrow \Delta\xi_\tau$; $p'_\tau \leftarrow p_\tau + \dot{p}_\tau \cdot \Delta\xi$;
  if $p'_\tau = 0$ **then**
    Basis Change $\leftarrow \mathcal{T}$ *to* $\mathcal{N}$;
  else
    Basis Change $\leftarrow \mathcal{T}$ *to* $\mathcal{D}$;
  end
end
return *Basis Change, and* $\Delta\xi$

Since the inner maximization is linear in $\boldsymbol{\pi}$, it is optimized at an extreme point of Δ^A. This allows us to re-express the optimization problem as

$$\min_{\boldsymbol{\xi}\in\mathbb{R}^A_+} \left\{ \max_{a\in\mathcal{A}} q_a(\xi_a) \mid \sum_{a\in\mathcal{A}} \xi_a \leq \kappa \right\} . \tag{A.26}$$

We can linearize the objective function in this problem by introducing the epigraphical variable $u \in \mathbb{R}$

$$\min_{u\in\mathbb{R}} \min_{\boldsymbol{\xi}\in\mathbb{R}^A_+} \left\{ u \mid \sum_{a\in\mathcal{A}} \xi_a \leq \kappa, u \geq \max_{a\in\mathcal{A}} [q_a(\xi_a)] \right\} \tag{A.27}$$

It can be readily seen that for a fixed u in the outer minimization, there is an optimal $\boldsymbol{\xi}$ in the inner minimization that minimizes each ξ_a a individually while satisfying $q_a(\xi_a) \leq u$ for all $a \in \mathcal{A}$. Define g_q as the a-th component of this optimal $\boldsymbol{\xi}$:

$$g_a(u) = \min_{\xi_a\in\mathbb{R}^A_+} \{\xi_a \mid q_a(\xi_a) \leq u\}. \tag{A.28}$$

We show that $g_a(u) = q_a^{-1}$. To see this, we substitude q_a in Equation (A.28) to get:

$$g_a(u) = \min_{\xi_a\in\mathbb{R}^A_+} \min_{\boldsymbol{p}_a\in\Delta^S} \left\{ \xi_a \mid \boldsymbol{p}_a^\mathsf{T} \boldsymbol{z}_a \leq u, \|\boldsymbol{p}_a - \bar{\boldsymbol{p}}_a\|_\infty \leq \xi_a \right\}. \tag{A.29}$$

The identity $g_a = q_a^{-1}$ then follows by realizing that the optimal $\xi_a^\star$ in the equation above must satisfy $\xi_a^\star = \|\boldsymbol{p}_a - \bar{\boldsymbol{p}}_a\|_\infty$. Finally, substituting the definition of g_a in Equation (A.28) into the problem (A.27) show that the optimization problem (5.5) is equivalent to Equation (5.4). □

Algorithm 11: Bisection method for the robust Bellman optimality operator [31].

Input: Precision ϵ, functions $q_a^{-1}, \forall a \in \mathcal{A}$
$u_{\min}$: maximum known u for which Equation (5.4) is infeasible ;
$u_{\max}$: minimum known u for which Equation (5.4) is feasible ;
Output: $\hat{u}$ such that $|u^\star - \hat{u}| \leq \epsilon$, where $u^\star$ is optimal in Equation (5.4)
while $u_{\max} - u_{\min} > 2\ \epsilon$ **do**
    Split interval $[u_{\min}, u_{\max}]$ in half: $u \leftarrow (u_{\min} + u_{\max})/2$;
    Calculate the budget required to achieve the mid-point u: $s \leftarrow \sum_{a \in \mathcal{A}} q_a^{-1}(u)$;
    if $s \leq \kappa$ **then**
        u is feasible: update the feasible upper bound: $u_{\max} \leftarrow u$;
    else
        u is infeasible: update the infeasible lower bound: $u_{\min} \leftarrow u$;
    end
end
return $(u_{\min} + u_{\max})/2$

A.3.4 Detailed Description of Domains

In this section, we provide a detailed description of five standard reinforcement domains that have been previously used to evaluate robustness.

As the primary metric, we compare the running time of our homotopy and bisection algorithm with Gurobi 9.1.2—a standard LP solver. In order to enable the comparison of the results among different domains, we also compare our results with the homotopy and bisection algorithm for L_1-constrained ambiguity sets in [31].

