Approximate Dynamic Programming via a Smoothed Linear Program

Vijay V. Desai
Industrial Engineering and Operations Research
Columbia University
email: vvd2101@columbia.edu

Vivek F. Farias
Sloan School of Management
Massachusetts Institute of Technology
email: vivekf@mit.edu

Ciamac C. Moallemi
Graduate School of Business
Columbia University
email: ciamac@gsb.columbia.edu

Initial Version: August 4, 2009
Current Revision: June 25, 2011

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 have typically relied on a natural 'projection' of a well studied linear program for exact dynamic programming. Such programs restrict attention to approximations that are lower bounds to the optimal cost-to-go function. Our program — the 'smoothed approximate linear program' — is distinct from such approaches and relaxes the restriction to lower bounding approximations in an appropriate fashion while remaining computationally tractable. Doing so appears to have several advantages: First, we demonstrate bounds on the quality of approximation to the optimal cost-to-go function afforded by our approach. These bounds are, in general, no worse than those available for extant LP approaches, and for specific problem instances can be shown to be arbitrarily stronger. Second, experiments with our approach on a pair of challenging problems (the game of Tetris and a queueing network control problem) show that the approach outperforms the existing LP approach (which has previously been shown to be competitive with several ADP algorithms) by a substantial margin.

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 programming optimal cost-to-go function within the span of some pre-specified set of basis functions.

ADP algorithms are typically motivated by exact algorithms for dynamic programming. The approximate linear programming (ALP) method is one such approach, motivated by the LP used for the computation of the optimal cost-to-go function. Introduced by Schweitzer and Seidmann (1985) and analyzed and further developed by de Farias and Van Roy (2003, 2004), this approach is attractive for a number of reasons. First, the availability of efficient solvers for linear programming makes the ALP approach easy to implement. Second, the approach offers attractive theoretical guarantees. In particular, the quality of the approximation to the cost-to-go function produced by the ALP approach can be shown to compete, in an appropriate sense, with the quality of the best possible approximation afforded by the set of basis functions used. A testament to the success of the ALP approach is the number of applications it has seen in recent years in large scale dynamic optimization problems. These applications range from the control of queueing networks to revenue management to the solution of large scale stochastic games.

The optimization program employed in the ALP approach is in some sense the most natural linear programming formulation for ADP. In particular, the ALP is identical to the linear program 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 LP implicitly restricts attention to approximations that are lower bounds to the optimal cost-to-go function. The structure of this program appears crucial in establishing guarantees on the quality of approximations produced by the approach (de Farias and Van Roy, 2003, 2004); these approximation guarantees were remarkable and a first for any ADP method. That said, the restriction to lower bounds naturally leads one to ask whether the program employed by the ALP approach is the 'right' math programming formulation for ADP. In particular, it may be advantageous to consider a generalization of the ALP approach that relaxes the lower bound requirement so as to allow for a better approximation, and, ultimately, better policy performance. Is there an alternative formulation that permits better approximations to the cost-to-go function while remaining computationally tractable? Motivated by this question, the present paper introduces a new linear program for ADP we call the 'smoothed' approximate linear program (or SALP). This program is a generalization of the ALP method. We believe that the SALP represents a useful new math programming formulation for ADP. In particular, we make the following contributions:

1. We are able to establish strong approximation and performance guarantees for approximations to the cost-to-go function produced by the SALP. Our analyses broadly follow the approach of de Farias and Van Roy (2003, 2004) for the ALP. The resultant guarantees are no worse than the corresponding guarantees for the ALP, and we demonstrate that they can be *substantially* stronger in certain cases.

2. 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. Moreover, we identify structural properties of the sampled SALP that can be exploited for fast optimization. Our sample complexity results and these structural observations allow us to conclude that the SALP scales similarly in computational complexity as existing LP formulations for ADP.

3. We present computational studies demonstrating the efficacy of our approach in the setting of two different challenging control problems. In the first study, we consider the game of Tetris. Tetris is a notoriously difficult, 'unstructured' dynamic optimization problem and has been used as a convenient testbed problem for numerous ADP approaches. The ALP has been demonstrated to be competitive with other ADP approaches for Tetris, such as temporal difference learning or policy gradient methods (see Farias and Van Roy, 2006). In detailed comparisons with the ALP, we show that the SALP provides an *order of magnitude* improvement over controllers designed via that approach for the game of Tetris. In the second computational study, we consider the optimal control of a 'criss-cross' queueing network. this is a challenging network control problem and a difficult test problem as witnessed by antecedent literature. Under several distinct parameter regimes, we show here that the SALP adds substantial value over the ALP approach.

   In addition to these results, the SALP method has recently been considered in other applications with favorable results: this includes work on a high-dimensional production optimization problem in oil exploration (Wen et al., 2011), and work studying large scale dynamic oligopoly models (Farias et al., 2011).

The literature on ADP algorithms is vast and we make no attempt to survey it here. Van Roy (2002) or Bertsekas (2007a, Chap. 6) provide good, brief overviews, while Bertsekas and Tsitsiklis (1996) and Powell (2007) are encyclopedic references on the topic. The exact LP for the solution of dynamic programs is attributed to Manne (1960). The ALP approach to ADP was introduced by Schweitzer and Seidmann (1985) and de Farias and Van Roy (2003, 2004). de Farias and Van Roy (2003) establish approximation guarantees for the ALP approach. These guarantees are especially strong if the basis spans suitable 'Lyapunov'-like functions. The approach we present yields strong bounds if any such Lyapunov function exists, whether or not it is spanned by the basis. de Farias and Van Roy (2006) introduce a program for average cost approximate dynamic programming that resembles the SALP; a critical difference is that their program requires the relative violation allowed across ALP constraints be specified as input. Contemporaneous with the present work, Petrik and Zilberstein (2009) propose a relaxed linear program for approximating the cost-to-go function of a dynamic program. This linear program is similar to the SALP program (14) herein. The crucial determinant of performance in either program is a certain choice of Lagrange multipliers. The present paper explicitly identifies such a choice, and for this choice, develops concrete approximation guarantees that compare favorably with guarantees available for the ALP. In addition, the choice of Lagrange multipliers identified also proves to be practically valuable as is borne out by our experiments. In contrast, Petrik and Zilberstein (2009) stop short of identifying this crucial input and thus provide neither approximation guarantees nor a 'generic' choice of multipliers for practical applications.

The remainder of this paper is organized as follows: In Section 2, we formulate the approximate dynamic programming setting and describe the ALP approach. The smoothed ALP is developed as a relaxation of the ALP in Section 3. Section 4 provides a theoretical analysis of the SALP, in terms of approximation and performance guarantees, as well as a sample complexity bound. In Section 5, we describe the practical implementation of the SALP method, illustrating how parameter choices can be made as well as how to efficiently solve the resulting optimization program. Section 6 contains the computational study of the game Tetris, while the computational study in Section 7 considers a queueing application. We conclude in Section 8.

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 $\mathcal{A}$. At time t, given the current state x_t and a choice of action 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 $\mu\colon \mathcal{X} \to \mathcal{A}$ is a mapping that determines the choice of 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

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

Here, $\alpha \in (0, 1)$ is the discount factor. The expectation is taken under the assumption that actions are selected according to the policy μ. In other words, at each time t, $a_t \triangleq \mu(x_t)$.

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 can be written in vector form as

$$J_\mu = \sum_{t=0}^{\infty} \alpha^t P_\mu^t g_\mu.$$

Further, 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 $T_\mu J = g_\mu + \alpha P_\mu J$.

Our goal is to find an optimal stationary policy μ^*, that is, a policy that minimizes the expected discounted cost from every state x. In particular,

$$\mu^*(x) \in \operatorname*{argmin}_{\mu} J_\mu(x), \quad \forall\ x \in \mathcal{X}.$$

The Bellman operator T is defined component-wise according to

$$(TJ)(x) \triangleq \min_{a \in \mathcal{A}} \; g(x,a) + \alpha \sum_{x' \in \mathcal{X}} P_a(x,x')J(x'), \quad \forall \; x \in \mathcal{X}.$$

Bellman's equation is then the fixed point equation

$$TJ = J. \tag{1}$$

Standard results in dynamic programming establish that the optimal cost-to-go function J^* is the unique solution to Bellman's equation (see, for example, Bertsekas, 2007a, Chap. 1). Further, if μ^* is a policy that is greedy with respect to J^* (i.e., μ^* satisfies $TJ^* = T_{\mu^*}J^*$), then μ^* is an optimal policy.

2.1. The Linear Programming Approach

A number of computational approaches are available for the solution of the Bellman equation. One approach involves solving the optimization program:

$$\begin{array}{ll} \underset{J}{\text{maximize}} & \nu^\top J \\ \text{subject to} & J \le TJ. \end{array} \tag{2}$$

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 for each state x, the constraint $J(x) \le (TJ)(x)$ is equivalent to the set of $|\mathcal{A}|$ linear constraints

$$J(x) \le g(x,a) + \alpha \sum_{x' \in \mathcal{X}} P_a(x,x')J(x'), \quad \forall \; a \in \mathcal{A}.$$

We refer to (2), which is credited to Manne (1960), as the *exact LP*. A simple argument, included here for completeness, establishes that J^* is the unique optimal solution: suppose that a vector J is feasible for the exact LP (2). Since $J \le TJ$, monotonicity of the Bellman operator implies that $J \le T^kJ$, for any integer $k \ge 1$. Since the Bellman operator T is a contraction, T^kJ must converge to the unique fixed point J^* as $k \to \infty$. Thus, we have that $J \le J^*$. Then, it is clear that every feasible point for (2) is a component-wise lower bound to J^*. Since J^* itself is feasible for (2), it must be that J^* is the unique optimal solution to the exact LP.

2.2. The Approximate Linear Program

In many problems, the size of the state space is enormous due to the curse of dimensionality. In such cases, it may be prohibitive to store, much less compute, the optimal cost-to-go function J^*. In approximate dynamic programming (ADP), the goal is to find tractable approximations to the optimal cost-to-go function J^*, with the hope that they will lead to good policies.

Specifically, consider a collection of *basis functions* $\{\phi_1, \ldots, \phi_K\}$ where each $\phi_i \colon \mathcal{X} \to \mathbb{R}$ is a real-valued function on the state space. ADP algorithms seek to find linear combinations of the basis functions that provide good approximations to the optimal cost-to-go function. In particular, we seek a vector of weights $r \in \mathbb{R}^K$ so that

$$J^*(x) \approx J_r(x) \triangleq \sum_{i=1}^{K} \phi_i(x) r_i = \Phi r(x).$$

Here, we define $\Phi \triangleq [\phi_1 \ \phi_2 \ \ldots \ \phi_K]$ to be a matrix with columns consisting of the basis functions. Given a vector of weights r and the corresponding value function approximation Φr, a policy μ_r is naturally defined as the 'greedy' policy with respect to Φr, i.e. as $T_{\mu_r} \Phi r = T \Phi r$.

One way to obtain a set of weights is to solve the exact LP (2), but restricting to the low-dimensional subspace of vectors spanned by the basis functions. This leads to the *approximate linear program* (ALP), introduced by Schweitzer and Seidmann (1985), which is defined by

$$\begin{array}{ll} \underset{r}{\text{maximize}} & \nu^\top \Phi r \\ \text{subject to} & \Phi r \leq T \Phi r. \end{array} \tag{3}$$

For the balance of the paper, we will make the following assumption:

Assumption 1. *Assume the ν is a probability distribution ($\nu \geq \mathbf{0}$, $\mathbf{1}^\top \nu = 1$), and that the constant function $\mathbf{1}$ is in the span of the basis functions Φ.*

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. By the discussion in Section 2.1, 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.

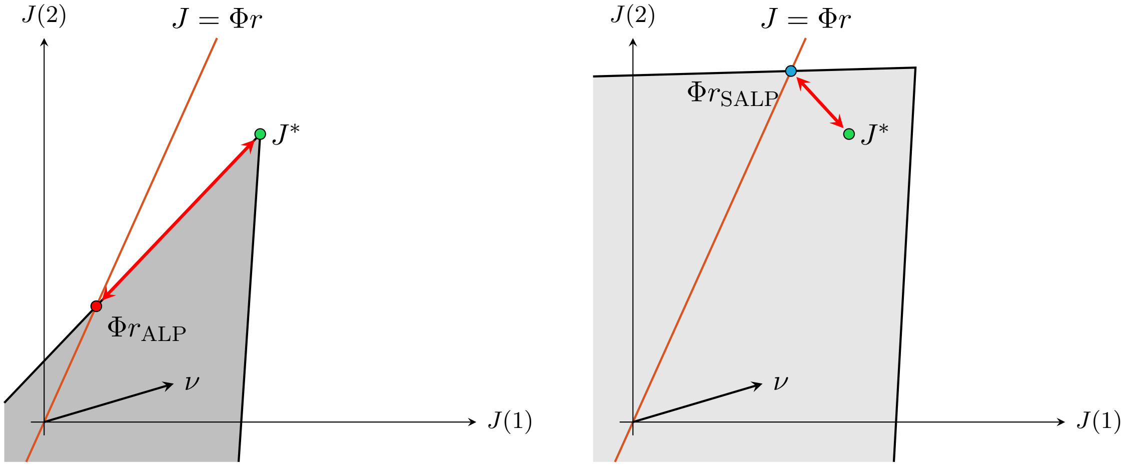
One can thus interpret the ALP solution r_{ALP} equivalently as the optimal solution to the program

$$\begin{array}{ll} \underset{r}{\text{minimize}} & \|J^* - \Phi r\|_{1,\nu} \\ \text{subject to} & \Phi r \leq T \Phi r. \end{array} \tag{4}$$

Here, the weighted 1-norm in the objective is defined by

$$\|J^* - \Phi r\|_{1,\nu} \triangleq \sum_{x \in \mathcal{X}} \nu(x) |J^*(x) - \Phi r(x)|.$$

This implies that the approximate LP will find the closest approximation (in the appropriate norm) to the optimal cost-to-go function, out of all approximations satisfying the constraints of the exact LP.



(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. It is easy to construct a concrete two state example with the above features.

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, the ultimate goal is not to find a *lower bound* on the optimal cost-to-go function, but rather to find a *good approximation*. In these instances, it may be 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(a) 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:

$$
\begin{aligned}
&\underset{r,s}{\text{maximize}} && \nu^\top \Phi r \\
&\text{subject to} && \Phi r \leq T\Phi r + s, \\
& && \pi^\top s \leq \theta, \quad s \geq \mathbf{0}.
\end{aligned}
\tag{5}
$$

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 SALP can be alternatively written as

$$
\begin{aligned}
&\underset{r}{\text{maximize}} && \nu^\top \Phi r \\
&\text{subject to} && \pi^\top (\Phi r - T\Phi r)^+ \leq \theta.
\end{aligned}
\tag{6}
$$

Here, given a vector J, $J^+(x) \triangleq \max(J(x), 0)$ is defined to be the component-wise positive part. Note that, when $\theta = 0$, the SALP is equivalent to the ALP. When $\theta > 0$, the SALP replaces the 'hard' constraints of the ALP with 'soft' constraints in the form of a hinge-loss function.

