Feature Selection Using Regularization in Approximate Linear Programs for Markov Decision Processes

Marek Petrik* PETRIK@CS.UMASS.EDU
Gavin Taylor† GVTAYLOR@CS.DUKE.EDU
Ron Parr† PARR@CS.DUKE.EDU
Shlomo Zilberstein* SHLOMO@CS.UMASS.EDU

* Department of Computer Science, University of Massachusetts, Amherst, MA 01003 USA

† Department of Computer Science, Duke University, Durham, NC 27708 USA

Abstract

Approximate dynamic programming has been used successfully in a large variety of domains, but it relies on a small set of provided approximation features to calculate solutions reliably. Large and rich sets of features can cause existing algorithms to overfit because of a limited number of samples. We address this shortcoming using L_1 regularization in approximate linear programming. Because the proposed method can automatically select the appropriate richness of features, its performance does not degrade with an increasing number of features. These results rely on new and stronger sampling bounds for regularized approximate linear programs. We also propose a computationally efficient homotopy method. The empirical evaluation of the approach shows that the proposed method performs well on simple MDPs and standard benchmark problems.

1. Introduction

Solving large Markov Decision Processes (MDPs) is a very useful, but computationally challenging problem addressed widely by reinforcement learning. It is widely accepted that large MDPs can only be solved approximately. This approximation is commonly done by relying on linear value function approximation, in which the value function is chosen from a small-dimensional vector space of features. While this framework offers computational benefits and protection from the overfitting in the training data, selecting an effective, small set of features is difficult and requires a deep understanding of the domain. Feature selection, therefore, seeks to automate this process in a way that may preserve the computational simplicity of linear approximation (Parr et al., 2007; Mahadevan, 2008). We show in this paper that L_1-regularized approximate linear programs (RALP) can be used with very rich feature spaces.

RALP relies, like other value function approximation methods, on samples of the state space. The value function error on states that are not sampled is known as the *sampling error*. This paper shows that regularization in RALP can guarantee small sampling error. The bounds on the sampling error require somewhat limiting assumptions on the structure of the MDPs, as any guarantees must, but this framework can be used to derive tighter bounds for specific problems in the future. The relatively simple bounds can be used to determine automatically the regularization coefficient to balance the expressivity of the features with the sampling error.

We derive the approach with the L_1 norm, but it could be used with other regularizations with small modifications. The L_1 norm is advantageous for two main reasons. First, the L_1 norm encourages the sparse solutions, which can reduce the computational requirements. Second, the L_1 norm preserves the linearity of RALPs; the L_2 norm would require quadratic optimization.

Regularization using the L_1 norm has been widely used in regression problems by methods such as LASSO (Tibshirani, 1996) and Dantzig selector (Candes & Tao, 2007). The value-function approximation setting is, however, quite different and the regression methods are not *directly* applicable. Regu-

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

larization has been previously used in value function approximation (Taylor & Parr, 2009; Farahmand et al., 2008; Kolter & Ng, 2009). In comparison with LARS-TD (Kolter & Ng, 2009), an L_1 regularized value function approximation method, we explicitly show the influence of regularization on the sampling error, provide a well-founded method for selecting the regularization parameter, and solve the full control problem. In comparison with existing sampling bounds for ALP (de Farias & Van Roy, 2001), we do not assume that the optimal policy is available, make more general assumptions, and derive bounds that are independent of the number of features.

Our approach is based on approximate linear programming (ALP), which offers stronger theoretical guarantees than some other value function approximation algorithms. We describe ALP in Section 3 and RALP and its basic properties in Section 4. RALP, unlike ordinary ALPs, is guaranteed to compute bounded solutions. We also briefly describe a homotopy algorithm for solving RALP, which exhibits anytime behavior by gradually increasing the norm of feature weights. To develop methods that automatically select features with generalization guarantees, we propose general sampling bounds in Section 5. These sampling bounds, coupled with the homotopy method, can automatically choose the complexity of the features to minimize overfitting. Our experimental results in Section 6 show that the proposed approach with large feature sets is competitive with LSPI when performed even with small feature spaces hand selected for standard benchmark problems. Section 7 concludes with future work and a more detailed relationship with other methods.

2. Framework and Notation

In this section, we formally define Markov decision processes and linear value function approximation. A *Markov Decision Process* is a tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the possibly infinite set of states, and $\mathcal{A}$ is the finite set of actions. $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ is the transition function, where $P(s', s, a)$ represents the probability of transiting to state s' from state s, given action a. The function $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, and γ is the discount factor. P_a and r_a are used to denote the probabilistic transition matrix and reward vector for action a.

