Efficient Policy Iteration for Robust Markov Decision Processes via Regularization

Navdeep Kumar[1], Kfir Levy[1], Kaixin Wang[2], and Shie Mannor[1]

[1]Technion
[2]National University of Singapore

May 31, 2022

Abstract

Robust Markov decision processes (MDPs) provide a general framework to model decision problems where the system dynamics are changing or only partially known. Recent work established the equivalence between $\mathtt{s}$ rectangular L_p robust MDPs and regularized MDPs, and derived a regularized policy iteration scheme that enjoys the same level of efficiency as standard MDPs. However, there lacks a clear understanding of the policy improvement step. For example, we know the greedy policy can be stochastic but have little clue how each action affects this greedy policy. In this work, we focus on the policy improvement step and derive concrete forms for the greedy policy and the optimal robust Bellman operators. We find that the greedy policy is closely related to some combination of the top k actions, which provides a novel characterization of its stochasticity. The exact nature of the combination depends on the shape of the uncertainty set. Furthermore, our results allow us to efficiently compute the policy improvement step by a simple binary search, without turning to an external optimization subroutine. Moreover, for L_1, L_2, and L_∞ robust MDPs, we can even get rid of the binary search and evaluate the optimal robust Bellman operators exactly. Our work greatly extends existing results on solving $\mathtt{s}$-rectangular L_p robust MDPs via regularized policy iteration and can be readily adapted to sample-based model-free algorithms.

1 Introduction

In Markov Decision Processes (MDPs), an agent interacts with the environment and learns to optimally behave in it [15]. Nevertheless, an MDP solution may be very sensitive to the model parameters [11, 18, 13], implying that one should be cautious when the model is changing or when there is uncertainty in the model parameters. Robust MDPs (RMDPs) on the other hand, allow some room for the uncertainty in the model parameters as it allows the model parameters to belong to some uncertainty set [6, 16, 9].

Solving robust MDPs is NP-hard for general uncertainty sets [12], hence the uncertainty set is popularly assumed to be *rectangular* which enables tractability [9, 12, 7]. The uncertainty set is called rectangular if the uncertainty in model parameters (*i.e.*, reward and transition kernel) in one state is not coupled with the uncertainty in model parameters in

different state. Under this rectangularity assumption, many structural properties of MDPs remain intact[9, 12] and methods such as robust value iteration [2, 17], robust modified policy iteration [10], partial robust policy iteration [7] etc can be used to solve it. It is also known that uncertainty in the reward can be easily handled, whereas handling uncertainty in transition kernel is much more challenging [3]. When an uncertainty set has a polyhedral structure (as in L_1/L_∞ robust MDPs) then it can be solved using nested Linear programming, that is, both policy evaluation and policy improvement are done using linear programming exactly or inexactly[10, 17]. Further in this direction, partial policy iteration approach to solve L_1-Robust MDPs by [7] uses bags of tricks such as homotopy, bisection methods etc. But these methods are computationally expensive or limited to certain cases.

Several previous works have investigated robust MDPs from a regularization perspective [3, 4, 8, 5]. While this approach provides an interesting insight into the the problem at hand, it has so far did not lead to scalable methods for solving RMDPs. Along this direction and closest to our work is by Derman et. al [3] which showed that robust MDPs can be viewed as value and policy regularized MDPs. Particularly, for (`sa`)-rectangular case, they reduced robust policy iteration (both policy evaluation and improvement) to reward regularized non-robust MDPs. The reward regularizer comes from the uncertainty in reward and transition kernel. Nevertheless their assumption on kernel noise is unrealistic. For `s`-rectangular case, they obtained only policy evaluation as regularized non-robust MDPs, and for policy improvement they go for external black box solver (projection onto simplex solver). Our work differs from [3] in the following respect: **a)** We correct unrealistic assumption made in the kernel noise (in both `s` and (`sa`) cases) by [3] that has a significant consequences on the results. **b)** For `s`- rectangular case, our work provide explicit characterization of the policy improvement step. For general p, policy improvement step can be approximated using binary search, and for $p = 1, 2, \infty$, it can be done exactly in either closed form or by some simple algorithm.

Contributions:

- This paper provides an efficient method to solving both `s` and (`sa`)-rectangular L_p robust MDPs whose complexity is very close (up to some log factors) to non-robust MDPs (in some special cases, it is exactly the same).
- Our proposed robust policy iteration variant for `s` rectangular L_p robust MDPs is very similar to the non-robust MDPs variant with some interesting differences: **a)** It penalizes the states proportional to its uncertainty and proportional to the *variance* of value function **b)** Instead of choosing the best action, it hedges itself by choosing top k actions proportional to their Q-values where k depends on the level of uncertainty in the state and *variance* of value function.

2 Preliminary

2.1 Notations

For a set $\mathcal{S}$, $|\mathcal{S}|$ denotes its cardinality. $\langle u, v\rangle := \sum_{s\in\mathcal{S}} u(s)v(s)$ denotes the dot product between functions $u, v : \mathcal{S} \to \mathbb{R}$. $\|v\|_p^q := (\sum_s |v(s)|^p)^{\frac{q}{p}}$ denotes the qth power of L_p norm of function v, and we use $\|v\|_p := \|v\|_p^1$ and $\|v\| := \|v\|_2$ as shorthand. For a set $\mathcal{C}$, $\Delta_{\mathcal{C}} := \{a : \mathcal{C} \to \mathbb{R} | a(s) \geq 0, \forall s \sum_{c\in\mathcal{C}} a_c = 1\}$ is the probability simplex over $\mathcal{C}$ $\mathbf{0}, \mathbf{1}$ denotes all zero vector and all ones vector/function respectively of appropriate dimension/domain. $\mathbb{1}(a = b) := 1$ if $a = b$, 0 otherwise, is indicator function.

2.2 Markov Decision Processes

A Markov Decision Process (MDP) is defined by $(\mathcal{S}, \mathcal{A}, P, R, \gamma, \mu)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ is the transition kernel, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\mu \in \Delta_{\mathcal{S}}$ is the initial distribution over states and $\gamma \in [0, 1)$ is the discount factor [15]. A stationary policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ maps states to probability distribution over actions. And $\pi(a|s), P(s'|s, a), R(s, a)$ denotes the probability of selecting action a in state s, transition probability to state s' in state s under action a, and reward in state $\mathtt{s}$ under action a respectively. We denote $S = |\mathcal{S}|$, and $A = |\mathcal{A}|$ as a shorthand. The objective in a MDP is to maximize the expected discounted cumulative reward, defined as

$$\mathbb{E}\left[\sum_{n=0}^{\infty} \gamma^n R(s_n, a_n) \;\middle|\; s_0 \sim \mu, a_n \sim \pi(\cdot|s_n), s_{n+1} \sim P(\cdot|s_n, a_n)\right] = \langle R, d_P^\pi \rangle = \langle \mu, v_{P,R}^\pi \rangle, \quad (1)$$

where d_P^π is occupation measure of policy π with initial distribution of sates μ, and kernel P, defined as

$$d_P^\pi(s, a) := \mathbb{E}\left[\sum_{n=0}^{\infty} \gamma^n \mathbb{1}(s_n = s, a_n = a) \;\middle|\; s_0 \sim \mu, a_n \sim \pi(\cdot|s_n), s_{n+1} \sim P(\cdot|s_n, a_n)\right],$$

and $v_{P,R}^\pi$ is the value function (dual formulation[14]) under policy π with transition kernel P and reward vector R, defined as

$$v_{P,R}^\pi(s) := \mathbb{E}\left[\sum_{n=0}^{\infty} \gamma^n R(s_n, a_n) \;\middle|\; s_0 = s, a_n \sim \pi(\cdot|s_n), s_{n+1} \sim P(\cdot|s_n, a_n)\right]. \quad (2)$$

The value function $v_{P,R}^\pi$ for policy π, is the fixed point of the Bellmen operator $\mathcal{T}_{P,R}^\pi$ [15], defined as

$$(\mathcal{T}_{P,R}^\pi v)(s) = \sum_a \pi(a|s) \left[R(s, a) + \gamma \sum_{s'} P(s'|s, a) v(s') \right]. \quad (3)$$

Similarly, the optimal value function $v_{P,R}^* := \max_\pi v_{P,R}^\pi$ is well defined and is the fixed point of the optimal Bellman operator $\mathcal{T}_{P,R}^*$ [15], defined as

$$(\mathcal{T}_{P,R}^* v)(s) = \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s'} P(s'|s, a) v(s') \right]. \quad (4)$$

The optimal Bellman operator $\mathcal{T}_{P,R}^*$ and Bellman operators $\mathcal{T}_{P,R}^\pi$ for all policy π, are γ-contraction maps[15].

2.3 Robust Markov Decision Processes

In most practical cases, the system dynamics (transition kernel P and reward function R) are not known precisely. Instead, we have an access to a nominal reward function R_0 and a nominal transition kernel P_0 that may have some uncertainty. To capture this, the uncertainty (or ambiguity) set is defined as $\mathcal{U} := (R_0 + \mathcal{R}) \times (P_0 + \mathcal{P})$, where $\mathcal{R}$, $\mathcal{P}$ are the respective uncertainties in the reward function and transition kernel [9].The robust performance of a policy π is defined to be its worst performance on the entire uncertainty set $\mathcal{U}$. And our objective is to maximize the robust performance, that is

$$\max_\pi \min_{R,P \in \mathcal{U}} \langle R, d_P^\pi \rangle, \qquad \text{equivalently} \qquad \max_\pi \min_{R,P \in \mathcal{U}} \langle \mu, v_{P,R}^\pi \rangle. \quad (5)$$

Solving the above robust objective is NP-hard in general[9, 17]. Hence, the uncertainty set $\mathcal{U}$ is commonly assumed to be $\mathtt{s}$-rectangular, that is $\mathcal{R}$ and $\mathcal{P}$ can be decomposed state wise as $\mathcal{R} = \times_{s\in\mathcal{S}}\mathcal{R}_s$ and $\mathcal{P} = \times_{s\in\mathcal{S}}\mathcal{P}_s$. Sometimes, the uncertainty set $\mathcal{U}$ is assumed to decompose state-action wise as $\mathcal{R} = \times_{s\in\mathcal{S},a\in\mathcal{A}}\mathcal{R}_{s,a}$ and $\mathcal{P} = \times_{s\in\mathcal{S},a\in\mathcal{A}}\mathcal{P}_{s,a}$, known as ($\mathtt{sa}$)-rectangular uncertainty set. Observe that it is a special case of $\mathtt{s}$-rectangular uncertainty set. Under the $\mathtt{s}$-rectangularity assumption, many structural properties of MDPs stay intact, and the problem becomes tractable [17, 9, 12].Throughout the paper, we assume that the uncertainty set $\mathcal{U}$ is $\mathtt{s}$-rectangular (or ($\mathtt{sa}$)-rectangular) unless stated otherwise. Under $\mathtt{s}$-rectangularity, the robust value function is well defined [12, 17, 9] as

$$v^\pi_\mathcal{U} := \min_{R,P\in\mathcal{U}} v^\pi_{P,R}. \tag{6}$$

Using the robust value function, robust policy performance can be rewritten as

$$\min_{R,P\in\mathcal{U}} \langle \mu, v^\pi_{P,R}\rangle = \langle \mu, \min_{R,P\in\mathcal{U}} v^\pi_{P,R}\rangle = \langle \mu, v^\pi_\mathcal{U}\rangle. \tag{7}$$

The robust value function $v^\pi_\mathcal{U}$ is the fixed point of the robust Bellmen operator $\mathcal{T}^\pi_\mathcal{U}$ [17, 9], defined as

$$(\mathcal{T}^\pi_\mathcal{U} v)(s) = \min_{R,P\in\mathcal{U}} \sum_a \pi(a|s)\left[R(s,a) + \gamma \sum_{s'} P(s'|s,a)v(s')\right]. \tag{8}$$

Moreover, the optimal robust value function $v^*_\mathcal{U} := \max_\pi v^\pi_\mathcal{U}$ is well defined and is the fixed point of the optimal robust Bellman operator $\mathcal{T}^*_\mathcal{U}$ [9, 17], defined as

$$(\mathcal{T}^*_\mathcal{U} v)(s) = \max_{\pi_s\in\Delta_\mathcal{A}} \min_{R,P\in\mathcal{U}} \sum_a \pi_s(a)\ \left[\ R(s,a) + \gamma \sum_{s'} P(s'|s,a)v(s')\ \right]. \tag{9}$$

The optimal robust Bellman operator $\mathcal{T}^*_\mathcal{U}$ and robust Bellman operators $\mathcal{T}^\pi_\mathcal{U}$ are γ contraction maps for all policy π (see theorem 3.2 of [9]), that is

$$\|\mathcal{T}^*_\mathcal{U} v - \mathcal{T}^*_\mathcal{U} u\|_\infty \le \gamma\|u-v\|_\infty, \qquad \|\mathcal{T}^\pi_\mathcal{U} v - \mathcal{T}^\pi_\mathcal{U} u\|_\infty \le \gamma\|u-v\|_\infty, \qquad \forall \pi, u, v. \tag{10}$$

So for all initial values v^π_0, v^*_0, sequences defined as

$$v^\pi_{n+1} := \mathcal{T}^\pi_\mathcal{U} v^\pi_n, \qquad v^*_{n+1} := \mathcal{T}^*_\mathcal{U} v^*_n \tag{11}$$

converges linearly to their respective fixed points, that is $v^\pi_n \to v^\pi_\mathcal{U}$ and $v^*_n \to v^*_\mathcal{U}$. This makes the robust value iteration an attractive method for solving robust MDPs.

3 Method

In this section, we study $\mathtt{s}$ and ($\mathtt{sa}$)-rectangular L_p robust MDPs where the uncertainty set is constrained by L_p balls. We will see in theorem 1 that for the ($\mathtt{sa}$)-rectangular case, both policy evaluation and improvement can be done similar to non-robust MDPs with an additional reward penalty. Theorem 2 states that policy evaluation for the $\mathtt{s}$-rectangular case, is similar to ($\mathtt{sa}$)-rectangular case except the reward penalty has extra dependence on policy. This dependence on the policy makes policy improvement a challenging task, and the rest of the section is devoted to efficiently solving it. Theorem 3 provides two methods for the policy improvement for $\mathtt{s}$-rectangular case, one being binary search that

Table 1: p-variance

x	$\kappa_x(v)$	remark
p	$\|v-\omega_p\|_p$	ω_p can be found by binary search
∞	$\frac{1}{2}\left(\max_s v(s) - \min_s v(s)\right)$	Peak to peak difference
2	$\sqrt{\sum_s \left(v(s) - \frac{\sum_s v(s)}{S}\right)^2}$	Variance
1	$\sum_{i=1}^{\lfloor (S+1)/2 \rfloor} v(s_i) - \sum_{i=\lceil (S+1)/2 \rceil}^{S} v(s_i))$	Top half minus bottom half

where v is sorted, i.e. $v(s_i) \geq v(s_{i+1}) \quad \forall i$.

is well suited for general p, and the other is an algorithmic method that is tailor made for $p = 1, 2$. Theorem 4 characterizes the stochasticity in the greedy policy. That is followed by some useful properties. All the results are summarized in the table 2 3.
We begin by making a few useful definitions. Let q be such that it satisfies the Holder's equality, i.e. $\frac{1}{p}+\frac{1}{q} = 1$. Let p-variance function $\kappa_p : \mathcal{S} \to \mathbb{R}$ and p-mean function $\omega_p : \mathcal{S} \to \mathbb{R}$ be defined as

$$\kappa_p(v) := \min_{\omega \in \mathbb{R}} \|v - \omega \mathbf{1}\|_p, \qquad \omega_p(v) := arg\min_{\omega \in \mathbb{R}} \|v - \mathbf{1}\|_p. \tag{12}$$

$\omega_p(v)$ can be calculated by binary search in the range $[\min_s v(s), \max_s v(s)]$ and can then be used to approximate $\kappa_p(v)$ (see appendix B). Observe that for $p = 1, 2, \infty$, the p-variance function κ_p can also be computed in closed form, see table 1 for summary and proofs are in appendix B. And proofs of all upcoming results in this section can be found in the appendix D.