The balance of the paper is concerned with establishing that the SALP forms the basis of a useful approximate dynamic programming 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 can provide substantial benefits over ALP as evidenced in application of SALP to the game of Tetris and queueing network control problem.

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 some instances, the bounds we provide will be directly comparable to bounds that have been developed for the ALP method. As such, a relative consideration of the bounds in these two cases can provide a theoretical comparison between the ALP and SALP methods. In addition, our analysis will serve as a crucial guide to practical implementation of the SALP as will be described in Section 5. In particular, the theoretical analysis presented here provides intuition as to how to select parameters such as the state-relevance weights and the constraint violation distribution. We note that all of our bounds are relative to a measure of how well the approximation architecture employed is capable of approximating the optimal cost-to-go function; it is unreasonable to expect non-trivial bounds that are independent of the architecture used.

Our analysis will present three types of results:

- Approximation guarantees (Sections 4.2–4.4): We establish bounds on the distance between approximations computed by the SALP and the optimal value function J^*, relative to the distance between the best possible approximation afforded by the chosen basis functions and J^*. These guarantees will indicate that the SALP computes approximations that are of comparable quality to the projection[1] of J^* on to the linear span of Φ. We explicitly demonstrate our approximation guarantees in the context of a simple, concrete queueing example, and show that they can be much stronger than corresponding guarantees for the ALP.

- Performance bounds (Section 4.5): While it is desirable to approximate J^* as closely as possible, an important concern is the quality of the policies generated by acting greedily according to such approximations, as measured by their performance. We present bounds on the performance loss incurred, relative to the optimal policy, in using an SALP approximation.

- Sample complexity results (Section 4.6): The SALP is a linear program with a large number of constraints as well as variables. In practical implementations, one may consider a 'sampled' version of this program that has a manageable number of variables and constraints. We present sample complexity guarantees that establish bounds on the number of samples required to produce a good approximation to the solution of the SALP. These bounds scale linearly with the number of basis function K and are independent of the size of the state space $\mathcal{X}$.

4.1. Idealized Assumptions

Our analysis of the SALP in this section is predicated on the knowledge of an idealized probability distribution over states. In particular, letting μ^* be an optimal policy and P_{μ^*} the associated transition matrix, we will require knowledge of the distribution $\pi_{\mu^*,\nu}$ given by

$$\pi_{\mu^*,\nu}^\top \triangleq (1-\alpha)\nu^\top (I - \alpha P_{\mu^*})^{-1} = (1-\alpha)\sum_{t=0}^{\infty} \alpha^t \nu^\top P_{\mu^*}^t. \tag{7}$$

Here, ν is an initial distribution over states satisfying Assumption 1. The distribution $\pi_{\mu^*,\nu}$ may be interpreted as yielding the discounted expected frequency of visits to a given state when the

[1] Note that it is intractable to directly compute the projection since J^* is unknown.

initial state is distributed according to ν and the system runs under the policy μ^*. The distribution $\pi_{\mu^*,\nu}$ will be used as the SALP constraint violation distribution in order to develop approximation bounds (Theorems 1–2) and a performance bound (Theorem 3), and as a sampling distribution in our analysis of sample complexity (Theorem 4).

We note that assumptions such as knowledge of the idealized distribution described in the preceding paragraph are not unusual in the analysis of ADP algorithms. In the case of the ALP, one either assumes the the ability to solve a linear program with as many constraints as there are states, or absent that, the 'sampled' ALP introduced by de Farias and Van Roy (2004) requires access to states sampled according to precisely this distribution. Theoretical analyses of other approaches to approximate DP such as approximate value iteration and temporal difference learning similarly rely on the knowledge of specialized sampling distributions that cannot be obtained tractably (see de Farias and Van Roy, 2000).

4.2. A Simple Approximation Guarantee

This section presents a first, simple approximation guarantee for the following specialization of the SALP in (5),

$$\begin{array}{ll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r \\ \text{subject to} & \Phi r \leq T\Phi r + s, \\ & \pi_{\mu^*,\nu}^\top s \leq \theta, \quad s \geq \mathbf{0}. \end{array} \tag{8}$$

Here, the constraint violation distribution is set to be $\pi_{\mu^*,\nu}$.

Before we state our approximation guarantee, consider the following function:

$$\begin{array}{lll} \ell(r,\theta) \triangleq & \underset{s,\gamma}{\text{minimize}} & \gamma/(1-\alpha) \\ & \text{subject to} & \Phi r \leq T\Phi r + s + \gamma \mathbf{1}, \\ & & \pi_{\mu^*,\nu}^\top s \leq \theta, \quad s \geq \mathbf{0}. \end{array} \tag{9}$$

We will denote by $s(r,\theta)$ the s component of the solution to (9). Armed with this definition, we are now in a position to state our first, crude approximation guarantee:

Theorem 1. *Suppose that r_{SALP} is an optimal solution to the SALP (8), and let r^* satisfy*

$$r^* \in \underset{r}{\text{argmin}} \ \|J^* - \Phi r\|_\infty.$$

Then,

$$\|J^* - \Phi r_{SALP}\|_{1,\nu} \leq \|J^* - \Phi r^*\|_\infty + \ell(r^*,\theta) + \frac{2\theta}{1-\alpha}. \tag{10}$$

As we will see shortly in the proof of Theorem 1, given a vector r of basis function weights and a violation budget θ, the quantity $\ell(r,\theta)$ obtained by solving (9) defines the minimal translation (in the direction of the constant vector $\mathbf{1}$) of r that yields a feasible solution for (8). The above theorem allows us to interpret $\ell(r^*,\theta) + 2\theta/(1-\alpha)$ as an upper bound to the approximation error (in the $\|\cdot\|_{1,\nu}$ norm) associated with the SALP solution r_{SALP}, relative to the error of the *best* approximation r^* (in the $\|\cdot\|_\infty$ norm). Note that this upper bound cannot be computed, in general, since r^* is unknown.

Theorem 1 provides justification for the intuition, described in Section 3, that a relaxation of the feasible region of the ALP will result in better value function approximations. To see this, first consider the following lemma (whose proof may be found in Appendix A) that characterizes the function $\ell(r,\theta)$:

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

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

(ii)

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

(iii) The right partial derivative of $\ell(r,\theta)$ with respect to θ satisfies

$$\frac{\partial^+}{\partial\theta^+}\ell(r,0) = -\left((1-\alpha)\sum_{x\in\Omega(r)} \pi_{\mu^*,\nu}(x)\right)^{-1},$$

where

$$\Omega(r) \triangleq \underset{\{x\in\mathcal{X}\,:\,\pi_{\mu^*,\nu}(x)>0\}}{\operatorname{argmax}} \quad \Phi r(x) - T\Phi r(x).$$

Then, we have the following corollary:

Corollary 1. *Define $U_{SALP}(\theta)$ to be the upper bound in* (10)*, i.e.,*

$$U_{SALP}(\theta) \triangleq \|J^* - \Phi r^*\|_\infty + \ell(r^*,\theta) + \frac{2\theta}{1-\alpha}.$$

Then:

(i)

$$U_{SALP}(0) \leq \frac{2}{1-\alpha}\|J^* - \Phi r^*\|_\infty.$$

(ii) The right partial derivative of $U_{SALP}(\theta)$ with respect to θ satisfies

$$\frac{d^+}{d\theta^+}U_{SALP}(0) = \frac{1}{1-\alpha}\left[2 - \left(\sum_{x\in\Omega(r^*)} \pi_{\mu^*,\nu}(x)\right)^{-1}\right].$$

Proof. The result follows immediately from Parts (ii) and (iii) of Lemma 1. ■

Suppose that $\theta = 0$, in which case the SALP (8) is identical to the ALP (3), thus, $r_{\text{SALP}} = r_{\text{ALP}}$. Applying Part (i) of Corollary 1, we have, for the ALP, the approximation error bound

$$\|J^* - \Phi r_{\text{ALP}}\|_{1,\nu} \leq \frac{2}{1-\alpha}\|J^* - \Phi r^*\|_\infty. \tag{11}$$

This is precisely Theorem 2 of de Farias and Van Roy (2003); we recover their approximation guarantee for the ALP.

Now observe that, from Part (ii) of Corollary 1, if the set $\Omega(r^*)$ is of very small probability according to the distribution $\pi_{\mu^*,\nu}$, we expect that the upper bound $U_{\text{SALP}}(\theta)$ may decrease rapidly as θ is increased from 0.[2] In other words, if the Bellman error $\Phi r^*(x) - T\Phi r^*(x)$ produced by r^* is maximized at states x that are collectively of very small probability, then we expect to have a choice of $\theta > 0$ for which $U_{\text{SALP}}(\theta) < U_{\text{SALP}}(0)$. In this case, the bound (10) on the SALP solution will be an improvement over the bound (11) on the ALP solution.

Before we present the proof of Theorem 1 we present an auxiliary claim that we will have several opportunities to use. The proof can be found in Appendix A.

Lemma 2. *Suppose that the vectors $J \in \mathbb{R}^{\mathcal{X}}$ and $s \in \mathbb{R}^{\mathcal{X}}$ satisfy*

$$J \leq T_{\mu^*}J + s.$$

Then,

$$J \leq J^* + \Delta^* s,$$

where

$$\Delta^* \triangleq \sum_{k=0}^{\infty}(\alpha P_{\mu^*})^k = (I - \alpha P_{\mu^*})^{-1},$$

and P_{μ^*} is the transition probability matrix corresponding to an optimal policy.

A feasible solution to the ALP is necessarily a lower bound to the optimal cost-to-go function, J^*. This is no longer the case for the SALP; the above lemma characterizes the extent to which this restriction is relaxed. In particular, if (r, s) is feasible for the SALP (8), then,

$$\Phi r \leq J^* + \Delta^* s.$$

We now proceed with the proof of Theorem 1:

Proof of Theorem 1. Define the weight vector $\tilde{r} \in \mathbb{R}^m$ according to

$$\Phi\tilde{r} = \Phi r^* - \ell(r^*, \theta)\mathbf{1}.$$

[2] Already if $\pi_{\mu^*,\nu}(\Omega(r^*)) < 1/2$, then $\frac{d^+}{d\theta^+}U_{\text{SALP}}(0) < 0$.

Note that $\tilde{r}$ is well-defined since $\mathbf{1} \in \operatorname{span}(\Phi)$. Set $\tilde{s} = s(r^*, \theta)$, the s-component of a solution to the LP (9) with parameters r^* and θ. We will demonstrate that $(\tilde{r}, \tilde{s})$ is feasible for (8). Observe that, by the definition of the LP (9),

$$\Phi r^* \le T\Phi r^* + \tilde{s} + (1-\alpha)\ell(r^*, \theta)\mathbf{1}.$$

Then,

$$\begin{aligned} T\Phi\tilde{r} &= T\Phi r^* - \alpha\ell(r^*, \theta)\mathbf{1} \\ &\ge \Phi r^* - \tilde{s} - (1-\alpha)\ell(r^*, \theta)\mathbf{1} - \alpha\ell(r^*, \theta)\mathbf{1} \\ &= \Phi\tilde{r} - \tilde{s}. \end{aligned}$$

Now, let $(r_{\text{SALP}}, \bar{s})$ be a solution to the SALP (8). By Lemma 2,

$$\begin{aligned} \|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} &\le \|J^* - \Phi r_{\text{SALP}} + \Delta^*\bar{s}\|_{1,\nu} + \|\Delta^*\bar{s}\|_{1,\nu} \\ &= \nu^\top(J^* - \Phi r_{\text{SALP}} + \Delta^*\bar{s}) + \nu^\top\Delta^*\bar{s} \\ &= \nu^\top(J^* - \Phi r_{\text{SALP}}) + \frac{2\pi_{\mu^*,\nu}^\top\bar{s}}{1-\alpha} \\ &\le \nu^\top(J^* - \Phi r_{\text{SALP}}) + \frac{2\theta}{1-\alpha}. \end{aligned} \tag{12}$$

Since $(\tilde{r}, \tilde{s})$ is feasible for (8), we have that

$$\begin{aligned} \|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} &\le \nu^\top(J^* - \Phi\tilde{r}) + \frac{2\theta}{1-\alpha} \\ &= \nu^\top(J^* - \Phi r^*) + \nu^\top(\Phi r^* - \Phi\tilde{r}) + \frac{2\theta}{1-\alpha} \\ &= \nu^\top(J^* - \Phi r^*) + \ell(r^*, \theta) + \frac{2\theta}{1-\alpha} \\ &\le \|J^* - \Phi r^*\|_\infty + \ell(r^*, \theta) + \frac{2\theta}{1-\alpha}, \end{aligned} \tag{13}$$

as desired. ■

While Theorem 1 reinforces the intuition (shown via Figure 1) that the SALP will permit closer approximations to J^* than the ALP, the bound leaves room for improvement:

1. The right hand side of our bound measures projection error, $\|J^* - \Phi r^*\|_\infty$ in the $\|\cdot\|_\infty$ 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. As suggested by (4), the choice of state relevance weights can significantly influence the solution. In particular, it 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 parameter θ.

The next section will present a substantially refined approximation bound, that will address these issues.

4.3. A Stronger 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 the approximation error $J^* - \Phi r^*$ in the $\| \cdot \|_\infty$ norm as we did for our previous bounds, we will use a weighted max norm defined according to:

$$\|J\|_{\infty,1/\psi} \triangleq \max_{x\in\mathcal{X}} \frac{|J(x)|}{\psi(x)}.$$

Here, $\psi\colon \mathcal{X} \to [1,\infty)$ 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 are less relevant. We define Ψ to be the set of all weighting functions, i.e.,

$$\Psi \triangleq \left\{ \psi \in \mathbb{R}^{\mathcal{X}} \ : \ \psi \geq \mathbf{1} \right\}.$$

Given a particular $\psi \in \Psi$, we define a scalar

$$\beta(\psi) \triangleq \max_{x,a} \frac{\sum_{x'} P_a(x,x')\psi(x')}{\psi(x)}.$$

Note that $\beta(\psi)$ is an upper bound on the one-step expected value of ψ relative to the current value when evaluated along a state trajectory under an arbitrary policy, i.e.,

$$\mathsf{E}\left[\psi(x_{t+1}) \mid x_t = x,\ a_t = a\right] \leq \beta(\psi)\psi(x), \quad \forall\, x \in \mathcal{X},\ a \in \mathcal{A}.$$

When $\beta(\psi)$ is small, then $\psi(x_{t+1})$ is expected to be small relative to $\psi(x_t)$, hence $\beta(\psi)$ can be interpreted as a measure of system 'stability'.

