Fast Bellman Updates for Robust MDPs

Chin Pang Ho [1 *] **Marek Petrik** [2 *] **Wolfram Wiesemann** [1 *]

Abstract

We describe two efficient, and exact, algorithms for computing Bellman updates in robust Markov decision processes (MDPs). The first algorithm uses a homotopy continuation method to compute updates for L_1-constrained s, a-*rectangular* ambiguity sets. It runs in quasi-linear time for plain L_1 norms and also generalizes to weighted L_1 norms. The second algorithm uses bisection to compute updates for robust MDPs with s-*rectangular* ambiguity sets. This algorithm, when combined with the homotopy method, also has a quasi-linear runtime. Unlike previous methods, our algorithms compute the primal solution in addition to the optimal objective value, which makes them useful in policy iteration methods. Our experimental results indicate that the proposed methods are over 1,000 times faster than Gurobi, a state-of-the-art commercial optimization package, for small instances, and the performance gap grows considerably with problem size.

1. Introduction

Markov decision processes (MDPs) provide a versatile methodology for modeling dynamic decision problems under uncertainty (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998; Puterman, 2005). By assuming that transition probabilities and rewards are known precisely, however, MDPs are sensitive to model and sample errors. In recent years, the reinforcement learning literature has studied robust MDPs (RMDPs), which assume that the transition probabilities and/or rewards are uncertain and can take on any plausible value from a so-called *ambiguity set* (also known as an *uncertainty set*) to mitigate the errors (Xu & Mannor, 2006; 2009; Mannor et al., 2012; Petrik, 2012; Hanasusanto & Kuhn, 2013; Tamar et al., 2014; Delgado et al., 2016; Petrik et al., 2016). RMDPs are reminiscent of dynamic zero-sum games: the decision maker chooses the best actions, while an adversarial nature chooses the worst plausible transition probabilities.

The majority of the RMDP literature assumes that the ambiguity set is *rectangular*, in which case the analysis and computation become particularly convenient (Iyengar, 2005; Nilim & El Ghaoui, 2005; Le Tallec, 2007; Kaufman & Schaefer, 2013; Wiesemann et al., 2013). RMDPs with rectangular ambiguity sets are optimized by stationary (that is, history-independent) policies, and they satisfy a robust Bellman optimality equation which allows the optimal policy to be computed using value or policy iteration in polynomial time (Hansen et al., 2013). Polynomial time complexity is, however, often insufficient. In all but the smallest RMDPs, computing the worst-case realization of the transition probabilities involves solving at least a linear program (LP). The runtime of LP solvers can grow cubically with the number of states. Although modern LP solvers are very efficient, solving an LP to compute the Bellman update for each state and iteration becomes prohibitively expensive even for problems with only a few hundred states.

In this paper, we develop new algorithms that mitigate the computational concerns for RMDPs with ambiguity sets constrained by plain and weighted L_1 norms. Such ambiguity sets are common in reinforcement learning and operations research for two main reasons (Iyengar, 2005; Strehl et al., 2009; Jaksch et al., 2010; Petrik & Subramanian, 2014; Taleghan et al., 2015; Petrik et al., 2016). First, it is easy to construct them from samples using Hoeffding-style bounds (Weissman et al., 2003). Second, they are convenient to work with computationally, since the worst transition probabilities can be computed using LPs.

Our main contributions are two efficient, and exact, algorithms for computing *Bellman updates* in RMDPs. The first algorithm uses a *homotopy* continuation approach (Vanderbei, 2001) for RMDPs with so-called s, a-rectangular ambiguity sets that are constrained by weighted L_1 norms. s, a-rectangular ambiguity sets assume that nature can observe the decision-maker's actions before choosing the worst plausible realization of the transition probabilities (Le Tallec, 2007; Wiesemann et al., 2013). Using these sets resem-

[*] Equal contribution [1] Imperial College London [2] Department of Computer Science, University of New Hampshire. Correspondence to: Chin Pang Ho <c.ho12@imperial.ac.uk>, Marek Petrik <mpetrik@cs.unh.edu>, Wolfram Wiesemann <ww@imperial.ac.uk>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

bles the adaptive offline adversary model. The homotopy method starts with a singleton ambiguity set, for which computing the worst response is trivial, and then it traces nature's response with the increasing ambiguity set size. Its computational complexity is $\mathcal{O}(SA \log S)$, where S is the number of states and A is the number of actions.

Our second algorithm uses a novel *bisection* approach to solve RMDPs with s-rectangular ambiguity sets. These ambiguity sets assume a weaker nature that must commit to a realization of the transition probabilities before observing the decision-maker's actions (Le Tallec, 2007; Wiesemann et al., 2013). Using s-rectangular ambiguity sets resembles the oblivious adversary model and provides less conservative solutions. They are, however, much more computationally challenging since the decision maker's optimal policy can be randomized (Wiesemann et al., 2013). When the bisection method is combined with our homotopy method for L_1-constrained ambiguity sets, its time complexity is $\mathcal{O}(SA \log(SA))$. We emphasize that the complexity is *independent* of any approximation constant ϵ which is unusual for bisection methods.

Problem-specific optimization methods are often needed to solve machine learning problems, which are large but exhibit simple structure. Quasi-linear-time algorithms for computing Bellman updates for RMDPs with L_1-constrained s, a-rectangular ambiguity sets have been proposed before (Iyengar, 2005; Strehl et al., 2009; Petrik & Subramanian, 2014). Our methods, in comparison, generalize to weighted L_1 norms and s-rectangular ambiguity sets, which are important in preventing overly conservative solutions in many data-driven settings (Nilim & El Ghaoui, 2005; Le Tallec, 2007; Wiesemann et al., 2013). The algorithms can also be used with (modified) robust policy iteration (Kaufman & Schaefer, 2013) because they compute the worst probability realizations, unlike prior work. Modified policy iteration can solve RMDPs many times faster than value iteration (Kaufman & Schaefer, 2013).

The proposed homotopy method is also related to LARS, a homotopy method for solving the LASSO problem (Drori & Donoho, 2006; Hastie et al., 2009; Murphy, 2012), and also to fast methods for computing efficient projections onto the L_1 ball (Duchi et al., 2008; Thai et al., 2015). Unlike this prior work, computing the worst transition probabilities is complicated by the need to respect constraints on transitions probabilities. Our homotopy method also works in the more general case of weighted L_1 norms, which have not been tackled previously and have a very different solution structure from the plain L_1 case. A bisection method has been previously proposed for robust MDPs, but that algorithm solves a different s, a-rectangular problem (Nilim & El Ghaoui, 2005).

The remainder of the paper is organized as follows. Section 2 describes basic RMDP models. Section 3 then introduces the new homotopy method for s, a-rectangular ambiguity sets. Section 4 describes the new bisection method for s-rectangular ambiguity sets. The bisection method takes advantage of the optimal solution paths generated by the homotopy method. Finally, Section 5 compares our algorithms with Gurobi, one of the leading commercial LP solvers. While we describe and evaluate the methods in the context of a tabular value function, they easily generalize to robust value function approximation methods (Tamar et al., 2014).

Notation: We use Δ^S to denote the probability simplex in $\mathbb{R}^S_+$. Symbols $\mathbf{1}$ and $\mathbf{0}$ denote vectors of all ones and zeros, respectively, of the size appropriate to their context.

2. Robust Bellman Updates

We consider a Robust Markov Decision Process (RMDP) with a finite number of states $\mathscr{S} = \{1, \ldots, S\}$ and actions $\mathscr{A} = \{1, \ldots, A\}$. Every action can be taken in every state. Choosing an action $a \in \mathscr{A}$ in a state $s \in \mathscr{S}$ yields a reward $r_{s,a} \in \mathbb{R}$ and results in a stochastic transition to a new state s' according to the transition probabilities $p_{s,a} \in \Delta^S$. The probability p is, however, unknown and is constrained to be in the *ambiguity set* $\mathscr{P}$ (which is also sometimes referred to as an *uncertainty set*).

A common objective in RMDPs is to compute a stationary randomized policy $\pi : \mathscr{S} \to \Delta^A$ that maximizes the return ρ for the worst plausible realization of transition probabilities:

$$\max_{\pi \in \Pi_R} \min_{p \in \mathscr{P}} \rho(\pi, p), \tag{1}$$

where Π_R is the set of all stationary randomized policies and $\rho(\pi, p)$ is the return. When assuming an infinite-horizon γ-discounted objective, the return is $\rho(\pi, p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r_{S_t, A_t}\right]$ where $S_0 = s_0$ is the initial state, the state random variables $S_0, \ldots$ are distributed according to p and actions $A_0, \ldots$ are distributed according to π. But our results also apply directly to finite-horizon problems.

The problem (1) is NP-hard in general but solvable in polynomial time when $\mathscr{P}$ is s, a-rectangular (Nilim & El Ghaoui, 2005; Iyengar, 2005). In s, a-rectangular RMDPs, ambiguity sets are defined independently for each state s and action a: $p_{s,a} \in \mathscr{P}_{s,a}$ rather than $p \in \mathscr{P}$.