We are concerned with finding a value function v that maps each state $s \in \mathcal{S}$ to the expected total γ-discounted reward for the process. Value functions can be useful in creating or analyzing a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ such that for all $s \in \mathcal{S}$, $\sum_{a \in \mathcal{A}} \pi(s, a) = 1$. The transition and reward functions for a given policy are denoted by P_π and r_π. The value function update for a policy π is denoted by L_π, and the Bellman operator is denoted by L. That is:

$$L_\pi v = \gamma P_\pi v + r_\pi \qquad Lv = \max_{\pi \in \Pi} L_\pi v.$$

The optimal value function v^* satisfies $Lv^* = v^*$.

We focus on *linear value function approximation* for discounted infinite-horizon problems, in which the value function is represented as a linear combination of *nonlinear basis functions (vectors)*. For each state s, we define a vector $\phi(s)$ of features. The rows of the basis matrix Φ correspond to $\phi(s)$, and the approximation space is generated by the columns of the matrix. That is, the basis matrix Φ, and the value function v are represented as:

$$\Phi = \begin{pmatrix} — & \phi(s_1)^\mathsf{T} & — \\ & \vdots & \end{pmatrix} \qquad v = \Phi w.$$

This form of linear representation allows for the calculation of an approximate value function in a lower-dimensional space, which provides significant computational benefits over using a complete basis; if the number of features is small, this framework can also guard against overfitting noise in the samples.

Definition 1. A value function, v, is *representable* if $v \in \mathcal{M} \subseteq \mathbb{R}^{|\mathcal{S}|}$, where $\mathcal{M} = \text{colspan}(\Phi)$. The set of *$\epsilon$-transitive-feasible* value functions is defined for $\epsilon \geq 0$ as follows: $\mathcal{K}(\epsilon) = \{v \in \mathbb{R}^{|\mathcal{S}|} \mid v \geq Lv - \epsilon \mathbf{1}$. Here $\mathbf{1}$ is a vector of all ones. A value function is *transitive-feasible* when $v \geq Lv$ and the set of transitive-feasible functions is defined as $\mathcal{K} = \mathcal{K}(0)$.

Notice that the optimal value function v^* is transitive-feasible, and that $\mathcal{M}$ is a linear space.

3. Approximate Linear Programming

The approximate linear programming (ALP) framework is an approach for calculating a value function approximation for large MDP with a set of features Φ that define a linear space $\mathcal{M}$ (Schweitzer & Seidmann, 1985; de Farias & Van Roy, 2003). The ALP takes the following form:

$$\begin{aligned} \min_{v \in \mathcal{M}} \quad & \sum_{s \in \mathcal{S}} \rho(s) v(s) \\ \text{s.t.} \quad & r(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s', s, a) v(s') \leq v(s) \end{aligned}$$

where ρ is a distribution over the initial states and the constraints are for all $(s, a) \in (\mathcal{S}, \mathcal{A})$; that is $\sum_{s \in \mathcal{S}} \rho(s) = 1$. To ensure feasibility, one of the features is assumed to be constant. Therefore, in the remainder of the paper, we make the following standard

assumption (Schweitzer & Seidmann, 1985), which can be satisfied by setting the first column of $\mathcal{M}$ to $\mathbf{1}$.

Assumption 2. For all $k \in \mathbb{R}$, we have that $k \cdot \mathbf{1} \in \mathcal{M}$, where $\mathbf{1}$ is a vector of all ones.

For simplicity and generality of notation, we use $\mathcal{L}$ to denote the ALP constraint matrix, so $\mathcal{L}v \leq v$ is equal to the set of constraints $\{L_a v \leq v : \forall a \in \mathcal{A}\}$. Then, we can rewrite the ALP as follows:

$$\begin{array}{ll} \min\limits_{v} & \rho^{\mathsf{T}} v \\ \text{s.t.} & \mathcal{L}v \leq v \quad v \in \mathcal{M} \end{array} \tag{1}$$

Notice that the constraints in the ALP correspond to the definition of transitive-feasible functions in Definition 1. A succinct notation of the ALP constraints can then use the set of transitive-feasible functions as $v \in \mathcal{M} \cap \mathcal{K}$.