3.1 (Sa)-rectangular L_p robust Markov Decision Processes

In accordance with [3], we define (sa)-rectangular L_p constrained uncertainty set as

$$\mathcal{U}_p^{\mathtt{sa}} := (R_0 + \mathcal{R}) \times (P_0 + \mathcal{P}) \quad \text{where}$$

$$\mathcal{R} = \times_{s \in \mathcal{S}, a \in \mathcal{A}} \mathcal{R}_{s,a}, \qquad \mathcal{R}_{s,a} = \{r_{s,a} \in \mathbb{R} \mid \|r_{s,a}\|_p \leq \alpha_{s,a}\}$$

$$\mathcal{P} = \times_{s \in \mathcal{S}, a \in \mathcal{A}} \mathcal{P}_{s,a} \qquad \mathcal{P}_{s,a} = \{p_{s,a} : \mathcal{S} \to \mathbb{R} \mid \underbrace{\sum_{s'} p_{s,a}(s') = 0}_{\text{condition A}}, \|p_{s,a}\|_p \leq \beta_{s,a}\},$$

and $\alpha_{s,a}, \beta_{s,a} \in \mathbb{R}$ are reward noise radius and transition kernel noise radius respectively. These are chosen small enough so that all the transition kernels in $(P_0 + \mathcal{P})$ are well defined. Since the sum of rows of valid transition kernel must be one, hence the sum of rows of noise kernel must be zero (condition A), that we ensured in the above definition of $\mathcal{P}_{s,a}$. It makes our result differ from [3], as they did not impose this condition (condition A) on their kernel noise. This renders their setting unrealistic (see corollary 4.1 in [3]) as not all transition kernels in their uncertainty set, satisfies the properties of transition kernels. And we see next and later, this condition on the noise makes reward regularizer dependent on the $\kappa_q(v)$ (q-variance of value function) in our result instead of $\|v\|_q$ (qth norm of value function v) in [3].

Theorem 1. (sa)*-rectangular L_p robust Bellman operator is equivalent to reward regularized (non-robust) Bellman operator. That is, using κ_p defined in* (12)*, we have*

$$(\mathcal{T}^{\pi}_{\mathcal{U}^{\texttt{sa}}_p} v)(s) = \sum_a \pi(a|s)[-\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v) + R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')],$$
$$(\mathcal{T}^{*}_{\mathcal{U}^{\texttt{sa}}_p} v)(s) = \max_{a\in\mathcal{A}}[-\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v) + R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')].$$

Proof. The proof mainly consists of two parts: a) Separating the noise from nominal values, along the lines of [3]. b) Then the noise to yields the term $-\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v)$. □

The above result states that robust value iteration (both policy evaluation and improvement) can be done as easily as non-robust value iteration with reward penalty. Notably, the reward penalty is not only proportional to the uncertainty radiuses but also to the $\kappa_p(v)$ that measures the variance in value function. We retrieve non-robust value iteration by putting uncertainty radiuses (i.e. $\alpha_{s,a}, \beta_{s,a}$) to zero, in the above results. Furthermore, same is true for all subsequent robust results in this paper. Model based Q-learning based on the above result, is discussed in section F, and implemented in algorithm 5 in appendix.

3.2 S-rectangular L_p robust Markov Decision Processes

We define s-rectangular L_p constrained uncertainty set as

$$\mathcal{U}^{\texttt{s}}_p = (R_0 + \mathcal{R}) \times (P_0 + \mathcal{P}), \quad \text{where} \quad \mathcal{R} = \times_{s\in\mathcal{S}}\mathcal{R}_s, \quad \mathcal{R}_s = \{r_s : \mathcal{A} \to \mathbb{R} \;\Big|\; \|r_s\|_p \le \alpha_s\},$$
$$\mathcal{P} = \times_{s\in\mathcal{S}}\mathcal{P}_s, \qquad \mathcal{P}_s = \{p_s : \mathcal{S}\times\mathcal{A} \to \mathbb{R} \;\Big|\; \sum_{s'} p_s(s',a) = 0, \forall a, \|p_s\|_p \le \beta_s\},$$

and $\alpha_s, \beta_s \in \mathbb{R}$ are reward and transition kernel respectively noise radius that are chosen small enough so that all the transition kernels in $P_0 + \mathcal{P}$ are well defined. S-rectangularity is more challenging to deal than (sa)-rectangularity, so we begin by robust policy evaluation.

Theorem 2. *(Policy Evaluation)* $\mathcal{S}$*-rectangular L_p robust Bellman operator is equivalent to reward regularized (non-robust) Bellman operator, that is*

$$(\mathcal{T}^{\pi}_{\mathcal{U}^{\texttt{s}}_p} v)(s) = -\Big(\alpha_s + \gamma\beta_s\kappa_q(v)\Big)\|\pi(\cdot|s)\|_q + \sum_a \pi(a|s)\Big(R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')\Big)$$

where κ_p is defined in (12) *and $\|\pi(\cdot|s)\|_q$ is q-norm of the vector $\pi(\cdot|s) \in \Delta_{\mathcal{A}}$.*

Proof. Proof techniques are similar to as its (sa)-rectangular counterpart. □

Observe that S-rectangular L_p robust policy evaluation is same as its (sa) counterpart except here the reward penalty has an extra dependence on the policy. This makes the policy improvement a challenging task. Before we move to policy improvement, let us define a function χ_p as

$$\chi_p(v,s) := \max\{k \mid \sum_{i=1}^{k} \big(\, Q(s,a_i) - Q(s,a_k)\,\big)^p \le \sigma^p\}, \tag{13}$$

where $\sigma = \alpha_s + \gamma\beta_s\kappa_q(v), Q(s,a) = R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')$, and $Q(s,a_1) \ge Q(s,a_2), \ge \cdots \ge Q(s,a_A)$. We will see shortly that $\chi_p(v,s)$ denotes the number of active actions at state s and for value function v.

Theorem 3. *(Policy improvement) The optimal robust Bellman operator can be evaluated in following ways.*

1. $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$ *is the solution of the following equation that can be found using binary search between* $\left[\, \max_a Q(s,a) - \sigma, \max_a Q(s,a) \,\right]$,

$$\sum_a \left(\, Q(s,a) - x \,\right)^p \mathbf{1}\left(\, Q(s,a) \geq x \,\right) = \sigma^p. \tag{14}$$

2. $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$ *and* $\chi_p(v,s)$ *can also be computed through algorithm 1.*

where $\sigma = \alpha_s + \gamma\beta_s\kappa_q(v)$, *and* $Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')$

Proof. The proof of proof of the first part, has the following main steps,

$$(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) = \max_{\pi_s \in \Delta_{\mathcal{A}}} \min_{R,P \in \mathcal{U}^s_p} \sum_a \pi_s(a)[R(s,a) + \gamma \sum_{s'} P(s'|s,a)v(s')], \qquad \text{(from definition)}$$

By solving inner loop (policy evaluation) using lemma 2, we have

$$= \max_{\pi_s \in \Delta_{\mathcal{A}}} \Big[-\left(\, \alpha_s + \gamma\beta_s\kappa_q(v) \,\right) \|\pi_s\|_q + \sum_a \pi_s(a) \left(\, R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s') \,\right) \Big]$$

$$= \max_{\pi_s \in \Delta_{\mathcal{A}}} -\alpha\|\pi_s\|_q + \langle \pi_s, b \rangle \qquad (\text{ for appropriate values of } \alpha, b).$$

The rest follows from the solution of the above optimization problem. The proof of the second part is more technical so referred to the appendix. For $p = 2$, the proof can be found in [1]. □

Algorithm 1 Algorithm to compute `s`-rectangular L_p robust optimal Bellman Operator

Input: $\sigma = \alpha_s + \gamma\beta_s\kappa_q(v), \qquad Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')$.
Output $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s), \chi_p(v,s)$
Sort $Q(s,\cdot)$ and label actions such that $Q(s,a_1) \geq Q(s,a_2), \cdots$.
Set initial value guess $\lambda_1 = Q(s,a_1) - \sigma$ and counter $k = 1$.
while $k \leq A - 1$ and $\lambda_k \leq Q(s,a_k)$ **do**
    Increment counter: $k = k + 1$
    Take λ_k to be a solution of the following

$$\sum_{i=1}^{k} \left(\, Q(s,a_i) - x \,\right)^p = \sigma^p, \quad \text{and} \quad x \leq Q(s,a_k). \tag{15}$$

end while
Return: λ_k, k

We want to note here that the algorithm 1 requires solving (15) at each iteration that can be solved using binary search too. But this seems unnecessary as we can directly solve (14) instead. The silver lining being that (15) can solve exactly in closed form for $p = 1, 2$ as it would be become linear and quadratic equation respectively. Its exact implementation is shown in algorithm 4 and 3 in appendix. Moreover, for $p = 1, \infty$, optimal robust Bellman operator can be calculated in closed form similar to (`sa`)-rectangular case above and non-robust MDPs, see table 2.

Theorem 4. *(Go To Policy) The greedy policy π w.r.t. value function v, defined as $\mathcal{T}^*_{\mathcal{U}^s_p} v = \mathcal{T}^\pi_{\mathcal{U}^s_p} v$ is a threshold policy. It takes only those actions that has positive advantage, with probability proportional to $(p-1)$th power of its advantage. That is*

$$\pi(a|s) \propto (A(s,a))^{p-1} \mathbf{1}(A(s,a) \geq 0),$$

where $A(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') - (\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$.

Proof. The proof follows from the proof of the (14). Notable special cases are $p = 1, \infty$ where it takes actions with uniform probability and the best action respectively. □

The advantage function A defined in the above theorem, is not usual advantage function that is defined as $Q(s,a) - v(s)$. But it bears a good resemblance to it, and for convenience, we name it advantage function. The above result states that the greedy policy takes actions that has positive advantage, and the property below states that the greedy policy takes top $\chi_p(v,s)$ actions in state s at value function v.

Property 1. *$\chi_p(v,s)$ is number of actions that has positive advantage, that is*

$$\chi_p(v,s) = \left| \left\{ \, a \mid (\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) \leq R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \, \right\} \right|.$$

Recall the definition of χ_p in (13), when uncertainty radiuses (α_s, β_s) are zero (essentially $\sigma = 0$ in (13)), then $\chi_p(v,s) = 1, \forall v, s$, that means, greedy policy taking the best action. In other words, all the robust results reduces to non-robust results as discussed in section 2.2 as uncertainty radius becomes zero. Now the next property shows another use of χ_p. It shows that $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$ depends on $\chi_p(v,s)$th and $(\chi_p(v,s)+1)$th best value (according to Q-value), as opposed to the best value in non-robust MDPs.

Property 2. *(Value vs Q-value) $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$ is bounded by the Q-value of χth and $(\chi+1)$th actions. That is*

$$Q(s, a_{\chi+1}) < (\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) \leq Q(s, a_\chi), \qquad \textit{where} \quad \chi = \chi_p(v,s),$$

$Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s')$, *and* $Q(s,a_1) \geq Q(s,a_2), \cdots Q(s,a_A)$.

To summarize, the robust optimal Bellman operator $\mathcal{T}^*_{\mathcal{U}^s_p}$ can be computed exactly in closed form for $p = 1, \infty$, by algorithm 3 for $p = 2$. And for general p, it can be approximated through binary search. The optimal policy π_p w.r.t. value function v for general p, is easy to obtain once the robust optimal Bellman operator $\mathcal{T}^*_{\mathcal{U}^s_p} v$ is computed. Interesting enough, for $p = 1, \infty$, it can be computed directly. Results from the above discussion are summarized in tables 1,3,2.

Model based Q-learning algorithm based on the above results, is implemented in algorithm 2 for $p = 1$ and for general p, it is presented in 6 in appendix G.

All the results in this section, we presented for arbitrary value function v and greedy policy π (w.r.t. v). But we can take $v = v^*_{\mathcal{U}}$ and $\pi = \pi^*_{\mathcal{U}}$ to be the optimal robust value function and the optimal robust policy, in the above results, to get characteristics about optimal robust value and policy, for different uncertainty sets $\mathcal{U}$.

Table 2: Optimal robust Bellman operator evaluation

$\mathcal{U}$	$(\mathcal{T}_{\mathcal{U}}^* v)(s)$	remark
$\mathcal{U}_p^s$	$\sum_a \left(Q(s,a) - (\mathcal{T}_{\mathcal{U}}^* v)(s) \right)^p \mathbf{1}(Q(s,a) \geq (\mathcal{T}_{\mathcal{U}}^* v)(s))$	Solve by binary search
	$=(\alpha_s + \gamma \beta_s \kappa_q(v))^p$	
$\mathcal{U}_1^s$	$\max_k \frac{1}{k} \left(\sum_i^k Q(s,a_i) - \alpha_s - \gamma \beta_s \kappa_\infty(v) \right)$	Highest penalized average
$\mathcal{U}_2^s$	Through algorithm 3	High average with high variance
$\mathcal{U}_\infty^s$	$-\alpha_s - \gamma \beta_s \kappa_1(v) + \max_{a \in \mathcal{A}} Q(s,a)$	Best response
$\mathcal{U}_p^{\mathtt{sa}}$	$\max_{a \in \mathcal{A}} [-\alpha_{sa} - \gamma \beta_{sa} \kappa_q(v) + Q(s,a)]$	Best penalized response

where $Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s')$ and $Q(s,a_1) \geq \cdots \geq Q(s,a_A)$

Table 3: Greedy policy at value function v

$\mathcal{U}$	$\pi(a\|s)$ s.t. $\mathcal{T}_{\mathcal{U}}^* v = \mathcal{T}_{\mathcal{U}}^\pi v$	remark
$\mathcal{U}_p^s$	$\frac{(A(s,a))^{p-1} \mathbf{1}(A(s,a) \geq 0}{\sum_a (A(s,a))^{p-1} \mathbf{1}(A(s,a) \geq 0)}$	top $\chi_p(v,s)$ actions proportional to $(p-1)$th power of its advantage
$\mathcal{U}_1^s$	$\frac{\mathbf{1}(A(s,a) \geq 0)}{\sum_a \mathbf{1}(A(s,a) \geq 0)}$	top $\chi_1(v,s)$ actions with uniform probability
$\mathcal{U}_2^s$	$\frac{A(s,a) \mathbf{1} A(s,a) \geq 0)}{\sum_a A(s,a) \mathbf{1}(A(s,a) \geq 0)}$	top $\chi_2(v,s)$ actions proportion to its advantage
$\mathcal{U}_\infty^s$	$arg\max_{a \in \mathcal{A}} Q(s,a)$	best action
$\mathcal{U}_p^{\mathtt{sa}}$	$arg\max_a [-\alpha_{sa} - \gamma \beta_{sa} \kappa_q(v) + Q(s,a)]$	best action

where $A(s,a) = Q(s,a) - (\mathcal{T}_{\mathcal{U}}^* v)(s)$ and $Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s')$.