In addition to specifying the sampling distribution π, as we did in Section 4.2, we will specify (implicitly) a particular choice of the violation budget θ. In particular, we will consider solving the following SALP:

$$\begin{array}{ll} \underset{r,s}{\text{maximize}} & \nu^\top \Phi r - \dfrac{2\pi_{\mu^*,\nu}^\top s}{1-\alpha} \\ \text{subject to} & \Phi r \leq T\Phi r + s, \quad s \geq \mathbf{0}. \end{array} \tag{14}$$

Note that (14) is a Lagrangian relaxation of (8). It is clear that (14) and (8) are equivalent in the sense that there exists a specific choice of θ so any optimal solution to (14) is an optimal solution to (8) (for a formal statement and proof of this fact see Lemma 4 in Appendix A). We then have:

Theorem 2. *If r_{SALP} is an optimal solution to* (14)*, then*

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

Before presenting a proof for this approximation guarantee, 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 (2003) for the ALP. They demonstrate that a solution r_{ALP} to the ALP (3) satisfies

$$\|J^* - \Phi r_{\text{ALP}}\|_{1,\nu} \leq \inf_{r,\psi\in\bar{\Psi}} \ \|J^* - \Phi r\|_{\infty,1/\psi} \frac{2\nu^\top\psi}{1-\alpha\beta(\psi)}, \tag{15}$$

where

$$\bar{\Psi} \triangleq \{\psi \in \Psi \ : \ \psi \in \text{span}(\Phi), \ \alpha\beta(\psi) < 1\}.$$

Note that (15) provides a bound over a collection of weighting functions ψ that are within the span of the basis Φ and satisfy a 'Lyapunov' condition $\beta(\psi) < 1/\alpha$. Suppose that there is a particular Lyapunov function ψ such that under the $\|\cdot\|_{\infty,1/\psi}$ norm, J^* is well approximated by a function in the span of Ψ, i.e., $\inf_r \ \|J^* - \Phi r\|_{\infty,1/\psi}$ is small. In order for the left-hand side of (15) also to be small and hence guarantee a small approximation error for the ALP, it must be the case that ψ is contained in the basis. Hence, being able to select a basis that spans suitable Lyapunov functions is viewed to be an important task in ensuring good approximation guarantees for the ALP. This often requires a good deal of problem specific analysis; de Farias and Van Roy (2003) identify appropriate ψ for several queueing models. To contrast with the SALP, the guarantee we present holds over *all possible* ψ (including those ψ that do not satisfy the Lyapunov condition $\beta(\psi) < 1/\alpha$, and that are not necessarily in the span of Φ). As we will see in Section 4.4, this difference can be significant.

To provide another comparison, let us focus attention on a particular choice of ν, namely $\nu = \pi_{\mu^*} \triangleq \pi_*$, the stationary distribution induced under an optimal policy μ^*. In this case, restricting attention to the set of weighting functions $\bar{\Psi}$ so as to make the two bounds comparable, Theorem 2 guarantees that

$$\begin{aligned} \|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} &\leq \inf_{r,\psi\in\bar{\Psi}} \ \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \left(\pi_*^\top \psi + \frac{2(\pi_*^\top \psi)(\alpha\beta(\psi)+1)}{1-\alpha} \right) \\ &\leq \inf_{r,\psi\in\bar{\Psi}} \ \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \frac{5\pi_*^\top\psi}{1-\alpha}. \end{aligned} \tag{16}$$

On the other hand, observing that $\beta(\psi) \geq 1$ for all $\psi \in \Psi$, the right hand side for the ALP bound

(15) is at least

$$\inf_{r,\psi\in\bar{\Psi}} \; \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \frac{2\pi_*^\top \psi}{1-\alpha}.$$

Thus, the approximation guarantee of Theorem 2 is at most a constant factor of 5/2 worse than the guarantee (15) for the ALP, and can be significantly better since it allows for the consideration of weighting functions outside the span of the basis.

Proof of Theorem 2. Let $r \in \mathbb{R}^m$ and $\psi \in \Psi$ be arbitrary. Define the vector $\tilde{s} \in \mathbb{R}^{\mathcal{X}}$ componentwise by

$$\tilde{s}(x) \triangleq \big((\Phi r)(x) - (T\Phi r)(x)\big)^+.$$

Observe that $(r, \tilde{s})$ is feasible for (14). Furthermore,

$$\pi_{\mu^*,\nu}^\top \tilde{s} \leq (\pi_{\mu^*,\nu}^\top \psi)\|\tilde{s}\|_{\infty,\mathbf{1}/\psi} \leq (\pi_{\mu^*,\nu}^\top \psi)\|T\Phi r - \Phi r\|_{\infty,\mathbf{1}/\psi}.$$

Finally, note that

$$\nu^\top (J^* - \Phi r) \leq (\nu^\top \psi)\|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi}.$$

Now, suppose that $(r_{\text{SALP}}, \bar{s})$ is an optimal solution to the SALP (14). We have from the inequalities in (12) in the proof of Theorem 1 and the above observations,

$$\begin{aligned}
\|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} &\leq \nu^\top (J^* - \Phi r_{\text{SALP}}) + \frac{2\pi_{\mu^*,\nu}^\top \bar{s}}{1-\alpha} \\
&\leq \nu^\top (J^* - \Phi r) + \frac{2\pi_{\mu^*,\nu}^\top \tilde{s}}{1-\alpha} \\
&\leq (\nu^\top \psi)\|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} + \|T\Phi r - \Phi r\|_{\infty,1/\psi} \frac{2\pi_{\mu^*,\nu}^\top \psi}{1-\alpha}.
\end{aligned} \tag{17}$$

Since our choice of r and ψ were arbitrary, we have:

$$\|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} \leq \inf_{r,\psi\in\Psi} \; (\nu^\top \psi)\|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} + \|T\Phi r - \Phi r\|_{\infty,1/\psi} \frac{2\pi_{\mu^*,\nu}^\top \psi}{1-\alpha}. \tag{18}$$

We would like to relate the Bellman error term $T\Phi r - \Phi r$ on the right hand side of (18) to the approximation error $J^* - \Phi r$. In order to do so, first note that, for any vectors $F_1, F_2 \in \mathbb{R}^{\mathcal{A}}$ with $a_1 \in \operatorname{argmin}_a \; F_1(a)$ and $a_2 \in \operatorname{argmin}_a \; F_2(a)$,

$$\min_a \; F_1(a) - \min_a \; F_2(a) = F_1(a_1) - F_2(a_2) \leq F_1(a_2) - F_2(a_2) \leq \max_a \; |F_1(a) - F_2(a)|\,.$$

By swapping the roles of F_1 and F_2, it is easy to see that

$$\left|\min_a \; F_1(a) - \min_a \; F_2(a)\right| \leq \max_a \; |F_1(a) - F_2(a)|\,.$$

Examining the definition of the Bellman operator T, this implies that, for any vectors $J, \bar{J} \in \mathbb{R}^{\mathcal{X}}$ and any $x \in \mathcal{X}$,

$$|TJ(x) - T\bar{J}(x)| \leq \alpha \max_a \sum_{x' \in \mathcal{X}} P_a(x, x')|J(x') - \bar{J}(x')|.$$

Therefore,

$$\begin{aligned} \|T\Phi r - J^*\|_{\infty, \mathbf{1}/\psi} &\leq \alpha \max_{x,a} \frac{\sum_{x'} P_a(x,x')|\Phi r(x') - J^*(x')|}{\psi(x)} \\ &\leq \alpha \max_{x,a} \frac{\sum_{x'} P_a(x,x')\psi(x')\frac{|\Phi r(x') - J^*(x')|}{\psi(x')}}{\psi(x)} \\ &\leq \alpha\beta(\psi)\|J^* - \Phi r\|_{\infty,1/\psi}. \end{aligned}$$

Thus,

$$\begin{aligned} \|T\Phi r - \Phi r\|_{\infty, \mathbf{1}/\psi} &\leq \|T\Phi r - J^*\|_{\infty,\mathbf{1}/\psi} + \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi} \\ &\leq \|J^* - \Phi r\|_{\infty,1/\psi}(1 + \alpha\beta(\psi)). \end{aligned} \tag{19}$$

Combining (18) and (19), we get the desired result. ■

4.4. Approximation Guarantee: A Queueing Example

In this section, we will examine the strength of the approximation guarantee we have provided for the SALP (Theorem 2) in a simple, concrete model studied in the context of the ALP by de Farias and Van Roy (2003). In particular, we consider an autonomous queue whose queue-length dynamics evolve over the state space $\mathcal{X} \triangleq \{0, 1, \ldots, N-1\}$ according to

$$x_{t+1} = \begin{cases} \max(x_t - 1, 0) & \text{w.p. } 1-p, \\ \min(x_t + 1, N-1) & \text{w.p. } p. \end{cases}$$

Here, we assume that $p \in (0, 1/2)$ and $N \geq 1$ is the buffer size. For convenience so as to avoid integrality issues, we will assume that $N - 1$ is a multiple of 4. For $0 < x < N-1$, define the cost function $g(x) \triangleq x^2$. As in de Farias and Van Roy (2003), we may and will select $g(0)$ and $g(N-1)$ so that $J^*(x) = \rho_2 x^2 + \rho_1 x + \rho_0$ for constants $\rho_2 > 0$, ρ_1, and $\rho_0 > 0$ that depend only on p and the discount factor α. We take ν to be the steady-state distribution over states of the resulting birth-death chain, i.e., for all $x \in \mathcal{X}$,

$$\nu(x) = \frac{1-q}{1-q^N} q^x, \quad \text{where} \quad q \triangleq \frac{p}{1-p}.$$

Note that since this system is uncontrolled, we have $\pi_{\mu^*,\nu} = \nu$.

Assume we have a constant basis function and a linear basis function, i.e., $\phi_1(x) \triangleq 1$ and $\phi_2(x) \triangleq x$, for $x \in \mathcal{X}$. Note that this is different than the example studied by de Farias and

Van Roy (2003), which assumed basis functions $\phi_1(x) \triangleq 1$ and $\phi_2(x) \triangleq x^2$. Nonetheless, the best possible approximation to J^* within this architecture continues to have an approximation error that is uniformly bounded in N. In particular, we have that

$$\begin{aligned}\inf_r \ \|J^* - \Phi r\|_{1,\nu} \leq \|J^* - (\rho_0\phi_1 + \rho_1\phi_2)\|_{1,\nu} &= \rho_2 \frac{1-q}{1-q^N} \sum_{x=0}^{N-1} q^x x^2 \\ &\leq \rho_2 \sum_{x=0}^{\infty} q^x x^2 = \frac{\rho_2 q}{(1-q)^3}.\end{aligned}$$

We make two principal claims for this problem setting:

(a) Theorem 2 in fact shows that the SALP is guaranteed to find an approximation in the span of the basis functions with an approximation error that is also uniformly bounded in N.

(b) We will see that the corresponding guarantee, (15), for the ALP (de Farias and Van Roy, 2003, Theorem 4.2) can guarantee at best an approximation error that scales linearly in N.

The broad idea used in establishing the above claims is as follows: For (a), we utilize a (quadratic) Lyapunov function identified by de Farias and Van Roy (2003) for the very problem here to produce an upper bound on the approximation guarantee we have developed for the SALP; we are careful to exclude this Lyapunov function from our basis. We then consider the ALP with the same basis, and absent the ability to utilize the quadratic Lyapunov function alluded to, show that the bound in de Farias and Van Roy (2003) must scale at least linearly in N. We now present the details.

The broad idea used in establishing the above claims is as follows: For (a), we utilize a (quadratic) Lyapunov function identified by de Farias and Van Roy (2003) for the very problem here to produce an upper bound on the approximation guarantee we have developed for the SALP; we are careful to exclude this Lyapunov function from our basis. We then consider the ALP with the same basis. We show that the bound in de Farias and Van Roy (2003), without the ability to utilize the quadratic Lyapunov function alluded to, must scale at least linearly in N. This establishes (b). We now present the details.

First, consider claim (a). To see the first claim, we consider the weighting function $\psi(x) \triangleq x^2 + 2/(1-\alpha)$, for $x \in \mathcal{X}$. Notice that this weighting function is *not* in the span of Φ but still permissible for the bound in Theorem 2. For this choice of ψ, we have

$$\inf_r \ \|J^* - \Phi r\|_{\infty,1/\psi} \leq \max_{0 \leq x < N} \ \frac{\rho_2 x^2}{x^2 + 2/(1-\alpha)} \leq \rho_2. \tag{20}$$

Moreover, de Farias and Van Roy (2003) show that for this choice of ψ,

$$\beta(\psi) \leq \frac{1+\alpha}{2\alpha}, \qquad \nu^\top \psi \leq \frac{1-p}{1-2p}\left(\frac{2}{1-\alpha} + 2\frac{p^2}{(1-2p)^2} + \frac{p}{1-2p}\right). \tag{21}$$

Combining (20)–(21), Theorem 2, and, in particular, (16), yields the (uniform in N) upper bound

$$\|J^* - \Phi r_{\text{SALP}}\|_{1,\nu} \leq \frac{5\rho_2(1-p)}{(1-\alpha)(1-2p)} \left(\frac{2}{1-\alpha} + 2\frac{p^2}{(1-2p)^2} + \frac{p}{1-2p} \right).$$

The analysis of de Farias and Van Roy (2003) applies identically to the more complex settings considered in that work (namely the controlled queue and queueing network considered there) to yield uniform approximation guarantees for SALP approximations.

The following lemma, whose proof may be found in Appendix A, demonstrates that the right-hand side of (15) must increase linearly with N, establishing (b). The proof of the lemma reveals that this behavior is driven primarily by the fact that the basis does not span an appropriate weighting function ψ.

Lemma 3. *For the autonomous queue with basis functions $\phi_1(x) \triangleq 1$ and $\phi_2(x) \triangleq x$, if N is sufficiently large, then*

$$\inf_{r,\psi \in \bar{\Psi}} \frac{2\nu^\top \psi}{1 - \alpha\beta(\psi)} \|J^* - \Phi r\|_{\infty,1/\psi} \geq \frac{3\rho_2 q}{32(1-q)}(N-1).$$

4.5. A Performance Bound

The analytical results provided in Sections 4.2 and 4.3 provide bounds on the quality of the approximation provided by the SALP solution to J^*. In this section, we derive performance bounds with the intent of understanding the increase in expected cost incurred in using a control policy that is greedy with respect to the SALP approximation in lieu of the optimal policy. In particular, we will momentarily present a result that will allow us to interpret the objective of the SALP (14) as an upper bound on the performance loss of a greedy policy with respect to the SALP solution.