The constraint $v \in \mathcal{M}$ implies that $v = \Phi w$ and therefore the number of variables in (1) corresponds to the number of features. Typically, this is a small number. However, the number of required constraints in ALP is $|\mathcal{S}| \times |\mathcal{A}|$, which is oftentimes impractically large or infinite. The standard solution is to sample a small set of constraints according to a given distribution (de Farias & Van Roy, 2003). It is then possible to bound the probability of violating a randomly chosen constraint. There are, however, a few difficulties with this approach. First, leaving constraints out can lead to an unbounded linear program. Second, in practice the distribution over the constraints can be very different from the distribution assumed by the theory. Finally, the bound provides no guarantees on the solution quality.

ALP has often under-performed ADP methods in practice; this issue has been recently studied and partially remedied (Petrik & Zilberstein, 2009; Desai et al., 2009). Because these methods are independent of the proposed modifications, we only focus on standard approximate linear programs.

We show next that RALP with sampled constraints not only guarantees that the solution is bounded and provides worst-case error bounds on the value function, but also is independent of the number of features. As a result, the ALP formulation does not require a small number of features to be selected in advance.

4. Regularized Approximate Linear Programming

In this section, we introduce L_1-regularized ALP (RALP) as an approach to automate feature selection and alleviate the need for all constraints in standard ALP. Adding L_1 regularization to ALP permits the user to supply an arbitrarily rich set of features without the risk of overfitting.

The RALP for basis Φ and L_1 constraint ψ is defined as follows:

$$\begin{array}{lll} \min\limits_{w} & \rho^{\mathsf{T}} \Phi w & \\ \text{s.t.} & \mathcal{L}\Phi w \leq \Phi w & \|w\|_{1,e} \leq \psi, \end{array} \tag{2}$$

where $\|w\|_{1,e} = \sum_i e(i) w(i)$. Note that RALP is a generalization of ALP; when ψ approaches infinity, the RALP solution approaches the ALP solution. The objective value of (2) as a function of ψ is denoted as $\theta(\psi)$.

We generally use $e = \mathbf{1}_{-1}$, which is a vector of all ones except the first position, which is 0; because the first feature is the constant feature, we do not include it in the regularization. The main reasons for excluding the constant feature are that the policy is independent of the constant shifts, and the homotopy method we propose requires that the linear program is easy to solve when $\psi = 0$.

Alternatively, we can formulate RALP in (2) as a minor modification of ALP in equation (1). This is by modifying $\mathcal{M}$ to satisfy the L_1 norm as:

$$\mathcal{M}(\psi) = \{\Phi w \,|\, \|w\|_{1,e} \leq \psi\}.$$

Notice that RALP introduces an additional parameter ψ over ALP. As with L_1 regularization for regression, this raises some concerns about a method for choosing the regularization parameter. Practical methods, such as cross-validation may be used to address this issue. We also propose an automated method for choosing ψ in Section 5 based on the problem and sampling parameters.

5. Sampling Bounds

The purpose of this section is to show that RALP offers two main benefits over ALP. First, even when the constraints are sampled and incomplete, it is guaranteed to provide a feasible solution. Since feasibility does not imply that the solution is close to optimal, we then show that under specific assumptions — such as smooth reward and transition functions — RALP guarantees that the error due to the missing constraints is small.

To bound the error from sampling, we must formally define the samples and how they are used to construct ALPs. We consider the following two types of samples $\tilde{\Sigma}$ and $\bar{\Sigma}$ defined as follows.

Definition 3. *One-step simple samples* are defined as follows: $\tilde{\Sigma} \subseteq \{(s, a, (s_1 \dots s_n), r(s,a)) \,|\, s \in \mathcal{S},\ a \in \mathcal{A}\}$, where $s_1 \dots s_n$ are selected i.i.d. from the distribution $P(s, a, \cdot)$. *One-step samples with expectation* are defined as follows: $\bar{\Sigma} \subseteq \{(s, a, P(s,a,\cdot), r(s,a)) \mid s \in \mathcal{S},\ a \in \mathcal{A}\}$.

Often the samples only include state transitions, as $\tilde{\Sigma}$ defines. The more informative samples $\bar{\Sigma}$ include the probability distribution of the states that follow the given state and action, as follows:

$$\bar{L}(v)(\bar{s}) = r(\bar{s}, a) + \gamma \sum_{s' \in \mathcal{S}} P(\bar{s}, a, s') v(s'), \qquad (3)$$

where $(\bar{s}, a, P(\bar{s}, a, \cdot), r(\bar{s}, a)) \in \bar{\Sigma}$. The less-informative $\tilde{\Sigma}$ can be used as follows:

$$\tilde{L}(v)(\tilde{s}) = r(\tilde{s}, a) + \gamma \frac{1}{n} \sum_{i=1}^{n} v(s_i), \qquad (4)$$

where $(\tilde{s}, a, (s_1 \dots s_n), r(\tilde{s}, a)) \in \tilde{\Sigma}$. The corresponding transitive-feasible sets $\bar{\mathcal{K}}$ and $\tilde{\mathcal{K}}$ are defined similarly. The ALPs can be constructed based on samples as Figure 1 shows. *Full ALP* corresponds to the RALP formulation in (2), when $\mathcal{M}$ is constricted with L_1 regularization. In comparison, *sampled ALP* is missing some of the constraints while *estimated ALP* is both missing some constraints, and the included constraints may be estimated imprecisely.