We focus particularly on ambiguity sets defined with respect to a norm-bounded distance from a known *nominal transition probability* $\bar{p}$. Formally, the s, a-rectangular ambiguity set for $s \in \mathscr{S}$ and $a \in \mathscr{A}$ is $\mathscr{P}_{s,a} = \{p \in \Delta^S : \|\bar{p}_{s,a} - p\| \leq \kappa_{s,a}\}$ for a given $\kappa_{s,a} \geq 0$ and a norm $\|\cdot\|$. Recall that the Q-function in regular MDPs is defined as $q_{s,a} = r_{s,a} + \gamma \cdot p_{s,a}^{\mathsf{T}} v$ for some value function $v \in \mathbb{R}^S$. The Q-function $q \in \mathbb{R}^{S \times A}$ for s, a-rectangular RMDPs is

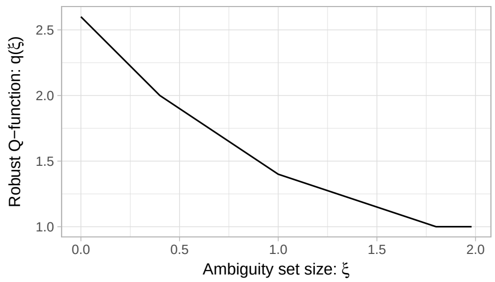


Figure 1. Example function $q_{s,a}(\xi)$ for an ambiguity set constrained by the L_1 norm.

defined as:

$$q_{s,a}(\xi) = \min_{p\in\Delta^S} \left\{ r_{s,a} + \gamma \cdot p^\mathsf{T} v \ : \ \|p - \bar{p}_{s,a}\| \le \xi \right\}. \quad (2)$$

Figure 1 depicts an example of the value of a q as a function of ξ. Finally, the optimal value function $v^\star \in \mathbb{R}^S$ in this RMDP model must satisfy the robust Bellman optimality equation (Iyengar, 2005):

$$v^\star_s = \max_{a\in\mathscr{A}} \min_{\xi \le \kappa_{s,a}} q^\star_{s,a}(\xi)\,, \quad (3)$$

where $q^\star$ is defined in terms of $v^\star$. We use ξ as an optimization parameter in the robust q function, and we use κ_s and $\kappa_{s,a}$ as robustness budgets of the RMDP.

In s-rectangular RMDPs, the ambiguity set is defined independently for each state s: $p_{s,\cdot} \in \mathscr{P}_s$. The norm constrained s-rectangular ambiguity set becomes: $\mathscr{P}_s = \{(p_1 \in \Delta^S, \ldots, p_A \in \Delta^S) \ : \ \sum_{a\in\mathscr{A}} \|\bar{p}_{s,a} - p_a\| \le \kappa_s\}$ for a given $\kappa_s \ge 0$. The optimal value function $v^\star \in \mathbb{R}^S$ must satisfy the following robust Bellman optimality equation (Wiesemann et al., 2013):

$$v^\star_s = \max_{d\in\Delta^A} \min_{\xi\in\mathbb{R}^A} \left\{ \sum_{a\in\mathscr{A}} d_a \cdot q^\star_{s,a}(\xi_a) \ : \ \sum_{a\in\mathscr{A}} \xi_a \le \kappa_s \right\} \quad (4)$$

Here, d_a represents the probability of taking an action a in this state: $\pi(s,a) = d_a$ for the state s above. Optimal policies in s-rectangular RMDPs may be randomized.

3. Homotopy for s, a-rectangular Sets

In this section, we consider the problem of efficiently computing the worst-case response of nature in RMDPs with s, a-rectangular ambiguity sets. This amounts to computing the function $q_{s,a}(\xi)$ in (2) for some state s and action a. Our homotopy method assumes that the ambiguity set emerges from the intersection of the probability simplex and a w-weighted L_1 norm ball, where the norm is defined as $\|x\|_{1,w} = \sum_{i=1}^n w_i |x_i|$. The weights $w > 0$ must be positive but need *not* sum to 1 or any other specific number.

Since our focus in this section is restricted to computing the function $q_{s,a}(\xi)$ for a fixed state s and action a, we drop the subscripts and refer to the function as $q(\xi)$, the nominal probabilities as $\bar{p} \in \Delta^S$, and the degree of ambiguity as $\kappa \in \mathbb{R}_+$. To further simplify notation, we define $z = r_{s,a}\mathbf{1} + \gamma\, v$ to obtain:

$$q(\xi) = \min_{p\in\Delta^S} \left\{ p^\mathsf{T} z \ : \ \|p - \bar{p}\|_{1,w} \le \xi \right\}. \quad (5)$$

Problem (5) can be readily formulated as a linear program:

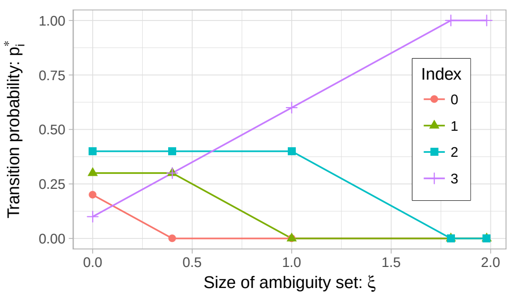
$$\begin{aligned} q(\xi) = \min_{p\in\mathbb{R}^S,\, l\in\mathbb{R}^S} \quad & z^\mathsf{T} p \\ \text{subject to} \quad & p - \bar{p} \le l && (\mathscr{U}) \\ & \bar{p} - p \le l && (\mathscr{L}) \\ & p \ge \mathbf{0} && (\mathscr{Z}) \\ & \mathbf{1}^\mathsf{T} p = 1, \quad w^\mathsf{T} l = \xi \end{aligned} \quad (6)$$

We assume in (6) that the constraint $w^\mathsf{T} l \le \xi$ is binding, which will be the case for all ξ of interest to our homotopy method. Note that when the constraint $w^\mathsf{T} l \le \xi$ is not binding, $q(\xi)$ will not change for any greater value of ξ.

Algorithm 1: Homotopy method for $q(\xi)$.

```
Input: LP parameters: z, w, p̄
Initialize ξ ← 0.0, p ← p̄, X_1 = 0 and
  Q_1 = q(0) = p̄ᵀz, k ← 2;
// Derivatives for basic solutions
for donor i = 1...S do
    for receiver j = 1...S do
        Case C1 (i ∈ ℒ'): α_{i,j} ← (z_j − z_i)/(w_i + w_j);
        Case C2 (i ∈ 𝒰'): β_{i,j} ← (z_j − z_i)/(−w_i + w_j);
    end
end
Sort (α_{i,j} and β_{i,j}) in ascending order of their
  derivatives to get the bases B_1, ..., B_T;
for l = 1...T do
    if B_l is feasible then
        Compute maximum possible increase Δξ in
          ξ for B_l to remain feasible;
        if Δξ > 0 then
            Update ξ ← ξ + Δξ, update objective
              value using derivative;
            Record breakpoint:
              k ← k + 1, X_k ← ξ, Q_k ← q(ξ);
        end
    end
end
The remainder of the function q(ξ) will be constant:
  X_{k+1} ← ∞, Q_{k+1} ← Q_k, k ← k + 1;
return Breakpoints X_{1...k} and values Q_{1...k}.
```

The basic idea of homotopy methods is to trace the optimal solution to an optimization problem while increasing the



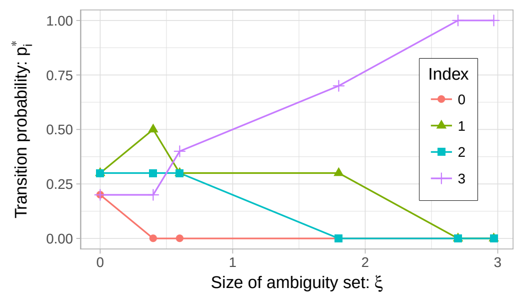


Figure 2. Example evolution of the minimizer in $q(\xi)$ when nature is limited by an unweighted (*left*) and weighted (*right*) L_1 norm. The plots show the optimal solution p for any ξ and not only where marked with points. The point markers indicate breakpoints where the solution switches to a new optimal basis.

value of some parameter (Garrigues & El Ghaoui, 2009; Asif & Romberg, 2009). One starts with a parameter value for which the problem is easy to solve. In our case, we choose $\xi = 0$ since the only feasible response of nature is $p = \bar{p}$, and solving $q(0)$ amounts to computing a Bellman update in a non-robust MDP. We then track the optimal p as the value of ξ increases. Implementing a homotopy method in the context of a linear program such as (6) is especially convenient since $q(\xi)$ and p are piecewise linear in ξ (Vanderbei, 2001). Intuitively, p is linear in ξ for each basic feasible solution in (6), and a breakpoint (or a 'knot') occurs whenever the currently optimal basis becomes infeasible. A similar argument shows that $q(\xi)$ is piecewise linear, too.

Before proving the correctness of our homotopy method, we describe the algorithm informally. The *bases* of interest, which correspond to a subset of the extreme points of the feasible set in (6), have a particularly simple structure. For each such basis, the value of exactly two components of p change as ξ increases. For p to be a valid probability distribution, it must sum to 1, and therefore one component increases and the other component decreases. We use the term *donor* for the component that decreases and donates some of its mass to the *receiver* component.

Given an optimal basis with a donor-receiver pair, we trace the optimal solution p as ξ increases. Once the basis becomes infeasible, we switch to a feasible basis whose objective value decreases fastest when increasing ξ. For example, consider a problem with uniform weights $w = \mathbf{1}$ and an increase of ξ by $\Delta\xi$. If i is the donor and j the receiver, then $\Delta p_i = -\Delta\xi/2$ and $\Delta p_j = \Delta\xi/2$, and the objective value changes by $(z_j - z_i)\Delta\xi/2$. The examples below illustrate the paths traced by the optimal solution p.

Example 1. *Consider the function $q(\xi)$ in (5) for an RMDP with 4 states, $z = [4, 3, 2, 1]$, $\bar{p} = [0.2, 0.3, 0.4, 0.1]$, and uniform weights $w = \mathbf{1}$. Figure 2(left) depicts the evolution of the optimal p as a function of ξ. The only receiver in all bases is component 3, and the donors are the components 0, 1, and 2 (in order of increasing ξ). Section 3.1 shows that for uniform w, the component with the smallest value of z is always the sole receiver.*

Example 2. *Consider the function $q(\xi)$ in (5) for an RMDP with 4 states, $z = [2.9, 0.9, 1.5, 0.0]$, $\bar{p} = [0.2, 0.3, 0.3, 0.2]$, and non-uniform weights $w = [1, 1, 2, 2]$. Figure 2(right) depicts the evolution of the optimal p as a function of ξ. The donor-receiver pairs are $(0, 1)$, $(1, 3)$, and $(2, 3)$. This example shows that when w is not uniform, several components can serve as receivers, and some components can be both receivers and donors for different values of ξ.*

The homotopy algorithm is described in Algorithm 1. We first prove the structure of the bases described above and then compute the derivatives of the optimal objective value. The section concludes by proving the optimality of the homotopy solution as well as analyzing its computational complexity.