To begin, we briefly introduce some relevant notation. For a given policy μ, we denote

$$\Delta_\mu \triangleq \sum_{k=0}^{\infty} (\alpha P_\mu)^k = (I - \alpha P_\mu)^{-1}.$$

Thus, $\Delta^* = \Delta_{\mu^*}$. Given a vector $J \in \mathbb{R}^{\mathcal{X}}$, let μ_J denote the greedy policy with respect to J. That is, μ_J satisfies $T_{\mu_J} J = TJ$. Recall that the policy of interest to us will be $\mu_{\Phi r_{\text{SALP}}}$ for a solution r_{SALP} to the SALP. Finally, for an arbitrary starting distribution over states η, we define $\nu(\eta, J)$ to be the 'discounted' expected frequency of visits to each state under the policy μ_J, i.e.,

$$\nu(\eta, J)^\top \triangleq (1-\alpha)\eta^\top \sum_{k=0}^{\infty} (\alpha P_{\mu_J})^k = (1-\alpha)\eta^\top \Delta_{\mu_J}.$$

We have the following upper bound on the increase in cost incurred by using μ_J in place of μ^*:

Theorem 3.

$$\|J_{\mu_J} - J^*\|_{1,\eta} \leq \frac{1}{1-\alpha}\left(\nu(\eta, J)^\top (J^* - J) + \frac{2}{1-\alpha}\pi_{\mu^*,\nu(\eta,J)}^\top (J - TJ)^+\right).$$

Theorem 3 applies to general approximations J and is not specific to approximations produced by the SALP. Theorem 3 indicates that if J is close to J^*, so that $(J - TJ)^+$ is also small, then the expected cost incurred in using a control policy that is greedy with respect to J will be close to optimal. The bound indicates the impact of approximation errors over differing parts of the state space on performance loss.

Suppose that $(r_{\text{SALP}}, \bar{s})$ is an optimal solution to the SALP (14). Then, examining the proof of Theorem 2 and, in particular, (17), reveals that

$$\begin{aligned} &\nu^\top (J^* - \Phi r_{\text{SALP}}) + \frac{2}{1-\alpha}\pi_{\mu^*,\nu}^\top \bar{s} \\ &\leq \inf_{r,\psi \in \Psi} \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi}\left(\nu^\top \psi + \frac{2(\pi_{\mu^*,\nu}^\top \psi)(\alpha\beta(\psi)+1)}{1-\alpha}\right). \end{aligned} \tag{22}$$

Assume that the state relevance weights ν in the SALP (14) satisfy

$$\nu = \nu(\eta, \Phi r_{\text{SALP}}). \tag{23}$$

Then, combining Theorem 3 and (22) yields

$$\|J_{\mu_{\Phi r_{\text{SALP}}}} - J^*\|_{1,\eta} \leq \frac{1}{1-\alpha}\left(\inf_{r,\psi \in \Psi} \|J^* - \Phi r\|_{\infty,\mathbf{1}/\psi}\left(\nu^\top \psi + \frac{2(\pi_{\mu^*,\nu}^\top \psi)(\alpha\beta(\psi)+1)}{1-\alpha}\right)\right). \tag{24}$$

This bound *directly* relates the performance loss of the SALP policy to the ability of the basis function architecture Φ to approximate J^*. Moreover, assuming (23), we can interpret the SALP as minimizing the upper bound on performance loss given by Theorem 3.

Unfortunately, it is not clear how to make an a-priori choice of the state relevance weights ν to satisfy (23), since the choice of ν determines the solution to the SALP r_{SALP}; this is essentially the situation one faces in performance analyses for approximate dynamic programming algorithms such as approximate value iteration and temporal difference learning (de Farias and Van Roy, 2000). Indeed, it is not clear that there exists a ν that solves the fixed point equation (23). On the other hand, given a choice of ν so that $\nu \approx \nu(\eta, \Phi r_{\text{SALP}})$, in the sense of a bounded Radon-Nikodym derivative between the two distributions, then the performance bound (24) will hold, inflated by the quantity

$$\max_{x \in \mathcal{X}} \frac{\nu(x)}{\nu(\eta, \Phi r_{\text{SALP}})(x)}.$$

As suggested by de Farias and Van Roy (2003) in the ALP case, one possibility for finding such a choice of state relevance weights is to iteratively re-solve the SALP, and at each time using the policy from the prior iteration to generate state relevance weights for the next iteration.

Proof of Theorem 3. Define $s \triangleq (J - TJ)^+$. From Lemma 2, we know that

$$J \leq J^* + \Delta^* s.$$

Using the fact that the operator T_{μ^*} is monotonic, we can apply T_{μ^*} to both sides to obtain

$$\begin{aligned} T_{\mu^*} J &\leq T_{\mu^*}(J^* + \Delta^* s) = g_{\mu^*} + \alpha P_{\mu^*}(J^* + \Delta^* s) = J^* + \alpha P_{\mu^*} \Delta^* s \\ &= J^* + \alpha P_{\mu^*}(I - \alpha P_{\mu^*})^{-1} s = J^* + \Delta^* s - s \leq J^* + \Delta^* s, \end{aligned}$$

so that

$$TJ \leq T_{\mu^*} J \leq J^* + \Delta^* s. \tag{25}$$

Then,

$$\begin{aligned} \eta^\top (J_{\mu_J} - J) &= \eta^\top \sum_{k=0}^{\infty} \alpha^k P_{\mu_J}^k (g_{\mu_J} + \alpha P_{\mu_J} J - J) \\ &= \eta^\top \Delta_{\mu_J}(TJ - J) \\ &\leq \eta^\top \Delta_{\mu_J}(J^* - J + \Delta^* s) \\ &= \frac{1}{1-\alpha} \nu(\eta, J)^\top (J^* - J + \Delta^* s). \end{aligned} \tag{26}$$

where the second equality is from the fact that $g_{\mu_J} + \alpha P_{\mu_J} J = T_{\mu_J} J = TJ$, and the inequality follows from (25).

Further,

$$\begin{aligned} \eta^\top (J - J^*) &\leq \eta^\top \Delta^* s \\ &\leq \eta^\top \Delta_{\mu_J} \Delta^* s \\ &= \frac{1}{1-\alpha} \nu(\eta, J)^\top \Delta^* s. \end{aligned} \tag{27}$$

where the second inequality follows from the fact that $\Delta^* s \geq \mathbf{0}$ and $\Delta_{\mu_J} = I + \sum_{k=1}^{\infty} \alpha^k P_{\mu_J}^k$.

It follows from (26) and (27) that

$$\begin{aligned} \eta^\top (J_{\mu_J} - J^*) &= \eta^\top (J_{\mu_J} - J) + \eta^\top (J - J^*) \\ &\leq \frac{1}{1-\alpha} \nu(\eta, J)^\top (J^* - J + 2\Delta^* s) \\ &= \frac{1}{1-\alpha} \left(\nu(\eta, J)^\top (J^* - J) + \frac{2}{1-\alpha} \pi_{\mu^*, \nu(\eta, J)}^\top s \right), \end{aligned}$$

which is the result. ■

4.6. Sample Complexity

Our analysis thus far has assumed we have the ability to solve the SALP. The number of constraints and variables in the SALP grows linearly with the size of the state space $\mathcal{X}$. Hence, this program will typically be intractable for problems of interest. One solution, which we describe here, is to consider a *sampled* variation of the SALP, where states and constraints are sampled rather than exhaustively considered. In this section, we will argue that the solution to the SALP is well approximated by the solution to a tractable, sampled variation.

In particular, let $\hat{\mathcal{X}}$ be a collection of S states drawn independently from the state space $\mathcal{X}$ according to the distribution $\pi_{\mu^*,\nu}$. Consider the following optimization program:

$$\begin{aligned} &\underset{r,s}{\text{maximize}} && \nu^\top \Phi r - \frac{2}{(1-\alpha)S} \sum_{x \in \hat{\mathcal{X}}} s(x) \\ &\text{subject to} && \Phi r(x) \leq T\Phi r(x) + s(x), \qquad \forall\, x \in \hat{\mathcal{X}}, \\ &&& s \geq \mathbf{0}, \quad r \in \mathcal{N}. \end{aligned} \tag{28}$$

Here, $\mathcal{N} \subset \mathbb{R}^K$ is a bounding set that restricts the magnitude of the sampled SALP solution, we will discuss the role of $\mathcal{N}$ shortly. Notice that (28) is a variation of (14), where only the decision variables and constraints corresponding to the sampled subset of states are retained. The resulting optimization program has $K + S$ decision variables and $S|\mathcal{A}|$ linear constraints. For a moderate number of samples S, this is easily solved. Even in scenarios where the size of the action space $\mathcal{A}$ is large, it is frequently possible to rewrite (28) as a compact linear program (Farias and Van Roy, 2007; Moallemi et al., 2008). The natural question, however, is whether the solution to the sampled SALP (28) is a good approximation to the solution provided by the SALP (14), for a 'tractable' number of samples S.

Here, we answer this question in the affirmative. 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 nearly satisfies, with high probability, the approximation guarantee presented for the SALP solution in Theorem 2.

In order to establish a sample complexity result, we require control over the magnitude of optimal solutions to the SALP (14). This control is provided by the bounding set $\mathcal{N}$. In particular, we will assume that $\mathcal{N}$ is large enough so that it contains an optimal solution to the SALP (14), and we define the constant

$$B \triangleq \sup_{r \in \mathcal{N}} \|(\Phi r - T\Phi r)^+\|_\infty. \tag{29}$$

This quantity is closely related to the diameter of the region $\mathcal{N}$. Our main sample complexity result can then be stated as follows:

Theorem 4. *Under the conditions of Theorem 2, let r_{SALP} be an optimal solution to the SALP* (14)*, and let $\hat{r}_{SALP}$ be an optimal solution to the sampled SALP* (28)*. 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 $1 - \delta - 2^{-383}\delta^{128}$,

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

The proof of Theorem 4 is based on bounding the pseudo-dimension of a certain class of functions, and is provided in Appendix B.

Theorem 4 establishes that the sampled SALP (28) provides a close approximation to the solution of the SALP (14), in the sense that the approximation guarantee we established for the SALP in Theorem 2 is approximately valid for the solution to the sampled SALP, with high probability. The theorem precisely specifies the number of samples required to accomplish this task. This number depends linearly on the number of basis functions and the diameter of the 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 (3). Recall that the ALP has a large number of constraints but a *small* number of variables;[3] 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 (2004) establish a sample complexity bound for a sampled version of the ALP analogous to the sampled SALP (28). 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.: the constant B and the number of basis functions K. The sample complexity in the ALP case is identical to the sample complexity bound established here, up to constants and a linear dependence on the ratio B/ϵ. This is as opposed to the quadratic dependence on B/ϵ of the sampled SALP. Although the two sample complexity bounds are within polynomial terms of each other, one may rightfully worry abut the practical implications of an additional factor of B/ϵ in the required number of samples. In the numerical study of Section 6, we will attempt to address this concern computationally.

Finally, note that the sampled SALP has $K + S$ variables and $S|\mathcal{A}|$ linear constraints whereas the sampled ALP has merely K variables and $S|\mathcal{A}|$ linear constraints. Nonetheless, we will show in the Section 5.1 that the special structure of the Hessian associated with the sampled SALP affords us a linear computational complexity dependence on S when applying interior point methods.

[3] Since the ALP has a small number of variables, it may be possible to solve exactly the ALP without resorting to constraint sampling by using a cutting-plane method or by applying column generation to the dual problem. In general, this would require some form of problem-specific analysis. The SALP, on the other hand, has many variables and constraints, and thus some form of sampling seems necessary.

An alternative sample complexity bound of a similar flavor can be developed using results from the stochastic programming literature. The key idea is that the SALP (14) can be reformulated as the following convex stochastic programming problem:

$$\underset{r\in\mathcal{N}}{\text{maximize}}\ \mathsf{E}_{\nu,\pi_{\mu^*,\nu}}\left[\Phi r(x_0) - \frac{2}{1-\alpha}(\Phi r(x) - T\Phi r(x))^+\right], \tag{30}$$

where $x_0, x \in \mathcal{X}$ have distributions ν and $\pi_{\mu^*,\nu}$, respectively. Interpreting the sampled SALP (28) as a sample average approximation of (30), a sample complexity bound can be developed using the methodology of Shapiro et al. (2009, Chap. 5), for example. This proof is simpler than the one presented here, but yields a cruder estimate that is not as easily compared with those available for the ALP.

5. Practical Implementation

The analysis in Section 4 applies to certain 'idealized' SALP variants, as discussed in Section 4.1. In particular, our main approximation guarantees focused on the linear program (14), and the 'sampled' version on that program (28). (14) is equivalent to the SALP (5) for a specialized choice of the violation budget θ and an idealized choice of the distribution π, namely $\pi_{\mu^*,\nu}$. As such (14) is not implementable: $\pi_{\mu^*,\nu}$ is not available and the number of constraints and variables scales linearly with the size of $\mathcal{X}$ which will typically be prohibitively large for interesting problems. The sampled variant of that program, (28), requires access to the same idealized sampling distribution and the guarantees pertaining to that program require knowledge of a bounding set for the optimal solution to (14), $\mathcal{N}$. As such, this program is not directly implementable either. Finally, the specialized choice of θ implicit in both (14) and (28) may not yield the best policies.

Thus, the bounds in Section 4 do not apply directly in the practical settings we will consider. Nonetheless, they do provide some insights that allow us to codify a recipe for a practical and implementable variation that we discuss below.

Consider the following algorithm:

1. Sample S states independently from the state space $\mathcal{X}$ according to a sampling distribution ρ. Denote the set of sampled states by $\hat{\mathcal{X}}$.

2. Perform a line search over increasing choices of $\theta \geq 0$. For each choice of θ,

(a) Solve the *sampled* SALP:

$$
\begin{aligned}
\text{(31)} \qquad & \underset{r,s}{\text{maximize}} && \frac{1}{S} \sum_{x \in \hat{\mathcal{X}}} (\Phi r)(x) \\
& \text{subject to} && \Phi r(x) \leq T\Phi r(x) + s(x), \quad \forall\, x \in \hat{\mathcal{X}}, \\
& && \frac{1}{S} \sum_{x \in \hat{\mathcal{X}}} s(x) \leq \theta, \\
& && s \geq \mathbf{0}.
\end{aligned}
$$

(b) Evaluate the performance of the policy resulting from (31) via Monte Carlo simulation.

3. Select the best of the evaluated policies over different choices of θ.

Note that our algorithm does not require the specific choice of the violation budget θ implicit in the program (14), since we optimize with a line search so as to guarantee the *best* possible choice of θ. Note that, in such a line search, the sampled SALP (31) can be efficiently re-solved for increasing values of θ via a 'warm-start' procedure. Here, the optimal solution of the sampled SALP given previous value of θ is used as a starting point for the solver in a subsequent round of optimization. Using this method we observe that, in practice, the total solution time for a series of sampled SALP instances that vary by their values of θ grows sub-linearly with the number of instances. However, the policy for each solution instance must be evaluated via Monte Carlo simulation, which may be a time-consuming task.

Barring a line search, however, note that a reasonable choice of θ is implicitly suggested by the SALP (14) considered in Section 4.3. Thus, alternatively, the line search in Steps 2 and 3 can be replaced with the solution of single LP as follows:

2'. Solve the *sampled* SALP:

$$
\begin{aligned}
\text{(32)} \qquad & \underset{r,s}{\text{maximize}} && \frac{1}{S} \sum_{x \in \hat{\mathcal{X}}} (\Phi r)(x) - \frac{2}{(1-\alpha)S} \sum_{x \in \hat{\mathcal{X}}} s(x) \\
& \text{subject to} && \Phi r(x) \leq T\Phi r(x) + s(x), \qquad \forall\, x \in \hat{\mathcal{X}}, \\
& && s \geq \mathbf{0}.
\end{aligned}
$$

Note that the sampled SALP (32) is equivalent to (31) for a specific, implicitly determined choice of θ (cf. Lemma 4 in Appendix A).