As the first benchmark, we employ Inventory Management (IM), a classic inventory management problem [84], with discrete inventory levels $0, \ldots, S = 30$. The purchase cost, sale price, and holding cost are $2.49, 3.99$, and 0.03, respectively. The demand is sampled from a normal distribution with a mean $S/4$ and a standard deviation of $S/6$. The initial state is 0 (empty stock). It also uses a Dirichlet prior. Table 5.3 summarizes the run-time for computed guaranteed returns of different methods at 0.95 confidence levels.

The second domain is RiverSwim (RS) which is a standard benchmark [74], which is an MDP consisting of six states and two actions. The process follows by sampling synthetic datasets from the true model and then computing the guaranteed robust returns for different methods. The prior is a uniform Dirichlet distribution over reachable states.

Moreover, Machine Replacement (MR) is a small benchmark MDP problem with $S = 10$ states that models progressive deterioration of a mechanical device [12]. Two repair actions $A = 2$ are available and restore the machine's state.

A.3.5 Fast Algorithm for Nature Response with Fixed ξ

Let us consider the optimization problem (2.3):

$$\min_{\boldsymbol{p}\in\Delta^S}\{\boldsymbol{p}^\mathsf{T}\boldsymbol{z} : \|\bar{\boldsymbol{p}}-\boldsymbol{p}\|_\infty \le \xi\}, \tag{A.30}$$

As expressed earlier, the problem can be formulated as the following LP problem:

$$\begin{aligned} q(\xi) = \quad & \min_{\boldsymbol{p}\in\mathbb{R}^{\mathcal{S}}} \quad \boldsymbol{z}^\mathsf{T}\boldsymbol{p} \\ & \text{s. t.} \quad \boldsymbol{p}-\bar{\boldsymbol{p}} \le \boldsymbol{\xi} \\ & \qquad\quad \bar{\boldsymbol{p}}-\boldsymbol{p} \le \boldsymbol{\xi} \\ & \qquad\quad \boldsymbol{p} \ge \mathbf{0} \\ & \qquad\quad \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1\ . \end{aligned} \tag{A.31}$$

The problem is analogous to minimizing a linear function over a rectangle.

$$\begin{aligned} & \min_{\boldsymbol{p}\in\mathbb{R}^{\mathcal{S}}} \quad \boldsymbol{z}^\mathsf{T}\boldsymbol{p} \\ & \text{s. t.} \quad -\boldsymbol{\xi} \le \boldsymbol{p}-\bar{\boldsymbol{p}} \le \boldsymbol{\xi} \\ & \qquad\quad \boldsymbol{p} \ge \mathbf{0} \\ & \qquad\quad \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1\ , \end{aligned} \tag{A.32}$$

where the objective and the constrains are separable. The objective is a sum of terms $z_i p_i$ that each term depends only on one variable. We can therefore solve the problem by minimizing over each component of $\boldsymbol{p}$ independently.

$$\begin{aligned} & \min_{\boldsymbol{p}\in\mathbb{R}^{\mathcal{S}}} \quad \boldsymbol{z}^\mathsf{T}\boldsymbol{p} \\ & \text{s. t.} \quad -\boldsymbol{\xi}+\bar{\boldsymbol{p}} \le \boldsymbol{p} \le \boldsymbol{\xi}+\bar{\boldsymbol{p}} \\ & \qquad\quad \boldsymbol{p} \ge \mathbf{0} \\ & \qquad\quad \mathbf{1}^\mathsf{T}\boldsymbol{p} = 1\ . \end{aligned} \tag{A.33}$$

let $-\boldsymbol{\xi}+\bar{\boldsymbol{p}}=\boldsymbol{l}$ and $\boldsymbol{\xi}+\bar{\boldsymbol{p}}=\boldsymbol{u}$.

$$\begin{aligned} \min_{\boldsymbol{p}\in\mathbb{R}^{\mathcal{S}}} \quad & \boldsymbol{z}^{\mathsf{T}}\boldsymbol{p} \\ \text{s. t.} \quad & \boldsymbol{l}\leq\boldsymbol{p}\leq\boldsymbol{u} \\ & \boldsymbol{p}\geq\mathbf{0} \\ & \mathbf{1}^{\mathsf{T}}\boldsymbol{p}=1\ . \end{aligned} \tag{A.34}$$