In the following, we use the sets $\mathscr{U}, \mathscr{L}, \mathscr{Z} \subseteq \{1, \ldots, S\}$ to denote which inequalities in (6) are active in a particular basis. For example, $i \in \mathscr{U}$ indicates that $p_i - \bar{p}_i = l_i$, and $j \in \mathscr{Z}$ indicates that $p_j = 0$. The letter $\mathscr{U}$ ($\mathscr{L}$) stands for upper (lower) bounds on p, and $\mathscr{Z}$ for zero. Note that some constraints may be inactive *in the basis* and still hold with equality or even be violated.

Our homotopy approach is based on tracing the *basic feasible solutions*. Since (6) has $2S$ variables (p and l), each set of $2S$ constraints (inequalities and/or equalities) in (6) that are satisfied as equalities and that are linearly independent define a *basis*, see, *e.g.*, Definition 2.9 in Bertsimas & Tsitsiklis (1997). In particular, each basis is uniquely defined by the elements in the sets $\mathscr{U}$, $\mathscr{L}$, and $\mathscr{Z}$. We let $\mathscr{O} = \mathscr{U} \cap \mathscr{L}$ denote the components i for which both the lower and the upper bounds hold with equality, that is, for which $p_i = \bar{p}_i$. Moreover, we let $\mathscr{U}' = \mathscr{U} \setminus (\mathscr{O} \cup \mathscr{Z})$ denote the components i for which the upper bounds (but not the lower bounds) hold with equality and for which $p_i > 0$, that is, for which $p_i > \bar{p}_i$. Likewise, we let $\mathscr{L}' = \mathscr{L} \setminus (\mathscr{O} \cup \mathscr{Z})$ denote the components i for which the lower bounds (but not the upper bounds) hold with equality and for which $p_i > 0$, that is, for which $\bar{p}_i > p_i > 0$. We first show that any basis to (6) satisfies $|\mathscr{U}'| + |\mathscr{L}'| \le 2$.

Lemma 1. *Assume a* basis *(possibly infeasible) to* (6) *is defined by $\mathscr{U}$, $\mathscr{L}$ and $\mathscr{Z}$. We then have $|\mathscr{U}'| + |\mathscr{L}'| \le 2$.*

The lemma follows from algebraic manipulation and the fact that $\mathscr{O} \cap \mathscr{Z} = \emptyset$, which in turn follows from the linear independence of the constraints defining a basis. The full proof is technical and is deferred to Appendix A.1.

We now show that $\mathscr{U}' \cup \mathscr{L}'$ precisely contains the donors and receivers. From Lemma 1 we can then conclude that in any basis, there is at most one donor-receiver pair.

Lemma 2. *Assume a* basis *to* (6) *is defined by $\mathscr{U}$, $\mathscr{L}$ and $\mathscr{Z}$, and let $\dot{p}$ and $\dot{q}$ be the derivatives of p and q with respect to ξ for this basis. We then have:*

(C1) *If $\mathscr{L}' = \{i\}$ and $\mathscr{U}' = \{j\}$, then*

$$\dot{q} = \frac{z_j - z_i}{w_i + w_j}, \quad \dot{p}_i = \frac{-1}{w_i + w_j}, \quad \dot{p}_j = \frac{1}{w_i + w_j}.$$

(C2) *If $\mathscr{L}' = \emptyset$ and $\mathscr{U}' = \{i, j\}$, then*

$$\dot{q} = \frac{z_j - z_i}{-w_i + w_j}, \quad \dot{p}_i = \frac{-1}{w_i - w_j}, \quad \dot{p}_j = \frac{1}{w_i - w_j}.$$

The lemma follows from algebraic manipulation of the constraints that define a particular basis. We defer the proof to Appendix A.2. We note that there are potential bases to (6) with $\mathscr{U}' = \emptyset, \mathscr{L}' = \{i, j\}$ (Case C3), and $|\mathscr{L}'| + |\mathscr{U}'| < 2$ (Case C4) as well. This is the case when $\xi = 0$, when $\bar{p}$ solves (6) for all $\xi \in \mathbb{R}_+$, at breakpoints of the function $q(\xi)$ or when the optimal solution p satisfies $\|p - \bar{p}\|_{1,w} < \xi$. Since none of these cases is relevant for our homotopy method, we do not further elaborate on them.

Algorithm 1 summarizes the homotopy method. As explained above, it follows each basis as ξ increases as long as it is feasible. The basis becomes infeasible either because some p_i or some l_i is reduced to 0. In that case, the algorithm determines the new optimal basis. Since the function $q(\xi)$ is convex, it is only necessary to search for bases that have derivatives $\dot{q}$ no smaller than the previous basis. For ease of exposition, Algorithm 1 assumes that there are no ties between the derivatives. It is straightforward but tedious to generalize the proofs to the presence of ties.

Note that we designed Algorithm 1 to generate the entire solution path of $q(\xi)$. If the goal is to compute the function q for a particular value of ξ, this is not necessary. However, the bisection method that we describe in the next section will require the entire path in order to compute solutions to s-rectangular problems.

The following theorem states the correctness of the proposed homotopy algorithm. It shows that the function q is a piecewise linear function defined by the output generated by Algorithm 1.

Theorem 1. *Let $X_{1\ldots n}$ and $Q_{1\ldots n}$ be the output of Algorithm 1. Then, $q(\xi)$ is a piecewise linear function with breakpoints X_l that satisfies $q(X_l) = Q_l$, $l = 1, \ldots, n$.*

A detailed proof of Theorem 1 is deferred to Appendix A.3. Broadly speaking, the theorem can be proved by contradiction. Since each point X_l in the algorithm corresponds to the value of ξ for some feasible basis, the output generated by Algorithm 1 provides an upper bound on the function $q(\xi)$. Assume to the contrary that the output does not coincide point-wise with the function $q(\xi)$. In that case, there must be a basis that is not considered by the homotopy method and that has a strictly smaller derivative than all other feasible bases for some value of ξ. This, however, contradicts the way in which bases are chosen by the algorithm.

3.1. Complexity

A naive implementation of Algorithm 1 has a computational complexity of S^2 because it enumerates all pairs of indexes. Although this would be an improvement over the typical $\mathcal{O}(S^3)$ time complexity of LP algorithms, having a close to linear time algorithm is preferable. In fact, we observed numerically that the naive implementation performs on par with LP solvers and sometimes even slower. In this section, we describe a way to take advantage of a simple structural property to dramatically speed up Algorithm 1.

The homotopy method transfers probability mass along the components with the greatest possible difference in z value and the smallest cumulative weight. A component i, therefore, cannot be a receiver if there is another component j with a smaller z_j and w_j. The component i is said to be *dominated* and can be eliminated from the set of receivers as the following lemma states.

Lemma 3. *Consider a component $i \in \mathscr{S}$ such that there is a component $j \in \mathscr{S} \setminus \{i\}$ with (1) $z_j < z_i$, and (2) $w_j \le w_i$. Then, Algorithm 1 will never choose to follow a basis such that $i \in \mathscr{U}'$.*

The proof shows that using j instead of i as a receiver leads to a steeper decrease in the objective value and does not introduce any infeasibility. The proof is in Appendix A.4.

In case of ties, when multiple components satisfy $z_i = z_j$ and $w_i = w_j$, it is sufficient to choose one of them as a possible receiver and eliminate others. The components that are not dominated and can be receivers can easily be identified in $\mathcal{O}(S \log S)$ using Algorithm 3 (in Appendix B.1).

It can be shown for uniform w that only the smallest component of z is not dominated and can serve as a receiver. In this case, our the homotopy method takes at most S steps. More generally, if the weights w come from at most C distinct

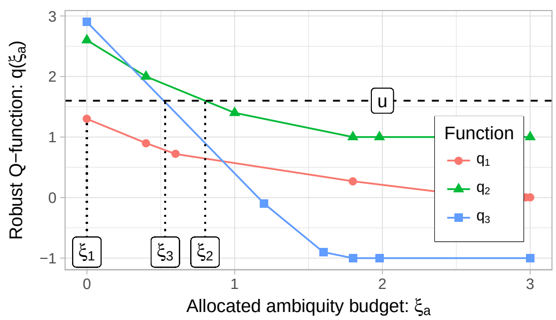


Figure 3. Visualization of the s-rectangular Bellman update with the response functions q_1, q_2, q_3 for 3 actions.

classes, then each class contains at most one receiver. The following corollary summarizes this fact.

Corollary 1. *If $w_{1,\ldots,S} \in \mathscr{C}$, Algorithm 1 and 3 run in total time $\mathcal{O}(|\mathscr{C}| \cdot S \log(|\mathscr{C}| \cdot S))$ and output $X_{1,\ldots,n}$ with $n \le |\mathscr{C}| \cdot S$.*

4. Bisection for s-rectangular Sets

In this section, we turn to RMDPs with s-rectangular ambiguity sets. Building upon the fact that $q(\xi)$ is piecewise linear, as detailed in Section 3, we propose an efficient bisection method for computing the s-rectangular Bellman update (4). Although we focus on ambiguity sets constrained by the L_1 norm, our approach readily generalizes to other norms.

For the remainder of this section, our goal is to compute the Bellman update for an arbitrary, but fixed, state $s \in \mathscr{S}$. To simplify notation, we drop the subscript for this state s. The nominal transition probabilities under action a are $\bar{p}_a \in \Delta^S$, the reward is $r_a \in \mathbb{R}$, the L_1 normed weight vector is $w_a \in \mathbb{R}^S$, and the degree of ambiguity is κ.