Table 4: Time complexity

	reward cost O	action cost O	Total cost O
Non-Robust MDP	1	A	$\log(1/\epsilon)S^2A$
$\mathcal{U}_1^{\mathtt{sa}}$	S	A	$\log(1/\epsilon)S^2A$
$\mathcal{U}_2^{\mathtt{sa}}$	S	A	$\log(1/\epsilon)S^2A$
$\mathcal{U}_\infty^{\mathtt{sa}}$	S	A	$\log(1/\epsilon)S^2A$
$\mathcal{U}_1^{\mathtt{s}}$	S	$A\log(A)$	$\log(1/\epsilon)(S^2A + SA\log(A))$
$\mathcal{U}_2^{\mathtt{s}}$	S	$A\log(A)$	$\log(1/\epsilon)(S^2A + SA\log(A))$
$\mathcal{U}_\infty^{\mathtt{s}}$	S	A	$\log(1/\epsilon)S^2A$
$\mathcal{U}_p^{\mathtt{sa}}$	$S\log(S/\epsilon)$	A	$\log(1/\epsilon)\ \big(\ S^2A + S\log(S/\epsilon)\ \big)$
$\mathcal{U}_p^{\mathtt{s}}$	$S\log(S/\epsilon)$	$A\log(A/\epsilon)$	$\log(1/\epsilon)\ \big(\ S^2A + SA\log(A/\epsilon)\ \big)$

4 Time complexity

Here we study the time complexity of the value iteration of algorithms (algorithm 5 and algorithm 6) for L_p robust MDPs (when nominal reward and nominal transition kernel are known) and compare it to the non-robust counterpart. Precisely, we compare the number of iterations required for value iteration to converge to ϵ close to the optimal value function for different robust MDPs.

Value (or Q-value) iterations discussed above, have mainly two parts where they differ: a) reward cost: effective reward calculation (essentially κ_p) b) action cost: choosing the optimal actions. Non-robust MDPs requires constant time to calculate reward as there is no penalty, and it takes $O(A)$ time to compute action as we need to find the best action. For $p = 1, 2, \infty$, calculation of κ_p and evaluation of robust Bellman operator is done exactly. Different robust MDPs requires different operations such as sorting of actions, sorting of value functions, calculation of best action, variance of value function ,etc that accounts for different complexity. For general p, both evaluation of κ_p and robust Bellman operator is done approximately through binary search. That leads to an increased complexity, yet the increase is not significant. The results are summarized in the comparison table 4 and the proofs are in appendix E. Comparing the complexity in table 4, shows that the L_p robust MDPs are not harder than non-robust MDPs.

5 Conclusion and future work

This work develops methodology for policy improvement for $\mathtt{s}$-rectangular L_p robust MDPs, and fully characterizes the greedy policy. It also corrects the noise condition in transition kernel that leads to a different reward regularizer that depends on q-variance of the value function for L_p robust MDPs. The work concludes that L_p robust MDPs are computationally as easy as non-robust MDPs, and adds a concrete link between between regularized MDPs and L_p robust MDPs. Further, the p-mean function κ_p, for $p = 2, \infty$, can be easily estimated in online fashion. This paves the path for sample based (model free) algorithms for ($\mathtt{sa}$)/$\mathtt{s}$-rectangular L_1/L_2 robust MDPs that can be implemented using deep neural networks (such as DQN). We leave its analysis for the future work. We studied L_p constrained rectangular uncertainty set, we leave for the future work to extend this work to the other uncertainty sets.

Algorithm 2 Model based algorithm for **s**-recantangular L_1 robust MDPs

Take initial value function v_0 randomly and start the counter $n = 0$.
while not converged **do**
  Calculate q-variance: $\kappa_n = \frac{1}{2}\left[\max_s v_n(s) - \min_s v_n(s)\right]$
  for $s \in \mathcal{S}$ **do**
    for $a \in \mathcal{A}$ **do**
      Update Q-value as

$$Q_n(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v_n(s'). \tag{16}$$

    end for
    Sort actions in state s, in decreasing order of the Q-value, that is

$$Q_n(s,a_1) \geq Q_n(s,a_2), \cdots \geq Q_n(s,a_A). \tag{17}$$

    Value evaluation:

$$v_{n+1}(s) = \max_m \frac{\sum_{i=1}^m Q_n(s,a_i) - \alpha_s - \beta_s \gamma \kappa_n}{m}. \tag{18}$$

    Value evaluation can also be done using algorithm 4.
  end for

$$n \to n+1$$

end while

References

[1] Oren Anava and Kfir Levy. k*-nearest neighbors: From global to local. *Advances in neural information processing systems*, 29, 2016.

[2] J. Andrew Bagnell, Andrew Y. Ng, and Jeff G. Schneider. Solving uncertain markov decision processes. Technical report, Carnegie Mellon University, 2001.

[3] Esther Derman, Matthieu Geist, and Shie Mannor. Twice regularized mdps and the equivalence between robustness and regularization, 2021.

[4] Esther Derman and Shie Mannor. Distributional robustness and regularization in reinforcement learning, 2020.

[5] Benjamin Eysenbach and Sergey Levine. Maximum entropy rl (provably) solves some robust rl problems, 2021.

[6] Grani Adiwena Hanasusanto and Daniel Kuhn. Robust data-driven dynamic programming. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[7] Chin Pang Ho, Marek Petrik, and Wolfram Wiesemann. Partial policy iteration for l1-robust markov decision processes, 2020.

[8] Hisham Husain, Kamil Ciosek, and Ryota Tomioka. Regularized policies are reward robust, 2021.

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

[10] David L. Kaufman and Andrew J. Schaefer. Robust modified policy iteration. *INFORMS J. Comput.*, 25:396–410, 2013.

[11] Shie Mannor, Duncan Simester, Peng Sun, and John N. Tsitsiklis. Bias and variance in value function estimation. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 72, New York, NY, USA, 2004. Association for Computing Machinery.

[12] Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Oper. Res.*, 53:780–798, 2005.

[13] Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krähenbühl, Vladlen Koltun, and Dawn Song. Assessing generalization in deep reinforcement learning, 2018.

[14] Martin L. Puterman. Markov decision processes: Discrete stochastic dynamic programming. In *Wiley Series in Probability and Statistics*, 1994.

[15] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[16] Aviv Tamar, Shie Mannor, and Huan Xu. Scaling up robust mdps using function approximation. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 181–189, Bejing, China, 22–24 Jun 2014. PMLR.

[17] Berç Rustem Wolfram Wiesemann, Daniel Kuhn. Robust markov decision processes. *Mathematics of Operations Research 38(1):153-183*, 2012.

[18] Chenyang Zhao, Olivier Sigaud, Freek Stulp, and Timothy M. Hospedales. Investigating generalisation in continuous deep reinforcement learning, 2019.

A How to read appendix

1. All the proofs of the main body of the paper is presented in the section D and E.
2. Section B contains helper results for section D. Particularly, it discusses p-mean function ω_p and p-variance function κ_p.
3. Section C contains helper results for section D. Particularly, it discusses L_p water pouring lemma, necessary to evaluate robust optimal Bellman operator (learning) for $\mathtt{s}$-rectangular L_p robust MDPs.
4. Section E contains time complexity proof for model based algorithms.
5. Section F develops Q-learning machinary for ($\mathtt{sa}$)-rectangular L_p robust MDPs based on the results in the main section. It is not used in the main body or anywhere else, but this provides a good understanding for algorithms proposed in section G for ($\mathtt{sa}$)-rectangular case.
6. Section G contains model based algorithms for $\mathtt{s}$ and ($\mathtt{sa}$)-rectangular L_p robust MDPs. It also contains, remarks for special cases for $p = 1, 2, \infty$.

B p-variance

Recall that κ_p is defined as following

$$\kappa_p(v) = \min_w \|v - \omega \mathbf{1}\|_p = \|v - \omega_p\|_p.$$

Now, observe that

$$\begin{aligned} & \frac{\partial \|v-\omega\|_p}{\partial \omega} = 0 \\ \implies & \sum_s sign(v(s)-\omega)|v(s)-\omega|^{p-1} = 0, \\ \implies & \sum_s sign(v(s)-\omega_p(v))|v(s)-\omega_q(v)|^{p-1} = 0. \end{aligned} \tag{19}$$

For $p = \infty$, we have

$$\lim_{p\to\infty} \Big| \sum_s sign\ \big(\ v(s) - \omega_\infty(v)\ \big) \big|\ v(s) - \omega_\infty(v)\ \big|^p \Big|^{\frac{1}{p}} = 0$$

$$\implies \big(\ \max_s |v(s) - \omega_\infty(v)|\ \big) \lim_{p\to\infty} \Big| \sum_s sign\ \big(\ v(s) - \omega_\infty(v)\ \big) \Big(\frac{|v(s) - \omega_\infty(v)\ \big|}{\max_s |v(s) - \omega_\infty(v)|}\ \Big)^p \Big|^{\frac{1}{p}} = 0$$

Assuming $\max_s |v(s) - \omega_\infty(v)| \neq 0$ otherwise $\omega_\infty = v(s) = v(s'), \quad \forall s, s'$

$$\implies \lim_{p\to\infty} \Big| \sum_s sign\ \big(\ v(s) - \omega_\infty(v)\ \big) \Big(\frac{|v(s) - \omega_\infty(v)\ \big|}{\max_s |v(s) - \omega_\infty(v)|}\ \Big)^p \Big|^{\frac{1}{p}} = 0$$

To avoid technical complication, we assume $\max_s v(s) > v(s) < \min_s v(s), \quad \forall s$

$$\implies \lim_{p\to\infty} |\max_s v(s) - \omega_\infty(v)| = \lim_{p\to\infty} |\min_s v(s) - \omega_\infty(v)|$$

$$\implies \max_s v(s) - \lim_{q\to\infty} \omega_\infty(v) = -(\min_s v(s) - \lim_{p\to\infty} \omega_\infty(v)), \qquad \text{(managing signs)}$$

$$\implies \lim_{p\to\infty} \omega_\infty(v) = \frac{\max_s v(s) + \min_s v(s)}{2}. \tag{20}$$

$$\begin{aligned} \kappa_\infty(v) =& \|v - \omega_\infty \mathbf{1}\|_\infty \\ =& \|v - \frac{\max_s v(s) + \min_s v(s)}{2}\mathbf{1}\|_\infty, \qquad \text{(putting in value of } \omega_\infty) \\ =& \frac{\max_s v(s) - \min_s v(s)}{2} \end{aligned} \tag{21}$$

For $p = 2$, we have

$$\begin{aligned} \kappa_2(v) =& \|v - \omega_2 \mathbf{1}\|_2 \\ =& \|v - \frac{\sum_s v(s)}{S}\mathbf{1}\|_2, \\ =& \sqrt{\sum_s (v(s) - \frac{\sum_s v(s)}{S})^2} \end{aligned} \tag{22}$$

For $p = 1$, we have

$$\sum_{s\in\mathcal{S}} sign\ \big(\ v(s) - \omega_1(v)\ \big) = 0 \tag{23}$$

Note that there may be more than one values of $\omega_1(v)$ that satisfies the above equation and each solution does equally good job (as we will see later). So we will pick one (is median of v) according to our convenience as

$$\omega_1(v) = \frac{v(s_{\lfloor (S+1)/2 \rfloor}) + v(s_{\lceil (S+1)/2 \rceil})}{2} \quad \text{where} \quad v(s_i) \geq v(s_{i+1}) \quad \forall i.$$

Table 5: p-mean, where $v(s_i) \geq v(s_{i+1}) \quad \forall i$.

x	$\omega_x(v)$	remark
p	$\sum_s sign(v(s)-\omega_p(v))\lvert v(s)-\omega_p(v)\rvert^{\frac{1}{p-1}} = 0$	Solve by binary search
1	$\frac{v(s_{\lfloor (S+1)/2 \rfloor})+v(s_{\lceil (S+1)/2 \rceil})}{2}$	Median
2	$\frac{\sum_s v(s)}{S}$	Mean
∞	$\frac{\max_s v(s)+\min_s v(s)}{2}$	Average of peaks

$$
\begin{aligned}
\kappa_1(v) =& \|v-\omega_1\mathbf{1}\|_1 \\
=& \|v-med(v)\mathbf{1}\|_1, \qquad \text{(putting in value of } \omega_0\text{, see table 5)} \\
=& \sum_s |v(s)-med(v)| \\
=& \sum_{i=1}^{\lfloor (S+1)/2 \rfloor} (v(s)-med(v)) + \sum_{\lceil (S+1)/2 \rceil}^{S} (med(v)-v(s)) \\
=& \sum_{i=1}^{\lfloor (S+1)/2 \rfloor} v(s) - \sum_{\lceil (S+1)/2 \rceil}^{S} v(s)
\end{aligned} \tag{24}
$$

where $med(v) := \frac{v(s_{\lfloor (S+1)/2 \rfloor})+v(s_{\lceil (S+1)/2 \rceil})}{2}$ where $v(s_i) \geq v(s_{i+1}) \quad \forall i$ is median of v. The results are summarized in table 1 and 5.