The constraint $\boldsymbol{l}\leq\boldsymbol{p}$ and $0\leq\boldsymbol{p}$ can be combined as $\boldsymbol{l}'\leq\boldsymbol{p}$ where $l_i'=\max\{0,l_i\}$. We also know that $p_i\leq 1,\forall i\in\mathcal{S}$. So $\boldsymbol{p}\leq\boldsymbol{u}$ also can be replaced by $\boldsymbol{p}\leq\boldsymbol{u}'$ where $u_i'=\min\{1,u_i\}$. Now we have

$$\begin{aligned} \min_{\boldsymbol{p}\in\mathbb{R}^{\mathcal{S}}} \quad & \boldsymbol{z}^{\mathsf{T}}\boldsymbol{p} \\ \text{s. t.} \quad & \boldsymbol{l}'\leq\boldsymbol{p}\leq\boldsymbol{u}' \\ & \mathbf{1}^{\mathsf{T}}\boldsymbol{p}=1\ . \end{aligned} \tag{A.35}$$

The problem (A.35) is a bounded resource allocation problem with continuous variables, where the objective function is convex and continuously differentiable. Without loss of generality we add the following restrictions:

First, $\boldsymbol{l}'<\boldsymbol{u}'$, since if $l_j'=u_j'$ for any $j\in\{1,\ldots,S\}$ implies that p_j is fixed and can be dropped from (A.35). Second, $\mathbf{1}^{\mathsf{T}}\boldsymbol{l}'<1<\mathbf{1}^{\mathsf{T}}\boldsymbol{u}'$. Otherwise the problems is either infeasible or trivially solvable. We consider the following equivalent problem, which obtained by change in variables $\boldsymbol{x}=\boldsymbol{p}-\boldsymbol{l}'$, and the modified upper bound $\boldsymbol{u}=\boldsymbol{u}'-\boldsymbol{l}'$. Let $\alpha=1-\mathbf{1}^{\mathsf{T}}\boldsymbol{l}'$:

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^{\mathcal{S}}} \quad & \boldsymbol{z}^{\mathsf{T}}\boldsymbol{x} \\ \text{s. t.} \quad & \mathbf{0}\leq\boldsymbol{x}\leq\boldsymbol{u} \\ & \mathbf{1}^{\mathsf{T}}\boldsymbol{x}=\alpha\ . \end{aligned} \tag{A.36}$$

For problem (A.36), the Kuhn-Tucker conditions are as follows:

$$z_j-\mu_j+\nu_j-\lambda=0, \qquad\qquad j=1,\ldots,S,$$

$$
\begin{aligned}
&\mu_j x_j = 0, && j = 1, \ldots, S, \\
&\nu_j (x_j - u_j) = 0, && j = 1, \ldots, S, \\
&\mu_j, \nu_j \geq 0, && j = 1, \ldots, S, \\
&\sum_{j=1}^{S} x_j = \alpha, && \\
&0 \leq x_j \leq u_j, && j = 1, \ldots, S,
\end{aligned}
$$

where λ, μ_j, ν_j are Lagrange multipliers associated with constraints $\mathbf{1}^\mathsf{T}\boldsymbol{x} = \alpha$, $\mathbf{0} \leq \boldsymbol{x}$, and $\boldsymbol{x} - \boldsymbol{u} \leq \mathbf{0}$, respectively. These conditions are equivalent to:

$$
\begin{aligned}
0 < x_j < u_j &\implies z_j = \lambda, \\
x_j = 0 &\implies z_j \geq \lambda, \\
x_j = u_j &\implies z_j \leq \lambda, \\
\sum_{j=1}^{S} x_j &= \alpha, \\
0 \leq x_j \leq u_j, &\quad j = 1, \ldots, S,
\end{aligned}
$$

Next, we show the relationship between the optimal solution to A.37, and the following relaxed problem:

$$
\begin{aligned}
\min_{\boldsymbol{x} \in \mathbb{R}^{\mathcal{S}}} \quad & \boldsymbol{z}^\mathsf{T}\boldsymbol{x} \\
\text{s. t.} \quad & \mathbf{0} \leq \boldsymbol{x} \\
& \mathbf{1}^\mathsf{T}\boldsymbol{x} = \alpha \ .
\end{aligned}
\tag{A.37}
$$

Lemma A.3.4. *Let $\hat{\boldsymbol{x}} = (\hat{x}_1, \ldots, \hat{x}_n)$ be the optimal solution of A.37. Then $\hat{x}_j \geq u_j$ implies that $x_j^\star = u_j$ holds in an optimal solution $\boldsymbol{x}^\star$ of* (A.36).