The programs (31) and (32) do not employ a specialized choice of π, and the use of the bounding set $\mathcal{N}$ is omitted. In addition, (31) does not require the specific choice of violation budget θ implicit in (14) and (28). As such, our main approximation guarantees do not apply to these programs.

Our algorithm takes as inputs the following parameters:

- Φ, a collection of K basis functions.

- S, the number of states to sample. By sampling S states, we limit the number of variables and constraints in the sampled SALP (31). Thus, by keeping S small, the sampled SALP becomes tractable to solve numerically. On the other hand, the quality of the approximation provided by the sampled SALP may suffer if S is chosen to be too small. The sample complexity theory developed in Section 4.6 suggests that S can be chosen to grow linearly with K, the size of the basis set. In particular, a reasonable choice of S need not depend on the size of the underlying state space.

  In practice, we choose $S \gg K$ to be as large as possible subject to limits on the CPU time and memory required to solve (31). In Section 5.1, we will discuss how the program (31) can be solved efficiently via barrier methods for large choices of S. Finally, note that a larger sample size can be used in the evaluation of the objective of the sampled SALP (31) than in the construction of constraints. In other words, the objective in (31) can be constructed from a set of states distinct from $\hat{\mathcal{X}}$, since this does not increase the size of the LP.

- ρ, a sampling distribution on the state space $\mathcal{X}$. The distribution ρ is used, via Monte Carlo sampling, in place of both the distributions ν and π in the SALP (5).

  Recall that the bounds in Theorems 1 and 2 provide approximation guarantees in a ν-weighted 1-norm. This suggests that ν should be chosen to emphasize regions of the state space where the quality of approximation is most important. The important regions could be, for example, regions of the state space where the process spends the most time under a baseline policy, and they could emphasized by setting ν to be the stationary distribution induced by the baseline policy. Similarly, the theory in Section 4 also suggests that the distribution π should be chosen to be the discounted expected frequency of visits to each state given an initial distribution ν under the *optimal* policy. Such a choice of distribution is clearly impossible to compute. In its place, however, if ν is the stationary distribution under a baseline policy, it seems reasonable to use the same distribution for π.

  In practice, we choose ρ to be the stationary distribution under some baseline policy. States are then sampled from ρ via Monte Carlo simulation of the baseline policy. This baseline policy can correspond, for example, to a heuristic control policy for the system. More sophisticated procedures such as 'bootstrapping' can also be considered (Farias and Van Roy, 2006). Here, one starts with a heuristic policy to be used for sampling states. Given the sampled states, the application of our algorithm will result in a new control policy. The new control policy can then be used for state sampling in a subsequent round of optimization, and the process can be repeated.

5.1. Efficient Linear Programming Solution

In this section, we will discuss the efficient solution of the sampled SALP (31) via linear programming. Note that the discussion here applies to the variant (32) as well. To begin, note that (31) can be written explicitly in the form of a linear program as

$$\begin{aligned} &\underset{r,s}{\text{maximize}} && c^\top r \\ &\text{subject to} && \begin{bmatrix} A_{11} & A_{12} \\ 0 & d^\top \end{bmatrix} \begin{bmatrix} r \\ s \end{bmatrix} \leq b, \\ &&& s \geq \mathbf{0}. \end{aligned} \tag{33}$$

Here, $b \in \mathbb{R}^{S|\mathcal{A}|+1}$, $c \in \mathbb{R}^K$, and $d \in \mathbb{R}^S$ are vectors, $A_{11} \in \mathbb{R}^{S|\mathcal{A}|\times K}$ is a dense matrix, and $A_{12} \in \mathbb{R}^{S|\mathcal{A}|\times S}$ is a sparse matrix. This LP has $K + S$ decision variables and $S|\mathcal{A}| + 1$ linear constraints.

Typically, the number of sampled states S will be quite large. For example, in Section 6, we will discuss an example where $K = 22$ and $S = 300{,}000$. The resulting LP has approximately 300,000 variables and 6,600,000 constraints. In such cases, with many variables *and* many constraints, one might expect the LP to be difficult to solve. However, the sparsity structure of the constraint matrix in (33) and, especially, that of the sub-matrix A_{12}, allows efficient optimization of this LP.

In particular, imagine solving the LP (33) with a barrier method. The computational bottleneck of such a method is the inner Newton step to compute a central point (see, for example, Boyd and Vandenberghe, 2004). This step involves the solution of a system of linear equations of the form

$$H \begin{bmatrix} \Delta r \\ \Delta s \end{bmatrix} = -g. \tag{34}$$

Here, $g \in \mathbb{R}^{K+S}$ is a vector and $H \in \mathbb{R}^{(K+S)\times(K+S)}$ is the Hessian matrix of the barrier function. Without exploiting the structure of the matrix H, this linear system can be solved with $O((K+S)^3)$ floating point operations. For large values of S, this may be prohibitive.

Fortunately, the Hessian matrix H can be decomposed according to the block structure

$$H \triangleq \begin{bmatrix} H_{11} & H_{12} \\ H_{12}^\top & H_{22} \end{bmatrix},$$

where $H_{11} \in \mathbb{R}^{K\times K}$, $H_{12} \in \mathbb{R}^{K\times S}$, and $H_{22} \in \mathbb{R}^{S\times S}$. In the case of the LP (33), it is not difficult to see that the sparsity structure of the sub-matrix A_{12} ensures that the sub-matrix H_{22} takes the form of a diagonal matrix plus a rank-one matrix. This allows the linear system (34) to be solved with $O(K^2 S + K^3)$ floating point operations. This is linear in S, the number of sampled states. Note that this is the same computational complexity as that of an inner Newton step for the ALP, despite the fact that the SALP has more variables than the ALP. This is because the added slack variables in the SALP are 'local' and effectively do not contribute to the dimension of the problem.

6. Case Study: Tetris

Tetris is a popular video game designed and developed by Alexey Pazhitnov in 1985. The Tetris board, illustrated in Figure 2, consists of a two-dimensional grid of 20 rows and 10 columns. The game starts with an empty grid and pieces fall randomly one after another. Each piece consists of four blocks and the player can rotate and translate it in the plane before it touches the 'floor'. The pieces come in seven different shapes and the next piece to fall is chosen from among these with equal probability. Whenever the pieces are placed such that there is an entire horizontal row or line of contiguous blocks formed, a point is earned and the line gets cleared. Once the board has enough blocks such that the incoming piece cannot be placed for all translations and rotations, the game terminates. Hence the goal of the player is to clear maximum number of lines before the board gets full.

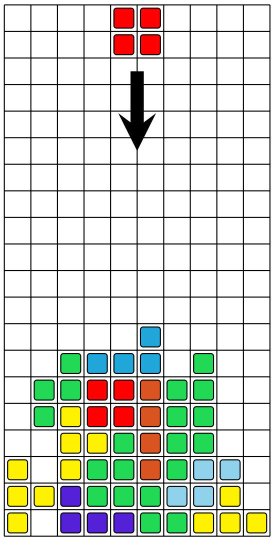


Figure 2: Example of a Tetris board configuration.

Our interest in Tetris as a case study for the SALP algorithm is motivated by several facts. First, theoretical results suggest that design of an optimal Tetris player is a difficult problem. Brzustowski (1992) and Burgiel (1997) have shown that the game of Tetris has to end with probability one, under all policies. They demonstrate a sequence of pieces, which leads to termination state of game for all possible actions. Demaine et al. (2003) consider the offline version of Tetris and provide computational complexity results for 'optimally' playing Tetris. They show that when the sequence of pieces is known beforehand it is NP-complete to maximize the number of cleared lines, minimize the maximum height of an occupied square, or maximize the number of pieces placed before the game ends. This suggests that the online version should be computationally difficult.

Second, 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. One of the important steps in applying these techniques is the choice of basis functions. Fortunately, there is a *fixed set of basis functions*, to be described shortly, which have been used by researchers while applying temporal-difference learning (Bertsekas and Ioffe, 1996; Bertsekas and Tsitsiklis, 1996), policy gradient methods (Kakade, 2002), and approximate linear programming (Farias and Van Roy, 2006). Hence, application of SALP to Tetris allows us to make a clear comparison to other ADP methods.

The SALP methodology described in Section 5 was applied as follows:

- **MDP formulation.** We used the formulation of Tetris as a Markov decision problem of Farias and Van Roy (2006). Here, the 'state' at a particular time encodes the current board configuration and the shape of the next falling piece, while the 'action' determines the placement of the falling piece. Thus, given a state and an action, the subsequent state is determined by the new configuration of the board following placement, and the shape of a new falling piece that is selected uniformly at random.

- **Reward structure.** The objective of Tetris is to maximize reward, where, given a state and an action, the per stage reward is defined to be the number of rows that are cleared following the placement of the falling piece.

  Note that since every game of Tetris must ultimately end, Tetris is most naturally formulated with the objective of maximizing the expected total number of rows cleared, i.e., a maximum *total* reward formulation. Indeed, in the existing literature, performance is reported in terms of total reward. In order to accommodate the SALP setting, we will apply our methodology to a maximum *discounted* reward formulation with a discount factor[4] of $\alpha = 0.9$. When evaluating the performance of resulting policies, however, we will report both total reward (in order to allow comparison with the literature) and discounted reward (to be consistent with the SALP objective).

- **Basis functions.** We employed the 22 basis functions originally introduced by Bertsekas and Ioffe (1996). Each basis function takes a Tetris board configuration as its argument. The functions are as follows:

  - Ten basis functions, $\phi_0, \ldots, \phi_9$, mapping the state to the height h_k of each of the ten columns.
  - Nine basis functions, $\phi_{10}, \ldots, \phi_{18}$, each mapping the state to the absolute difference between heights of successive columns: $|h_{k+1} - h_k|$, $k = 1, \ldots, 9$.
  - One basis function, ϕ_{19}, that maps state to the maximum column height: $\max_k h_k$
  - One basis function, ϕ_{20}, that maps state to the number of 'holes' in the board.

[4] The introduction of an artificial discount factor into an average cost problem is akin to analyzing a perturbed problem with a limited time horizon, a common feature in many ADP schemes (e.g., de Farias and Van Roy, 2006).

  - One basis function, ϕ_{21}, that is equal to 1 in every state.

- **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 i.i.d. fashion from the stationary distribution of a (rather poor) baseline policy.[5] For each choice of sample size S, ten different collections of S samples were generated.

- **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 (31) was solved. Separately, the optimization program (32), which implicitly defines a reasonable choice of θ, was also employed. The CPLEX 11.0.0 optimization package was used to solve the resulting linear programs.

- **Policy evaluation.** Given a vector of weights $\hat{r}$, the performance of the corresponding policy was evaluated using Monte Carlo simulation. We estimate the expected reward of the policy $\mu_{\hat{r}}$ over a series of 3,000 games. The sequence of pieces in each of the 3,000 games was fixed across the evaluation of different policies in order to reduce the Monte Carlo error in estimated performance differences.

  Performance is measured in two ways, starting from empty board configuration. The total reward is computed, as the expected total number of lines eliminated in a single game and the discounted reward is computed, as the expected discounted number of lines eliminated in a single game

For each pair (S, θ), the resulting *average* performance (averaged over each of the 10 policies arising from the different sets of sampled states) in terms of expected total lines cleared is shown in Figure 3. Note that the $\theta = 0$ curve in Figure 3 corresponds to the original ALP algorithm. Figure 3 provides experimental evidence for the intuition expressed in Section 3 and the analytic result of Theorem 1: Relaxing the constraints of the ALP even slightly, by allowing for a small slack budget, allows for better policy performance. As the slack budget θ is increased from 0, performance dramatically improves. At the peak value of $\theta = 0.0205$, the SALP generates policies with performance that is an order of magnitude better than ALP. Beyond this value, the performance of the SALP begins to degrade, as shown by the $\theta = 0.041$ curve. Hence, we did not explore larger values of θ.

As suggested in Section 5, instead of doing a line search over θ, one can consider solving the sampled SALP (32), which implicitly makes a choice of θ. We denote this implicit choice by $\theta = \theta^*$ in Figure 3. The results of solving (32) are given by the $\theta = \theta^*$ curve in Figure 3. We observe that, in our experiments, the results obtained by solving (32) are quite similar to the best results obtained by doing a line search over choices of θ. In fact, across these experiments, θ^* is observed

[5] Our baseline policy had an expected total reward of 113 lines cleared.

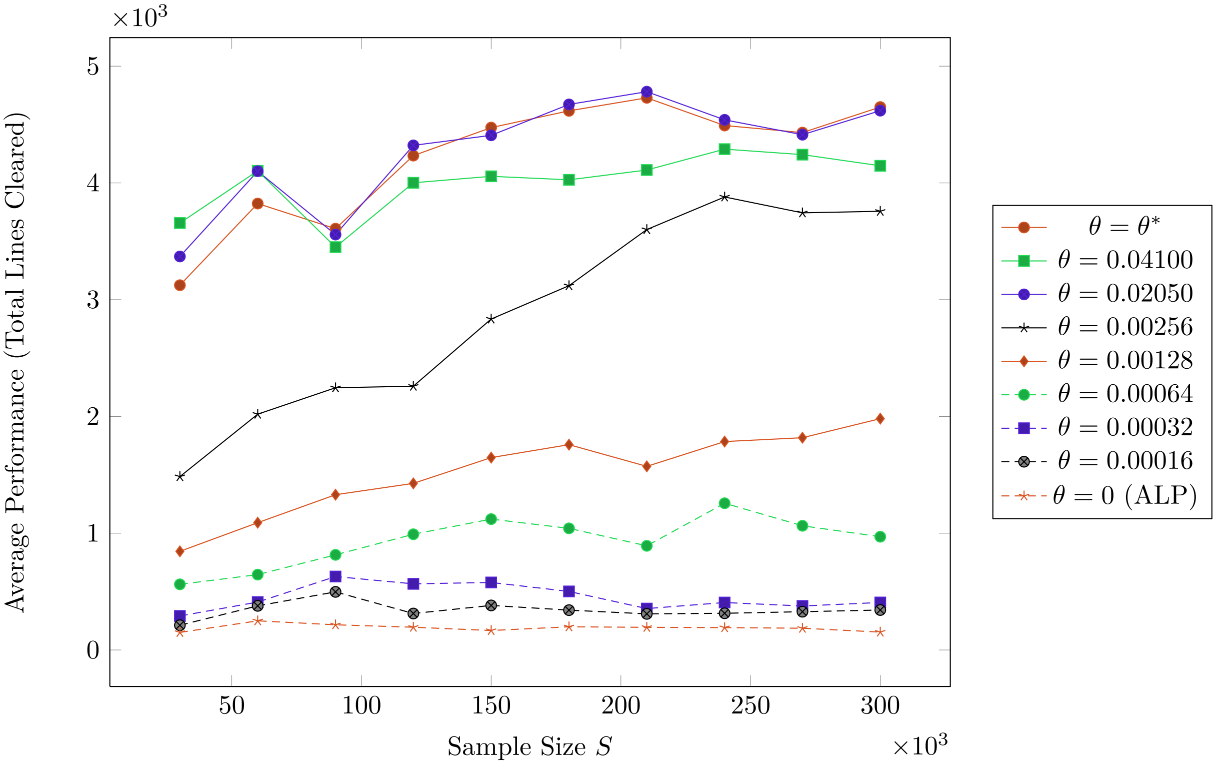


Figure 3: Expected total reward of the average SALP policy for different values of the number of sampled states S and the violation budget θ. Values for θ were chosen in an increasing fashion starting from 0, until the resulting average performance began to degrade. The $\theta = \theta^*$ curve corresponds to the implicit choice of θ made by solving (32).

to be roughly constant as a function of the sample size S, and approximately equal to 0.02. This is very close to the best values of θ found via line search.