The following two assumptions quantify the behavior of the ALP with respect to missing and imprecise constraints respectively. The first assumption limits the error due to missing transitions in the sampled Bellman operator $\bar{L}$.

Assumption 4 (Constraint Sampling Behavior)**.** The representable value functions satisfy that:

$$\mathcal{K} \cap \mathcal{M}(\psi) \subseteq \bar{\mathcal{K}} \cap \mathcal{M}(\psi) \subseteq \mathcal{K}(\epsilon_p) \cap \mathcal{M}(\psi),$$

and that for all representable value functions $v \in \mathcal{M}(\psi)$ we have that $|(\rho - \bar{\rho})^\mathsf{T} v| \leq \epsilon_c(\psi)$.

The constant ϵ_p bounds the potential violation of the ALP constraints on states that are not provided as a part of the sample. In addition, all value functions that are transitive-feasible for the full Bellman operator are transitive-feasible in the sampled version; the sampling only removes constraints on the set. The constant ϵ_c essentially represents the maximal error in estimating the objective value of ALP for any representable value function.

The next assumption quantifies the error on the estimation of the transitions of the estimated Bellman operator $\tilde{L}$.

Assumption 5 (Constraint Estimation Behavior)**.** The representable value functions satisfy that:

$$\bar{\mathcal{K}}(-\epsilon_s) \cap \mathcal{M}(\psi) \subseteq \tilde{\mathcal{K}} \cap \mathcal{M}(\psi) \subseteq \bar{\mathcal{K}}(\epsilon_s) \cap \mathcal{M}(\psi),$$

where $\bar{\Sigma}$ and $\tilde{\Sigma}$ (and therefore $\bar{\mathcal{K}}$ and $\tilde{\mathcal{K}}$) are defined for identical sets of states.

These assumptions are quite generic in order to apply in a wide range of scenarios. The main idea behind the assumptions is to bound by how much a feasible solution in the sampled or estimated ALP can violate the true ALP constraints. These assumptions may be easily satisfied, for example, by making the following Lipschitz continuity assumptions.

Assumption 6. Let $k : \mathcal{S} \to \mathbb{R}^n$ be a map of the state-space to a normed vector space. Then for all $x, y, z \in \mathcal{S}$ and all features (columns) $\phi_i \in \Phi$, we define K_r, K_P, and K_ϕ such that

$$\begin{aligned} |r(x) - r(y)| &\leq K_r \|k(x) - k(y)\| \\ |p(z|x,a) - p(z|y,a)| &\leq K_P \|k(x) - k(y)\| \\ |\phi_i(x) - \phi_i(y)| &\leq K_\phi \|k(x) - k(y)\| \end{aligned}$$

Proposition 7. *Assume Assumption 6 and that for any $s \in \mathcal{S}$ there exists a state $\bar{s} \in \bar{\Sigma}$ such that $\|\bar{s} - s\| \leq c$. Then Assumption 4 and Assumption 5 hold with $\epsilon_p(\psi) = cK_r + c\psi(K_\phi + \gamma K_P)$*

Because of the technical nature of the proof, it is omitted and can be found in (Petrik et al., 2010).

The importance of this bound is that the violation on constraints that were not sampled grows linearly with the increasing coefficient ψ. As we show below, this fact can be used to determine the optimal value of ψ. For the sake of brevity, we do not discuss the estimation error bounds ϵ_s in more detail, which can be easily derived from existing results (Petrik & Zilberstein, 2009).

We are now ready to state the following general bounds on the approximation error of a RALP.