B.1 p-variance function κ_p and kernel noise

Lemma 1. *q-variance function κ_q is the solution of the following optimization problem (kernel noise),*

$$\kappa_q(v) = -\frac{1}{\epsilon}\min_c \langle c, v\rangle, \qquad \|c\|_p \leq \epsilon, \qquad \sum_s c(s) = 0.$$

Proof. Writing Lagrangian L, as

$$L := \sum_s c(s)v(s) + \lambda \sum_s c(s) + \mu(\sum_s |c(s)|^p - \epsilon^p),$$

where $\lambda \in \mathbb{R}$ is the multiplier for the constraint $\sum_s c(s) = 0$ and $\mu \geq 0$ is the multiplier for the inequality constraint $\|c\|_q \leq \epsilon$. Taking its derivative, we have

$$\frac{\partial L}{\partial c(s)} = v(s) + \lambda + \mu p |c(s)|^{p-1} \frac{c(s)}{|c(s)|} \tag{25}$$

From the KKT (stationarity) condition, the solution c^* has zero derivative, that is

$$v(s) + \lambda + \mu p |c^*(s)|^{p-1} \frac{c^*(s)}{|c^*(s)|} = 0, \qquad \forall s \in \mathcal{S}. \tag{26}$$

Using Lagrangian derivative equation (26), we have

$$
\begin{aligned}
& v(s) + \lambda + \mu p |c^*(s)|^{p-1} \frac{c^*(s)}{|c^*(s)|} = 0 \\
\implies & \sum_s c^*(s)[v(s) + \lambda + \mu p |c^*(s)|^{p-1} \frac{c^*(s)}{|c^*(s)|}] = 0, \qquad \text{(multiply with } c^*(s) \text{ and summing)} \\
\implies & \sum_s c^*(s) v(s) + \lambda \sum_s c^*(s) + \mu p \sum_s |c^*(s)|^{p-1} \frac{(c^*(s))^2}{|c^*(s)|} = 0 \\
\implies & \langle c^*, v \rangle + \mu p \sum_s |c^*(s)|^p = 0 \qquad \text{(using } \sum_s c^*(s) = 0 \text{ and } (c^*(s))^2 = |c^*(s)|^2 \text{)} \\
\implies & \langle c^*, v \rangle = -\mu p \epsilon^p, \qquad \text{(using } \sum_s |c^*(s)|^p = \epsilon^p \text{).}
\end{aligned}
\tag{27}
$$

It is easy to see that $\mu \geq 0$, as minimum value of the objective must not be positive (at $c = 0$, the objective value is zero). Again we use Lagrangian derivative (26) and try to get the objective value $(-\mu p \epsilon^p)$ in terms of λ, as

$$
\begin{aligned}
& v(s) + \lambda + \mu p |c^*(s)|^{p-1} \frac{c^*(s)}{|c^*(s)|} = 0 \\
\implies & |c^*(s)|^{p-2} c^*(s) = -\frac{v(s) + \lambda}{\mu p}, \qquad \text{(re-arranging terms)} \\
\implies & \sum_s |(|c^*(s)|^{p-2} c^*(s))|^{\frac{p}{p-1}} = \sum_s | - \frac{v(s) + \lambda}{\mu p}|^{\frac{p}{p-1}}, \qquad \text{(doing } \sum_s |\cdot|^{\frac{p}{p-1}} \text{)} \\
\implies & \|c^*\|_p^p = \sum_s | - \frac{v(s) + \lambda}{\mu p}|^{\frac{p}{p-1}} = \sum_s |\frac{v(s) + \lambda}{\mu p}|^q = \frac{\|v + \lambda\|_q^q}{|\mu p|^q} \\
\implies & |\mu p|^q \|c^*\|_p^p = \|v + \lambda\|_q^q, \qquad \text{(re-arranging terms)} \\
\implies & |\mu p|^q \epsilon^p = \|v + \lambda\|_q^q, \qquad \text{(using } \sum_s |c^*(s)|^p = \epsilon^p \text{)} \\
\implies & \epsilon(\mu p \epsilon^{p/q}) = \epsilon \|v + \lambda\|_q \qquad \text{(taking } \frac{1}{q} \text{the power then multiplying with } \epsilon) \\
\implies & \mu p \epsilon^p = \epsilon \|v + \lambda\|_q.
\end{aligned}
\tag{28}
$$

Again, using Lagrangian derivative (26) to solve for λ, we have

$$\begin{aligned}
&v(s) + \lambda + \mu p |c^*(s)|^{p-1} \frac{c^*(s)}{|c^*(s)|} = 0 \\
\implies &|c^*(s)|^{p-2} c^*(s) = -\frac{v(s)+\lambda}{\mu p}, \qquad \text{(re-arranging terms)} \\
\implies &|c^*(s)| = |\frac{v(s)+\lambda}{\mu p}|^{\frac{1}{p-1}}, \quad \text{(looking at absolute value)} \\
&\text{and} \quad \frac{c^*(s)}{|c^*(s)|} = -\frac{v(s)+\lambda}{|v(s)+\lambda|}, \quad \text{(looking at sign: and note } \mu, p \geq 0) \\
\implies &\sum_s \frac{c^*(s)}{|c^*(s)|}|c^*(s)| = -\sum_s \frac{v(s)+\lambda}{|v(s)+\lambda|}|\frac{v(s)+\lambda}{\mu p}|^{\frac{1}{p-1}}, \qquad \text{(putting back)} \\
\implies &\sum_s c^*(s) = -\sum_s \frac{v(s)+\lambda}{|v(s)+\lambda|}|\frac{v(s)+\lambda}{\mu p}|^{\frac{1}{p-1}}, \\
\implies &\sum_s \frac{v(s)+\lambda}{|v(s)+\lambda|}|v(s)+\lambda|^{\frac{1}{p-1}} = 0, \qquad (\text{ using } \sum_i c^*(s) = 0)
\end{aligned} \tag{29}$$

Combining everything, we have

$$\begin{aligned}
&-\frac{1}{\epsilon}\min_c \langle c, v\rangle, \qquad \|c\|_p \leq \epsilon, \qquad \sum_s c(s) = 0 \\
=&\|v-\lambda\|_q, \quad \text{such that} \quad \sum_s sign(v(s)-\lambda)|v(s)-\lambda|^{\frac{1}{p-1}} = 0.
\end{aligned} \tag{30}$$

Now, observe that

$$\begin{aligned}
&\frac{\partial \|v-\lambda\|_q}{\partial \lambda} = 0 \\
\implies &\sum_s sign(v(s)-\lambda)|v(s)-\lambda|^{\frac{1}{p-1}} = 0, \\
\implies &\kappa_q(v) = \|v-\lambda\|_q, \quad \text{such that} \quad \sum_s sign(v(s)-\lambda)|v(s)-\lambda|^{\frac{1}{p-1}} = 0.
\end{aligned} \tag{31}$$

The last equality follows from the convexity of p-norm $\|\cdot\|_q$, where every local minima is global minima.

For the sanity check, we re-derive things for $p = 1$ from scratch. For $p = 1$, we have

$$\begin{aligned}
&-\frac{1}{\epsilon}\min_c \langle c, v\rangle, \qquad \|c\|_1 \leq \epsilon, \qquad \sum_s c(s) = 0. \\
=&-\frac{1}{2}(\min_s v(s) - \max_s v(s)) \\
=&\kappa_1(v).
\end{aligned} \tag{32}$$

It is easy to see the above result, just by inspection. □

B.2 Binary search for ω_p and estimation of κ_p

If the function $f : [-B/2, B/2] \to \mathbb{R}, B \in \mathbb{R}$ is monotonic (WLOG let it be monotonically decreasing) in a bounded domain, and it has a unique root x^* s.t. $f(x^*) = 0$. Then we can

find x that is an ϵ-approximation x^* (i.e. $\|x - x^*\| \leq \epsilon$) in $O(B/\epsilon)$ iterations. Why? Let $x_0 = 0$ and

$$x_{n+1} := \begin{cases} \frac{-B+x_n}{2} & \text{if} \quad f(x_n) > 0 \\ \frac{B+x_n}{2} & \text{if} \quad f(x_n) < 0 \\ x_n & \text{if} \quad f(x_n) = 0 \end{cases}.$$

It is easy to observe that $\|x_n - x^*\| \leq B(1/2)^n$. This is proves the above claim. This observation will be referred to many times.
Now, we move to the main claims of the section.

Proposition 1. *The function*

$$h_p(\lambda) := \sum_s sign\ \big(\ v(s) - \lambda\ \big) \big|\ v(s) - \lambda\ \big|^p$$

is monotonically strictly decreasing and also has a root in the range $[\min_s v(s), \max_s v(s)]$.

Proof.

$$\begin{aligned} h_p(\lambda) &= \sum_s \frac{v(s) - \lambda}{|v(s) - \lambda|} |v(s) - \lambda|^p \\ \frac{dh_p}{d\lambda}(\lambda) &= -p \sum_s |v(s) - \lambda|^{p-1} \leq 0, \qquad \forall p \geq 0. \end{aligned} \tag{33}$$

Now, observe that $h_p(\max_s v(s)) \leq 0$ and $h_p(\min_s v(s)) \geq 0$, hence by h_p must have a root in the range $[\min_s v(s), \max_s v(s)]$ as the function is continuous. □

The above proposition ensures that a root $\omega_p(v)$ can be easily found by binary search between $[\min_s v(s), \max_s v(s)]$. Precisely, ϵ approximation of $\omega_p(v)$ can be found in $O(\log(\frac{\max_s v(s) - \min_s v(s)}{\epsilon}))$ number of iterations of binary search. And one evaluation of the function h_p requires $O(S)$ iterations. And we have finite state-action space and bounded reward hence WLOG we can assume $|\max_s v(s)|, |\min_s v(s)|$ are bounded by a constant. Hence, the complexity to approximate ω_p is $O(S \log(\frac{1}{\epsilon}))$.

Let $\hat{\omega}_p(v)$ be an ϵ-approximation of $\omega_p(v)$, that is

$$\big|\ \omega_p(v) - \hat{\omega}_p(v)\ \big| \leq \epsilon.$$

And let $\hat{\kappa}_p(v)$ be approximation of $\kappa_p(v)$ using approximated mean, that is,

$$\hat{\kappa}_p(v) := \|v - \hat{\omega}_p(v)\mathbf{1}\|_p.$$

Now we will show that ϵ error in calculation of p-mean ω_p, induces $O(\epsilon)$ error in estimation of p-variance κ_p. Precisely,

$$\begin{aligned} \Big|\ \kappa_p(v) - \hat{\kappa}_p(v)\ \Big| &= \Big| \big\|\ v - \omega_p(v)\mathbf{1}\ \big\|_p - \big\|\ v - \hat{\omega}_p(v)\mathbf{1}\ \big\|_p \Big| \\ &\leq \big\|\ \omega_p(v)\mathbf{1} - \hat{\omega}_p(v)\mathbf{1}\ \big\|_p, \qquad \text{(reverse triangle inequality)} \\ &= \big\|\ \mathbf{1}\ \big\|_p \big|\ \omega_p(v) - \hat{\omega}_p(v)\ \big| \\ &\leq \big\|\ \mathbf{1}\ \big\|_p\ \epsilon \\ &= S^{\frac{1}{p}} \epsilon \leq S\epsilon. \end{aligned} \tag{34}$$

For general p, an ϵ approximation of $\kappa_p(v)$ can be calculated in $O(S\log(\frac{S}{\epsilon})$ iterations. Why? We will estimate mean ω_p to an ϵ/S tolerance (with cost $O(S\log(\frac{S}{\epsilon})$) and then approximate the κ_p with this approximated mean (cost $O(S)$).

C L_p Water Pouring lemma

In this section, we are going to discuss the following optimization problem,

$$\max_c -\alpha\|c\|_q + \langle c, b\rangle \qquad \text{such that} \qquad \sum_{i=1}^{A} c_i = 1, \quad c_i \geq 0, \quad \forall i$$

where $\alpha \geq 0$, referred as L_p-water pouring problem. We are going to assume WLOG that b is sorted component wise, that is $b_1 \geq b_2, \cdots \geq b_A$. The above problem for $p = 2$, is studied in [1]. The approach we are going to solve the problem is as follows: a) Write Lagrangian b) Since the problem is convex, any solutions of KKT condition is global maximum. c) Obtain conditions using KKT conditions.

Lemma 2. *Let $b \in \mathbb{R}^A$ be such that its components are in decreasing order (i,e $b_i \geq b_{i+1}$), $\alpha \geq 0$ be any non-negative constant, and*

$$\zeta_p := \max_c -\alpha\|c\|_q + \langle c, b\rangle \qquad \textit{such that} \qquad \sum_{i=1}^{A} c_i = 1, \quad c_i \geq 0, \quad \forall i, \tag{35}$$

and let c^* be a solution to the above problem. Then

1. *Higher components of b, gets higher weight in c^*. In other words, c^* is also sorted component wise in descending order, that is*

$$c_1^* \geq c_2^*, \cdots, \geq c_A^*.$$

2. *The value ζ_p satisfies the following equation*

$$\alpha^p = \sum_{b_i \geq \zeta_p} (b_i - \zeta_p)^p$$

3. *The solution c of (35), is related to ζ_p as*

$$c_i = \frac{(b_i - \zeta_p)^{p-1}\mathbf{1}(b_i \geq \zeta_p)}{\sum_s (b_i - \zeta_p)^{p-1}\mathbf{1}(b_i \geq \zeta_p)}$$

4. *Observe that the top $\chi_p := \max\{i | b_i \geq \zeta_p\}$ actions are active and rest are passive. The number of active actions can be calculated as*

$$\{k | \alpha^p \geq \sum_{i=1}^{k} (b_i - b_k)^p\} = \{1, 2, \cdots, \chi_p\}.$$

5. *Things can be re-written as*

$$c_i \propto \begin{cases} (b_i - \zeta_p)^{p-1} & \textit{if} \quad i \leq \chi_p \\ 0 & \textit{else} \end{cases} \qquad \textit{and} \qquad \alpha^p = \sum_{i=1}^{\chi_p} (b_i - \zeta_p)^p$$

6. *The function* $\sum_{b_i \geq x}(b_i - x)^p$ *is monotonically decreasing in* x*, hence the root* ζ_p *can be calculated efficiently by binary search between* $[b_1 - \alpha, b_1]$.

7. *Solution is sandwiched as follows*

$$b_{\chi_p+1} \leq \zeta_p \leq b_{\chi_p}$$

8. $k \leq \chi_p$ *if and only if there exist the solution of the following,*

$$\sum_{i=1}^{k}(b_i - x)^p = \alpha^p \quad \textit{and} \quad x \leq b_k.$$

9. *If action* k *is active and there is greedy increment hope then action* $k+1$ *is also active. That is*

$$k \leq \chi_p \quad \textit{and} \quad \lambda_k \leq b_{k+1} \implies k+1 \leq \chi_p,$$

where

$$\sum_{i=1}^{k}(b_i - \lambda_k)^p = \alpha^p \quad \textit{and} \quad \lambda_k \leq b_k.$$

10. *If action* k *is active, and there is no greedy hope and then action* $k+1$ *is not active. That is,*

$$k \leq \chi_p \quad \textit{and} \quad \lambda_k > b_{k+1} \implies k+1 > \chi_p,$$

where

$$\sum_{i=1}^{k}(b_i - \lambda_k)^p = \alpha^p \quad \textit{and} \quad \lambda_k \leq b_k.$$

And this implies $k = \chi_p$.

Proof. 1. Let

$$f(c) := -\alpha\|c\|_q + \langle b, c\rangle.$$

Let c be any vector, and c' be rearrangement c in descending order. Precisely,

$$c'_k := c_{i_k}, \quad \text{where} \quad c_{i_1} \geq c_{i_2}, \cdots, \geq c_{i_A}.$$

Then it is easy to see that $f(c') \geq f(c)$. And the claim follows.