In order to allow a comparison of our results with those reported elsewhere in the literature, Table 1 summarizes the expected total reward of the *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. Here, the reported ALP and SALP performance corresponds to that of the best performing policy among all of policies computed for Figure 3. Note that the best performance result of SALP is a factor of 2 better than the nearest competitors.

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 global optimization method, Szita and Lőrincz (2006) report finding policies with a remarkable average performance of 350,000. Their method is very 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. More generally, global optimization

Algorithm	**Best Performance** (Total Lines Cleared)	**CPU Time**
ALP	698.4	hours
TD-Learning (Bertsekas and Ioffe, 1996)	3,183	minutes
ALP with bootstrapping (Farias and Van Roy, 2006)	4,274	hours
TD-Learning (Bertsekas and Tsitsiklis, 1996)	4,471	minutes
Policy gradient (Kakade, 2002)	5,500	days
SALP	11,574	hours

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

methods typically require significant trial and error and other problem specific experimentation in order to work well.

In Table 2, we see the effect of the choice of the discount factor α on the performance of the ALP and SALP methods. Here, we show both the expected discounted reward and the expected total reward, for different values of the discount factor α and the violation budget θ. Here, the policies were constructed using $S = 200{,}000$ sampled states. We find that:

1. For all discount factors, the SALP dominates the ALP. The performance improvement of the SALP relative to the ALP increases dramatically at high discount factors.

2. The absolute performance of both schemes degrades at high discount factors. This is consistent with our approximation guarantees, which degrade as $\alpha \to 1$, as well as prior theory that has been developed for the average cost ALP (de Farias and Van Roy, 2006). However, observe that the ALP degradation is drastic (scores in single digits) while the SALP degradation relatively mild (scores remain in the thousands).

7. Case Study: A Queueing Network

In this section, we study the application of SALP and ALP to control of queueing networks. In particular, we consider a *criss-cross queueing network*, which has been considered extensively in the literature (e.g., Harrison and Wein, 1989; Kushner and Martins, 1996; Martins et al., 1996). Optimal control of a criss-cross network is a standard example of a challenging network control problem, and has eluded attempts to find an analytical solution (Kumar and Muthuraman, 2004).

The cross-cross queueing network consists of two servers and three queues connected as shown in Figure 4. There are two classes of jobs in this system. The first class of jobs takes a vertical path through the system, arriving to queue 1 according to a Poisson process of rate λ_1. The second class of jobs takes a horizontal path through the system, arriving at queue 2 according to a Poisson process of rate λ_2. Server 1 can work on jobs in either queue 1 or queue 2, with service times

Violation	**Expected Total Reward**				**Expected Discounted Reward**			
Budget	Discount Factor α				Discount Factor α			
θ	0.9	0.95	0.99	0.999	0.9	0.95	0.99	0.999
0 (ALP)	169.1	367.9	240.0	1.9	2.150	5.454	30.410	1.870
0.00002	201.7	844.6	295.9	44.1	2.111	5.767	34.063	39.317
0.00008	308.5	1091.7	355.7	93.9	2.249	5.943	34.603	79.086
0.00032	380.2	1460.2	792.1	137.4	2.261	6.011	35.969	108.554
0.00128	1587.4	2750.4	752.1	189.0	2.351	6.055	36.032	138.329
0.00512	5023.9	4069.9	612.5	355.1	2.356	6.116	35.954	202.640
0.01024	5149.7	4607.7	1198.6	1342.5	2.281	6.115	36.472	318.532
0.02048	4664.6	3662.3	1844.6	2227.4	2.216	6.081	36.552	340.718
0.04096	4089.9	2959.7	1523.3	694.5	2.200	6.044	36.324	262.462
0.08192	3085.9	2236.8	901.7	360.5	2.192	5.975	35.772	200.861
0.32768	1601.6	855.4	357.5	145.4	2.247	5.613	34.025	112.427
θ^*	4739.2	4473.7	663.5	138.7	2.213	6.114	35.827	109.341
Average θ^*	0.0204	0.0062	0.0008	0.0003	0.0204	0.0062	0.0008	0.0003

Table 2: Expected discounted reward and expected total reward for different values of the discount factor α and the violation budget θ. Here, the policies were constructed using $S = 200{,}000$ sampled states. The last row reports average value of the implicit violation budget θ^* for different values of the discount factor α.

distributed exponentially with rate $\mu_1 \triangleq 2$ and $\mu_2 \triangleq 2$ respectively. Vertical jobs exit the system after service, while horizontal jobs proceed to queue 3. There, they await service by server 2. The service times at server 2 are exponentially distributed with rate $\mu_3 \triangleq 1$. Given a common arrival rate $\lambda_1 \triangleq \lambda_2 \triangleq \lambda$, by analysis of the static planning LP (Harrison, 1988) associated with the network, it is straight forward to derive that the load of the network takes the form

$$\rho = \max\left(\frac{\lambda_1}{\mu_1} + \frac{\lambda_2}{\mu_2}, \frac{\lambda_2}{\mu_3}\right) = \lambda.$$

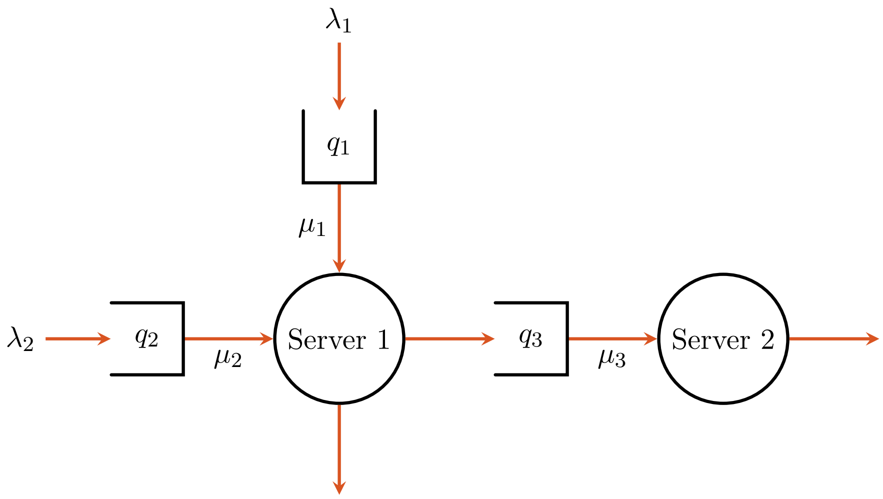


Figure 4: A criss-cross queueing network consisting of three queues and two servers. One class of jobs arrives to the system at queue 1, and departs after service by server 1. The second class of jobs arrives to the system at queue 2, and departs after sequential service from server 1 followed by server 2.

The SALP and ALP methodologies were applied to this queueing network as follows:

- **MDP formulation.** The evolution of this queuing network is described by a continuous time Markov chain with the state $q \in \mathbb{Z}_+^3$ corresponding to the queue lengths. Via a standard uniformization construction,[6] we consider an equivalent discrete time formulation, where $q_t \in \mathbb{Z}_+^3$ is the vector of queue lengths after the tth event, for $t \in \{0, 1, \ldots\}$. At each time, the choice of action corresponds to an assignment of each server to an associated non-empty queue, and idling is allowed.

- **Reward structure.** We seek find a control policy that optimizes the discounted infinite horizon cost objective

$$\text{minimize } \mathsf{E}\left[\sum_{t=0}^{\infty} \alpha^t c^\top q_t\right].$$

[6] See, for example, Bertsekas (2007b), or Moallemi et al. (2008) for an explicit construction in this context.

Here, the vector $c \in \mathbb{R}^3_+$ denotes the holding costs associated with each queue, and α is a discount factor. We use $\alpha = 0.98$ in our numerical experiments.

- **Basis functions.** Four basis functions were used: the constant function, and, for each queue, a quadratic function in the queue length. In other words, our basis function architecture is given by $\Phi(q) \triangleq [1\ q_1^2\ q_2^2\ q_3^2]$.
- **State sampling.** States were sampled from the stationary distribution of a policy which acts greedily according to the value function surrogate[7] $V(q) \triangleq \|q\|_2^2$. We use a collection $\hat{\mathcal{X}}$ of $S = 40{,}000$ sampled states as input to SALP. The results are averaged over 10 different collections of S samples.
- **Optimization.** The sampled states $\hat{\mathcal{X}}$ were used as input to optimization program (31). For increasing choices of violation budget $\theta \geq 0$, the linear program was solved to obtain policies. A policy corresponding to the implicit choice $\theta = \theta^*$ was obtained by separately solving linear program (32). Our implementation used CPLEX 11.0.0 to solve the resulting linear programs.
- **Policy evaluation.** Given a value function approximation, the expected discounted performance of the corresponding policy was evaluated by simulating 100 sample paths starting from an empty state ($q = 0$). Each sample path was evaluated over 50,000,000 time steps to compute the discounted cost.

We first consider the case where the holding costs are given by the vector $c \triangleq (1, 1, 3)$. This corresponds to Case IIB as considered by Martins et al. (1996), and the associated stochastic control problem is known to be challenging (Kumar and Muthuraman, 2004). These particular parameter settings are difficult because of the fact the holding costs for queue 3 are so much higher than those for queue 2. Hence, it may be optimal for server 1 to idle even if there are jobs in queue 2 so as to keep jobs in the cheaper buffer. On the other hand, too much idling at server 1 could lead to an empty queue 3, which would force idling at server 2. Hence, the policy decision for server 1 also depends on the downstream queue length.

Observe that our queueing problem has a countably infinite state space and it is not possible to exactly determine the optimal cost via a standard dynamic programming approach. We compute lower bounds on the optimal cost by considering a problem with a truncated state space, obtained by limiting the maximum queue length to size 30. Further, the transition probabilities are modified so that arrivals to a queue at maximum capacity result in self-transitions. For this modified problem, we enumerate all possible states and solve the exact linear program given by (2). The solution to this linear program yields the optimal cost, starting from the empty state. It is not hard to argue that this value should be a lower bound on the original, untruncated problem.

[7] This corresponds approximately to a ‘maximum pressure’ policy (Tassiulas and Ephremides, 1992, 1993; Dai and Lin, 2005).

In Table 3(a), we see the resulting performance of policies by solving SALP for various values of θ and for the ALP (i.e., $\theta = 0$). We also compute lower bounds on the optimal cost by solving the exact dynamic program for the aforementioned truncated state space problem. We report an optimality gap, defined as the performance normalized relative to the truncated lower bound. The results are shown for various levels of the load ρ. Overall, we observe a significant reduction in cost by policies generated via SALP in comparison to ALP. Using a line search to find the best choice of θ yields an SALP policy with 15% optimality gap as opposed to ALP policy, which results in 95% optimality gap. The policy corresponding to the implicit choice of $\theta = \theta^*$, obtained by solving LP (32), has an optimality gap of 40%.

In Table 3(b), we consider the case when the holding costs are given by the vector $c \triangleq (1, 1, 1)$. This is a considerably easier control problem, since there is no need for server 1 to idle. In this case, the SALP is still a significant improvement over the ALP, however the magnitude of the improvement is smaller.

8. Conclusion

The approximate linear programming (ALP) approach to approximate DP is interesting at the outset for two reasons. First, it gives us the ability to leverage commercial linear programming software to solve large ADP problems, and second, the ability to prove rigorous approximation guarantees and performance bounds. This paper asked whether the formulation considered in the ALP approach was the ideal formulation. In particular, we asked whether certain strong restrictions imposed on approximations produced by the approach can be relaxed in a tractable fashion and whether such a relaxation has a beneficial impact on the quality of the approximation produced. We have answered both of these questions in the affirmative. In particular, we have presented a novel linear programming formulation that, while remaining tractable, appears to yield substantial performance gains relative to the ALP. Further, our formulation permits us to prove approximation guarantees that are in general as strong as those available for the ALP, while being substantially stronger in particular problem instances.

There are a number of interesting algorithmic directions that warrant exploration. For instance, notice that from (30), that the SALP may be written as an unconstrained stochastic optimization problem. Such problems suggest natural *online* update rules for the weights r, based on stochastic gradient methods, yielding 'data-driven' ADP methods. The menagerie of online ADP algorithms available at present are effectively iterative methods for solving a projected version of Bellman's equation. TD-learning is a good representative of this type of approach and, as can be seen from Table 1, is not among the highest performing algorithms in our computational study. An online update rule that effectively solves the SALP promises policies that will perform on par with the SALP solution, while at the same time retaining the benefits of an online ADP algorithm. A second interesting algorithmic direction worth exploring is an extension of the smoothed linear

(a) Expected discounted cost for varying values of the load ρ, with holding costs $c = (1, 1, 3)$.

Violation Budget	**Expected Discounted Cost**					
	$\rho = 0.98$		$\rho = 0.95$		$\rho = 0.90$	
θ	Cost	Normalized	Cost	Normalized	Cost	Normalized
0 (ALP)	560.0	1.940	542.8	1.960	514.3	1.996
0.0001	560.0	1.940	542.8	1.959	514.4	1.996
0.0010	559.7	1.939	542.5	1.959	514.2	1.995
0.0100	588.7	2.039	570.7	2.060	541.2	2.100
0.1000	584.3	2.024	566.9	2.046	538.1	2.088
1.0000	502.8	1.742	486.1	1.755	459.0	1.781
θ^*	412.5	1.429	398.2	1.437	373.0	1.447
25.000	332.2	1.151	318.7	1.151	295.8	1.148
50.000	334.0	1.157	320.5	1.157	296.8	1.152
75.000	337.5	1.169	323.6	1.168	301.4	1.170
100.00	337.5	1.169	323.6	1.168	301.4	1.170
Lower Bound	288.7	1.000	277.0	1.000	257.7	1.000
Average θ^*	17.79		17.73		17.67	

(b) Expected discounted cost for load $\rho = 0.98$, with holding costs $c = (1, 1, 1)$.