Theorem 8 (Offline Error Bound)**.** *Assume Assumptions 2, 4, and 5. Let $\hat{v}$, $\bar{v}$, $\tilde{v}$ be the optimal solutions of (5), (6), and (7), respectively (see Figure 1). Let $\epsilon = \frac{2}{1-\gamma} \min_{v \in \mathcal{M}(\psi)} \|v - v^*\|_\infty$ Then, the following inequalities hold:*

$$\begin{aligned} \|\hat{v} - v^*\|_{1,\rho} &\leq \epsilon \\ \|\bar{v} - v^*\|_{1,\rho} &\leq \epsilon + 2\epsilon_c(\psi) + 2\frac{\epsilon_p(\psi)}{1-\gamma} \\ \|\tilde{v} - v^*\|_{1,\rho} &\leq \epsilon + 2\epsilon_c(\psi) + \frac{3\epsilon_s(\psi) + 2\epsilon_p(\psi)}{1-\gamma} \end{aligned}$$

Full ALP

$$\rho = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}}\phi(s)$$

$$\min_{v} \quad \rho^{\mathsf{T}} v \qquad \text{s.t.} \quad v \in \mathcal{K} \quad v \in \mathcal{M}(\psi) \tag{5}$$

Sampled ALP

$$\bar{\rho} = \frac{1}{|\bar{\Sigma}|}\sum_{(s,\ldots)\in\bar{\Sigma}}\phi(s)$$

$$\min_{v} \quad \bar{\rho}^{\mathsf{T}} v \qquad \text{s.t.} \quad v \in \bar{\mathcal{K}} \quad v \in \mathcal{M}(\psi) \tag{6}$$

Estimated ALP

$$\tilde{\rho} = \frac{1}{|\tilde{\Sigma}|}\sum_{(s,\ldots)\in\tilde{\Sigma}}\phi(s)$$

$$\min_{v} \quad \tilde{\rho}^{\mathsf{T}} v \qquad \text{s.t.} \quad v \in \tilde{\mathcal{K}} \quad v \in \mathcal{M}(\psi) \tag{7}$$

Figure 1. Constructing ALP From Samples

Because of the technical nature of the proof, it is omitted and can be found in (Petrik et al., 2010).

Notice that because the weight ρ is often chosen arbitrarily, the bounds may be simply derived for $\|\cdot\|_{1,\bar{\rho}}$. In that case, $\epsilon_c = 0$. Unlike most of the existing ALP bounds, we focus on bounding the error of the value function, instead of bounding the number of violated constraints.

Consider the implications of these bounds combined with the Lipschitz assumptions of Proposition 7. It is clear that reducing ψ tightens Theorem 8, but causes the set $\mathcal{M}(\psi)$ to shrink and become more restrictive; this suggests a tradeoff to be considered when setting the regularization parameter. The bound also illustrates the importance of covering the space with samples; as the distance between the samples c approaches zero, the bound tightens. In short, the Lipschitz assumptions limit how quickly the constraints can change between sampled states. As sampled states get closer or the reward, feature, and probability functions become smoother, constraints between samples become more and more restricted; however, smoother basis functions may mean a less expressive space. Similar tradeoffs are likely to appear however Assumption 4 and Assumption 5 are fulfilled.

The offline error bounds in Theorem 8 can be used to guarantee the performance of a RALP for a fixed number of samples and the regularization coefficient ψ. It does not, however, prescribe how to choose the regularization coefficient for a given set of samples. To do that, we have to derive bounds for an actual value function v. When the samples are known, these bounds are typically tighter than the offline error bound.

Theorem 9 (Online Error Bound)**.** *Assume Assumption 2 and let $v \in \tilde{\mathcal{K}} \cap \mathcal{M}(\psi)$ be an arbitrary feasible solution of the estimated ALP (7). Then:*

$$\|v^* - v\|_{1,\rho} \leq \bar{\rho}^{\mathsf{T}} v - \rho^{\mathsf{T}} v^* + \epsilon_c(\psi) + 2\frac{\epsilon_s(\psi) + \epsilon_p(\psi)}{1-\gamma}.$$

Because of the technical nature of the proof, it is omitted and can be found in (Petrik et al., 2010).

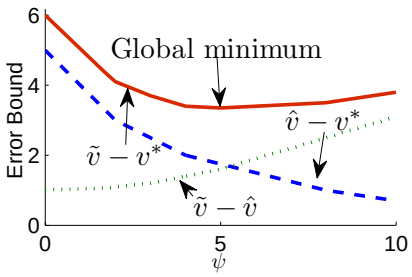

Figure 2. Sketch of error *bounds* as a function of the regularization coefficient. Here, $\hat{v}$ is the value function of the full ALP, $\tilde{v}$ is the value function of the estimated ALP, and v^* is the optimal value function.