2. Writting Lagrangian of the optimization problem, and its derivative,

$$\begin{aligned} L &= -\alpha\|c\|_q + \langle c, b\rangle + \lambda(\sum_i c_i - 1) + \theta_i c_i \\ \frac{\partial L}{\partial c_i} &= -\alpha\|c\|_q^{1-q}|c_i|^{q-2}c_i + b_i + \lambda + \theta_i, \end{aligned} \tag{36}$$

$\lambda \in \mathbb{R}$ is multiplier for equality constraint $\sum_i c_i = 1$ and $\theta_1, \cdots, \theta_A \geq 0$ are multipliers for inequality constraints $c_i \geq 0, \quad \forall i \in [A]$. Using KKT (stationarity) condition, we have

$$-\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* + b_i + \lambda + \theta_i = 0 \tag{37}$$

Let $\mathcal{B} := \{i | c_i^* > 0\}$, then

$$\begin{aligned}
&\sum_{i\in\mathcal{B}} c_i^*[-\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* + b_i + \lambda] = 0 \\
\implies & -\alpha\|c^*\|_q^{1-q}\|c^*\|_q^q + \langle c^*, b\rangle + \lambda = 0, \qquad (\text{using } \sum_i c_i^* = 1 \text{ and } (c_i^*)^2 = |c_i^*|^2) \\
\implies & -\alpha\|c^*\|_q + \langle c^*, b\rangle + \lambda = 0 \\
\implies & -\alpha\|c^*\|_q + \langle c^*, b\rangle = -\lambda, \qquad (\text{re-arranging})
\end{aligned} \tag{38}$$

Now again using (37), we have

$$\begin{aligned}
&-\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* + b_i + \lambda + \theta_i = 0 \\
\implies &\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* = b_i + \lambda + \theta_i, \qquad \forall i, \qquad (\text{re-arranging})
\end{aligned} \tag{39}$$

Now, if $i \in \mathcal{B}$ then $\theta_i = 0$ from complimentry slackness, so we have

$$\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* = b_i + \lambda > 0, \qquad \forall i \in \mathcal{B}$$

by definition of $\mathcal{B}$. Now, if for some i, $b_i + \lambda > 0$ then $b_i + \lambda + \theta_i > 0$ as $\theta_i \geq 0$, that implies

$$\begin{aligned}
&\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* = b_i + \lambda + \theta_i > 0 \\
&\implies c_i^* > 0 \implies i \in \mathcal{B}.
\end{aligned}$$

So, we have,

$$i \in \mathcal{B} \iff b_i + \lambda > 0.$$

To summarize, we have

$$\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* = (b_i + \lambda)\mathbf{1}(b_i \geq -\lambda), \quad \forall i, \tag{40}$$

$$\begin{aligned}
&\implies \sum_i \alpha^{\frac{q}{q-1}}\|c^*\|_q^{-q}(c_i^*)^q = \sum_i (b_i + \lambda)^{\frac{q}{q-1}}\mathbf{1}(b_i \geq -\lambda), \qquad (\text{taking } q/(q-1)\text{th power and summming}) \\
&\implies \alpha^p = \sum_{i=1}^{A}(b_i + \lambda)^p\mathbf{1}(b_i \geq -\lambda).
\end{aligned} \tag{41}$$

So, we have,

$$\begin{aligned}
&\zeta_p = -\lambda \quad \text{such that} \quad \alpha^p = \sum_{b_i \geq \lambda}(b_i + \lambda)^p. \\
&\implies \alpha^p = \sum_{b_i \geq \zeta_p}(b_i - \zeta_p)^p
\end{aligned} \tag{42}$$

3. Furthermore, using (40), we have

$$\begin{aligned}
&\alpha\|c^*\|_q^{1-q}|c_i^*|^{q-2}c_i^* = (b_i + \lambda)\mathbf{1}(b_i \geq -\lambda) = (b_i - \zeta_p)\mathbf{1}(b_i \geq \zeta_p) \quad \forall i, \\
&\implies c_i^* \propto (b_i - \zeta_p)^{\frac{1}{q-1}}\mathbf{1}(b_i \geq \zeta_p) = \frac{(b_i - \zeta_p)^{p-1}\mathbf{1}(b_i \geq \zeta_p)}{\sum_i (b_i - \zeta_p)^{p-1}\mathbf{1}(b_i \geq \zeta_p)}, \qquad (\text{using } \sum_i c_i^* = 1).
\end{aligned} \tag{43}$$

4. Now, we move on to calculate the number of active actions χ_p. Observe that the function

$$f(\lambda) := \sum_{i=1}^{A} (b_i - \lambda)^p \mathbf{1}(b_i \geq \lambda) - \alpha^p \tag{44}$$

is monotonically decreasing in λ and ζ_p is a root of f. This implies

$$\begin{aligned} & f(x) \leq 0 \iff x \geq \zeta_p \\ \implies & f(b_i) \leq 0 \iff b_i \geq \zeta_p \\ \implies & \{i | b_i \geq \zeta_p\} = \{i | f(b_i) \leq 0\} \\ \implies & \chi_p = \max\{i | b_i \geq \zeta_p\} = \max\{i | f(b_i) \leq 0\}. \end{aligned} \tag{45}$$

Hence, things follows by putting back in the definition of f.

5. We have,

$$\alpha^p = \sum_{i=1}^{A} (b_i - \zeta_p)^p \mathbf{1}(b_i \geq \zeta_p), \quad \text{and} \quad \chi_p = \max\{i | b_i \geq \zeta_p\}.$$

Combining both we have

$$\alpha^p = \sum_{i=1}^{\chi_p} (b_i - \zeta_p)^p.$$

And the other part follows directly.

6. Continuity and montonocity of the function $\sum_{b_i \geq x} (b_i - x)^p$ is trivial. Now observe that $\sum_{b_i \geq b_1} (b_i - b_1)^p = 0$ and $\sum_{b_i \geq b_1 - \alpha} (b_i - (b_1 - \alpha))^p \geq \alpha^p$, so it implies that it is equal to α^p in the range $[b_1 - \alpha, b_1]$.

7. Recall that the ζ_p is the solution to the following equation

$$\alpha^p = \sum_{b_i \geq x} (b_i - x)^p.$$

And from the definition of χ_p, we have

$$\alpha^p < \sum_{i=1}^{\chi_p + 1} (b_i - b_{\chi_p + 1})^p = \sum_{b_i \geq b_{\chi_p + 1}} (b_i - b_{\chi_p + 1})^p, \quad \text{and}$$

$$\alpha^p \geq \sum_{i=1}^{\chi_p} (b_i - b_{\chi_p})^p = \sum_{b_i \geq b_{\chi_p}} (b_i - b_{\chi_p})^p.$$

So from continuity, we infer the root ζ_p must lie between $[b_{\chi_p + 1}, b_\chi]$.

8. We prove the first direction, and assume we have

$$\begin{aligned} & k \leq \chi_p \\ \implies & \sum_{i=1}^{k} (b_i - b_k)^p \leq \alpha^p \qquad \text{(from definition of } \chi_p\text{)}. \end{aligned} \tag{46}$$

Observe the function $f(x) := \sum_{i=1}^{k}(b_i - x)^p$ is monotically decreasing in the range $(-\infty, b_k]$. Further, $f(b_k) \leq \alpha^p$ and $\lim_{x\to-\infty} f(x) = \infty$, so from the continuity argument there must exist a value $y \in (-\infty, b_k]$ such that $f(y) = \alpha^p$. This implies that

$$\sum_{i=1}^{k}(b_i - y)^p \leq \alpha^p, \quad \text{and} \quad y \leq b_k.$$

Hence, explicitly showed the existence of the solution. Now, we move on to the second direction, and assume there exist x such that

$$\sum_{i=1}^{k}(b_i - x)^p = \alpha^p, \quad \text{and} \quad x \leq b_k.$$

$$\implies \sum_{i=1}^{k}(b_i - b_k)^p \leq \alpha^p, \qquad (\text{as } x \leq b_k \leq b_{k-1} \cdots \leq b_1)$$

$$\implies k \leq \chi_p.$$

9. We have $k \leq \chi_p$ and λ_k such that

$$\begin{aligned}
\alpha^p &= \sum_{i=1}^{k}(b_i - \lambda_k)^p, \quad \text{and} \quad \lambda_k \leq b_k, \qquad (\text{from above item}) \\
&\geq \sum_{i=1}^{k}(b_i - b_{k+1})^p, \qquad (\text{as } \lambda_k \leq b_{k+1} \leq b_k) \qquad (47) \\
&\geq \sum_{i=1}^{k+1}(b_i - b_{k+1})^p, \qquad (\text{addition of 0}).
\end{aligned}$$

From the definition of χ_p, we get $k + 1 \leq \chi_p$.

10. We are given

$$\sum_{i=1}^{k}(b_i - \lambda_k)^p = \alpha^p$$

$$\implies \sum_{i=1}^{k}(b_i - b_{k+1})^p > \alpha^p, \qquad (\text{as } \lambda_k > b_{k+1})$$

$$\implies \sum_{i=1}^{k+1}(b_i - b_{k+1})^p > \alpha^p, \qquad (\text{addition of zero})$$

$$\implies k + 1 > \chi_p.$$

□

C.0.1 Special case: $p = 1$

For $p = 1$, by definition, we have

$$\zeta_1 = \max_c -\alpha\|c\|_\infty + \langle c, b\rangle \qquad \text{such that} \qquad \sum_{a\in\mathcal{A}} c_a = 1, \quad c \succeq 0. \tag{48}$$

And χ_1 is the optimal number of actions, that is

$$\alpha = \sum_{i=1}^{\chi_1}(b_i - \zeta_1)$$

$$\implies \zeta_1 = \frac{\sum_{i=1}^{\chi_1} b_i - \alpha}{\chi_1}.$$

Let λ_k be the such that

$$\alpha = \sum_{i=1}^{k}(b_i - \lambda_k)$$

$$\implies \lambda_k = \frac{\sum_{i=1}^{k} b_i - \alpha}{k}.$$

Proposition 2.

$$\zeta_1 = \max_k \lambda_k$$

Proof. From lemma 2, we have

$$\lambda_1 \le \lambda_2 \cdots \le \lambda_{\chi_1}.$$

Now, we have

$$\begin{aligned}
\lambda_k - \lambda_{k+m} &= \frac{\sum_{i=1}^{k} b_i - \alpha}{k} - \frac{\sum_{i=1}^{k+m} b_i - \alpha}{k+m} \\
&= \frac{\sum_{i=1}^{k} b_i - \alpha}{k} - \frac{\sum_{i=1}^{k} b_i - \alpha}{k+m} - \frac{\sum_{i=1}^{m} b_{k+i}}{k+m} \\
&= \frac{m(\sum_{i=1}^{k} b_i - \alpha}{k(k+m))} - \frac{\sum_{i=1}^{m} b_{k+i}}{k+m} \\
&= \frac{m}{k+m}(\frac{\sum_{i=1}^{k} b_i - \alpha}{k} - \frac{\sum_{i=1}^{m} b_{k+i}}{m}) \\
&= \frac{m}{k+m}(\lambda_k - \frac{\sum_{i=1}^{m} b_{k+i}}{m})
\end{aligned} \tag{49}$$

From lemma 2, we also know the stopping criteria for χ_1, that is

$$\lambda_{\chi_1} > b_{\chi_1+1}$$

$$\implies \lambda_{\chi_1} > b_{\chi_1+i}, \qquad i \ge 1, \qquad \text{(as } b_i \text{ are in descending order)}$$

$$\implies \lambda_{\chi_1} > \frac{\sum_{i=1}^{m} b_{\chi_1+i}}{m}, \qquad \forall m \ge 1.$$

Combining it with the (49), for all $m \geq 0$, we get

$$\begin{aligned} \lambda_{\chi_1} - \lambda_{\chi_1+m} &= \frac{m}{\chi_1 + m}(\lambda_{\chi_1} - \frac{\sum_{i=1}^{m} b_{\chi_1+i}}{m}) \\ &\geq 0 \\ \implies \lambda_{\chi_1} &\geq \lambda_{\chi_1+m} \end{aligned} \tag{50}$$

Hence, we get the desired result,

$$\zeta_1 = \lambda_{\chi_1} = \max_k \lambda_k.$$

□

C.0.2 Special case: $p = \infty$

For $p = \infty$, by definition, we have

$$\begin{aligned} \zeta_\infty(b) &= \max_c -\alpha\|c\|_1 + \langle c, b\rangle \qquad \text{such that} \qquad \sum_{a\in\mathcal{A}} c_a = 1, \quad c \succeq 0. \\ &= \max_c -\alpha + \langle c, b\rangle \qquad \text{such that} \qquad \sum_{a\in\mathcal{A}} c_a = 1, \quad c \succeq 0. \\ &= -\alpha + \max_i b_i \end{aligned} \tag{51}$$

C.0.3 Special case: $p = 2$

The problem is discussed in great details in [1], here we outline the proof. For $p = 2$, we have

$$\zeta_2 = \max_c -\alpha\|c\|_2 + \langle c, b\rangle \qquad \text{such that} \qquad \sum_{a\in\mathcal{A}} c_a = 1, \quad c \succeq 0. \tag{52}$$

Let λ_k be the solution of the following equation

$$\begin{aligned} \alpha^2 &= \sum_{i=1}^{k}(b_i - \lambda)^2, \qquad \lambda \leq b_k \\ &= k\lambda^2 - 2\sum_{i=1}^{k}\lambda b_i. + \sum_{i=1}^{k}(b_i)^2, \qquad \lambda \leq b_k \\ \implies \lambda_k &= \frac{\sum_{i=1}^{k} b_i \pm \sqrt{(\sum_{i=1}^{k} b_i)^2 - k(\sum_{i=1}^{k}(b_i)^2 - \alpha^2)}}{k}, \quad \text{and} \qquad \lambda_k \leq b_k \\ &= \frac{\sum_{i=1}^{k} b_i - \sqrt{(\sum_{i=1}^{k} b_i)^2 - k(\sum_{i=1}^{k}(b_i)^2 - \alpha^2)}}{k} \\ &= \frac{\sum_{i=1}^{k} b_i}{k} - \sqrt{\alpha^2 - \sum_{i=1}^{k}(b_i - \frac{\sum_{i=1}^{k} b_i}{k})^2} \end{aligned} \tag{53}$$

From lemma 2, we know

$$\lambda_1 \leq \lambda_2 \cdots \leq \lambda_{\chi_2} = \zeta_2$$

where χ_2 calculated in two ways: a)

$$\chi_2 = \max_m \{m | \sum_{i=1}^{m} (b_i - b_m)^2 \leq \alpha^2\}$$

b)

$$\chi_2 = \min_m \{m | \lambda_m \leq b_{m+1}\}$$

We proceed greedily until stopping condition is met in lemma 2. Concretely, it is illustrated in algorithm 3.

C.1 L_1 Water Pouring lemma

In this section, we re-derive the above water pouring lemma for $p = 1$ from scratch, just for sanity check. As in the above proof, there is a possibility of some breakdown, as we had take limits $q \to \infty$. We will see that all the above results for $p = 1$ too.

Let $b \in \mathbb{R}^A$ be such that its components are in decreasing order, i,e $b_i \geq b_{i+1}$ and

$$\zeta_1 := \max_c -\alpha \|c\|_\infty + \langle c, b \rangle \qquad \text{such that} \qquad \sum_{i=1}^{A} c_i = 1, \quad c_i \geq 0, \quad \forall i. \tag{54}$$

Lets fix any vector $c \in \mathbb{R}^A$, and let $k_1 := \lfloor \frac{1}{\max_i c_i} \rfloor$ and let

$$c_i^1 = \begin{cases} \max_i c_i & \text{if} \quad i \leq k_1 \\ 1 - k_1 \max_i c_i & \text{if} \quad i = k_1 + 1 \\ 0 & \text{else} \end{cases}$$

Then we have,

$$\begin{aligned} -\alpha \|c\|_\infty + \langle c, b \rangle &= -\alpha \max_i c_i + \sum_{i=1}^{A} c_i b_i \\ &\leq -\alpha \max_i c_i + \sum_{i=1}^{A} c_i^1 b_i, \qquad \text{(recall } b_i \text{ is in decreasing order)} \\ &= -\alpha \|c^1\|_\infty + \langle c^1, b \rangle \end{aligned} \tag{55}$$

Now, lets define $c^2 \in \mathbb{R}^A$. Let

$$k_2 = \begin{cases} k_1 + 1 & \text{if} \quad \frac{\sum_{i=1}^{k_1} b_i - \alpha}{k_1} \leq b_{k+1} \\ k_1 & \text{else} \end{cases}$$

and let $c_i^2 = \frac{\mathbf{1}(i \le k_2)}{k_2}$. Then we have,

$$\begin{aligned}
-\alpha\|c^1\|_\infty + \langle c^1, b\rangle &= -\alpha \max_i c_i + \sum_{i=1}^{A} c_i^1 b_i \\
&= -\alpha \max_i c_i + \sum_{i=1}^{k_1} \max_i c_i b_i + (1 - k_1 \max_i c_i) b_{k_1+1}, \qquad \text{(by definition of } c^1\text{)} \\
&= (\frac{-\alpha + \sum_{i=1}^{k_1} b_i}{k_1}) k_1 \max_i c_i + b_{k_1+1}(1 - k_1 \max_i c_i), \qquad \text{(re-arranging)} \\
&\le \frac{-\alpha + \sum_{i=1}^{k_2} b_i}{k_2} \\
&= -\alpha\|c^2\|_\infty + \langle c^2, b\rangle
\end{aligned} \tag{56}$$

The last inequality comes from the definition of k_2 and c^2. So we conclude that a optimal solution is uniform over some actions, that is

$$\begin{aligned}
\zeta_1 &= \max_{c \in \mathcal{C}} -\alpha\|c\|_\infty + \langle c, b\rangle \\
&= \max_k \left(\frac{-\alpha + \sum_{i=1}^{k} b_i}{k} \right)
\end{aligned} \tag{57}$$

where $\mathcal{C} := \{c^k \in \mathbb{R}^A | c_i^k = \frac{\mathbf{1}(i \le k)}{k}\}$ is set of uniform actions. Rest all the properties follows same as L_p water pouring lemma.