We will show below that for s-rectangular ambiguity sets, the following optimization problem is an equivalent reformulation of the robust Bellman update in (4):

$$\min_{u \in \mathbb{R}} \left\{ u \;:\; \sum_a q_a^{-1}(u) \le \kappa \right\}, \tag{7}$$

where q_a^{-1} is defined as the following optimization problem:

$$q_a^{-1}(u) = \min_{p \in \Delta^S} \left\{ \|p - \bar{p}_a\|_{1,w_a} \;:\; r_a + \gamma\, p^{\mathsf{T}} v \le u \right\}. \tag{8}$$

The intuition behind (7) is as follows. In the robust Bellman update (4), the adversarial nature chooses the transition probabilities p_a, $a \in \mathscr{A}$, so as to minimize the value of the Bellman update $\sum_a d_a \cdot (r_a + \gamma\, p_a^{\mathsf{T}} v)$ while adhering to the ambiguity budget via $\sum_a \xi_a \le \kappa$ for $\xi_a = \|p_a - \bar{p}_a\|_{1,w_a}$. In problem (8), $q_a^{-1}(u)$ can be interpreted as the minimum ambiguity budget $\|p - \bar{p}_a\|_{1,w_a}$ assigned to action $a \in \mathscr{A}$ that allows nature to ensure that a results in a value to-go $r_a + \gamma\, p^{\mathsf{T}} v$ not exceeding u. Any value of u that is feasible in (7) thus implies that within the specified overall ambiguity budget of κ, nature can ensure that *every* action $a \in \mathscr{A}$ results in a value to-go not exceeding u. Minimizing u in (7) thus determines the transition probabilities that lead to the lowest value to-go under *any* decision rule d_a, which in turn is tantamount to solving the robust Bellman update (4).

Figure 3 shows an example with 3 actions and q_1, q_2, q_3. If nature wants to achieve the value of u depicted in the figure, then the *smallest* values for ξ_i such that $q(\xi_i) \le u$, $i = 1, 2, 3$, are indicated with points, and a budget of $\kappa = \xi_1 + \xi_2 + \xi_3$ is required to ensure a value to-go of at most u.

Algorithm 2: Bisection algorithm to solve (7)

Input: ϵ: desired precision,
$u_{\min}$: maximum known u for which (7) is *infeasible*
$u_{\max}$: minimum known u for which (7) is *feasible*
// Assumption: $u_{\min} \le u^\star \le u_{\max}$
while $u_{\max} - u_{\min} > 2\,\epsilon$ **do**
  Split interval $[u_{\min}, u_{\max}]$ in half:
   $u \leftarrow (u_{\min} + u_{\max})/2$;
  Check feasibility of the mid point u:
   $s \leftarrow \sum_{a \in \mathscr{A}} q_a^{-1}(u)$;
  if $s \le \kappa$ **then**
    When u is *feasible* update the feasible upper bound: $u_{\max} \leftarrow u$;
  else
    When u *infeasible* update the infeasible lower bound: $u_{\min} \leftarrow u$;
  end
end
return $(u_{\min} + u_{\max})/2$;

The reformulation (7) can be solved by a bisection algorithm, as summarized in Algorithm 2. Bisection is a natural approach for the one-dimensional optimization problem in (7). It is efficient because the functions $q^{-1}(u)$ are piecewise linear with a small number of segments when the ambiguity sets are constrained by the L_1 norm. That is, $q(\xi)$ is piecewise linear with breakpoints $X_{1,\ldots,n}$ and values $Q_l = q(X_l)$ with $n \in \mathcal{O}(|\mathscr{C}| X)$ (from Corollary 1). The inverse function $q^{-1}(u)$ is also piecewise linear with breakpoints $Q_{1,\ldots,n}$ and corresponding values of $X_l = q^{-1}(Q_l)$; care needs to be taken to define $q^{-1}(u) = \infty$ for $u < Q_n$. The following theorem states the correctness of (7).

Theorem 2. *The optimal objective values of (4) and (7) coincide.*

The proof is deferred to Appendix A.5. It employs strong linear programming duality and algebraic manipulation.

Algorithm 2 only computes the objective value of the robust Bellman update and not the optimal d or p values. These values are necessary when implementing variations of policy iteration which are vastly more efficient than value iteration (Kaufman & Schaefer, 2013). As we show, the value of d can be computed from the derivatives of $q_a^{-1}(\xi)$ and the optimal $u^\star$ in linear time. As this procedure is simple but technical, we defer it to Appendix A.6.

4.1. Computational Complexity

The runtime of Algorithm 2 depends on the desired level of precision. It is possible to eliminate this dependence by taking advantage of the piecewise linearity of $q_a(\xi)$ and $q_a^{-1}(u)$. Because the extension is simple, but tedious, we defer it to Appendix B.2. The idea is to choose the *median breakpoint* between $u_{\min}$ and $u_{\max}$ instead of the mean value. Bisections then continue until q_a^{-1} are affine on the interval $[u_{\min}, u_{\max}]$ for all a. Then, $u^\star$ can be obtained by solving two equations with two unknowns. Appendix B.2 describes this method in Algorithm 4 and proves the following complexity statement.

Theorem 3. *Assuming that all q_a^{-1} are piecewise linear with at most $|\mathscr{C}|\, S$ segments, then the worst-case time complexity of Algorithm 4 is $\mathcal{O}\big(|\mathscr{C}|\, S\, A\, \log(|\mathscr{C}|\, S\, A)\big)$.*

5. Numerical Results

In this section, we evaluate the numerical performance of the proposed algorithms by measuring their times to compute a single Bellman update. Note that since the methods are exact, they have no effect on the number of iterations taken by the (approximate) policy or value iteration method.

We focus on three sets of domains with very disparate characteristics. The first set of problems are generated randomly. The transition probabilities are sampled from a uniform distribution supported on $[0, 1]$ and are subsequently normalized. The rewards are zero, and the value functions are sampled i.i.d. from the uniform distribution on $[0, 1]$. The weights w_s of the L_1 norm are sampled i.i.d. from the uniform distribution on $[0.5, 2]$ (very large/small weights are omitted since they simplify the optimization). These random problems have dense transition probabilities, and many actions have similar Q-values, which makes them particularly challenging.

We use the classic inventory management problem (Zipkin, 2000) to generate the second set of instances. The holding cost, purchase cost, and sale price are 0.1, 1.0, and 1.6, respectively. There are no backlogs, and the inventory is limited by the number of states S. The demand is sampled from a normal distribution with mean $S/2$ and standard deviation $S/5$. The weights for the L_1 norm are set to $w_i = 10/\bar{p}_i$ (as suggested for the L_2 norm in Iyengar (2005)) and clamped to the interval $[0.3, 3.0]$. The initial state is 0 (no inventory), and the value function is linear with slope 1. Inventory problems are more structured and sparse, and their Q-values are more diverse.

The third set of instances involves simple reinforcement learning benchmark problems. These numerical results are reported in Appendix D.

We compare the proposed methods with Gurobi 7.5, a state-of-the-art commercial LP solver. A comparison with related algorithms, such as those in Petrik & Subramanian (2014) and Iyengar (2005), is omitted because they are special cases of the homotopy method for the plain L_1 norm and do not generalize. We are unaware of any prior fast method for computing s-rectangular Bellman updates.

A number of methods have been proposed for choosing appropriate values of κ in RMDPs and reinforcement learning (Weissman et al., 2003; Wiesemann et al., 2013; Taleghan et al., 2015; Petrik et al., 2016). Instead of using a specific value of κ, we examine the runtime of all methods on a range of possible κ values.

The remainder of the section presents timing results first for the s, a-rectangular homotopy method and then for the s-rectangular bisection method. The homotopy method uses Algorithm 3 to eliminate dominated donor-receiver pairs. The bisection method is implemented as in Algorithm 2, except that it stops if all q_a^{-1} are affine and computes $u^\star$ as in Algorithm 4. All results were generated on a PC with i7-6700 3.4 GHz CPU with 32 GB RAM. All algorithms were implemented in C++ and the code is available from the publications section of `http://cs.unh.edu/~mpetrik`.

5.1. s, a-rectangular Ambiguity

In our first benchmarks, we compute $q_{s,a}(\xi)$ for a single state s and action a. We vary the number of states from 50 to 400 in increments of 50. Since the L_1 norm distance between two distributions is always between 0 and 2, we consider $\kappa \in \{0.0, 0.25, 0.5, 0.75, \ldots, 1.75, 2.0\}$ and report average performances over those sizes. The same κ values are used for the weighted L_1 norm although they may be greater than 2. The results are averaged over 5 runs.

Figure 4 shows the timing results for the two domains. To enhance clarity, we omit confidence intervals which are very small. The figure compares the times needed to solve the robust Bellman update for both the weighted and the plain L_1 norms in the same plot.

Several important conclusions can be drawn from Figure 4. First, the homotopy method for the plain L_1 norm is about 1,000 times faster than Gurobi, and the generic homotopy method for weighted L_1 norms is about 100 times faster. Second, there is virtually no difference between Gurobi's runtimes for the weighted and unweighted problems. Weights slow down the homotopy method signifi-

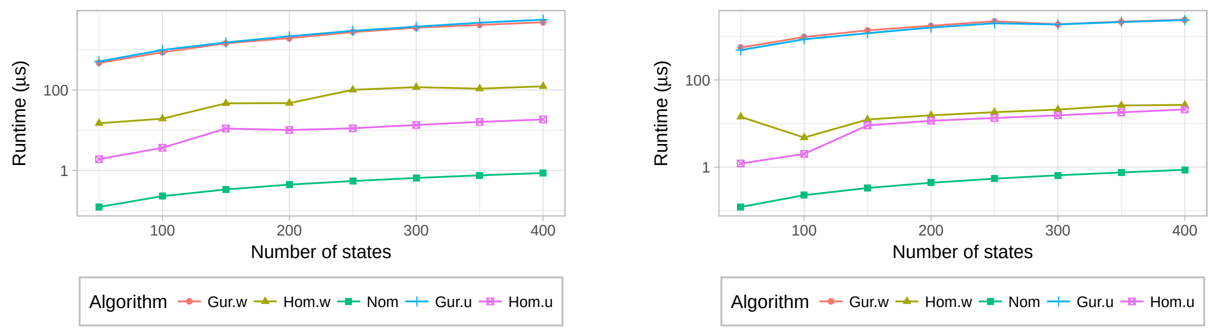


Figure 4. Comparison of the time to compute $q(\xi)$ with the homotopy method (Hom), Gurobi (Gur), and the nominal MDP (Nom). The suffixes ".w" and ".u" refer to the weighted and unweighted L_1 norms, respectively. The runtimes correspond to randomly generated (*left*) and inventory management problems (*right*).