Here we briefly introduce a homotopy method that not only speeds the computation of the RALP solution, but also can be used to find the optimal value of ψ. Because the homotopy method only relies on standard linear programming analysis and is somewhat technical, the detailed description is provided in (Petrik et al., 2010).

The main idea of the homotopy method is to first calculate $\theta(0)$ and then trace the optimal solution for increasing values of ψ. The optimal solution $w(\psi)$ of the RALP (2) is a piecewise linear function of ψ. At any point in time, the algorithm keeps track of a set of active – or non-zero – variables w and a set of active constraints, which are satisfied with equality. In the linear segments, the number of active constraints and variables are identical, and the non-linearity arises when variables and constraints become active or inactive. Therefore, the linear segments are traced until a variable becomes inactive or a constraint becomes active. Then, the dual solution is traced until a constraint becomes inactive or a variable becomes active.

Since the homotopy algorithm solves for the optimal value of the RALP (2) for all values of the regularization coefficient ψ, it is possible to increase the coefficient ψ until the error increase between sampled constraints balances out the decrease in the error due to the restricted feature space as defined in Theorem 8.

That is, we can calculate the objective value of the linear program (2) for any value of ψ.

It is easy to find ψ that minimizes the bounds in this section. As the following corollary shows, to find the global minimum of the bounds, it is sufficient to use the homotopy method to trace $\theta(\psi)$ while its derivative is less than the increase in the error due to the sampling ($\|\hat{v} - \tilde{v}\|_{1,\rho}$). Let $v(\psi)$ be an optimal solution of (7) as a function of the regularization coefficient ψ.

Corollary 10. *Assume that $\epsilon_c(\psi)$, $\epsilon_p(\psi)$, and $\epsilon_s(\psi)$ are convex functions of ψ. Then, the error bound $\|v(\psi) - v^*\|_{1,\rho} \leq f(\psi)$ for any $v(\psi)$ is:*

$$f(\psi) = \theta(\psi) - \rho^\mathsf{T} v^* + \epsilon_c(\psi) + 2\frac{\epsilon_s(\psi) + \epsilon_p(\psi)}{1-\gamma}.$$

The function $f(\psi)$ is convex and its sub-differential[1] *$\nabla_\psi f$ is independent of v^*. Therefore, a global minimum ψ^* of f is attained when $0 \in \nabla_\psi f(\psi^*)$ or when $\psi^* = 0$.*

The corollary follows directly from Theorem 9 and the convexity of the optimal objective value of (2) as a function ψ. Figure 2 illustrates the functions in the corollary. Notice that Proposition 7 is sufficient to satisfy the conditions of this corollary. In particular, the functions $\epsilon_s(\psi), \epsilon_p(\psi), \epsilon_c(\psi)$ are linear in ψ.

6. Experimental Results

In this section, we present results indicating that RALP effectively selects from rich feature spaces to outperform ALP and other common algorithms, such as LSPI, on several example problems, including the balanced pendulum and the bicycle problems. We also demonstrate the speed and effectiveness of the homotopy method in choosing a value of ψ.

Benefits of Regularization First, we demonstrate and analyze the properties of RALP on a simple chain problem with 200 states, in which the transitions move to the right by one step with a centered Gaussian noise with standard deviation 3. The reward for reaching the right-most state was +1 and the reward in the 20th state was -3. This problem is small to enable calculation of the optimal value function and to control sampling. We uniformly selected every fourth state on the chain and estimated the sampling bound $\epsilon_p(\psi) = 0.05\psi$. The approximation basis in this problem is represented by piecewise linear features, of the form $\phi(s_i) = [i - c]_+$, for c from 1 to 200; these features were chosen due to their strong guarantees for the sampling bounds. The experimental results were obtained using the proposed homotopy algorithm.

Figure 3 demonstrates the solution quality of RALP on the chain problem as a function of the regularization coefficient ψ. The figure shows that although the objective of RALP keeps decreasing as ψ increases, the sampling error overtakes that reduction. It is clear that a proper selection of ψ improves the quality of the resulting approximation. To demonstrate the benefits of regularization as it relates to overfitting, we compare the performance of ALP and RALP as a function of the number of available features in Figure 5. While ALP performance improves initially, it degrades severely with more features. The value ψ in RALP is selected automatically using Corollary 10 and the sampling bound of $\epsilon_p(\psi) = 0.05\psi$. Figure 4 demonstrates that RALP may also overfit, or perform poorly when the regularization coefficient ψ is not selected properly.

