A Smoothed Approximate Linear Program

Vijay V. Desai
IEOR, Columbia University
vvd2101@columbia.edu

Vivek F. Farias
MIT Sloan
vivekf@mit.edu

Ciamac C. Moallemi
GSB, Columbia University
ciamac@gsb.columbia.edu

Abstract

We present a novel linear program for the approximation of the dynamic programming cost-to-go function in high-dimensional stochastic control problems. LP approaches to approximate DP naturally restrict attention to approximations that are lower bounds to the optimal cost-to-go function. Our program – the 'smoothed approximate linear program' – relaxes this restriction in an appropriate fashion while remaining computationally tractable. Doing so appears to have several advantages: First, we demonstrate superior bounds on the quality of approximation to the optimal cost-to-go function afforded by our approach. Second, experiments with our approach on a challenging problem (the game of Tetris) show that the approach outperforms the existing LP approach (which has previously been shown to be competitive with several ADP algorithms) by an order of magnitude.

1 Introduction

Many dynamic optimization problems can be cast as Markov decision problems (MDPs) and solved, in principle, via dynamic programming. Unfortunately, this approach is frequently untenable due to the 'curse of dimensionality'. Approximate dynamic programming (ADP) is an approach which attempts to address this difficulty. ADP algorithms seek to compute good approximations to the dynamic programing optimal cost-to-go function within the span of some pre-specified set of basis functions. The approximate linear programming (ALP) approach to ADP [1, 2] is one such well-recognized approach.

The program employed in the ALP approach is identical to the LP used for exact computation of the optimal cost-to-go function, with further constraints limiting solutions to the low-dimensional subspace spanned by the basis functions used. The resulting low dimensional LP implicitly restricts attention to approximations that are lower bounds on the optimal cost-to-go function. While the structure of this program appears crucial in establishing approximation guarantees for the approach, the restriction to lower bounds leads one to ask whether the ALP is the 'right' LP. In particular, could an appropriate relaxation of the feasible region of the ALP allow for better approximations to the cost-to-go function while remaining computationally tractable? Motivated by this question, the present paper presents a new linear program for ADP we call the 'smoothed' ALP (or SALP).

The SALP may be viewed as a relaxation of the ALP wherein one is allowed to violate the ALP constraints for any given state. A user defined 'violation budget' parameter controls the 'expected' violation across states; a budget of 0 thus yields the original ALP. We specify a choice of this violation budget that yields a relaxation with attractive properties. In particular, we are able to establish strong approximation guarantees for the SALP; these guarantees are *substantially* stronger than the corresponding guarantees for the ALP.

The number of constraints and variables in the SALP scale with the size of the MDP state space. We nonetheless establish sample complexity bounds that demonstrate that an

appropriate 'sampled' SALP provides a good approximation to the SALP solution with a tractable number of sampled MDP states. This sampled program is no more complex than the 'sampled' ALP and, as such, we demonstrate that the SALP is essentially no harder to solve than the ALP.

We present a computational study demonstrating the efficacy of our approach on the game of Tetris. The ALP has been demonstrated to be competitive with several ADP approaches for Tetris (see [3]). In detailed comparisons with the ALP, we estimate that the SALP provides an *order of magnitude* improvement over controllers designed via that approach for the game of Tetris.

2 Problem Formulation

Our setting is that of a discrete-time, discounted infinite-horizon, cost-minimizing MDP with a finite state space $\mathcal{X}$ and finite action space $\mathcal{A}$. Given the state and action at time t, x_t and a_t, a per-stage cost $g(x_t, a_t)$ is incurred. The subsequent state x_{t+1} is determined according to the transition probability kernel $P_{a_t}(x_t, \cdot)$. A stationary policy $\mu : \mathcal{X} \to \mathcal{A}$ is a mapping that determines the action at each time as a function of the state. Given each initial state $x_0 = x$, the expected discounted cost (cost-to-go function) of the policy μ is given by

$$J_\mu(x) \triangleq \mathsf{E}_\mu \left[\sum_{t=0}^{\infty} \alpha^t g(x_t, \mu(x_t)) \;\middle|\; x_0 = x \right].$$