D Robust Value Iteration (Main)

In this section, we will discuss main results from the paper except time complexity results. It contains the proofs of the results presented in the main body and also some other corollaries/special cases.

D.1 (sa)-rectangular robust policy evaluation and improvement

Theorem. (sa)*-rectangular L_p robust Bellman operator is equivalent to reward regularized (non-robust) Bellman operator, that is*

$$\begin{aligned}
(\mathcal{T}^{\pi}_{\mathcal{U}^{\text{sa}}_p} v)(s) &= \sum_a \pi(a|s)[-\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')], \qquad \textit{and} \\
(\mathcal{T}^{*}_{\mathcal{U}^{\text{sa}}_p} v)(s) &= \max_{a \in \mathcal{A}}[-\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')],
\end{aligned}$$

where κ_p is defined in (12)*.*

Proof. From definition robust Bellman operator and $\mathcal{U}_p^{\mathtt{sa}} = (R_0 + \mathcal{R}) \times (P_0 + \mathcal{P})$, we have,

$$\begin{aligned}
(\mathcal{T}^{\pi}_{\mathcal{U}_p^{\mathtt{sa}}} v)(s) &= \min_{R,P \in \mathcal{U}_p^{\mathtt{sa}}} \sum_a \pi(a|s) \Big[R(s,a) + \gamma \sum_{s'} P(s'|s,a) v(s') \Big] \\
&= \sum_a \pi(a|s) \Big[R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big] + \\
&\qquad \min_{p \in \mathcal{P}, r \in \mathcal{R}} \sum_a \pi(a|s) \Big[r(s,a) + \gamma \sum_{s'} p(s'|s,a) v(s') \Big], \\
&\quad \text{(from (sa)-rectangularity, we get)} \\
&= \sum_a \pi(a|s) \Big[R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big] + \\
&\qquad \sum_a \pi(a|s) \underbrace{\min_{p_{s,a} \in \mathcal{P}_{sa}, r_{s,a} \in \mathcal{R}_{s,a}} \Big[r_{s,a} + \gamma \sum_{s'} p_{s,a}(s') v(s') \Big]}_{:= \Omega_{sa}(v)}
\end{aligned} \tag{58}$$

Now we focus on regularizer function Ω, as follows

$$\begin{aligned}
\Omega_{sa}(v) &= \min_{p_{s,a} \in \mathcal{P}_{s,a}, r_{s,a} \in \mathcal{R}_{s,a}} \Big[r_{s,a} + \gamma \sum_{s'} p_{s,a}(s') v(s') \Big] \\
&= \min_{r_{s,a} \in \mathcal{R}_{s,a}} r_{s,a} + \gamma \min_{p_{s,a} \in \mathcal{P}_{sa}} \sum_{s'} p_{s,a}(s') v(s') \\
&= -\alpha_{s,a} + \gamma \min_{\|p_{sa}\|_p \le \beta_{s,a}, \sum_{s'} p_{sa}(s') = 0} \langle p_{s,a}, v \rangle, \\
&= -\alpha_{s,a} - \gamma \beta_{s,a} \kappa_q(v), \qquad \text{(from lemma 1).}
\end{aligned} \tag{59}$$

Putting back, we have

$$(\mathcal{T}^{\pi}_{\mathcal{U}_p^{\mathtt{sa}}} v)(s) = \sum_a \pi(a|s) \Big[-\alpha_{s,a} - \gamma \beta_{s,a} \kappa_q(v) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big]$$

Again, reusing above results in optimal robust operator, we have

$$\begin{aligned}
(\mathcal{T}^{*}_{\mathcal{U}_p^{\mathtt{sa}}} v)(s) &= \max_{\pi_s \in \Delta_{\mathcal{A}}} \min_{R,P \in \mathcal{U}_p^{\mathtt{sa}}} \sum_a \pi_s(a) \Big[R(s,a) + \gamma \sum_{s'} P(s'|s,a) v(s') \Big] \\
&= \max_{\pi_s \in \Delta_{\mathcal{A}}} \sum_a \pi_s(a) \Big[-\alpha_{s,a} - \gamma \beta_{s,a} \kappa_p(v) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big] \\
&= \max_{a \in \mathcal{A}} \Big[-\alpha_{s,a} - \gamma \beta_{s,a} \kappa_q(v) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big]
\end{aligned} \tag{60}$$

The claim is proved. □

D.2 S-rectangular robust policy evaluation

Theorem. S*-rectangular L_p robust Bellman operator is equivalent to reward regularized (non-robust) Bellman operator, that is*

$$(\mathcal{T}^{\pi}_{\mathcal{U}_p^{\mathtt{s}}} v)(s) = -\Big(\alpha_s + \gamma \beta_s \kappa_q(v) \Big) \|\pi(\cdot|s)\|_q + \sum_a \pi(a|s) \Big(R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big)$$

where κ_p is defined in (12) *and $\|\pi(\cdot|s)\|_q$ is q-norm of the vector $\pi(\cdot|s) \in \Delta_{\mathcal{A}}$.*

Proof. From definition of robust Bellman operator and $\mathcal{U}_p^{\mathtt{s}} = (R_0 + \mathcal{R}) \times (P_0 + \mathcal{P})$, we have

$$\begin{aligned}
(\mathcal{T}^{\pi}_{\mathcal{U}_p^s} v)(s) &= \min_{R,P \in \mathcal{U}_p^s} \sum_a \pi(a|s) \Big[R(s,a) + \gamma \sum_{s'} P(s'|s,a) v(s') \Big] \\
&= \sum_a \pi(a|s) \Big[\underbrace{R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s')}_{\text{nominal values}} \Big] \\
&\qquad + \min_{p \in \mathcal{P}, r \in \mathcal{R}} \sum_a \pi(a|s) \Big[r(s,a) + \gamma \sum_{s'} p(s'|s,a) v(s') \Big] \\
&\qquad \text{(from } \mathtt{s}\text{-rectangularity we have)} \\
&= \sum_a \pi(a|s) \Big[R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big] \\
&\qquad + \underbrace{\min_{p_s \in \mathcal{P}_s, r_s \in \mathcal{R}_s} \sum_a \pi(a|s) \Big[r_s(a) + \gamma \sum_{s'} p_s(s'|a) v(s') \Big]}_{:= \Omega_s(\pi_s, v)}
\end{aligned} \tag{61}$$

where we denote $\pi_s(a) = \pi(a|s)$ as a shorthand. Now we calculate the regularizer function as follows

$$\begin{aligned}
\Omega_s(\pi_s, v) &:= \min_{r_s \in \mathcal{R}_s, p_s \in \mathcal{P}_s} \langle r_s + \gamma v^T p_s, \pi_s \rangle = \min_{r_s \in \mathcal{R}_s} \langle r_s, \pi_s \rangle + \gamma \min_{p_s \in \mathcal{P}_s} v^T p_s \pi_s \\
&= -\alpha_s \|\pi_s\|_q + \gamma \min_{p_s \in \mathcal{P}_s} v^T p_s \pi_s, \qquad \text{(using Holders inequality, where } \frac{1}{p} + \frac{1}{q} = 1 \text{)} \\
&= -\alpha_s \|\pi_s\|_q + \gamma \min_{p_s \in \mathcal{P}_s} \sum_a \pi_s(a) \langle p_{s,a}, v \rangle \\
&= -\alpha_s \|\pi_s\|_q + \gamma \min_{\sum_a (\beta_{s,a})^p \le (\beta_s)^p} \quad \min_{\|p_{sa}\|_p \le \beta_{s,a}, \sum_{s'} p_{sa}(s') = 0} \quad \sum_a \pi_s(a) \langle p_{s,a}, v \rangle \\
&= -\alpha_s \|\pi_s\|_q + \gamma \min_{\sum_a (\beta_{s,a})^p \le (\beta_s)^p} \sum_a \pi_s(a) \quad \min_{\|p_{sa}\|_p \le \beta_{s,a}, \sum_{s'} p_{sa}(s') = 0} \quad \langle p_{s,a}, v \rangle \\
&= -\alpha_s \|\pi_s\|_q + \gamma \min_{\sum_a (\beta_{sa})^p \le (\beta_s)^p} \sum_a \pi_s(a) (-\beta_{sa} \kappa_p(v)) \qquad \text{(from lemma 1)} \\
&= -\alpha_s \|\pi_s\|_q - \gamma \kappa_q(v) \max_{\sum_a (\beta_{sa})^p \le (\beta_s)^p} \sum_a \pi_s(a) \beta_{sa} \\
&= -\alpha_s \|\pi_s\|_q - \gamma \kappa_p(v) \|\pi_s\|_q \beta_s \qquad \text{(using Holders)} \\
&= -(\alpha_s + \gamma \beta_s \kappa_q(v)) \|\pi_s\|_q.
\end{aligned} \tag{62}$$

Now putting above values in robust operator, we have

$$(\mathcal{T}^{\pi}_{\mathcal{U}_p^s} v)(s) = -\Big(\alpha_s + \gamma \beta_s \kappa_q(v) \Big) \|\pi(\cdot|s)\|_q + \sum_a \pi(a|s) \Big(R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \Big).$$

□

D.3 s-rectangular robust policy improvement

Reusing robust policy evaluation results in section D.2, we have

$$
\begin{aligned}
(\mathcal{T}^*_{\mathcal{U}^{\mathtt{s}}_p} v)(s) &= \max_{\pi_s \in \Delta_{\mathcal{A}}} \min_{R,P \in \mathcal{U}^{\mathtt{sa}}_p} \sum_a \pi_s(a) \Big[R(s,a) + \gamma \sum_{s'} P(s'|s,a) v(s') \Big] \\
&= \max_{\pi_s \in \Delta_{\mathcal{A}}} \Big[-(\alpha_s + \gamma \beta_s \kappa_q(v)) \|\pi_s\|_q + \sum_a \pi_s(a) (R(s,a) + \gamma \sum_{s'} P(s'|s,a) v(s')) \Big] .
\end{aligned} \tag{63}
$$

Observe that, we have the following form

$$
(\mathcal{T}^*_{\mathcal{U}^{\mathtt{s}}_p} v)(s) = \max_c -\alpha \|c\|_q + \langle c, b \rangle \qquad \text{such that} \qquad \sum_{i=1}^{A} c_i = 1, \quad c \succeq 0, \tag{64}
$$

where $\alpha = \alpha_s + \gamma \beta_s \kappa_q(v)$ and $b_i = R(s,a_i) + \gamma \sum_{s'} P(s'|s,a_i) v(s')$. Now all the results below, follows from water pouring lemma (lemma 2).

Theorem. *(Policy improvement) The optimal robust Bellman operator can be evaluated in following ways.*

1. $(\mathcal{T}^*_{\mathcal{U}^{s}_p} v)(s)$ *is the solution of the following equation that can be found using binary search between* $\big[\max_a Q(s,a) - \sigma, \max_a Q(s,a) \big]$,

$$
\sum_a \big(Q(s,a) - x \big)^p \, \mathbf{1} \big(Q(s,a) \geq x \big) = \sigma^p . \tag{65}
$$

2. $(\mathcal{T}^*_{\mathcal{U}^{s}_p} v)(s)$ *and* $\chi_p(v,s)$ *can also be computed through algorithm 1.*

where $\sigma = \alpha_s + \gamma \beta_s \kappa_q(v)$, *and* $Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s')$.

Proof. The first part follows from lemma 2, point 2. The second part follows from lemma 2, point 9 (greedy inclusion) and point 10 (stopping condition). □

Theorem. *(Go To Policy) The greedy policy* π *w.r.t. value function* v, *defined as* $\mathcal{T}^*_{\mathcal{U}^{s}_p} v = \mathcal{T}^{\pi}_{\mathcal{U}^{s}_p} v$ *is a threshold policy. It takes only those actions that has positive advantage, with probability proportional to* $(p-1)$*th power of its advantage. That is*

$$
\pi(a|s) \propto (A(s,a))^{p-1} \mathbf{1}(A(s,a) \geq 0),
$$

where $A(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') - (\mathcal{T}^*_{\mathcal{U}^{s}_p} v)(s)$.

Proof. Follows from lemma 2, point 3. □

Property. $\chi_p(v,s)$ *is number of actions that has positive advantage, that is*

$$
\chi_p(v,s) = \Big| \big\{ \, a \mid (\mathcal{T}^*_{\mathcal{U}^{s}_p} v)(s) \leq R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v(s') \, \big\} \Big| .
$$

Proof. Follows from lemma 2, point 4. □

Property. *(Value vs Q-value)* $(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s)$ *is bounded by the Q-value of* χ*th and* $(\chi+1)$*th actions. That is*

$$Q(s, a_{\chi+1}) < (\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) \leq Q(s, a_\chi), \qquad \textit{where} \quad \chi = \chi_p(v, s),$$

$Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')$*, and* $Q(s, a_1) \geq Q(s, a_2), \cdots Q(s, a_A)$.

Proof. Follows from lemma 2, point 7. □

Corollary 1. *For* $p = 1$*, the optimal policy* π_1 *w.r.t. value function* v *and uncertainty set* $\mathcal{U}^s_1$*, can be computed directly using* $\chi_1(s)$ *without calculating advantage function. That is*

$$\pi_1(a^s_i|s) = \frac{\mathbf{1}(i \leq \chi_1(s))}{\chi_1(s)}.$$

Proof. Follows from theorem 4 by putting $p = 1$. Note that it can be directly obtained using L_1 water pouring lemma (see section C.1) □

Corollary 2. *(For* $p = \infty$*) The optimal policy* π *w.r.t. value function* v *and uncertainty set* $\mathcal{U}^s_\infty$ *(precisely* $\mathcal{T}^*_{\mathcal{U}^s_\infty} v = \mathcal{T}^\pi_{\mathcal{U}^s_\infty} v$*), is to play the best response, that is*

$$\pi(a|s) = \frac{\mathbf{1}(a \in arg\max_a Q(s,a))}{\mid arg\max_a Q(s,a) \mid}.$$

In case of tie in the best response, it is optimal to play any of the best responses with any probability.