To find the proper value of ψ, as described in Corollary 10, the problem needs to be solved using the homotopy method. We show that the homotopy method performs significantly faster than a commercially available linear program solver Mosek. Figure 6 compares the computational time of homotopy method and Mosek, when solving the problem for multiple values of ψ in increments of 0.5 on the standard mountain car problem (Barto & Sutton, 1998) with 901 piecewise linear features and 6000 samples. Even for any *single* value ψ, the homotopy method solves the linear program about 3 times faster than Mosek. The next two experiments, however, do *not* use the homotopy method. In practice, RALP often works much better than what is suggested by our bounds, which can be loose for sparsely sampled large state spaces. In the following experiments, we determined ψ empirically by solving the RALP for several different values of ψ and selecting the one that produced the best policy. This was practical because we could solve the large RALPs in just a few minutes using constraint generation.

Inverted Pendulum We now offer experimental results demonstrating RALP's ability to create effective value functions in balancing an inverted pendulum, a standard benchmark problem in reinforcement learning (Wang et al., 1996; Lagoudakis & Parr, 2003). Samples of the form (s, a, r, s') were drawn from every action on states drawn from random trajectories with the pendulum starting in an upright state, referred to as episodes. Features were kernels on 650 states randomly selected from the training data, consisting of Gaussian kernels of standard deviation 0.5, 1, and 1.5, and a 6^{th} degree polynomial kernel. ψ was 1.4, and an average of 25 features had non-zero weights.

[1] Function f may be non-differentiable

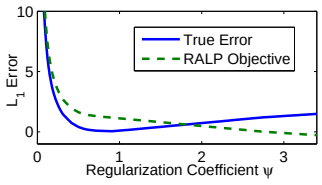

Figure 3. Comparison of the objective value of RALP with the true error.

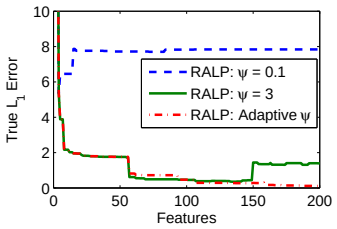

Figure 4. Comparison of the performance RALP with two values of ψ and the one chosen adaptively (Corollary 10).

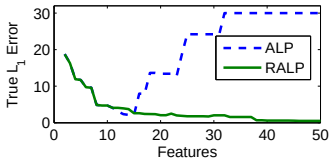

Figure 5. Average of 45 runs of ALP and RALP as a function of the number of features. Coefficient ψ was selected using Corollary 10.

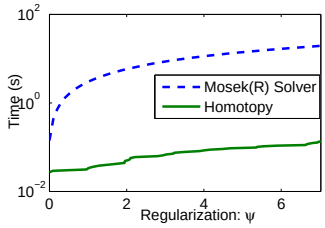

Figure 6. Comparison of performance of homotopy method versus Mosek as a function of ψ in the mountain car domain.

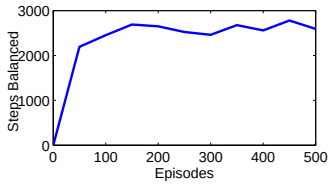

Figure 7. RALP performance on pendulum as a function on the number of episodes.

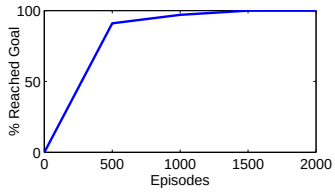

Figure 8. RALP performance on bicycle as a function on the number of episodes.

The constraints in the RALP were based on a single sample for each state and action pair. The policy was evaluated based on the number of steps it could balance the pendulum, with an upper limit of 3000. This served to evaluate the policy resulting from the approximate value function. We plot the average number of steps the pendulum was balanced as a function of the number of training episodes in Figure 7, as an average of 100 runs. It is clear the controller produced by RALP was effective for small amounts of data, balancing the pendulum for the maximum number of steps nearly all of the time, even with only 50 training episodes. Similarly, it was able to leverage the larger number of available features to construct an effective controller with fewer trajectories than LSPI, which needed 450 training episodes before achieving an average of 2500 balanced steps (Lagoudakis & Parr, 2003).

Bicycle Balancing and Riding We also present experimental results for the bicycle problem, in which the goal is to learn to balance and ride a bicycle to a target position (Randløv & Alstrøm, 1998; Lagoudakis & Parr, 2003). This is a challenging benchmark domain in reinforcement learning. Training data consisted of samples for every action on states drawn from trajectories resulting from random actions up to 35 states long, similar to the inverted pendulum domain. The feature set consisted of monomials up to degree 4 on the individual dimensions and products of monomials up to degree 3, for a total of 159 features. ψ was 0.03, and an average of 34 features had nonzero weights. We plot the number of runs out of 100 in which the bicycle reached the goal region as a function of number of training episodes in Figure 8. Again, a high percentage of runs were successful, even with only 500 training episodes. In comparison, LSPI required 1500 training episodes to pass 80% success. It is worth pointing out that due to sampling every action at each state, RALP is using more samples than LSPI, but far fewer trajectories.