where, $\alpha \in (0,1)$ is the discount factor. Denote by $P_\mu \in \mathbb{R}^{\mathcal{X}\times\mathcal{X}}$ the transition probability matrix for the policy μ, whose (x,x')th entry is $P_{\mu(x)}(x,x')$. Denote by $g_\mu \in \mathbb{R}^{\mathcal{X}}$ the vector whose xth entry is $g(x,\mu(x))$. Then, the cost-to-go function J_μ is the unique solution to the equation $T_\mu J = J$, where the operator T_μ is defined by $T_\mu J = g_\mu + \alpha P_\mu J$.

The Bellman operator T can be defined according to $TJ = \min_\mu T_\mu J$. Bellman's equation is then the fixed point equation, $TJ = J$. It is readily shown that the optimal cost-to-go function J^* is the unique solution to Bellman's equation and that a corresponding optimal policy μ^* is greedy with respect to J^*; i.e., μ^* satisfies $TJ^* = T_{\mu^*}J^*$.

Bellman's equation may be solved exactly via the following linear program:

$$\begin{array}{ll} \underset{J}{\text{maximize}} & \nu^\top J \\ \text{subject to} & J \leq TJ. \end{array} \tag{1}$$

Here, $\nu \in \mathbb{R}^{\mathcal{X}}$ is a vector with positive components that are known as the *state-relevance weights*. The above program is indeed an LP since the constraint $J(x) \leq (TJ)(x)$ is equivalent to the set of linear constraints $J(x) \leq g(x,a) + \alpha \sum_{x' \in \mathcal{X}} P_a(x,x')J(x'), \quad \forall\, a \in \mathcal{A}$. We refer to (1) as the *exact LP*.

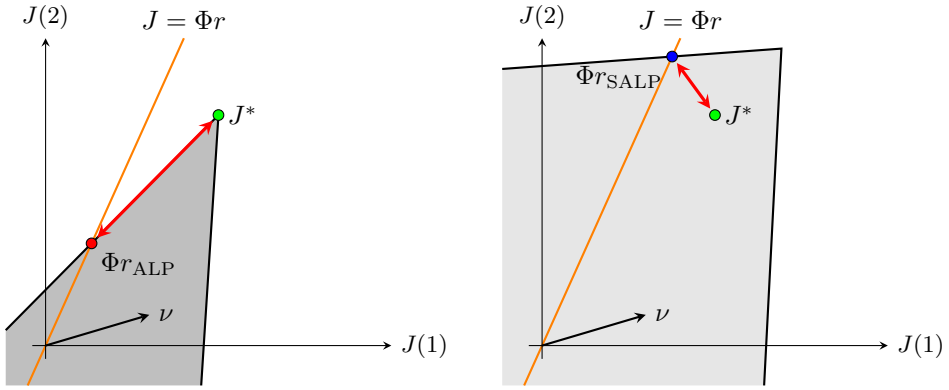
Note that if a vector J satisfies $J \leq TJ$, then $J \leq T^k J$ (by monotonicity of the Bellman operator), and thus $J \leq J^*$ (since the Bellman operator is a contraction with unique fixed point J^*). Then, every feasible point for (1) is a component-wise lower bound to J^*, and J^* is the unique optimal solution to the exact LP (1).

For problems where $\mathcal{X}$ is prohibitively large, an ADP algorithm seeks to find a good approximation to J^*. Specifically, one considers a collection of *basis functions* $\{\phi_1, \ldots, \phi_K\}$ where each $\phi_i : \mathcal{X} \to \mathbb{R}$. Defining $\Phi \triangleq [\phi_1 \phi_2 \ldots \phi_K]$ to be a matrix with columns consisting of basis functions, one seeks an approximation of the form $J_r = \Phi r$, with the hope that $J_r \sim J^*$. The ALP for this task is then simply

$$\begin{array}{ll} \underset{r}{\text{maximize}} & \nu^\top \Phi r \\ \text{subject to} & \Phi r \leq T\Phi r. \end{array} \tag{2}$$