The proof is provided by Ibaraki and Katoh [33]. The lemma makes it possible to have the following algorithm for (A.37):

Another approach for solving (A.36) is to suppose the component of $\boldsymbol{z}$ are sorted in increasing order with

$$z_1 \leq z_2 \leq \ldots \leq z_S$$

The problem is now similar to budget allocation. First we set $\boldsymbol{p} = \boldsymbol{l}$ to satisfy the lower bound constraint. Note that if $\mathbf{1}^\mathsf{T}\boldsymbol{l} > 1$ the problem is infeasible. The reaming budget $\alpha = 1 - \mathbf{1}^\mathsf{T}\boldsymbol{l}$ which needs to be allocated to satisfy $\mathbf{1}^\mathsf{T}\boldsymbol{p} = 1$ constraint. Each component p_i has the capacity of allocating $u_i - l_i$ of the budget α, so we have $0 \leq \alpha_i \leq u_i - l_i$. The optimal $p_i^\star$ minimizes $z_i p_i$, so we can now allocate α to the smallest elements of $\boldsymbol{z}$ until we run out of budget.

A.4 Fast Feature Selection for Reinforcement Learning

A.4.1 Proof of Theorem 6.2.1

Proof of Theorem 6.2.1. From the definition of Δ_P and P_Φ we get the following equality:

$$\Delta_P = U\Sigma V^\mathsf{T} U_1 - U_1 (U_1^\mathsf{T} U_1)^{-1} U_1^\mathsf{T} U \Sigma V^\mathsf{T} U_1 \ .$$

Recall that singular vectors are orthonormal which implies that $(U_1^\mathsf{T} U_1)^{-1} = \mathbf{I}$ and $U_1^\mathsf{T} U = \begin{bmatrix} \mathbf{I}_1 & 0 \end{bmatrix}$. Substituting these terms into the equality above, we get:

$$\begin{aligned} \|\Delta_P\|_2 &= \left\|(U\Sigma V^\mathsf{T} - U_1 \Sigma_1 V_1^\mathsf{T}) U_1\right\|_2 \\ &\leq \left\|U\Sigma V^\mathsf{T} - U_1 \Sigma_1 V_1^\mathsf{T}\right\|_2 \|U_1\|_2 \ . \end{aligned}$$

Simple algebraic manipulation shows that $\left\|U\Sigma V^\mathsf{T} - U_1 \Sigma_1 V_1^\mathsf{T}\right\|_2 = \|\Sigma_2\|_2$ and $\|U_1\|_2 = 1$ because U is an unitary matrix. This establishes the inequality for Δ_P; the result for Δ_r follows directly from the properties of orthogonal projection since $\boldsymbol{r}$ itself is included in the features. □

A.4.2 Proof of Theorem 6.2.2

Proof of Theorem 6.2.2. We show the result only for Δ_P; the result for Δ_r follows similarly. From the definition of Δ,

$$\left\|\Delta_P^{A\widehat{\Phi}}\right\|_2 = \left\|PA\widehat{\Phi} - A\widehat{\Phi} P_{A\widehat{\Phi}}\right\|_2 \ .$$

Now, by adding a zero ($AP_A\widehat{\Phi} - AP_A\widehat{\Phi}$) and applying the triangle inequality, we get:

$$\begin{aligned} \left\|\Delta_P^{A\widehat{\Phi}}\right\|_2 &= \left\|PA\widehat{\Phi} - AP_A\widehat{\Phi} + AP_A\widehat{\Phi} - A\widehat{\Phi} P_{A\widehat{\Phi}}\right\|_2 \leq \\ &\leq \left\|PA\widehat{\Phi} - AP_A\widehat{\Phi}\right\|_2 + \left\|AP_A\widehat{\Phi} - A\widehat{\Phi} P_{A\widehat{\Phi}}\right\|_2 \ . \end{aligned}$$

Given $(A\widehat{\Phi})^+ = \widehat{\Phi}^+ A^+$ and the property of the compressed transition matrix in Equation (2.9) we can show:

$$\begin{aligned}(P_A)_{\widehat{\Phi}} - P_{A\widehat{\Phi}} &= \widehat{\Phi}^+ P_A \widehat{\Phi} - (A\widehat{\Phi})^+ PA\widehat{\Phi} \\ &= \widehat{\Phi}^+ (P_A - A^+ PA)\widehat{\Phi} = 0\end{aligned}$$

$$\begin{aligned}\left\|\Delta_P^{A\widehat{\Phi}}\right\|_2 \leq \|PA - AP_A\|_2 \left\|\widehat{\Phi}\right\|_2 + \\ + \left\|P_A\widehat{\Phi} - \widehat{\Phi}(P_A)_{\widehat{\Phi}}\right\|_2 \|A\|_2 \ .\end{aligned}$$

The theorem then follows directly from algebraic manipulation and the fact that the features are normalized. □