7. Conclusion and Related Work

In this paper, we introduced L_1-regularized Approximate Linear Programming and demonstrated its properties for combined feature selection and value function approximation in reinforcement learning. RALP simultaneously addresses the feature selection, value function approximation, and policy determination problems; our experimental results demonstrate that it addresses these issues effectively for several sample problems, while our bounds explain the effects of sampling on the resulting approximation.

There are many additional issues that need to be addressed. The first is the construction of better bounds to guide the sampling. Our bounds explain the behavior of RALP approximation as it relates to the trade-off between the richness of the features with the number of available samples, but these bounds may at times be quite loose. Future work must identify conditions that can provide stronger guarantees. Additionally, a data-driven approach which can calculate a tighter bound online would be valuable. Finally, our analysis did not address the conditions that would guarantee sparse RALP solutions and, therefore, allow more computationally efficient solvers.

Acknowledgements

This work was supported in part by DARPA CSSG HR0011-06-1-0027, by NSF IIS-0713435, by the Air Force Office of Scientific Research under Grant No. FA9550-08-1-0171, and by the Duke University Center for Theoretical and Mathematical Sciences. We also thank the anonymous reviewers for their useful comments.

References

Barto, Andrew G. and Sutton, Richard S. *Reinforcement Learning: an Introduction*. MIT Press, 1998.

Candes, Emmanuel and Tao, Terence. The Dantzig selector:statistical estimation when p is much larger than n. *Annals of Statistics*, 35:2313–2351, 2007.

de Farias, Daniela Pucci and Van Roy, Benjamin. On constraint sampling for the linear programming approach to approximate dynamic programming. *Math. of Operations Res*, 2001.

de Farias, Daniela Pucci and Van Roy, Benjamin. The Linear Programming Approach to Approximate Dynamic Programming. *Operations Research*, 2003.

Desai, Vijay, Farias, Vivek, and Moallemi, Ciamac. A smoothed approximate linear program. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 459–467, 2009.

Farahmand, Amir Massoud, Ghavamzadeh, Mohammad, Szepesvari, Csaba, and Mannor, Shie. Regularized policy iteration. In *Advances in Neural Information Processing Systems*, volume 21, pp. 441–448, 2008.

Kolter, J. Zico and Ng, Andrew. Regularization and feature selection in least-squares temporal difference learning. In *International Conference on Machine Learning (ICML)*, pp. 521–528, 2009.

Lagoudakis, Michail G. and Parr, Ronald. Least-Squares Policy Iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

Mahadevan, Sridhar. Learning representation and control in markov decision processes: New frontiers. *Foundations and Trends in Machine Learning*, 1(4): 403–565, 2008.

Parr, Ronald, Painter-Wakefield, Christopher, Li, Lihong, and Littman, Michael. Analyzing feature generation for value-function approximation. In *International Conference on Machine Learning (ICML)*, pp. 744–751, 2007.

Petrik, Marek and Zilberstein, Shlomo. Constraint Relaxation in Approximate Linear Programs. In *International Conference on Machine Learning (ICML)*, pp. 809–816, 2009.

Petrik, Marek, Taylor, Gavin, Parr, Ron, and Zilberstein, Shlomo. Feature selection using regularization in approximate linear programs for markov decision processes. Technical report, arXiv, 2010.

Randløv, Jette and Alstrøm, Preben. Learning to drive a bicycle using reinforcement learning and shaping. In *International Conference on Machine Learning*, pp. 463–471, 1998.

Schweitzer, Paul J. and Seidmann, Abraham. Generalized polynomial approximations in Markovian decision processes. *Journal of mathematical analysis and applications*, 110(6):568–582, 1985.

Taylor, Gavin and Parr, Ronald. Kernelized Value Function Approximation for Reinforcement Learning. In *International Conference on Machine Learning*, pp. 1017–1024, Montreal, Canada, June 2009. Omnipress.

Tibshirani, Robert. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Wang, Hua O., Tanaka, Kazuo, and Griffin, Michael F. An approach to fuzzy control of nonlinear systems: Stability and design issues. *IEEE Transactions on Fuzzy Systems*, 4(1):14–23, 1996.