The geometric intuition behind the ALP is illustrated in Figure 1(a). Supposed that r_{ALP} is a vector that is optimal for the ALP. Then the approximate value function Φr_{ALP} will lie on the subspace spanned by the columns of Φ, as illustrated by the orange line. Φr_{ALP}

will also satisfy the constraints of the exact LP, illustrated by the dark gray region; this implies that $\Phi r_{\text{ALP}} \leq J^*$. In other words, the approximate cost-to-go function is necessarily a point-wise lower bound to the true cost-to-go function in the span of Φ.



(a) ALP case. (b) SALP case.

Figure 1: A cartoon illustrating the feasible set and optimal solution for the ALP and SALP, in the case of a two-state MDP. The axes correspond to the components of the value function. A careful relaxation from the feasible set of the ALP to that of the SALP can yield an improved approximation.

3 The Smoothed ALP

The $J \leq TJ$ constraints in the exact LP, which carry over to the ALP, impose a strong restriction on the cost-to-go function approximation: in particular they restrict us to approximations that are lower bounds to J^* at *every point in the state space*. In the case where the state space is very large, and the number of basis functions is (relatively) small, it may be the case that constraints arising from rarely visited or pathological states are binding and influence the optimal solution. In many cases, our ultimate goal is not to find a lower bound on the optimal cost-to-go function, but rather a good approximation to J^*. In these instances, it may be the case that relaxing the constraints in the ALP so as not to require a uniform lower bound may allow for better overall approximations to the optimal cost-to-go function. This is also illustrated in Figure 1. Relaxing the feasible region of the ALP in Figure 1(b) to the light gray region in Figure 1(b) would yield the point Φr_{SALP} as an optimal solution. The relaxation in this case is clearly beneficial; it allows us to compute a better approximation to J^* than the point Φr_{SALP}. Can we construct a fruitful relaxation of this sort in general? The *smoothed approximate linear program* (SALP) is given by:

$$\begin{array}{ll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r \\ \text{subject to} & \Phi r \leq T\Phi r + s, \\ & \pi^\top s \leq \theta, \quad s \geq 0. \end{array} \tag{3}$$

Here, a vector $s \in \mathbb{R}^{\mathcal{X}}$ of additional decision variables has been introduced. For each state x, $s(x)$ is a non-negative decision variable (a slack) that allows for violation of the corresponding ALP constraint. The parameter $\theta \geq 0$ is a non-negative scalar. The parameter $\pi \in \mathbb{R}^{\mathcal{X}}$ is a probability distribution known as the *constraint violation distribution*. The parameter θ is thus a *violation budget*: the expected violation of the $\Phi r \leq T\Phi r$ constraint, under the distribution π, must be less than θ. The balance of the paper is concerned with establishing that the SALP forms the basis of a useful ADP algorithm in large scale problems:

- We identify a concrete choice of violation budget θ and an idealized constraint violation distribution π for which the SALP provides a useful relaxation in that the optimal solution can be a better approximation to the optimal cost-to-go function. This brings the cartoon improvement in Figure 1 to fruition for general problems.

- We show that the SALP is tractable (i.e it is well approximated by an appropriate 'sampled' version) and present computational experiments for a hard problem (Tetris) illustrating an order of magnitude improvement over the ALP.

4 Analysis

This section is dedicated to a theoretical analysis of the SALP. The overarching objective of this analysis is to provide some assurance of the soundness of the proposed approach. In addition, our analysis will serve as a crucial guide to practical implementation of the SALP. Our analysis will present two types of results: First, we prove approximation guarantees (Sections 4.1 and 4.2) that will indicate that the SALP computes approximations that are of comparable quality to the projection of J^* on the linear span of Φ. Second, we show (Section 4.3) that an implementable 'sampled' version of the SALP may be used to approximate the SALP with a tractable number of samples. All proofs can be found in the technical appendix.