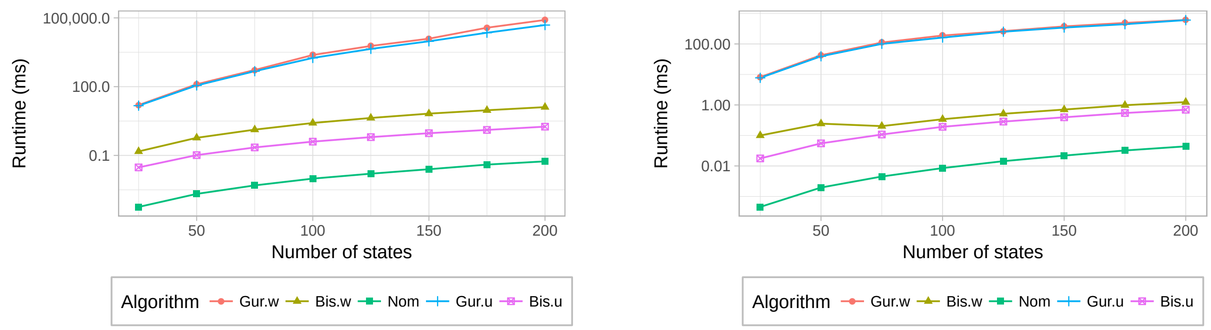


Figure 5. Comparison of the time to solve (4) with the bisection (Bis) method and the other approaches from Figure 4. We use the abbreviations from Figure 4. The presented runtimes correspond to randomly generated (*left*) and inventory management problems (*right*).

cantly in random problems but much less so in the inventory problem. This is caused by the uniformly distributed weights w in random problems, which imply that fewer components can be eliminated using Lemma 3. Finally, while the speedup over Gurobi is considerable, it does not increase significantly as the number of states S increases.

5.2. s-rectangular Ambiguity

In the s-rectangular case, the bisection method is used to compute (4) for a single state s. We consider problems with $S = 25, \ldots, 200$ states in increments of 25 and let $A = S$. We let the size of the ambiguity sets vary with the number of actions: $\kappa \in \{0.0, 0.25 \cdot A, 0.5 \cdot A, 0.75 \cdot A, \ldots, 1.75 \cdot A, 2.0 \cdot A\}$ for both the weighted and the unweighted L_1 norm. All algorithms are affected minimally by the choice of κ. The results are averaged over 5 runs.

Figure 5 shows the timing results for the two problem domains. We can make several observations. First, the bisection method is 1,000 to 10,000 times faster than Gurobi. Second, the speedup increases with S. With 200 states and actions, the bisection method is up to 1,500 times faster than Gurobi in the inventory domain and up to 49,000 times faster in the random domain and an intermediate choice of κ. Additional experimental results, which we omit here, show that the runtimes of all methods are quite insensitive to κ and the state-to-action ratio. Finally, the bisection method is about 10 times slower than the nominal solution.

6. Conclusion

We proposed two new methods for computing robust Bellman updates. Our new algorithms have a computational complexity of $\mathcal{O}(S\,A\,\log(S\,A))$ for plain and certain weighted L_1 norms, which is almost as efficient as the $\mathcal{O}(S\,A)$ complexity of Bellman updates in nominal, non-robust MDPs. While the worst-case complexity for weighted L_1 norms is quadratic, we proposed an elimination procedure to reduce the complexity in typical instances.

Our empirical results show significant speedups over a leading LP solver. We achieve meaningful speedups of 100-1,000 times for s, a-rectangular ambiguity sets, but they do not increase with problem size. For s-rectangular ambiguity sets, on the other hand, we achieve speedups of 1,000-10,000 times and they increase with problem size.

Future work should address extensions of the methods to generic L_p norms, Wasserstein balls, as well as their use in practical problems.

Acknowledgments

We thank the anonymous reviewers for comments that helped to improve this paper. This work was supported, in part, by the National Science Foundation under Grant No. IIS-1717368, by the Engineering and Physical Sciences Research Council under the Grants EP/M028240/1 and EP/M027856/1, and by the Imperial College Junior Research Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation and the Engineering and Physical Sciences Research Council.

References

Asif, M. Salman and Romberg, Justin. Dantzig selector homotopy with dynamic measurements. In *IS&T/SPIE Computational Imaging*, 2009.

Bertsekas, Dimitri P and Tsitsiklis, John N. *Neuro-dynamic programming*. 1996.

Bertsimas, Dimitris and Tsitsiklis, John N. *Introduction to linear optimization*. 1997.

Delgado, Karina V., De Barros, Leliane N., Dias, Daniel B., and Sanner, Scott. Real-time dynamic programming for Markov decision processes with imprecise probabilities. *Artificial Intelligence*, 230:192–223, 2016.

Drori, Iddo and Donoho, David L. Solution of ℓ_1 minimization problems by LARS/homotopy methods. In *Acoustics, Speech and Signal Processing (ICASSP)*, 2006.

Duchi, John, Shalev-Shwartz, Shai, Singer, Yoram, and Chandra, Tushar. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *International Conference of Machine Learning (ICML)*, 2008.

Garrigues, Pierre J and El Ghaoui, Laurent. An homotopy algorithm for the lasso with online observations. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 489–496, 2009.

Hanasusanto, Grani A and Kuhn, Daniel. Robust data-driven dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

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

Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. *The elements of statistical learning*. 2nd edition, 2009.

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

Jaksch, Thomas, Ortner, Ronald, and Auer, Peter. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(1):1563–1600, 2010.

Kaufman, David L and Schaefer, Andrew J. Robust modified policy iteration. *INFORMS Journal on Computing*, 25(3):396–410, 2013.

Le Tallec, Yann. *Robust, risk-sensitive, and data-driven control of Markov decision processes*. PhD thesis, MIT, 2007.

Mannor, Shie, Mebel, O, and Xu, H. Lightning does not strike twice: Robust MDPs with coupled uncertainty. In *International Conference on Machine Learning (ICML)*, 2012.

Murphy, Kevin. *Machine learning: A probabilistic perspective*. 2012.

Nilim, Arnab and El Ghaoui, Laurent. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

Petrik, Marek. Approximate dynamic programming by minimizing distributionally robust bounds. In *International Conference of Machine Learning (ICML)*, 2012.

Petrik, Marek and Subramanian, Dharmashankar. RAAM : The benefits of robustness in approximating aggregated MDPs in reinforcement learning. In *Neural Information Processing Systems (NIPS)*, 2014.

Petrik, Marek, Ghavamzadeh, Mohammad, and Chow, Yinlam. Safe policy improvement by minimizing robust baseline regret. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Puterman, Martin L. *Markov decision processes: Discrete stochastic dynamic programming*. 2005.

Strehl, Alexander L, Li, Lihong, and Littman, Michael. Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444, 2009.

Sutton, Richard S and Barto, Andrew. *Reinforcement learning*. 1998.

Taleghan, Majid Alkaee, Dietterich, Thomas G., Crowley, Mark, Hall, Kim, and Albers, H. Jo. PAC optimal MDP planning with application to invasive species management. *Journal of Machine Learning Research*, 16(1): 3877–3903, 2015.

Tamar, Aviv, Mannor, Shie, and Xu, Huan. Scaling up robust MDPs using function approximation. In *International Conference of Machine Learning (ICML)*, 2014.

Thai, Jerome, Wu, Cathy, Pozdnukhov, Alexey, and Bayen, Alexandre. Projected sub-gradient with $\ell 1$ or simplex constraints via isotonic regression. In *IEEE Conference on Decision and Control (CDC)*, pp. 2031–2036, 2015.

Vanderbei, Robert J. *Linear programming: Foundations and extensions*. Springer, 2nd edition, 2001.

Weissman, Tsachy, Ordentlich, Erik, Seroussi, Gadiel, Verdu, Sergio, and Weinberger, Marcelo J. Inequalities for the L1 deviation of the empirical distribution. 2003.

Wiesemann, Wolfram, Kuhn, Daniel, and Rustem, Berc. Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

Xu, Huan and Mannor, Shie. The robustness-performance tradeoff in Markov decision processes. *Advances in Neural Information Processing Systems (NIPS)*, 2006.

Xu, Huan and Mannor, Shie. Parametric regret in uncertain Markov decision processes. In *IEEE Conference on Decision and Control (CDC)*, pp. 3606–3613, 2009.

Zipkin, Paul H. *Foundations of inventory management*. 2000.

A. Technical Results

A.1. Proof of Lemma 1

Proof. The lemma relies on the fact that $\mathscr{O} \cap \mathscr{Z} = \emptyset$; otherwise, the constraints in the basis would be linearly dependent. In other words, although for a single element the two associated constraints in $\mathscr{O}$ and $\mathscr{Z}$ can both hold with equality, at most one of them can be included in the basis.