Violation Budget	**Expected Discounted Cost**	
	$\rho = 0.98$	
θ	Cost	Normalized
0 (ALP)	334.5	1.581
0.0001	334.5	1.581
0.0010	381.1	1.801
0.0100	284.4	1.344
0.1000	237.9	1.124
1.0000	246.7	1.166
θ^*	245.9	1.162
25.000	250.4	1.183
50.000	254.4	1.202
75.000	254.4	1.202
100.00	254.4	1.202
Lower Bound	211.6	1.000
Average θ^*	11.81	

Table 3: Expected discounted cost for different values of the violation budget θ, load ρ, and holding costs c. Lower bounds are computed by solving the exact dynamic program for a problem truncated to a maximum queue length of 30. An optimality gap is reported by normalizing the cost by the truncated lower bound. Here, the expected discounted cost is measured starting from an empty state. The last row reports the average value of the implicit violation budget θ^*, for different values of the load ρ.

programming approach to average cost dynamic programming problems.

As discussed in Section 4, theoretical guarantees for ADP algorithms typically rely on some sort of idealized assumption. For instance, in the case of the ALP, it is the ability to solve an LP with a potentially intractable number of states or else access to a set of sampled states, sampled according to some idealized sampling distribution. For the SALP, it is the latter of the two assumptions. It would be interesting to see whether this assumption can be loosened for some specific class of MDPs. An interesting class of MDPs in this vein are high dimensional optimal stopping problems. Yet another direction for research, is understanding the dynamics of 'bootstrapping' procedures, that solve a sequence of sampled versions of the SALP with samples for a given SALP in the sequence drawn according to a policy produced by the previous SALP in the sequence.

Acknowledgements

We wish to thank the anonymous reviewers for a number of helpful suggestions, in particular, a strengthening of Theorem 2.

References

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, Belmont, MA, 3rd edition, 2007a.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, Belmont, MA, 3rd edition, 2007b.

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.

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

S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, Cambridge, UK, 2004.

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

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

J. G. Dai and W. Lin. Maximum pressure policies in stochastic processing networks. *Operations Research*, 53:197–218, 2005.

D. P. de Farias and B. Van Roy. On the existence of fixed points for approximate value iteration and temporal-difference learning. *Journal of Optimization Theory and Applications*, 105(3), 2000.

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

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.

D. P. de Farias and B. Van Roy. A cost-shaping linear program for average-cost approximate dynamic programming with performance guarantees. *Mathematics of Operations Research*, 31(3):597–620, 2006.

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.

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.

V. F. Farias and B. Van Roy. An approximate dynamic programming approach to network revenue management. Working paper, 2007.

V. F. Farias, D. Saure, and G. Y. Weintraub. An approximate dynamic programming approach to solving dynamic oligopoly models. Working paper, 2011.

J. M. Harrison. Brownian models of queueing networks with heterogeneous customer populations. In *Stochastic Differential Systems, Stochastic Control Theory and Applications (Minneapolis, Minn., 1986)*, volume 10 of *IMA Vol. Math. Appl.*, pages 147–186. Springer, New York, 1988.

J. M. Harrison and L. M. Wein. Scheduling network of queues: Heavy traffic analysis of a simple open network. *Queueing Systems*, 5:265–280, 1989.

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

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

S. Kumar and K. Muthuraman. A numerical method for solving singular stochastic control problems. *Operations Research*, 52(4):563–582, 2004.

H. J. Kushner and L. F. Martins. Heavy traffic analysis of a controlled multiclass queueing network via weak convergence methods. *SIAM J. Control Optim.*, 34(5):1781–1797, 1996.

A. S. Manne. Linear programming and sequential decisions. *Management Science*, 60(3):259–267, 1960.

L. F. Martins, S. E. Shreve, and H. M. Soner. Heavy traffic convergence of a controlled multiclass queueing network. *SIAM J. Control Optim.*, 34(6):2133–2171, 1996.

C. C. Moallemi, S. Kumar, and B. Van Roy. Approximate and data-driven dynamic programming for queueing networks. Working paper, 2008.

M. Petrik and S. Zilberstein. Constraint relaxation in approximate linear programs. In *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley and Sons, 2007.

P. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110:568–582, 1985.

A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia, PA, 2009.

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

L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12): 1936–1948, December 1992.

L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, March 1993.

B. Van Roy. Neuro-dynamic programming: Overview and recent trends. In A. Shwartz E. Feinberg, editor, *Handbook of Markov Decision Processes*. Kluwer, Boston, 2002.

Z. Wen, L. J. Durlofsky, B. Van Roy, and K. Aziz. Use of approximate dynamic programming for production optimization. To appear in *Society of Petroleum Engineerins Proceedings*, 2011.

A. Proofs for Sections 4.2–4.4

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

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

(ii)

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

(iii) The right partial derivative of $\ell(r, \theta)$ with respect to θ satisfies

$$\frac{\partial^+}{\partial \theta^+} \ell(r, 0) = -\left((1-\alpha) \sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x) \right)^{-1},$$

where

$$\Omega(r) \triangleq \operatorname*{argmax}_{\{x \in \mathcal{X}\,:\, \pi_{\mu^*,\nu}(x) > 0\}} \Phi r(x) - T\Phi r(x).$$

Proof. (i) Given any r, clearly $\gamma \triangleq \|\Phi r - T\Phi r\|_\infty$, $s \triangleq \mathbf{0}$ is a feasible point for (9), so $\ell(r, \theta)$ is feasible. To see that the LP is bounded, suppose (s, γ) is feasible. Then, for any $x \in \mathcal{X}$ with $\pi_{\mu^*,\nu}(x) > 0$,

$$\gamma \geq \Phi r(x) - T\Phi r(x) - s(x) \geq \Phi r(x) - T\Phi r(x) - \theta/\pi_{\mu^*,\nu}(x) > -\infty.$$

Letting (γ_1, s_1) and (γ_2, s_2) represent optimal solutions for the LP (9) with parameters (r, θ_1) and (r, θ_2) respectively, it is easy to see that $((\gamma_1 + \gamma_2)/2, (s_1 + s_2)/2)$ is feasible for the LP with parameters $(r, (\theta_1 + \theta_2)/2)$. It follows that $\ell(r, (\theta_1 + \theta_2)/2) \leq (\ell(r, \theta_1) + \ell(r, \theta_2))/2$. The remaining properties are simple to check.

(ii) Let $\epsilon \triangleq \|J^* - \Phi r\|_\infty$. Then, since T is an α-contraction under the $\|\cdot\|_\infty$ norm,

$$\|T\Phi r - \Phi r\|_\infty \leq \|J^* - T\Phi r\|_\infty + \|J^* - \Phi r\|_\infty \leq \alpha \|J^* - \Phi r\|_\infty + \epsilon = (1+\alpha)\epsilon.$$

Since $\gamma \triangleq \|T\Phi r - \Phi r\|_\infty$, $s \triangleq \mathbf{0}$ is feasible for (9), the result follows.

(iii) Fix $r \in \mathbb{R}^K$, and define

$$\Delta \triangleq \max_{\{x \in \mathcal{X} \,:\, \pi_{\mu^*,\nu}(x) > 0\}} \big(\Phi r(x) - T\Phi r(x)\big) - \max_{\{x \in \mathcal{X} \setminus \Omega(r) \,:\, \pi_{\mu^*,\nu}(x) > 0\}} \big(\Phi r(x) - T\Phi r(x)\big) > 0.$$

Consider the program for $\ell(r, \delta)$. It is easy to verify that for $\delta \geq 0$ and sufficiently small, viz. $\delta \leq \Delta \sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x)$, $(\bar{s}_\delta, \bar{\gamma}_\delta)$ is an optimal solution to the program, where

$$\bar{s}_\delta(x) \triangleq \begin{cases} \frac{\delta}{\sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x)} & \text{if } x \in \Omega(r), \\ 0 & \text{otherwise}, \end{cases}$$

and

$$\bar{\gamma}_\delta \triangleq \gamma_0 - \frac{\delta}{\sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x)},$$

so that

$$\ell(r, \delta) = \ell(r, 0) - \frac{\delta}{(1-\alpha) \sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x)}.$$

Thus,

$$\frac{\ell(r, \delta) - \ell(r, 0)}{\delta} = -\left((1-\alpha) \sum_{x \in \Omega(r)} \pi_{\mu^*,\nu}(x) \right)^{-1}.$$

Taking a limit as $\delta \searrow 0$ yields the result. ■

Lemma 2. *Suppose that the vectors $J \in \mathbb{R}^{\mathcal{X}}$ and $s \in \mathbb{R}^{\mathcal{X}}$ satisfy*

$$J \leq T_{\mu^*} J + s.$$

Then,

$$J \leq J^* + \Delta^* s,$$

where

$$\Delta^* \triangleq \sum_{k=0}^{\infty} (\alpha P_{\mu^*})^k = (I - \alpha P_{\mu^*})^{-1},$$

and P_{μ^*} is the transition probability matrix corresponding to an optimal policy.

Proof. Note that the T_{μ^*}, the Bellman operator corresponding to the optimal policy μ^*, is monotonic and is a contraction. Then, repeatedly applying T_{μ^*} to the inequality $J \leq T_{\mu^*} J + s$ and using the fact that $T_{\mu^*}^k J \to J^*$, we obtain

$$J \leq J^* + \sum_{k=0}^{\infty} (\alpha P_{\mu^*})^k s = J^* + \Delta^* s.$$

■

Lemma 3. *For the autonomous queue with basis functions $\phi_1(x) \triangleq 1$ and $\phi_2(x) \triangleq x$, if N is sufficiently large, then*

$$\inf_{r,\psi\in\bar{\Psi}} \frac{2\nu^\top\psi}{1-\alpha\beta(\psi)}\|J^* - \Phi r\|_{\infty,1/\psi} \geq \frac{3\rho_2 q}{32(1-q)}(N-1).$$

Proof. We have:

$$\inf_{r,\psi\in\bar{\Psi}} \frac{2\nu^\top\psi}{1-\alpha\beta(\psi)}\|J^* - \Phi r\|_{\infty,1/\psi} \geq \inf_{\psi\in\bar{\Psi}} \frac{2\nu^\top\psi}{\|\psi\|_\infty} \inf_r \|J^* - \Phi r\|_\infty.$$

We will produce lower bounds on the two infima on the right-hand side above. Observe that

$$\begin{aligned}\inf_r \|J^* - \Phi r\|_\infty &= \inf_r \max_x |\rho_2 x^2 + \rho_1 x + \rho_0 - r_1 x - r_0| \\ &\geq \inf_r \max\left(\max_x |\rho_2 x^2 + (\rho_1 - r_1)x| - |\rho_0 - r_0|, |\rho_0 - r_0|\right) \\ &= \inf_{r_0} \max\left(\inf_{r_1} \max_x |\rho_2 x^2 + (\rho_1 - r_1)x| - |\rho_0 - r_0|, |\rho_0 - r_0|\right),\end{aligned}$$

which follows from the triangle inequality and the fact that

$$\max_x |\rho_2 x^2 + \rho_1 x + \rho_0 - r_1 x - r_0| \geq |\rho_0 - r_0|.$$

Routine algebra verifies that

$$\inf_{r_1}\max_x |\rho_2 x^2 + (\rho_1 - r_1)x| \geq \tfrac{3}{16}\rho_2(N-1)^2. \tag{35}$$

It thus follows that

$$\inf_r \|J^* - \Phi r\|_\infty \geq \inf_{r_0} \max\left(\tfrac{3}{16}\rho_2(N-1)^2 - |\rho_0 - r_0|, |\rho_0 - r_0|\right) \geq \tfrac{3}{32}\rho_2(N-1)^2.$$

We next note that any $\psi \in \tilde{\Psi}$ must satisfy $\psi \in \text{span}(\Phi)$ and $\psi \geq \mathbf{1}$. Thus, $\psi \in \tilde{\Psi}$ must take the form $\psi(x) = \alpha_1 x + \alpha_0$ with $\alpha_0 \geq 1$ and $\alpha_1 \geq (1-\alpha_0)/(N-1)$. Thus, $\|\psi\|_\infty = \max(\alpha_1(N-1) + \alpha_0, \alpha_0)$. Define $\kappa(N)$ to be the expected queue length under the distribution ν, i.e.,

$$\kappa(N) \triangleq \sum_{x=0}^{N-1} \nu(x)x = \frac{1-q}{1-q^N}\sum_{x=0}^{N-1} xq^x = \frac{q}{1-q}\left[\frac{1 - Nq^{N-1}(1-q) - q^N}{1-q^N}\right],$$

so that $\nu^\top\psi = \alpha_1\kappa(N) + \alpha_0$, Thus,

$$\inf_{\psi\in\tilde{\Psi}} \frac{2\nu^\top\psi}{\|\psi\|_\infty} \inf_r \|J^* - \Phi r\|_\infty \geq \tfrac{3}{16}\rho_2 \inf_{\substack{\alpha_0\geq 1 \\ \alpha_1 \geq \frac{1-\alpha_0}{N-1}}} \frac{\alpha_1\kappa(N) + \alpha_0}{\max(\alpha_1(N-1)+\alpha_0, \alpha_0)}(N-1)^2$$

When $(1-\alpha_0)/(N-1) \leq \alpha_1 \leq 0$, we have

$$\begin{aligned}\frac{\alpha_1\kappa(N)+\alpha_0}{\max(\alpha_1(N-1)+\alpha_0,\alpha_0)}(N-1)^2 &= \frac{\alpha_1\kappa(N)+\alpha_0}{\alpha_0}(N-1)^2 \\ &\geq \frac{(1-\alpha_0)\kappa(N)/(N-1)+\alpha_0}{\alpha_0}(N-1)^2 \\ &\geq \left(1-\frac{\kappa(N)}{N-1}\right)(N-1)^2.\end{aligned}$$

When $\alpha_1 > 0$, we have

$$\frac{\alpha_1\kappa(N)+\alpha_0}{\max(\alpha_1(N-1)+\alpha_0,\alpha_0)}(N-1)^2 = \frac{\alpha_1\kappa(N)+\alpha_0}{\alpha_1(N-1)+\alpha_0}(N-1)^2 \geq (N-1)\kappa(N),$$

where the inequality follows from the fact that $\kappa(N) \leq N-1$ and $\alpha_0 > 0$. It then follows that

$$\inf_{\psi\in\tilde{\Psi}} \frac{2\nu^\top\psi}{\|\psi\|_\infty} \inf_r \|J^* - \Phi r\|_\infty \geq \tfrac{3}{16}\rho_2 \min\left(\kappa(N)(N-1), \left(1-\frac{\kappa(N)}{N-1}\right)(N-1)^2\right).$$

Now, observe that $\kappa(N)$ is increasing in N. Also, by assumption, $p < 1/2$, so $q < 1$ and thus $\kappa(N) \to q/(1-q)$ as $N \to \infty$. Then, for N sufficiently large, $\frac{1}{2}q/(1-q) \leq \kappa(N) \leq q/(1-q)$. Therefore, for N sufficiently large,

$$\inf_{\psi\in\tilde{\Psi}} \frac{2\nu^\top\psi}{\|\psi\|_\infty} \inf_r \|J^* - \Phi r\|_\infty \geq \frac{3\rho_2 q}{32(1-q)}(N-1),$$

as desired. ■

Lemma 4. *For every $\lambda \geq 0$, there exists a $\hat{\theta} \geq 0$ such that an optimal solution (r^*, s^*) to*

$$\begin{array}{ll}\underset{r,s}{\text{maximize}} & \nu^\top\Phi r - \lambda\pi_{\mu^*,\nu}^\top s \\ \text{subject to} & \Phi r \leq T\Phi r + s, \quad s \geq \mathbf{0}.\end{array} \tag{36}$$

is also an optimal solution the SALP (8) with $\theta = \hat{\theta}$.