Idealized Assumptions: Given the broad scope of problems addressed by ADP algorithms, analyses of such algorithms typically rely on an 'idealized' assumption of some sort. In the case of the ALP, one either assumes the ability to solve a linear program with as many constraints as there are states, or absent that, knowledge of a certain idealized sampling distribution, so that one can then proceed with solving a 'sampled' version of the ALP. Our analysis of the SALP in this section is predicated on the knowledge of an idealized constraint violation distribution, which is this same idealized sampling distribution. In particular, we will require access to samples drawn according to the distribution $\pi_{\mu^*,\nu}$ given by $\pi_{\mu^*,\nu}^\top \triangleq (1-\alpha)\nu^\top(I-\alpha P_{\mu^*})^{-1}$. Here ν is an arbitrary initial distribution over states. The distribution $\pi_{\mu^*,\nu}$ may be interpreted as yielding the discounted expected frequency of visits to a given state when the initial state is distributed according to ν and the system runs under the optimal policy μ^*. We note that the 'sampled' ALP introduced by de Farias and Van Roy [2] requires access to states sampled according to precisely this distribution.

4.1 A Simple Approximation Guarantee

We present a first, simple approximation guarantee for the following specialization of the SALP in (3):

$$\begin{array}{lll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r & \\ \text{subject to} & \Phi r \leq T\Phi r + s, & \\ & \pi_{\mu^*,\nu}^\top s \leq \theta, & s \geq 0. \end{array} \tag{4}$$

Before we proceed to state our result, we define a useful function:

$$\begin{array}{lll} \ell(r,\theta) \triangleq & \underset{s,\gamma}{\text{minimize}} & \gamma \\ & \text{subject to} & \Phi r - T\Phi r \leq s + \gamma \mathbf{1}, \\ & & \pi_{\mu^*,\nu}^\top s \leq \theta, \quad s \geq 0. \end{array} \tag{5}$$

$\ell(r,\theta)$ is the minimum translation (in the direction of the vector $\mathbf{1}$) of an arbitrary weight vector r so as to result in a feasible vector for (4). We will denote by $s(r,\theta)$ the s component of the solution to (5). The following Lemma characterizes $l(r,\theta)$:

Lemma 1. *For any $r \in \mathbb{R}^K$ and $\theta \geq 0$:*

(i) $\ell(r,\theta)$ is a bounded, decreasing, piecewise linear, convex function of θ.

(ii) $\ell(r,\theta) \leq (1+\alpha)\|J^* - \Phi r\|_\infty$.

(iii) $\frac{\partial}{\partial r}\ell(r,0) = -\frac{1}{\sum_{x\in\Omega(r)} \pi_{\mu^*,\nu}(x)}$, where $\Omega(r) = \text{argmax}_{x\in\mathcal{X}}\, \Phi r(x) - T\Phi r(x)$.

Armed with this definition, we are now in a position to state our first, crude approximation guarantee:

Theorem 1. *Let* $\mathbf{1}$ *be in the span of* Φ *and* ν *be a probability distribution. Let* $\bar{r}$ *be an optimal solution to the SALP* (4)*. Moreover, let* r^* *satisfy* $r^* \in \operatorname{argmin}_r \|J^* - \Phi r\|_\infty$*. Then,*

$$\|J^* - \Phi\bar{r}\|_{1,\nu} \leq \|J^* - \Phi r^*\|_\infty + \frac{l(r^*, \theta) + 2\theta}{1-\alpha}.$$