Since the basis is defined by at least $2\,S - 2$ of the inequalities holding with equality (other constraints may be active or violated, but they do not participate in the definition of the basis), we have that

$$|\mathscr{U}| + |\mathscr{L}| + |\mathscr{Z}| \geq 2\,S - 2\,, \tag{9}$$

where the subtraction of 2 accounts for the additional two equality constraints in the LP. From the definition of $\mathscr{O}, \mathscr{L}', \mathscr{U}'$, the equality above and the fact that the constraints in a basis are linearly independent, we obtain the following identity:

$$|\mathscr{U}| + |\mathscr{L}| + |\mathscr{Z}| = |\mathscr{U}' \cup \mathscr{O} \cup (\mathscr{Z} \cap \mathscr{U})| + |\mathscr{L}' \cup \mathscr{O} \cup (\mathscr{Z} \cap \mathscr{L})| + |\mathscr{Z}|\,.$$

Since $\mathscr{Z} \cap \mathscr{O} = \emptyset$, the sets in the unions $\mathscr{U}' \cup \mathscr{O} \cup (\mathscr{Z} \cap \mathscr{U})$ and $\mathscr{L}' \cup \mathscr{O} \cup (\mathscr{Z} \cap \mathscr{L})$ do not intersect (i.e. $\mathscr{O} \cap (\mathscr{Z} \cap \mathscr{U}) = \emptyset$). Therefore, we obtain that

$$|\mathscr{U}| + |\mathscr{L}| + |\mathscr{Z}| = |\mathscr{U}'| + |\mathscr{L}'| + 2\,|\mathscr{O}| + |\mathscr{Z} \cap \mathscr{U}| + |\mathscr{Z} \cap \mathscr{L}| + |\mathscr{Z}|\,,$$

which in turn simplifies to:

$$|\mathscr{U}| + |\mathscr{L}| + |\mathscr{Z}| = |\mathscr{U}'| + |\mathscr{L}'| + 2\,|\mathscr{O}| + |\mathscr{Z} \cap (\mathscr{U} \cup \mathscr{L})| - |\mathscr{Z} \cap \mathscr{U} \cap \mathscr{L}| + |\mathscr{Z}| \geq 2\,S - 2\,. \tag{10}$$

Next, we observe that

$$|\mathscr{U}'| + |\mathscr{L}'| + |\mathscr{O}| + |\mathscr{Z}| = |\mathscr{U}' \cup \mathscr{L}' \cup \mathscr{O} \cup \mathscr{Z}| \leq S\,, \tag{11}$$

where the equality follows from the fact that the index sets do not overlap, and the inequality holds since $\mathscr{U}', \mathscr{L}', \mathscr{O}$ and $\mathscr{Z}$ are all subsets of $\{1, \ldots, S\}$. Now, combining (10) and (11) we obtain:

$$|\mathscr{U}'| + |\mathscr{L}'| + 2\,|\mathscr{O}| + |\mathscr{Z} \cap (\mathscr{U} \cup \mathscr{L})| - |\mathscr{Z} \cap \mathscr{U} \cap \mathscr{L}| + |\mathscr{Z}| + 2 \geq 2(|\mathscr{U}'| + |\mathscr{L}'| + |\mathscr{O}| + |\mathscr{Z}|)\,.$$

A simple algebraic manipulation gives us

$$|\mathscr{Z} \cap (\mathscr{U} \cup \mathscr{L})| - |\mathscr{Z} \cap \mathscr{U} \cap \mathscr{L}| - |\mathscr{Z}| + 2 \geq |\mathscr{U}'| + |\mathscr{L}'|\,,$$

and finally because $|\mathscr{Z} \cap (\mathscr{U} \cup \mathscr{L})| - |\mathscr{Z}| \leq 0$, we get the desired inequality:

$$2 \geq |\mathscr{U}'| + |\mathscr{L}'|\,.$$

□

A.2. Proof of Lemma 2

Proof. Note that $p_{\mathscr{Z}} = \mathbf{0}$ and $p_{\mathscr{O}} = \bar{p}_{\mathscr{O}}$. Consider some $i, j \in \mathscr{L}' \cup \mathscr{U}'$ for any of the cases C1-C4. Then, algebraic manipulation of the constraint $\mathbf{1}^\mathsf{T} p = 1$ readily implies that:

$$p_i + p_j = 1 - \bar{p}_{\mathscr{O}}\,.$$

Therefore, as p_i and p_j sum to a constant, we must have that

$$\dot{p}_i + \dot{p}_j = 0\,.$$

Similarly simplifying the constraints that define the basic feasible solutions for each of the cases C1 - C4 gives us:

Case C1: $\mathscr{L}' = \{i\}$, $\mathscr{U}' = \{j\}$: Combining the constraint $w^\mathsf{T} l = \xi$ with $\bar{p}_i - p_i = l_i$ and with $p_j - \bar{p}_j = l_j$ yields:

$$w_i(\bar{p}_i - p_i) + w_j(p_j - \bar{p}_j) = \xi - w_{\mathscr{Z}}^\mathsf{T} \bar{p}_{\mathscr{Z}}\,.$$

Then, $\dot{p}_i + \dot{p}_j = 0$ gives us:

$$\begin{aligned} w_i(\bar{p}_i - p_i) + w_j(p_j - \bar{p}_j) &= \xi - w_{\mathscr{X}}^\mathsf{T} \bar{p}_{\mathscr{X}} \\ -w_i\dot{p}_i + w_j\dot{p}_j &= 1 \\ -w_i\dot{p}_i - w_j\dot{p}_i &= 1 \\ \dot{p}_i &= -\frac{1}{w_i + w_j} \,. \end{aligned}$$

The other expressions follow directly from the above.

Case C2: $\mathscr{L}' = \emptyset$, $\mathscr{U}' = \{i, j\}$: Similarly to Case C1, combining the constraint $w^\mathsf{T} l = \xi$ with $p_i - \bar{p}_i = l_i$ and with $p_j - \bar{p}_j = l_j$ yields to:

$$w_i(p_i - \bar{p}_i) + w_j(p_j - \bar{p}_j) = \xi - w_{\mathscr{X}}^\mathsf{T} \bar{p}_{\mathscr{X}} \,.$$

The rest follows analogously to Case C1.

Case C3: $\mathscr{L}' = \{i, j\}$, $\mathscr{U}' = \emptyset$: This case is infeasible as any such basis violates the constraint that $\mathbf{1}^\top p = 1$.

Case C4: $|\mathscr{U}| + |\mathscr{L}| < 2$: We show that this basis can be feasible only for a singleton set and is thus not relevant for the homotopy method. One option that falls into this case is $\mathscr{U}' = \{i\}, \mathscr{L}' = \emptyset$. The following equality constraints are a part of the basis:

$$\begin{aligned} p_i &= 1 - \bar{p}_{\mathscr{O}} \\ p_i - \bar{p}_i &= l_i \,. \end{aligned}$$

Then, $w_i l_i = w_i(p_i - \bar{p}_i) = w_i(1 - \bar{p}_{\mathscr{O}} - \bar{p}_i) = \xi - w_{\mathscr{X}}^\mathsf{T} \bar{p}_{\mathscr{X}}$, which can be satisfied only for a particular value of ξ. The result follows similarly for other possibilities that fall within this case. □

A.3. Proof of Theorem 1

Proof. The piecewise linear function computed by Algorithm 1 can be defined as follows:

$$g(\xi) = \min_{\alpha \in \Delta^k} \left\{ \sum_{i=1}^{k} \alpha_i \cdot Q_i \; : \; \sum_{i=1}^{k} \alpha_i \cdot X_i = \xi \right\} \,. \tag{12}$$

Here, k is the number of linear segments.

To show the correctness of Algorithm 1, we need to show that $g(\xi) = q(\xi)$ for any $\xi \geq 0$.

Lemma 1 shows that the homotopy method generates valid bases and thus corresponding basic solutions. Hence, each value X_i corresponds to the value of ξ in a feasible basis and thus a basic feasible solution (there may be multiple bases for a single basic solution).

Because the optimal solution to a linear program can be expressed as a convex combination of basic feasible solutions, and the homotopy algorithm considers only one of them for any fixed ξ, we can readily show that:

$$q(\xi) \leq g(\xi) \,.$$

Now, to conclude that $q(\xi) \geq g(\xi)$, we must show that the homotopy method does not skip over any relevant bases. To derive a contradiction, assume that a basis B'' is feasible for some ξ'' and satisfies $q(\xi'') < g(\xi'')$. Also assume, without loss of generality, that it is such a basis with the smallest possible value of ξ''.

Then, let X_i be the largest element in $X_{1\dots k}$ such that $X_i \leq \xi''$. Also, let X_j be the smallest element in $X_{1\dots k}$ such that $X_j \geq \xi''$. Let B be the basic solution that X_i and X_j share. Then, from the fact that $q(\xi)$ is piecewise linear and *convex* with a finite number of segments for any linear program, we have for the derivative:

$$\frac{q(\xi'') - Q_i}{\xi'' - X_i} < \frac{g(\xi'') - Q_i}{\xi'' - X_i} = \frac{Q_j - Q_i}{X_j - X_i} = \dot{g}(\xi) \,.$$

However, by Lemma 1 and Lemma 2, the homotopy method selects the basis with the minimal derivative and thus:

$$\dot{q}(\xi) \geq \frac{Q_j - Q_i}{X_j - X_i} .$$

This is contradicts the strict inequality above.

The correctness of choosing the last value $X_k = \infty$ follows because for large ξ the function q is constant since the constraint $w^\top l = \xi$ is inactive. □

A.4. Proof of Lemma 3

Proof. To prove the lemma by contradiction, assume that a basis with $i \in \mathscr{U}'$ that satisfies the lemma's hypothesis is chosen by the homotopy algorithm. According to Lemma 2, the slopes for each basis are as follows:

C1 $\mathscr{L}' = \{k\}$ and $\mathscr{U}' = \{i\}$: Let q_i be the objective of the basis with $i \in \mathscr{U}'$, and let q_j be the objective of the same basis except with j instead of i: $j \in \mathscr{U}'$. Then:

$$\dot{q}_i = \frac{z_i - z_k}{w_i + w_k} > \frac{z_j - z_k}{w_j + w_k} = \dot{q}_j .$$

Note that the above inequality holds since $z_i - z_k < 0$ and $z_j - z_k < 0$, otherwise the two corresponding bases would not be selected by the homotopy method. For the case C1, the element in $\mathscr{U}'$ is always the receiver (otherwise it is impossible to increase ξ while keeping the solution feasible). Replacing i by j, therefore, does not affect the solution's feasibility as there are no upper bounds on p_i and p_j and the donor element remains the same.

Then, for each type of basis (C1 and C2), we show that using i instead of j will lead to a smaller derivative and a solution that is feasible.