Proof. Let $\hat{\theta} \triangleq \pi_{\mu^*,\nu}^\top s^*$. It is then clear that (r^*, s^*) is a feasible solution to (8) with $\theta = \hat{\theta}$. We claim that it is also an optimal solution. To see this, assume to the contrary that it is not an optimal solution, and let $(\tilde{r}, \tilde{s})$ be an optimal solution to (8). It must then be that $\pi_{\mu^*,\nu}^\top\tilde{s} \leq \hat{\theta} = \pi_{\mu^*,\nu}^\top s^*$ and moreover, $\nu^\top\Phi\tilde{r} > \nu^\top\Phi r^*$ so that

$$\nu^\top\Phi r^* - \lambda\pi_{\mu^*,\nu}^\top s^* < \nu^\top\Phi\tilde{r} - \lambda\pi_{\mu^*,\nu}^\top\tilde{s}.$$

This, in turn, contradicts the optimality of (r^*, s^*) for (36) and yields the result. ■

B. Proof of Theorem 4

Our proof of Theorem 4 is based on uniformly bounding the rate of convergence of sample averages of a certain class of functions. We begin with some definitions: consider a family $\mathcal{F}$ of functions from a set $\mathcal{S}$ to $\{0,1\}$. Define the *Vapnik-Chervonenkis (VC) dimension* $\dim_{\mathrm{VC}}(\mathcal{F})$ to be the cardinality d of the largest set $\{x_1, x_2, \ldots, x_d\} \subset \mathcal{S}$ satisfying:

$$\forall\ e \in \{0,1\}^d,\ \exists f \in \mathcal{F} \text{ such that } \forall\ i,\ \ f(x_i) = 1 \text{ iff } e_i = 1.$$

Now, let $\mathcal{F}$ be some set of *real*-valued functions mapping $\mathcal{S}$ to $[0, B]$. The *pseudo-dimension* $\dim_P(\mathcal{F})$ is the following generalization of VC dimension: for each function $f \in \mathcal{F}$ and scalar $c \in \mathbb{R}$, define a function $g\colon \mathcal{S} \times \mathbb{R} \to \{0,1\}$ according to:

$$g(x, c) \triangleq \mathbb{I}_{\{f(x)-c\geq 0\}}.$$

Let $\mathcal{G}$ denote the set of all such functions. Then, we define $\dim_P(\mathcal{F}) \triangleq \dim_{\mathrm{VC}}(\mathcal{G})$.

In order to prove Theorem 4, define the $\mathcal{F}$ to be the set of functions $f\colon \mathbb{R}^K \times \mathbb{R} \to [0, B]$, where, for all $x \in \mathbb{R}^K$ and $y \in \mathbb{R}$,

$$f(y, z) \triangleq \zeta\left(r^\top y + z\right).$$

Here, $\zeta(t) \triangleq \max\left(\min(t, B), 0\right)$, and $r \in \mathbb{R}^K$ is a vector that parameterizes f. We will show that $\dim_P(\mathcal{F}) \leq K + 2$. We will use the following standard result from convex geometry:

Lemma 5 (Radon's Lemma). *A set $A \subset \mathbb{R}^m$ of $m+2$ points can be partitioned into two disjoint sets A_1 and A_2, such that the convex hulls of A_1 and A_2 intersect.*

Lemma 6. $\dim_P(\mathcal{F}) \leq K + 2$

Proof. Assume, for the sake of contradiction, that $\dim_P(\mathcal{F}) > K + 2$. It must be that there exists a 'shattered' set

$$\left\{(y^{(1)}, z^{(1)}, c^{(1)}), (y^{(2)}, z^{(2)}, c^{(2)}), \ldots, (y^{(K+3)}, z^{(K+3)}, c^{(K+3)})\right\} \subset \mathbb{R}^K \times \mathbb{R} \times \mathbb{R},$$

such that, for all $e \in \{0,1\}^{K+3}$, there exists a vector $r_e \in \mathbb{R}^K$ with

$$\zeta\left(r_e^\top y^{(i)} + z^{(i)}\right) \geq c^{(i)} \text{ iff } e_i = 1, \quad \forall\ 1 \leq i \leq K+3.$$

Observe that we must have $c^{(i)} \in (0, B]$ for all i, since if $c^{(i)} \leq 0$ or $c^{(i)} > B$, then no such shattered set can be demonstrated. But if $c^{(i)} \in (0, B]$, for all $r \in \mathbb{R}^K$,

$$\zeta\left(r^\top y^{(i)} + z^{(i)}\right) \geq c^{(i)} \implies r_e^\top y^{(i)} \geq c^{(i)} - z^{(i)},$$

and

$$\zeta\left(r^\top y^{(i)} + z^{(i)}\right) < c^{(i)} \implies r_e^\top y^{(i)} < c^{(i)} - z^{(i)}.$$

For each $1 \leq i \leq K+3$, define $x^{(i)} \in \mathbb{R}^{K+1}$ component-wise according to

$$x_j^{(i)} \triangleq \begin{cases} y_j^{(i)} & \text{if } j < K+1, \\ c^{(i)} - z^{(i)} & \text{if } j = K+1. \end{cases}$$

Let $A = \{x^{(1)}, x^{(2)}, \ldots, x^{(K+3)}\} \subset \mathbb{R}^{K+1}$, and let A_1 and A_2 be subsets of A satisfying the conditions of Radon's lemma. Define a vector $\tilde{e} \in \{0,1\}^{K+3}$ component-wise according to

$$\tilde{e}_i \triangleq \mathbb{I}_{\left\{x^{(i)} \in A_1\right\}}.$$

Define the vector $\tilde{r} \triangleq r_{\tilde{e}}$. Then, we have

$$\sum_{j=1}^{K} \tilde{r}_j x_j \geq x_{K+1}, \quad \forall\, x \in A_1,$$

$$\sum_{j=1}^{K} \tilde{r}_j x_j < x_{K+1}, \quad \forall\, x \in A_2.$$

Now, let $\bar{x} \in \mathbb{R}^{K+1}$ be a point contained in both the convex hull of A_1 and the convex hull of A_2. Such a point must exist by Radon's lemma. By virtue of being contained in the convex hull of A_1, we must have

$$\sum_{j=1}^{K} \tilde{r}_j \bar{x}_j \geq \bar{x}_{K+1}.$$

Yet, by virtue of being contained in the convex hull of A_2, we must have

$$\sum_{j=1}^{K} \tilde{r}_j \bar{x}_j < \bar{x}_{K+1},$$

which is impossible. ■

With the above pseudo-dimension estimate, we can establish the following lemma, which provides a Chernoff bound for the *uniform* convergence of a certain class of functions:

Lemma 7. *Given a constant $B > 0$, define the function $\zeta\colon \mathbb{R} \to [0, B]$ by*

$$\zeta(t) \triangleq \max\left(\min(t, B), 0\right).$$

Consider a pair of random variables $(Y, Z) \in \mathbb{R}^K \times \mathbb{R}$. For each $i = 1, \ldots, n$, let the pair $\left(Y^{(i)}, Z^{(i)}\right)$

be an i.i.d. sample drawn according to the distribution of (Y, Z)*. Then, for all* $\epsilon \in (0, B]$,

$$\mathsf{P}\left(\sup_{r\in\mathbb{R}^K} \left| \frac{1}{n}\sum_{i=1}^{n} \zeta\left(r^\top Y^{(i)} + Z^{(i)}\right) - \mathsf{E}\left[\zeta\left(r^\top Y + Z\right)\right] \right| > \epsilon \right) \\ \leq 8\left(\frac{32eB}{\epsilon}\log\frac{32eB}{\epsilon}\right)^{K+2} \exp\left(-\frac{\epsilon^2 n}{64B^2}\right).$$

Moreover, given $\delta \in (0,1)$*, if*

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

then this probability is at most δ*.*

Proof. Given Lemma 6, this follows immediately from Corollary 2 of of Haussler (1992, Section 4). ■

We are now ready to prove Theorem 4.

Theorem 4. *Under the conditions of Theorem 2, let* r_{SALP} *be an optimal solution to the SALP* (14)*, and let* $\hat{r}_{SALP}$ *be an optimal solution to the sampled SALP* (28)*. 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 $1 - \delta - 2^{-383}\delta^{128}$*,*

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

Proof. Define the vectors

$$\hat{s}_{\mu^*} \triangleq \left(\Phi\hat{r}_{\text{SALP}} - T_{\mu^*}\Phi\hat{r}_{\text{SALP}}\right)^+, \quad \text{and} \quad \hat{s} \triangleq \left(\Phi\hat{r}_{\text{SALP}} - T\Phi\hat{r}_{\text{SALP}}\right)^+.$$

Note that $\hat{s}_{\mu^*} \leq \hat{s}$. One has, via Lemma 2, that

$$\Phi\hat{r}_{\text{SALP}} - J^* \leq \Delta^*\hat{s}_{\mu^*}$$

Thus, as in the last set of inequalities in the proof of Theorem 1, we have

$$\|J^* - \Phi\hat{r}_{\text{SALP}}\|_{1,\nu} \leq \nu^\top(J^* - \Phi\hat{r}_{\text{SALP}}) + \frac{2\pi_{\mu^*,\nu}^\top\hat{s}_{\mu^*}}{1-\alpha}. \tag{37}$$

Now, let $\hat{\pi}_{\mu^*,\nu}$ be the empirical measure induced by the collection of sampled states $\hat{\mathcal{X}}$. Given a state $x \in \mathcal{X}$, define a vector $Y(x) \in \mathbb{R}^K$ and a scalar $Z(x) \in \mathbb{R}$ according to

$$Y(x) \triangleq \Phi(x)^\top - \alpha P_{\mu^*} \Phi(x)^\top, \quad Z(x) \triangleq -g(x, \mu^*(x)),$$

so that, for any vector of weights $r \in \mathcal{N}$,

$$(\Phi r(x) - T_{\mu^*} \Phi r(x))^+ = \zeta \left(r^\top Y(x) + Z(x) \right).$$

Then,

$$\left| \hat{\pi}_{\mu^*,\nu}^\top \hat{s}_{\mu^*} - \pi_{\mu^*,\nu}^\top \hat{s}_{\mu^*} \right| \leq \sup_{r \in \mathcal{N}} \left| \frac{1}{S} \sum_{x \in \hat{\mathcal{X}}} \zeta \left(r^\top Y(x) + Z(x) \right) - \sum_{x \in \mathcal{X}} \pi_{\mu^*,\nu}(x) \zeta \left(r^\top Y(x) + Z(x) \right) \right|.$$

Applying Lemma 7, we have that

$$\mathsf{P} \left(\left| \hat{\pi}_{\mu^*,\nu}^\top \hat{s}_{\mu^*} - \pi_{\mu^*,\nu}^\top \hat{s}_{\mu^*} \right| > \epsilon \right) \leq \delta. \tag{38}$$

Next, suppose $(r_{\text{SALP}}, \bar{s})$ is an optimal solution to the SALP (14). Then, with probability at least $1 - \delta$,

$$\begin{aligned} \nu^\top (J^* - \Phi \hat{r}_{\text{SALP}}) + \frac{2\pi_{\mu^*,\nu}^\top \hat{s}_{\mu^*}}{1-\alpha} &\leq \nu^\top (J^* - \Phi \hat{r}_{\text{SALP}}) + \frac{2\hat{\pi}_{\mu^*,\nu}^\top \hat{s}_{\mu^*}}{1-\alpha} + \frac{2\epsilon}{1-\alpha} \\ &\leq \nu^\top (J^* - \Phi \hat{r}_{\text{SALP}}) + \frac{2\hat{\pi}_{\mu^*,\nu}^\top \hat{s}}{1-\alpha} + \frac{2\epsilon}{1-\alpha} \\ &\leq \nu^\top (J^* - \Phi r_{\text{SALP}}) + \frac{2\hat{\pi}_{\mu^*,\nu}^\top \bar{s}}{1-\alpha} + \frac{2\epsilon}{1-\alpha}, \end{aligned} \tag{39}$$

where the first inequality follows from (38), and the final inequality follows from the optimality of $(\hat{r}_{\text{SALP}}, \hat{s})$ for the sampled SALP (28).

Notice that, without loss of generality, we can assume that $\bar{s}(x) = (\Phi r_{\text{SALP}}(x) - T \Phi r_{\text{SALP}}(x))^+$, for each $x \in \mathcal{X}$. Thus, $0 \leq \bar{s}(x) \leq B$. Further,

$$\hat{\pi}_{\mu^*,\nu}^\top \bar{s} - \pi_{\mu^*,\nu}^\top \bar{s} = \frac{1}{S} \sum_{x \in \hat{\mathcal{X}}} \left(\bar{s}(x) - \pi_{\mu^*,\nu}^\top \bar{s} \right),$$

where the right-hand-side is of a sum of zero-mean bounded i.i.d. random variables. Applying Hoeffding's inequality,

$$\mathsf{P} \left(\left| \hat{\pi}_{\mu^*,\nu}^\top \bar{s} - \pi_{\mu^*,\nu}^\top \bar{s} \right| \geq \epsilon \right) \leq 2 \exp \left(-\frac{2S\epsilon^2}{B^2} \right) < 2^{-383} \delta^{128},$$

where final inequality follows from our choice of S. Combining this with (37) and (39), with probability at least $1-\delta-2^{-383}\delta^{128}$, we have

$$\begin{aligned}\|J^*-\Phi\hat{r}_{\mathrm{SALP}}\|_{1,\nu} &\leq \nu^\top(J^*-\Phi r_{\mathrm{SALP}})+\frac{2\hat{\pi}_{\mu^*,\nu}^\top\bar{s}}{1-\alpha}+\frac{2\epsilon}{1-\alpha}\\ &\leq \nu^\top(J^*-\Phi r_{\mathrm{SALP}})+\frac{2\pi_{\mu^*,\nu}^\top\bar{s}}{1-\alpha}+\frac{4\epsilon}{1-\alpha}.\end{aligned}$$

The result then follows from (17)–(19) in the proof of Theorem 2. ■