The above theorem allows us to interpret $\frac{\ell(r^*,\theta)+2\theta}{1-\alpha}$ as the approximation error associated with the SALP solution $\bar{r}$. Consider setting $\theta = 0$, in which case (4) is identical to the ALP. In this case, we have from Lemma 1 that $\ell(r^*, 0) \leq (1+\alpha)\|J^* - \Phi r^*\|_\infty$, so that the right hand side of our bound is at most $\frac{2}{1-\alpha}\|J^* - \Phi r^*\|_\infty$. This is precisely Theorem 2 in de Farias and Van Roy [1]; we recover their approximation guarantee for the ALP. Next observe that, from (iii), if the set $\Omega(r^*)$ is of small probability according to the distribution $\pi_{\mu^*,\nu}$, we expect that $\ell(r^*, \theta)$ will decrease dramatically as θ is increased from 0. In the event that $\Phi r^*(x) - T\Phi r^*(x)$ is large for only a small number of states (that is, the Bellman error of the approximation produced by r^* is large for only a small number of states), we thus expect to have a choice of θ for which $l(r^*, \theta) + 2\theta \ll l(r^*, 0)$. Thus, Theorem 1 reinforces the intuition (shown via Figure 1) that the SALP will permit closer approximations to J^* than the ALP.

The bound in Theorem 1 leaves room for improvement:

1. The right hand side of our bound measures projection error, $\|J^* - \Phi r^*\|_\infty$ in the L^∞-norm. Since it is unlikely that the basis functions Φ will provide a uniformly good approximation over the entire state space, the right hand side of our bound could be quite large.
2. The choice of state relevance weights can significantly influence the solution. While we do not show this here, this choice allows us to choose regions of the state space where we would like a better approximation of J^*. The right hand side of our bound, however, is independent of ν.
3. Our guarantee does not suggest a concrete choice of the violation budget, θ.

The next section will present a substantially refined approximation bound.

4.2 A Better Approximation Guarantee

With the intent of deriving stronger approximation guarantees, we begin this section by introducing a 'nicer' measure of the quality of approximation afforded by Φ. In particular, instead of measuring $\|J^* - \Phi r^*\|$ in the L^∞ norm as we did for our previous bounds, we will use a weighted max norm defined according to: $\|J\|_{\infty,1/\psi} \triangleq \max_{x\in\mathcal{X}} |J(x)|/\psi(x)$, where $\psi : \mathcal{X} \to [1, \infty)$ is a given *weighting function*. The weighting function ψ allows us to weight approximation error in a non-uniform fashion across the state space and in this manner potentially ignore approximation quality in regions of the state space that 'don't matter'. In addition to specifying the constraint violation distribution π as we did for our previous bound, we will specify (implicitly) a particular choice of the violation budget θ. In particular, we will consider solving the following SALP:

$$\begin{array}{ll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r - \frac{2\pi_{\mu^*,\nu}^\top s}{1-\alpha} \\ \text{subject to} & \Phi r \leq T\Phi r + s, \quad s \geq 0. \end{array} \tag{6}$$

It is clear that (6) is equivalent to (4) for a specific choice of θ. We then have:

Theorem 2. *Let* $\Psi \triangleq \{y \in \mathbb{R}^{|\mathcal{X}|} : y \geq \mathbf{1}\}$*. For every* $\psi \in \Psi$*, let* $\beta(\psi) = \max_\mu \left\| \frac{P_\mu \psi}{\psi} \right\|_\infty$*. Then, for an optimal solution* $(r_{SALP}, \bar{s})$ *to* (6)*, we have:*

$$\|J^* - \Phi r_{SALP}\|_{1,\nu} \leq \inf_{r,\psi\in\Psi} \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \left(\nu^\top \psi + \frac{2(\pi_{\mu^*,\nu}^\top \psi + 1)(\alpha\beta(\psi) + 1)}{1-\alpha} \right).$$

It is worth placing the result in context to understand its implications. For this, we recall a closely related result shown by de Farias and Van Roy [1] for the ALP. In particular, de Farias and Van Roy [1] showed that *given* an appropriate weighting (or in their context, 'Lyapunov') function ψ, one may solve an ALP, with ψ in the span of the basis functions Φ; the solution to such an ALP then satisfies:

$$\|J^* - \Phi\bar{r}\|_{1,\nu} \leq \inf_r \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \frac{2\nu^\top \psi}{1-\alpha\beta(\psi)}$$

provided $\beta(\psi) \leq 1/\alpha$. Selecting an appropriate ψ in their context is viewed to be an important task for practical performance and often requires a good deal of problem specific analysis; de Farias and Van Roy [1] identify appropriate ψ for several queueing models (note that this is equivalent to identifying a desirable basis function). In contrast, the guarantee we present optimizes over *all possible* ψ[1]. Thus, the approximation guarantee of Theorem 2 allows us to view the SALP as *automating* the critical procedure of identifying a good Lyapunov function for a given problem.