Proof. Follows from theorem 4 by taking limit $p \to \infty$. □

Corollary 3. *For* $p = \infty$*,* $\mathcal{T}^*_{\mathcal{U}^s_p} v$*, the robust optimal Bellman operator evaluation can be obtained in closed form. That is*

$$(\mathcal{T}^*_{\mathcal{U}^s_\infty} v)(s) = \max_a Q(s,a) - \sigma,$$

where $\sigma = \alpha_s + \gamma\beta_s\kappa_1(v), Q(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a)v(s')$.

Proof. Let π be such that

$$\mathcal{T}^*_{\mathcal{U}^s_\infty} v = \mathcal{T}^\pi_{\mathcal{U}^s_\infty} v.$$

This implies

$$\begin{aligned}(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) &= \min_{R,P \in \mathcal{U}^{\text{sa}}_p} \sum_a \pi(a|s) \Big[R(s,a) + \gamma \sum_{s'} P(s'|s,a)v(s') \Big] \\ &= -(\alpha_s + \gamma\beta_s\kappa_p(v))\|\pi(\cdot|s)\|_q + \sum_a \pi(a|s)(R(s,a) + \gamma \sum_{s'} P(s'|s,a)v(s')).\end{aligned} \tag{66}$$

From corollary 2, we know the that π is deterministic best response policy. Putting this we get the desired result.
There is a another way of proving this, using theorem 3 by taking limit $p \to \infty$ carefully as

$$\lim_{p\to\infty} \sum_a \Big(Q(s,a) - \mathcal{T}^*_{\mathcal{U}^s_p} v)(s)\Big)^p \mathbf{1}\Big(Q(s,a) \geq \mathcal{T}^*_{\mathcal{U}^s_p} v)(s)\Big))^{\frac{1}{p}} = \sigma, \tag{67}$$

where $\sigma = \alpha_s + \gamma\beta_s\kappa_1(v)$. □

Corollary 4. *For $p = 1$, the robust optimal Bellman operator $\mathcal{T}^*_{\mathcal{U}^s_p}$, can be computed in closed form. That is*

$$(\mathcal{T}^*_{\mathcal{U}^s_p} v)(s) = \max_k \frac{\sum_{i=1}^k Q(s,a_i) - \sigma}{k},$$

where $\sigma = \alpha_s + \gamma\beta_s\kappa_\infty(v), Q(s,a) = R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')$, and $Q(s,a_1) \geq Q(s,a_2), \geq \cdots \geq Q(s,a_A)$.

Proof. Follows from section C.0.1. □

Corollary 5. *The* `s` *rectangular L_p robust Bellman operator can be evaluated for $p = 1, 2$ by algorithm 4 and algorithm 3 respectively.*

Proof. It follows from the algorithm 1, where we solve the linear equation and quadratic equation for $p = 1, 2$ respectively. For $p = 2$, it can be found in [1]. □

Algorithm 3 Algorithm to compute S-rectangular L_2 robust optimal Bellman Operator

Input: $\sigma = \alpha_s + \gamma\beta_s\kappa_2(v), \qquad Q(s,a) = R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')$.
Output $(\mathcal{T}^*_{\mathcal{U}^s_2} v)(s), \chi_2(v,s)$
Sort $Q(s,\cdot)$ and label actions such that $Q(s,a_1) \geq Q(s,a_2), \cdots$.
Set initial value guess $\lambda_1 = Q(s,a_1) - \sigma$ and counter $k = 1$.
while $k \leq A-1$ and $\lambda_k \leq Q(s,a_k)$ **do**
    Increment counter: $k = k+1$
    Update value estimate:

$$\lambda_k = \frac{1}{k}\left[\sum_{i=1}^k Q(s,a_i) - \sqrt{k\sigma^2 + (\sum_{i=1}^k Q(s,a_i))^2 - k\sum_{i=1}^k (Q(s,a_i))^2}\right]$$

end while
Return: λ_k, k

Algorithm 4 Algorithm to compute S-rectangular L_1 robust optimal Bellman Operator

Input: $\sigma = \alpha_s + \gamma\beta_s\kappa_\infty(v), \qquad Q(s,a) = R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v(s')$.
Output $(\mathcal{T}^*_{\mathcal{U}^s_1} v)(s), \chi_1(v,s)$
Sort $Q(s,\cdot)$ and label actions such that $Q(s,a_1) \geq Q(s,a_2), \cdots$.
Set initial value guess $\lambda_1 = Q(s,a_1) - \sigma$ and counter $k = 1$.
while $k \leq A-1$ and $\lambda_k \leq Q(s,a_k)$ **do**
    Increment counter: $k = k+1$
    Update value estimate:

$$\lambda_k = \frac{1}{k}\left[\sum_{i=1}^k Q(s,a_i) - \sigma\right]$$

end while
Return: λ_k, k

E Time Complexity

In this section, we will discuss time complexity of various robust MDPs and compare it with time complexity of non-robust MDPs. We assume that we have the knowledge of nominal transition kernel and nominal reward function for robust MDPs, and in case of non-robust MDPs, we assume the knowledge of the transition kernel and reward function. We divide the discussion into various parts depending upon their similarity.

E.1 Exact Value Iteration: Best Response

In this section, we will discuss non-robust MDPs, (sa)-rectangular $L_1/L_2/L_\infty$ robust MDPs and s-rectangular L_∞ robust MDPs. They all have a common theme for value iteration as follows, for the value function v, their Bellman operator ($\mathcal{T}$) evaluation is done as

$$(\mathcal{T}v)(s) = \underbrace{\max_a}_{\text{action cost}} \Big[R(s,a) + \alpha_{s,a} \underbrace{\kappa(v)}_{\text{reward penalty/cost}} + \gamma \underbrace{\sum_{s'} P(s'|s,a)v(s')}_{\text{sweep}} \Big] . \tag{68}$$

'Sweep' requires $O(S)$ iterations and 'action cost' requires $O(A)$ iterations. Note that the reward penalty $\kappa(v)$ doesn't depend on state and action. It is calculated only once for value iteration for all states. The above value update has to be done for each states , so one full update requires

$$O\Big(S(\text{action cost})(\text{sweep cost}) + \text{reward cost} \Big) = O\Big(S^2A + \text{reward cost} \Big)$$

Since the value iteration is a contraction map, so to get ϵ-close to the optimal value, it requires $O(\log(\frac{1}{\epsilon}))$ full value update, so the complexity is

$$O\Big(\log(\frac{1}{\epsilon}) \ (\ S^2A + \text{reward cost} \ ) \Big) .$$

1. **Non-robust MDPs**: The cost of 'reward is zero as there is no regularizer to compute. The total complexity is

$$O\Big(\log(\frac{1}{\epsilon}) \ (\ S^2A + 0 \ ) \Big) = O\Big(\log(\frac{1}{\epsilon}) S^2A \Big) .$$

2. **(sa)-rectangular $L_1/L_2/L_\infty$ and s-rectangular L_∞ robust MDPs**: We need to calculate the reward penalty ($\kappa_1(v)/\kappa_2(v)/\kappa_\infty$) that takes $O(S)$ iterations. As calculation of mean, variance and median, all are linear time compute. Hence the complexity is

$$O\Big(\log(\frac{1}{\epsilon}) \ (\ S^2A + S \ ) \Big) = O\Big(\log(\frac{1}{\epsilon}) S^2A \Big) .$$

E.2 Exact Value iteration: Top k response

In this section, we discuss the time complexity of s-rectangular L_1/L_2 robust MDPs as in algorithm 6. We need to calculate the reward penalty ($\kappa_\infty(v)/\kappa_2(v)$ in (82)) that takes $O(S)$ iterations. Then for each state we do: sorting of Q-values in (84), value evaluation in (85),

update Q-value in (83) that takes $O(A\log(A)), O(A), O(SA)$ iterations respectively. Hence the complexity is

= total iteration(reward cost (82) + S(sorting (84) + value evaluation (85) +Q-value(83))

$$= \log(\frac{1}{\epsilon})(S + S(A\log(A) + A + SA)$$

$$O\left(\log(\frac{1}{\epsilon})\left(S^2A + SA\log(A)\right)\right).$$

For general p, we need little caution as $k_p(v)$ can't be calculated exactly but approximately by binary search. And it is the subject of discussion for the next sections.

E.3 Inexact Value Iteration: (sa)-rectangular L_p robust MDPs ($\mathcal{U}_p^{\mathtt{sa}}$)

In this section, we will study the time complexity for robust value iteration for (sa)-rectangular L_p robust MDPs for general p. Recall, that value iteration takes best penalized action, that is easy to compute. But reward penalization depends on p-variance measure $\kappa_p(v)$, that we will estimate by $\hat{\kappa}_p(v)$ through binary search. We have inexact value iterations as

$$v_{n+1}(s) := \max_{a\in\mathcal{A}}[\alpha_{sa} - \gamma\beta_{sa}\hat{\kappa}_q(v_n) + R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v_n(s')]$$

where $\hat{\kappa}_q(v_n)$ is a ϵ_1 approximation of $\kappa_q(v_n)$, that is $|\hat{\kappa}_q(v_n) - \kappa_q(v_n)| \le \epsilon_1$. Then it is easy to see that we have bounded error in robust value iteration, that is

$$\|v_{n+1} - \mathcal{T}^*_{\mathcal{U}_p^{\mathtt{sa}}} v_n\|_\infty \le \gamma\beta_{max}\epsilon_1$$

where $\beta_{max} := \max_{s,a}\beta_{s,a}$

Proposition 3. *Let $\mathcal{T}^*_{\mathcal{U}}$ be a γ contraction map, and v^* be its fixed point. And let $\{v_n, n \ge 0\}$ be approximate value iteration, that is*

$$\|v_{n+1} - \mathcal{T}^*_{\mathcal{U}} v_n\|_\infty \le \epsilon$$

then

$$\lim_{n\to\infty}\|v_n - v^*\|_\infty \le \frac{\epsilon}{1-\gamma}$$

moreover, it converges to the $\frac{\epsilon}{1-\gamma}$ radius ball linearly, that is

$$\|v_n - v^*\|_\infty - \frac{\epsilon}{1-\gamma} \le c\gamma^n$$

where $c = \frac{1}{1-\gamma}\epsilon + \|v_0 - v^*\|_\infty$.

Proof.

$$
\begin{aligned}
\|v_{n+1} - v^*\|_\infty =& \|v_{n+1} - \mathcal{T}^*_{\mathcal{U}} v^*\|_\infty \\
=& \|v_{n+1} - \mathcal{T}^*_{\mathcal{U}} v_n + \mathcal{T}^*_{\mathcal{U}} v_n - \mathcal{T}^*_{\mathcal{U}} v^*\|_\infty \\
\leq& \|v_{n+1} - \mathcal{T}^*_{\mathcal{U}} v_n\|_\infty + \|\mathcal{T}^*_{\mathcal{U}} v_n - \mathcal{T}^*_{\mathcal{U}} v^*\|_\infty \\
\leq& \|v_{n+1} - \mathcal{T}^*_{\mathcal{U}} v_n\|_\infty + \gamma \|v_n - v^*\|_\infty, \qquad \text{(contraction)} \\
\leq& \epsilon + \gamma \|v_n - v^*\|_\infty, \qquad \text{(approximate value iteration)} \\
\implies \|v_n - v^*\|_\infty =& \sum_{k=0}^{n-1} \gamma^k \epsilon + \gamma^n \|v_0 - v^*\|_\infty, \qquad \text{(unrolling above recursion)} \\
=& \frac{1-\gamma^n}{1-\gamma} \epsilon + \gamma^n \|v_0 - v^*\|_\infty \\
=& \gamma^n [\frac{1}{1-\gamma} \epsilon + \|v_0 - v^*\|_\infty] + \frac{\epsilon}{1-\gamma}
\end{aligned}
\tag{69}
$$

Taking limit $n \to \infty$ both sides, we get

$$\lim_{n\to\infty} \|v_n - v^*\|_\infty \leq \frac{\epsilon}{1-\gamma}.$$

□

Lemma 3. *For $\mathcal{U}_p^{\mathtt{sa}}$, the total iteration cost is $\log(\frac{1}{\epsilon})S^2A + (\log(\frac{1}{\epsilon}))^2$ to get ϵ close to the optimal robust value function.*

Proof. We calculate $\kappa_q(v)$ with $\epsilon_1 = \frac{(1-\gamma)\epsilon}{3}$ tolerance that takes $O(S\log(\frac{S}{\epsilon_1}))$ using binary search (see section B.2). Now, we do approximate value iteration for $n = \log(\frac{3\|v_0 - v^*\|_\infty}{\epsilon})$. Using the above lemma, we have

$$
\begin{aligned}
\|v_n - v^*_{\mathcal{U}_p^{\mathtt{sa}}}\|_\infty =& \gamma^n [\frac{1}{1-\gamma}\epsilon_1 + \|v_0 - v^*_{\mathcal{U}_p^{\mathtt{sa}}}\|_\infty] + \frac{\epsilon_1}{1-\gamma} \\
\leq& \gamma^n [\frac{\epsilon}{3} + \|v_0 - v^*_{\mathcal{U}_p^{\mathtt{sa}}}\|_\infty] + \frac{\epsilon}{3} \\
\leq& \gamma^n \frac{\epsilon}{3} + \frac{\epsilon}{3} + \frac{\epsilon}{3} \leq \epsilon.
\end{aligned}
\tag{70}
$$

In summary, we have action cost $O(A)$, reward cost $O(S\log(\frac{S}{\epsilon}))$, sweep cost $O(S)$ and total number of iterations $O(\log(\frac{1}{\epsilon}))$. So the complexity is

$$
\begin{aligned}
&\big(\text{number of iterations}\big)\big(\text{S(actions cost) (sweep cost) + reward cost}\big) \\
&= \log(\frac{1}{\epsilon}) \ \big(\ S^2A + S\log(\frac{S}{\epsilon})\ \big) = \log(\frac{1}{\epsilon})(S^2A + S\log(\frac{1}{\epsilon}) + S\log(S)) \\
&= \log(\frac{1}{\epsilon})S^2A + S(\log(\frac{1}{\epsilon}))^2
\end{aligned}
$$

□

E.4 Inexact Value Iteration: s-rectangular L_p robust MDPs ($\mathcal{U}_p^{\mathtt{s}}$)

In this section, we study the time complexity for robust value iteration for s-rectangular L_p robust MDPs for general p (algorithm 5). Recall, that value iteration takes regularized actions and penalized reward. And reward penalization depends on q-variance measure $\kappa_q(v)$, that we will estimate by $\hat{\kappa}_q(v)$ through binary search, then again we will calculate $\mathcal{T}^*_{\mathcal{U}_p^{\mathtt{sa}}}$ by binary search with approximated $\kappa_q(v)$. Here, we have two error sources ((82), (85)) as contrast to (sa)-rectangular cases, where there was only one error source from the estimation of κ_q.