C2 $\mathscr{U}' = \{i, k\}$, $\mathscr{L}' = \emptyset$: We need to consider two separate possibilities: (i) $z_k \geq z_i$, and (ii) $z_k \geq z_i$. When (i) is true, a contradiction follows by the same steps as C1. In case (ii), element i would serve as donor. However, using the same argument as in Case C3 in the proof of Lemma 2, it can be readily shown that $i \in \mathscr{U}'$ implies that $p_i \geq \bar{p}_i$. Thus, to make a step of non-zero length, we would have to have $p_i > \bar{p}_i$ which is impossible since i can never be a receiver in any of the cases.

□

A.5. Proof of Theorem 2

First, it can be readily seen that the functions $q_a(\xi)$ and $q_a^{-1}(u)$ are convex.

Lemma 4. *The functions $q_a(\xi)$ and $q_a^{-1}(u)$ are convex in ξ and u, respectively, for any norm used in the constraint.*

For weighted L_1 norms, the lemma follows directly from the dual formulation. For general norms, the lemma follows by a change of variables $e = p - \bar{p}$, dualizing the constraints $\mathbf{1}^\top p = 1$ and $p \geq \mathbf{0}$, and then dualizing the norms in the constraints. The lemma then follows from the convexity of the dual norm.

Now, we are ready to prove the theorem.

Proof. Letting ξ_a be the L_1 deviation from the nominal transition probability, the Bellman update in (4) simplifies to:

$$\max_{d \in \Delta^A} \min_{\xi \in \mathbb{R}_+^A} \left\{ \sum_{a \in \mathscr{A}} d_a \cdot q_a(\xi_a) \; : \; \sum_{a \in \mathscr{A}} \xi_a \leq \kappa \right\} .$$

Since the function $q_a(\xi)$ is convex (see Lemma 4), we can exchange the maximization and minimization terms:

$$\min_{\xi \in \mathbb{R}_+^A} \max_{d \in \Delta^A} \left\{ \sum_{a \in \mathscr{A}} d_a \cdot q_a(\xi_a) \; : \; \sum_{a \in \mathscr{A}} \xi_a \leq \kappa \right\} . \tag{13}$$

Now, the optimal policy d in the inner optimization problem is deterministic, which further simplifies it to

$$\min_{\xi \in \mathbb{R}_+^A} \left\{ \max_{a \in \mathscr{A}} q_a(\xi_a) \;:\; \sum_{a \in \mathscr{A}} \xi_a \le \kappa \right\} .$$

This optimization problem is still challenging because the variable ξ needs to be optimized over all actions simultaneously. Conditioning the objective value of some u, we get:

$$\min_{u \in \mathbb{R}} \min_{\xi \in \mathbb{R}_+^A} \left\{ u \;:\; \sum_{a \in \mathscr{A}} \xi_a \le \kappa,\; \max_{a \in \mathscr{A}} q_a(\xi_a) \le u \right\} . \tag{14}$$

It can be readily seen that for any given u, the optimal choice of ξ_a is one that satisfies $q_a(\xi_a) \le u$ and ξ_a is minimal to satisfy the κ-constrained sum. Let g_a be such a value:

$$g_a(u) = \min_{\xi_a \in \mathbb{R}_+} \{\xi_a \;:\; q_a(\xi_a) \le u\}. \tag{15}$$

Substituting the definition of g_a in (15) into problem (14) simplifies our optimization problem to (7). It remains to show that $g_a(u) = q_a^{-1}(u)$. To show this, we substitute q_a in the definition of g_a above to get:

$$g_a(u) = \min_{\substack{\xi_a \in \mathbb{R}_+, \\ p_a \in \Delta^S}} \left\{ \xi_a \;:\; p_a^{\mathsf{T}} z_a \le u,\; \|p_a - \bar{p}_a\| \le \xi_a \right\} .$$

The identity $g_a = q_a^{-1}$ then follows by solving for the optimal value of ξ_a. □

A.6. Action Probabilities for Bisection Method

The following theorem shows that the optimal solution can be computed easily once the optimal objective value in (4) is known.

Theorem 4. *Let $u^\star$ be the optimal objective value in (7). Then, the optimal solution $\xi^\star$ to (4) equals to:*

$$\xi_a^\star = q_a^{-1}(u^\star) \qquad \forall a \in \mathscr{A} .$$

Let ∂q_a be an arbitrary but fixed subgradient of q_a. If $\partial q_{a'}(\xi_{a'}^\star) = 0$ for some $a' \in \mathscr{A}$, then let:

$$e_a = \begin{cases} 1 & \text{if } \partial q_a(\xi_a^\star) = 0 \\ 0 & \text{otherwise}\,; \end{cases}$$

else let:

$$e_a = \begin{cases} \frac{1}{\partial q_a(\xi_a^\star)} & \text{if } q_a(\xi_a^\star) = u^\star \\ 0 & \text{otherwise}\ . \end{cases}$$

Then, the optimal solution $d^\star$ in (13) be

$$d_a^\star = \frac{e_a}{\sum_{a' \in \mathscr{A}} e_{a'}} .$$

The theorem shows that the values of optimal solutions can be computed efficiently. The optimal decision rule will randomize arbitrarily between actions with $\partial q_a = 0$ if there are some actions that satisfy this condition. Otherwise, the action probability is inversely proportional to the derivative $\partial q_a = 0$. We show in Section 3 that computing q_a, q_a^{-1}, ∂q_a, and $p_a^\star$ from $\xi_a^\star$ can be done in linear or close-to-linear time for a weighted L_1 norm.

Proof. The optimality of $\xi^\star$ follows directly from the proof of Theorem 2 and the reformulation of (13). The argument mirrors the conditions for the optimal strategies in zero-sum games.

Because of the simple constraints on ξ, the optimal decision rule d in (4) can also be computed easily. In particular, some $d^\star$ is optimal if and only if

$$u^\star \le \min_{\xi \in \mathbb{R}^A} \left\{ \sum_{a \in \mathscr{A}} d_a^\star \cdot q_a(\xi_a) \;:\; \sum_{a \in \mathscr{A}} \xi_a \le \kappa \right\} . \tag{16}$$

The constraint $\xi_a \geq 0$ is not necessary in the minimization in (16) since $q_a(\xi_a) = \infty$ for $\xi_a < 0$.

For $d^\star$ to satisfy the inequality (16), it is sufficient to find a $d^\star$ such that 1) the inequality holds for the optimal $\xi_a^\star$ *and* 2) $\xi_a^\star$ is the minimizer. Next we show that the solution $d^\star$ in the statement of the theorem satisfies both of these requirements.

a. Assume that $\partial q_{a'}(\xi_{a'}^\star) = 0$ for some $a' \in \mathscr{A}$.

1) Solution $\xi^\star$ satisfies the inequality in (16)*:* Formally, this requirement translates to:

$$u^\star \leq \sum_a d_a^\star \cdot q_a(\xi_a^\star) \,.$$

This condition can only be satisfied when $d_a^\star = 0$ for any $q_a(\xi_a^\star) < u^\star$. The reason is that the right-hand side is a convex combination of values that are all upper bounded by $u^\star$. In particular, $q_a(\xi_a^\star) \leq u^\star$ holds from the constraint in (15) and the construction of $\xi_a^\star$:

$$\xi_a^\star = q_a^{-1}(u^\star) = \min_{\xi \in \mathbb{R}_+} \{\xi \ : \ q_a(\xi) \leq u^\star\} \ .$$

The solution $d^\star$ in the statement of the theorem satisfies this requirement since $\partial q_{a'}(\xi_{a'}^\star) = 0$ implies $q_a(\xi_a^\star) = u^\star$.

2) Solution $\xi^\star$ is the minimizer in (16)*:* Because q_a is convex, the sufficient optimality condition for the minimization in (16) is:

$$\frac{\partial}{\partial \xi}\left(\sum_a d_a^\star q_a(\xi_a) - \lambda \sum_a \xi_a + \lambda\kappa\right)[\xi^\star] \ni 0 \,,$$

where λ is the Lagrange multiplier for the constraint $\sum_a \xi_a \leq \kappa$. This condition readily translates to:

$$d_a^\star \, \partial q_a(\xi_a^\star) = \lambda, \quad \forall a.$$

Since $\partial q_{a'}(\xi_{a'}^\star) = 0$ for some $a' \in \mathscr{A}$, then the above condition only holds when

$$\lambda = 0.$$

From the definition of $d^\star$ in the statement of the theorem, one can verify that

$$d_a^\star \, \partial q_a(\xi_a^\star) = \lambda = 0, \quad \forall a,$$

and for the given $d^\star$, $\xi^\star$ is the minimizer.

b. Assume that $\partial q_{a'}(\xi_{a'}^\star) \neq 0$ for $\forall a' \in \mathscr{A}$.

1) Solution $\xi^\star$ satisfies the inequality in (16)*:* Following the same argument as in Case (a.), by definition, the solution $d^\star$ in the statement of the theorem has a non-zero entry in the a^{th} element only when $q_a(\xi_a^\star) = u^\star$. Therefore, the solution $d^\star$ in the statement of the theorem satisfies this requirement.

2) Solution $\xi^\star$ is the minimizer in (16)*:* Note that $q_a(\xi_a^\star) < u^\star$ and $\partial q_{a'}(\xi_a^\star) \neq 0$ imply that

$$\xi_a^\star = 0.$$

Therefore, we denote $\mathscr{A}' = \{a : q_a(\xi_a^\star) < u^\star\} = \{a : \xi_a^\star = 0\}$, and we show that $\xi^\star$ is the minimizer of

$$\min_{\xi \in \mathbb{R}^A} \left\{ \sum_{a \in \mathscr{A}} d_a^\star \cdot q_a(\xi_a) \ : \ \sum_{a \in \mathscr{A}} \xi_a \leq \kappa \right\} = \min_{\xi \in \mathbb{R}^A} \left\{ \sum_{a \in \mathscr{A}'} d_a^\star \cdot q_a(\xi_a) \ : \ \sum_{a \in \mathscr{A}'} \xi_a \leq \kappa \right\}.$$

Consider the formulation on the right-hand side of the above equation. Following the same argument as in Case (a.), we need

$$d_a^\star \, \partial q_a(\xi_a^\star) = \lambda, \quad \forall a \in \mathscr{A}',$$

where $\partial q_a(\xi_a^\star) \neq 0, \forall a$. From the definition of $d^\star$ in the statement of the theorem, one can verify that

$$d_a^\star \, \partial q_a(\xi_a^\star) = c, \quad \forall a \in \mathscr{A}',$$

for some constant c. Since $d^\star \in \Delta^A$, c must be unique and can be determined using $d^\star$. Therefore, $\lambda = c$, and so the solution $d^\star$ in the statement of the theorem satisfies this requirement.