4.3 Sample Complexity

Our analysis thus far has assumed we have the ability to solve the SALP, a program with a potentially intractable number of constraints and variables. As it turns out, a solution to the SALP is well approximated by the solution to a certain 'sampled' program which we now describe: Let $\hat{\mathcal{X}} = \{x_1, x_2, \ldots, x_S\}$ be an ordered collection of S states drawn independently from $\mathcal{X}$ according to the distribution $\pi_{\mu^*,\nu}$. Let us consider solving the following program which we call the sampled SALP:

$$\begin{array}{ll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r - \frac{2}{(1-\alpha)S} \sum_{x\in\hat{\mathcal{X}}} s(x) \\ \text{subject to} & (\Phi r)(x) \leq (T\Phi r)(x) + s(x),\ \forall\ x \in \hat{\mathcal{X}}, \\ & r \in \mathcal{N},\ s \geq 0. \end{array} \tag{7}$$

Here $\mathcal{N} \in \mathbb{R}^m$ is a parameter set chosen to contain the optimal solution to the SALP (6), r_{SALP}. Notice that (7) is a linear program with S variables and $S|\mathcal{A}|$ constraints. For a moderate number of samples S, this is is easily solved. We will provide a sample complexity bound that indicates that for a number of samples S that scales linearly with the dimension of Φ, K, and that need not depend on the size of the state space, the solution to the sampled SALP satisfies, with high probability, the approximation guarantee presented for the SALP solution in Theorem 2.

Let us define the constant $B \triangleq \sup_{r\in\mathcal{N}} \|(\Phi r - T\Phi r)^+\|_\infty$. This quantity is closely related to the diameter of the region $\mathcal{N}$. We then have:

Theorem 3. *Under the conditions of Theorem 2, let r_{SALP} be an optimal solution to the SALP (6), and let $\hat{r}_{SALP}$ be an optimal solution to the sampled SALP (7). Assume that $r_{SALP} \in \mathcal{N}$. Further, given $\epsilon \in (0, B]$ and $\delta \in (0, 1/2]$, suppose that the number of sampled states S satisfies*

$$S \geq \frac{64B^2}{\epsilon^2}\left(2(K+2)\log\frac{16eB}{\epsilon} + \log\frac{8}{\delta}\right).$$

Then, with probability at least $1 - \delta - 2^{-383}\delta^{128}$,

$$\|J^* - \Phi\hat{r}_{SALP}\|_{1,\nu} \leq \inf_{\substack{r\in\mathcal{N} \\ \psi\in\Psi}} \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi}\left(\nu^\top\psi + \frac{2(\pi_{\mu^*,\nu}^\top\psi + 1)(\alpha\beta(\psi)+1)}{1-\alpha}\right) + \frac{4\epsilon}{1-\alpha}.$$

Theorem 3 establishes that the sampled SALP provides a close approximation to the solution of the SALP, in the sense that the approximation guarantees we established for the SALP are approximately valid for the solution to the sampled version with high probability. The number of samples we require to accomplish this task is specified precisely via the theorem. This number depends linearly on the number of basis functions and the diameter of the

[1]This includes those ψ that do not satisfy the Lyapunov condition $\beta(\psi) \leq 1/\alpha$.

feasible region, but is otherwise independent of the size of the state space for the MDP under consideration. It is worth juxtaposing our sample complexity result with that available for the ALP. In particular, we recall that the ALP has a large number of constraints but a *small* number of variables; the SALP is thus, at least superficially, a significantly more complex program. Exploiting the fact that the ALP has a small number of variables, de Farias and Van Roy [2] establish a sample complexity bound for a sampled version of the ALP analogous (7). The number of samples required for this sampled ALP to produce a good approximation to the ALP can be shown to depend on the same problem parameters we have identified here, viz. B and the number of basis functions K. The sample complexity in that case is identical to the sample complexity bound established here up to constants and an additional multiplicative factor of B/ϵ (for the sampled SALP). Thus, the two sample complexity bounds are within polynomial terms of each other and we have established that the SALP is essentially no harder to solve than the ALP.