First, we account for the error caused by the first source (κ_q). Here we do value iteration with approximated q-variance $\hat{\kappa}_q$, and exact action regularizer. We have

$$v_{n+1}(s) := \lambda \quad \text{s.t.} \qquad \alpha_s + \gamma\beta_s\hat{\kappa}_q(v) = (\sum_{Q(s,a)\geq\lambda} (Q(s,a)-\lambda)^p)^{\frac{1}{p}}$$

where $Q(s,a) = R_0(s,a) + \gamma\sum_{s'} P_0(s'|s,a)v_n(s')$, and $|\hat{\kappa}_q(v_n) - \kappa_q(v_n)| \leq \epsilon_1$. Then from the next result (proposition 4), we get

$$\|v_{n+1} - \mathcal{T}^*_{\mathcal{U}_p^{\mathtt{sa}}} v_n\|_\infty \leq \gamma\beta_{max}\epsilon_1$$

where $\beta_{max} := \max_{s,a}\beta_{s,a}$

Proposition 4. *Let $\hat{\kappa}$ be an an ϵ-approximation of κ, that is $|\hat{\kappa}-\kappa| \leq \epsilon$, and let $b \in \mathbb{R}^A$ be sorted component wise, that is, $b_1 \geq, \cdots, \geq b_A$. Let λ be the solution to the following equation with exact parameter κ,*

$$\alpha + \gamma\beta\kappa = (\sum_{b_i\geq\lambda} |b_i-\lambda|^p)^{\frac{1}{p}}$$

and let $\hat{\lambda}$ be the solution of the following equation with approximated parameter $\hat{\kappa}$,

$$\alpha + \gamma\beta\hat{\kappa} = (\sum_{b_i\geq\hat{\lambda}} |b_i-\hat{\lambda}|^p)^{\frac{1}{p}},$$

then $\hat{\lambda}$ is an $O(\epsilon)$-approximation of λ, that is

$$|\lambda - \hat{\lambda}| \leq \gamma\beta\epsilon.$$

Proof. Let the function $f : [b_A, b_1] \to \mathbb{R}$ be defined as

$$f(x) := (\sum_{b_i\geq x} |b_i - x|^p)^{\frac{1}{p}}.$$

We will show that derivative of f is bounded, implying its inverse is bounded and hence

Lipschitz, that will prove the claim. Let proceed

$$\begin{aligned}\frac{df(x)}{dx} &= -(\sum_{b_i \geq x} |b_i - x|^p)^{\frac{1}{p}-1} \sum_{b_i \geq x} |b_i - x|^{p-1} \\ &= -\frac{\sum_{b_i \geq x} |b_i - x|^{p-1}}{(\sum_{b_i \geq x} |b_i - x|^p)^{\frac{p-1}{p}}} \\ &= -\Big[\frac{(\sum_{b_i \geq x} |b_i - x|^{p-1})^{\frac{1}{p-1}}}{(\sum_{b_i \geq x} |b_i - x|^p)^{\frac{1}{p}}} \Big]^{p-1} \\ &\leq -1.\end{aligned} \tag{71}$$

The inequality follows from the following relation between L_p norm,

$$\|x\|_a \geq \|x\|_b, \qquad \forall 0 \leq a \leq b.$$

It is easy to see that the function f is strictly monotone in the range $b_A, b_1]$, so its inverse is well defined in the same range. Then derivative of the inverse of the function f is bounded as

$$0 \geq \frac{d}{dx} f^-(x) \geq -1.$$

Now, observe that $\lambda = f^-(\alpha + \gamma\beta\kappa)$ and $\hat{\lambda} = f^-(\alpha + \gamma\beta\hat{\kappa})$, then by Lipschitzcity, we have

$$|\lambda - \hat{\lambda}| = |f^-(\alpha + \gamma\beta\kappa) - f^-(\alpha + \gamma\beta\hat{\kappa})| \leq \gamma\beta| - \kappa - \hat{\kappa})| \leq \gamma\beta\epsilon.$$

□

Lemma 4. *For $\mathcal{U}_p^{\mathbf{s}}$, the total iteration cost is $O\left(\log(\frac{1}{\epsilon}) \left(S^2A + SA\log(\frac{A}{\epsilon}) \right) \right)$ to get ϵ close to the optimal robust value function.*

Proof. We calculate $\kappa_q(v)$ in (82) with $\epsilon_1 = \frac{(1-\gamma)\epsilon}{6}$ tolerance that takes $O(S\log(\frac{S}{\epsilon_1}))$ iterations using binary search (see section B.2). Then for every state, we sort the Q values (as in (84)) that costs $O(A\log(A))$ iterations. In each state, to update value, we do again binary search with approximate $\kappa_q(v)$ upto $\epsilon_2 := \frac{(1-\gamma)\epsilon}{6}$ tolerance, that takes $O(\log(\frac{1}{\epsilon_2}))$ search iterations and each iteration cost $O(A)$, altogether it costs $O(A\log(\frac{1}{\epsilon_2}))$ iterations. Sorting of actions and binary search adds upto $O(A\log(\frac{A}{\epsilon}))$ iterations (action cost). So we have (doubly) approximated value iteration as following,

$$|v_{n+1}(s) - \hat{\lambda}| \leq \epsilon_1 \tag{72}$$

where

$$(\alpha_s + \gamma\beta_s\hat{\kappa}_q(v_n))^p = \sum_{Q_n(s,a) \geq \hat{\lambda}} (Q_n(s,a) - \hat{\lambda})^p$$

and

$$Q_n(s,a) = R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v_n(s'), \qquad |\hat{\kappa}_q(v_n) - \kappa_q(v_n)| \leq \epsilon_1.$$

And we do this approximate value iteration for $n = \log(\frac{3\|v_0 - v^*\|_\infty}{\epsilon})$. Now, we do error analysis. By accumulating error, we have

$$\begin{aligned}
|v_{n+1}(s) - (\mathcal{T}^*_{\mathcal{U}^s_p} v_n)(s)| \leq & |v_{n+1}(s) - \hat{\lambda}| + |\hat{\lambda} - (\mathcal{T}^*_{\mathcal{U}^s_p} v_n)(s)| \\
\leq & \epsilon_1 + |\hat{\lambda} - (\mathcal{T}^*_{\mathcal{U}^s_p} v_n)(s)|, \qquad \text{(by definition)} \\
\leq & \epsilon_1 + \gamma \beta_{\max} \epsilon_1, \qquad \text{(from proposition 4)} \\
\leq & 2\epsilon_1.
\end{aligned} \tag{73}$$

where $\beta_{max} := \max_s \beta_s, \gamma \leq 1$.

Now, we do approximate value iteration, and from proposition 3, we get

$$\|v_n - v^*_{\mathcal{U}^s_p}\| \leq \frac{2\epsilon_1}{1-\gamma} + \gamma^n [\frac{1}{1-\gamma} 2\epsilon_1 + \|v_0 - v^*_{\mathcal{U}^s_p}\|_\infty] \tag{74}$$

Now, putting the value of n, we have

$$\begin{aligned}
\|v_n - v^*_{\mathcal{U}^s_p}\|_\infty = & \gamma^n [\frac{2\epsilon_1}{1-\gamma} + \|v_0 - v^*_{\mathcal{U}^s_p}\|_\infty] + \frac{2\epsilon_1}{1-\gamma} \\
\leq & \gamma^n [\frac{\epsilon}{3} + \|v_0 - v^*_{\mathcal{U}^s_p}\|_\infty] + \frac{\epsilon}{3} \\
\leq & \gamma^n \frac{\epsilon}{3} + \frac{\epsilon}{3} + \frac{\epsilon}{3} \leq \epsilon.
\end{aligned} \tag{75}$$

To summarize, we do $O(\log(\frac{1}{\epsilon}))$ full value iterations. Cost of evaluating reward penalty is $O(S \log(\frac{S}{\epsilon}))$. For each state: evaluation of Q-value from value function requires $O(SA)$ iterations, sorting the actions according Q-values requires $O(A \log(A))$ iterations, and binary search for evaluation of value requires $O(A \log(1/\epsilon)$. So the complexity is

O((total iterations)(reward cost + S(Q-value + sorting + binary search for value)))

$$\begin{aligned}
&= O\left(\log(\frac{1}{\epsilon}) \left(S \log(\frac{S}{\epsilon}) + S(SA + A\log(A) + A\log(\frac{1}{\epsilon})) \right) \right) \\
&= O\left(\log(\frac{1}{\epsilon}) \left(S \log(\frac{1}{\epsilon}) + S\log(S) + S^2 A + SA\log(A) + SA\log(\frac{1}{\epsilon}) \right) \right) \\
&= O\left(\log(\frac{1}{\epsilon}) \left(S^2 A + SA\log(A) + SA\log(\frac{1}{\epsilon}) \right) \right) \\
&= O\left(\log(\frac{1}{\epsilon}) \left(S^2 A + SA\log(\frac{A}{\epsilon}) \right) \right)
\end{aligned}$$

□

F Q-Learning for (sa)-rectangular MDPs

In view of theorem 1, we can define $Q^\pi_{\mathcal{U}^{\texttt{sa}}_p}$, the robust Q-values under policy π for (sa)-rectangular L_p constrained uncertainty set $\mathcal{U}^{\texttt{sa}}_p$ as

$$Q^\pi_{\mathcal{U}^{\texttt{sa}}_p}(s,a) := -\alpha_{s,a} - \gamma \beta_{s,a} \kappa_q(v^\pi_{\mathcal{U}^{\texttt{sa}}_p}) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v^\pi_{\mathcal{U}^{\texttt{sa}}_p}(s'). \tag{76}$$

This implies that we have the following relation between robust Q-values and robust value function, same as its non-robust counterparts,

$$v^{\pi}_{\mathcal{U}^{\mathtt{sa}}_p}(s) = \sum_a \pi(a|s) Q^{\pi}_{\mathcal{U}^{\mathtt{sa}}_p}(s,a). \tag{77}$$

Let $Q^*_{\mathcal{U}^{\mathtt{sa}}_p}$ denote the optimal robust Q-values associated with optimal robust value $v^*_{\mathcal{U}^{\mathtt{sa}}_p}$, given as

$$Q^*_{\mathcal{U}^{\mathtt{sa}}_p}(s,a) := -\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v^*_{\mathcal{U}^{\mathtt{sa}}_p}) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) v^*_{\mathcal{U}^{\mathtt{sa}}_p}(s'). \tag{78}$$

It is evident from theorem 1 that optimal robust value and optimal robust Q-values satisfies the following relation, same as its non-robust counterparts,

$$v^*_{\mathcal{U}^{\mathtt{sa}}_p}(s') = \max_{a\in\mathcal{A}} Q^*_{\mathcal{U}^{\mathtt{sa}}_p}(s,a). \tag{79}$$

Combining 79 and 78, we have optimal robust Q-value recursion as follows

$$Q^*_{\mathcal{U}^{\mathtt{sa}}_p}(s,a) = -\alpha_{s,a} - \gamma\beta_{s,a}\kappa_q(v^*_{\mathcal{U}^{\mathtt{sa}}_p}) + R_0(s,a) + \gamma \sum_{s'} P_0(s'|s,a) \max_{a\in\mathcal{A}} Q^*_{\mathcal{U}^{\mathtt{sa}}_p}(s,a). \tag{80}$$

The above robust Q-value recursion enjoys the similar properties as its non-robust counterparts.

Corollary 6. *((*$\mathtt{sa}$*)-rectangular L_p regularized Q-learning) Let*

$$Q_{n+1}(s,a) = R_0(s,a) - \alpha_{sa} - \gamma\beta_{sa}\kappa_q(v_n) + \gamma \sum_{s'} P_0(s'|s,a) \max_{a\in\mathcal{A}} Q_n(s',a),$$

where $v_n(s) = \max_{a\in\mathcal{A}} Q_n(s,a)$, then Q_n converges to $Q^*_{\mathcal{U}^{\mathtt{sa}}_p}$ linearly.

Observe that the above Q-learning equation is exactly the same as non-robust MDP except the reward penalty. Recall that $\kappa_1(v) = 0.5(\max_s v(s) - \min_s v(s))$ is difference between peak to peak values and $\kappa_2(v)$ is variance of v, that can be easily estimated. Hence, model free algorithms for ($\mathtt{sa}$)-rectangular L_p robust MDPs for $p = 1, 2$, can be derived easily from the above results. This implies that ($\mathtt{sa}$)-rectangular L_1 and L_2 robust MDPs are as easy as non-robust MDPs.

G Model Based Algorithms

In this section, we assume that we know the nominal transitional kernel and nominal reward function. Algorithm 5, algorithm 6 is model based algorithm for ($\mathtt{sa}$)-rectangular and $\mathtt{s}$ rectangular L_p robust MDPs respectively. It is explained in the algorithms, how to get deal with specail cases ($p = 1, 2, \infty$) in a easy way.

Algorithm 5 Model Based Q-Learning Algorithm for SA Recantangular L_p Robust MDP

Input: $\alpha_{s,a}, \beta_{s,a}$ are uncertainty radius in reward and transition kernel respectively in state $\mathbf{s}$ and action a. Transition kernel P and reward vector R. Take initial Q-values Q_0 randomly and $v_0(s) = \max_a Q_0(s,a)$.

while not converged **do**

Do binary search in $[\min_s v_n(s), \max_s v_n(s)]$ to get q-mean ω_n, such that

$$\sum_s \frac{(v_n(s) - \omega_n)}{|v_n(s) - \omega_n|} |v_n(s) - \omega_n|^{\frac{1}{p-1}} = 0. \tag{81}$$

Compute q-variance: $\kappa_n = \|v - \omega_n\|_q$.

Note: For $p = 1, 2, \infty$, we can compute κ_n exactly in closed from, see table 1.

for $s \in \mathcal{S}$ **do**

for $a \in \mathcal{A}$ **do**

Update Q-value as

$$Q_{n+1}(s,a) = R_0(s,a) - \alpha_{sa} - \gamma \beta_{sa} \kappa_n + \gamma \sum_{s'} P_0(s'|s,a) \max_a Q_n(s',a).$$

end for

Update value as

$$v_{n+1}(s) = \max_a Q_{n+1}(s,a).$$

end for

$$n \to n+1$$

end while

Algorithm 6 Model Based Algorithm for S Recantangular L_p Robust MDP

Take initial Q-values Q_0 and value function v_0 randomly.

Input: α_s, β_s are uncertainty radius in reward and transition kernel respectively in state $\mathbf{s}$.

while not converged **do**

Do binary search in $[\min_s v_n(s), \max_s v_n(s)]$ to get q-mean ω_n, such that

$$\sum_s \frac{(v_n(s) - \omega_n)}{|v_n(s) - \omega_n|} |v_n(s) - \omega_n|^{\frac{1}{p-1}} = 0. \tag{82}$$

Compute q-variance: $\kappa_n = \|v - \omega_n\|_q$.

Note: For $p = 1, 2, \infty$, we can compute κ_n exactly in closed from, see table 1.

for $s \in \mathcal{S}$ **do**

for $a \in \mathcal{A}$ **do**

Update Q-value as

$$Q_{n+1}(s, a) = R_0(s, a) + \gamma \sum_{s'} P_0(s'|s, a) v_{n+1}(s'). \tag{83}$$

end for

Sort actions in decreasing order of the Q-value, that is

$$Q_{n+1}(s, a_i) \geq Q_{n+1}(s, a_{i+1}). \tag{84}$$

Value evaluation:

$$v_{n+1}(s) = x \quad \text{such that} \quad (\alpha_s + \gamma \beta_s \kappa_n)^p = \sum_{Q_{n+1}(s,a_i) \geq x} |Q_{n+1}(s, a_i) - x|^p. \tag{85}$$

Note: We can compute $v_{n+1}(s)$ exactly in closed from for $p = \infty$ and for $p = 1, 2$, we can do the same using algorithm 4,3 respectively, see table 2.

end for

$$n \to n + 1$$

end while