□

B. Detailed Algorithm Specification

B.1. Eliminating Dominated Receivers

Algorithm 3: Identify non-dominated elements that can become receivers.

```
Input: Element values z_1, ..., z_S and their weights w_1, ..., w_S
Sort the elements so that z_1 ... z_S are increasing ;
Initialize the set of possible receivers ℛ ← {1} ;
for i = 2 ... S do
    if w_i < min_{k∈ℛ} w_k then
        Add i to ℛ: ℛ ← ℛ ∪ {i} ;
    end
end
return Possible receivers mapped back to their original position ℛ
```

B.2. Bisection Algorithm with Quasi-Linear Time Complexity

Assume that each function $q_a(\xi)$ is piecewise linear with breakpoints $X_{1\ldots k}$ and corresponding function values $Y_{1\ldots k}$:

$$q_a^{-1}(\xi) = \min_{\alpha \in \Delta^k} \left\{ \sum_{i=1}^{k} \alpha_i \cdot Y_i^a \ : \ \sum_{i=1}^{k} \alpha_i \cdot \xi_i^a = \xi \right\} .$$

Also assume that $|X_a| = |Y_a| \leq CS$, where $C = |\mathscr{C}|$ for each action $a \in \mathscr{A}$, S is the number of states and C is some constant. Algorithm 4 describes the quasi-linear time algorithm along with the computational complexity of each step.

C. State-rectangular Linear Program Formulation

Consider the s-rectangular Bellman update:

$$\max_{d \in \Delta^A} \min_{\xi \in \mathbb{R}^A} \left\{ \sum_{a \in \mathscr{A}} d_a \cdot q_{s,a}(\xi_a) \ : \ \sum_{a \in \mathscr{A}} \xi_a \leq \kappa_s \right\} .$$

Let $z_a = r_{s,a}\,\mathbf{1} + \gamma\, v$ in the definition of q (note that for any $d \in \Delta^A$ we have that $\mathbf{1}^\mathsf{T} d = 1$). The LP formulation of the s-rectangular optimization problem is:

$$\begin{array}{lll}
\max\limits_{d \in \mathbb{R}^A} & \min\limits_{p \in \mathbb{R}^{A \times S}} \sum\limits_{a \in \mathscr{A}} d_a \cdot z_a^\mathsf{T} p_a & \\
\text{subject to} & \mathbf{1}^\mathsf{T} d = 1 \quad d \geq \mathbf{0} & \\
& \mathbf{1}^\mathsf{T} p_a = 1 \quad p_a \geq \mathbf{0} & \forall a \in \mathscr{A} \\
& \sum\limits_{a \in \mathscr{A}} \|p_a - \bar{p}_a\|_{1,w_a} \leq \kappa &
\end{array}$$

The inner minimization problem can be reformulated as the following *linear* program. The dual variables corresponding each constraint are noted in parentheses.

$$\begin{array}{lll}
\min\limits_{p,\theta \in \mathbb{R}^{A \times S}} & \sum\limits_{a \in \mathscr{A}} (d_a \cdot z_a^\mathsf{T} p_a) & \\
\text{subject to} & \mathbf{1}^\mathsf{T} p_a = 1 & (\text{dual} : x_a) \\
& p_a - \bar{p}_a \geq -\theta_a & (\text{dual} : y_a^n) \\
& \bar{p}_a - p_a \geq -\theta_a & (\text{dual} : y_a^p) \\
& -\sum\limits_{a \in \mathscr{A}} w_a^\mathsf{T} \theta_a \geq -\kappa & (\text{dual} : \lambda) \\
& p \geq \mathbf{0}, \quad \theta \geq \mathbf{0} &
\end{array}$$

Algorithm 4: Quasi-linear time bisection algorithm to solve (7) with piecewise linear response functions.

```
Input: X^a: the sequence of x_i^a's, Y^a: the sequence of y_i^a's
X ← ∪_{a∈𝒜} X^a ;                                        // 𝒪(C S A)
Y ← ∪_{a∈𝒜} Y^a ;                                        // 𝒪(C S A)
sort(X, Y) with Y increasing ;                           // O(C S A log(C S A))
o ← |X| ;
k_min ← 1; k_max ← o ;
// Assumption: u_min ≤ u* ≤ u_max
while k_max − k_min > 1 do                               // log(C S A)×
    k ← round((k_min + k_max)/2) ;
    u ← Y;
    foreach a ∈ 𝒜 do                                     // A×
        t_a ← q_a^{-1}(u) ;                              // 𝒪(log(C S))
    end
    s ← Σ_{a∈𝒜} t_a ;                                    // 𝒪(A)
    if s ≤ κ then
        k_max ← k ;
    else
        k_min ← k ;
    end
end
// All q_a^{-1} are affine between u_min, u_max
u_min ← Y_{k_min}; u_max ← Y_{k_max} ;
s_min ← Σ_{a∈𝒜} q_a^{-1}(u_min) ;                        // 𝒪(A log(C S))
s_max ← Σ_{a∈𝒜} q_a^{-1}(u_max) ;                        // 𝒪(A log(C S))
α ← (κ − s_min)/(s_max − s_min) ;
u* ← (1 − α)u_min + αu_max;
return u*
```

Dualizing the inner optimization problem, we get the full linear program for computing s-rectangular Bellman updates:

$$
\begin{aligned}
\max_{\substack{d,x\in\mathbb{R}^A,\lambda\in\mathbb{R}\\ y^p,y^n\in\mathbb{R}^{S\times A}}} \quad & \sum_{a\in\mathscr{A}} \Big(x_a - \bar{p}_a^{\mathsf{T}}(y_a^n - y_a^p)\Big) - \kappa\cdot\lambda \\
\text{subject to} \quad & \mathbf{1}^{\mathsf{T}} d = 1 \quad d \geq \mathbf{0} \\
& -y_a^p + y_a^n + x\cdot\mathbf{1} \leq d_a z_a \qquad \forall a\in\mathscr{A} \\
& y_a^p + y_a^n - \lambda\cdot w_a \leq 0 \qquad \forall a\in\mathscr{A} \\
& y^p \geq \mathbf{0} \quad y^n \geq \mathbf{0} \\
& \lambda \geq 0
\end{aligned}
$$

D. Additional Benchmark Results

This section describes an evaluation of our algorithms on simple reinforcement learning benchmarks from the OpenAI gym (https://gym.openai.com/). We consider three benchmark problems: 1) cart-pole, 2) inverted pendulum, and 3) mountain car. Compared with our other two benchmark categories, these problems are markedly simpler: there are few actions and the transition probabilities are sparse (few non-zero transition probabilities).

Cart-pole is a control problem that involves balancing a pole on top of a cart that moves along a single dimension. The state space in the problem is 4-dimensional continuous and consists of cart's position and velocity, and pole's angle and angular velocity. There are three actions in this problem: apply a force to the cart from the left, from the right, or do nothing. We discretize the state space and estimate the transition probabilities from samples.

Inverted-pendulum is a simplified version of the cart-pole problem in which the cart is stationary and it the torque is applied

Benchmark	States	Actions	ψ/κ	Nom	Time s,a-rect. (ms) Gurobi	Homotopy	Time s-rect. (ms) Gurobi	Bisection
Cart-pole	702	2	0.001	5.64	23,277	43.1	20,832	106
			0.1		23,173	31.5	22,365	111
			0.5		21,389	26.8	23,302	113
			2.0		18,692	27.3	22,695	118
Pendulum	800	50	0.001	40.5	86,917	65.3	34,681	241
			0.1		84,856	84.2	44,837	254
			0.5		81,753	80.8	51,110	275
			2.0		73,137	88.8	55,104	300
Mountain car	145	3	0.001	5.67	12,916	15.7	9,490	42.2
			0.1		14,148	14.2	10,345	48.2
			0.5		13,234	12.2	11,191	47.8
			2.0		11,882	12.2	12,671	52.0

Table 1. Comparison of computation time of 100 step of value iteration for discretized versions of simple RL benchmarks from OpenAI Gym.

directly to the pole (inverted pendulum). The state-space in this problem is three-dimensional and includes the sine and cosine of pole's angle and its angular velocity. The action space is single-dimensional continuous. We discretize the state and action spaces and estimate the transition probabilities from samples.

Mountain-car is a classic simple reinforcement learning benchmark. It involves driving an underpowered car up a steep hill along a single dimension. In order to make it to the top of the hill, the car must gain a sufficient momentum by backing up an opposing hill. The state space is two-dimensional and it includes the position and velocity of the car. There are two actions: accelerate forward and accelerate backward. We discretize the state space and estimate the transition probabilities from samples.

Since all three domains we consider here are continuous, we discrete the state space and action space when applicable. The discretization is uniform in each direction and is from one extreme point to the other. We then run 100 iterations of value iteration for the given value of ψ or κ. The same ψ/κ is set uniformly for all states and actions regardless of the confidence in the transition probability.

The benchmark results are summarized in Table 1. The column "Nom" represent the time to compute 100 iterations of value iterations in the plain MDP (with no robustness). The number of states denotes the number of unique discrete state observed in the simulation of 10,000 episodes. Both s, a- and s-rectangular solution are computed with respect to an unweighted $L1$ norm. Our algorithms show significant speedups over Gurobi 7.5 in every single case.