This section places the SALP on solid theoretical ground by establishing strong approximation guarantees for the SALP that represent a substantial improvement over those available for the ALP and sample complexity results that indicated that the SALP was implementable via sampling. We next present a computational study that tests the SALP relative to other ADP methods (including the ALP) on a hard problem (the game of Tetris).

5 Case Study: Tetris

Our interest in Tetris as a case study for the SALP algorithm is motivated by several facts. Theoretical results suggest that design of an optimal Tetris player is a difficult problem [4–6]. Tetris represents precisely the kind of large and unstructured MDP for which it is difficult to design heuristic controllers, and hence policies designed by ADP algorithms are particularly relevant. Moreover, Tetris has been employed by a number of researchers as a testbed problem [3, 7–9]. We follow the formulation of Tetris as a MDP presented by Farias and Van Roy [3]. The SALP methodology was applied as follows:

Basis functions. We employed the 22 basis functions originally introduced in [7].

State sampling. Given a sample size S, a collection $\hat{\mathcal{X}} \subset \mathcal{X}$ of S states was sampled. These samples were generated in an IID fashion from the stationary distribution of a (rather poor) baseline policy[2].

Optimization. Given the collection $\hat{\mathcal{X}}$ of sampled states, an increasing sequence of choices of the violation budget $\theta \geq 0$ is considered. For each choice of θ, the optimization program

$$
\begin{array}{lll}
\underset{r,s}{\text{maximize}} & \frac{1}{S}\sum_{x\in\hat{\mathcal{X}}}(\Phi r)(x) & \\
\text{subject to} & \Phi r(x) \leq T\Phi r(x) + s(x), & \forall\ x \in \hat{\mathcal{X}}, \\
& \frac{1}{S}\sum_{x\in\hat{\mathcal{X}}} s(x) \leq \theta, & \\
& s(x) \geq 0, & \forall\ x \in \hat{\mathcal{X}},
\end{array}
\tag{8}
$$

was solved. This program is a version of the original SALP (3), but with sampled empirical distributions in place of the state-relevance weights ν and the constraint violation distribution π. Note that (8) has $K + S$ decision variables and $S|\mathcal{A}|$ linear constraints. Because of the sparsity structure of the constraints, however, it is amenable to efficient solution via barrier methods, even for large values of S.

Evaluation. Given a vector of weights obtained by solving (8), the performance of the corresponding policy is evaluated via Monte Carlo simulation over $3,000$ games of Tetris. Performance is measured in terms of the average number of lines cleared in a single game.

For each pair (S, θ), the resulting *average* performance (averaged over 10 different sets of sampled states) is shown in Figure 2. It provides experimental evidence for the intuition expressed in Section 3 and the analytic result of Theorem 1: Relaxing the constraints of the ALP by allowing for a violation budget allows for better policy performance. As the violation budget θ is increased from 0, performance dramatically improves. At $\theta = 0.16384$, the performance peaks, and we get policies that is an order of magnitude better than ALP, and beyond that the performance deteriorates.

[2]Our baseline policy had an average performance of 113 points.

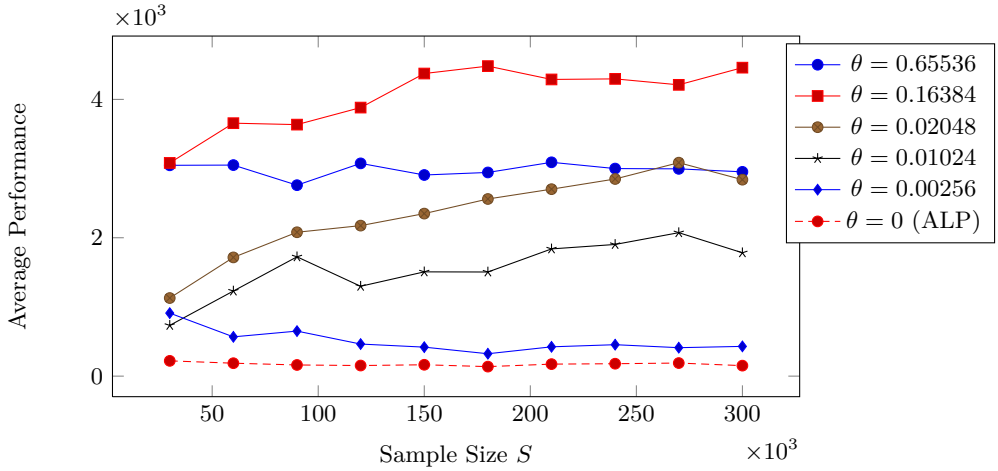


Figure 2: Average performance of SALP for different values of the number of sampled states S and the violation budget θ.

Table 1 summarizes the performance of best policies obtained by various ADP algorithms. Note that all of these algorithms employ the same basis function architecture. The ALP and SALP results are from our experiments, while the other results are from the literature. The best performance results of SALP is better by a factor of 2 in comparison to the competitors.

Algorithm	**Best Performance**	**CPU Time**
ALP	897	hours
TD-Learning [7]	3,183	minutes
ALP with bootstrapping [3]	4,274	hours
TD-Learning [8]	4,471	minutes
Policy gradient [9]	5,500	days
SALP	10,775	hours

Table 1: Comparison of the performance of the best policy found with various ADP methods.

Note that significantly better policies are possible with this basis function architecture than *any* of the ADP algorithms in Table 1 discover. Using a heuristic optimization method, Szita and Lőrincz [10] report policies with a remarkable average performance of 350,000. Their method is computationally intensive, however, requiring one month of CPU time. In addition, the approach employs a number of rather arbitrary Tetris specific 'modifications' that are ultimately seen to be critical to performance - in the absence of these modifications, the method is unable to find a policy for Tetris that scores above a few hundred points.

6 Future Directions

There are a number of interesting directions that remain to be explored. First, note that the bounds derived in Sections 4.1 and 4.2 are approximation guarantees, which provide bounds on the approximation error given by the SALP approach versus the best approximation possible with the particular set of basis functions. In preliminary work, we have also developed *performance guarantees*. These provide bounds on the performance of the resulting SALP policies, as a function of the basis architecture. Second, note that sample path variations of the SALP are possible. Rather than solving a large linear program, such an algorithm would optimize a policy in an online fashion along a single system trajectory. This would be in a manner reminiscent of stochastic approximation algorithms like TD-learning. However, a sample path SALP variation would inherit all of the theoretical bounds developed here. The design and analysis of such an algorithm is an exciting future direction.

References

[1] D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[2] D. P. de Farias and B. Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 293(3):462–478, 2004.

[3] V. F. Farias and B. Van Roy. Tetris: A study of randomized constraint sampling. In *Probabilistic and Randomized Methods for Design Under Uncertainty*. Springer-Verlag, 2006.

[4] J. Brzustowski. Can you win at Tetris? Master's thesis, University of British Columbia, 1992.

[5] H. Burgiel. How to lose at Tetris. *Mathematical Gazette*, page 194, 1997.

[6] E. D. Demaine, S. Hohenberger, and D. Liben-Nowell. Tetris is hard, even to approximate. In *Proceedings of the 9th International Computing and Combinatorics Conference*, 2003.

[7] D. P. Bertsekas and S. Ioffe. Temporal differences–based policy iteration and applications in neuro–dynamic programming. Technical Report LIDS–P–2349, MIT Laboratory for Information and Decision Systems, 1996.

[8] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[9] S. Kakade. A natural policy gradient. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[10] I. Szita and A. Lőrincz. Learning Tetris using the noisy cross-entropy method. *Neural Computation*, 18:2936–2941, 2006.

[11] D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.