Kernel-Based Reinforcement Learning

DIRK ORMONEIT AND ŚAUNAK SEN ormoneit@stat.stanford.edu
Department of Statistics, Stanford University, Stanford, CA 94305-4065

Abstract. We present a kernel-based approach to reinforcement learning that overcomes the stability problems of temporal-difference learning in continuous state-spaces. First, our algorithm converges to a unique solution of an approximate Bellman's equation regardless of its initialization values. Second, the method is consistent in the sense that the resulting policy converges asymptotically to the optimal policy. Parametric value function estimates such as neural networks do not possess this property. Our kernel-based approach also allows us to show that the limiting distribution of the value function estimate is a Gaussian process. This information is useful in studying the bias-variance tradeoff in reinforcement learning. We find that all reinforcement learning approaches to estimating the value function, parametric or non-parametric, are subject to a bias. This bias is typically larger in reinforcement learning than in a comparable regression problem.

Keywords: Reinforcement Learning, Markov Decision Process, Kernel-Based Learning, Kernel Smoothing, Local Averaging, Lazy Learning

1. Introduction

Reinforcement learning has been applied successfully to a variety of practical applications including prominent examples such as Tesauro's Neurogammon or Singh and Bertsekas' dynamic channel allocation algorithms [29, 24]. A fundamental obstacle to a widespread application of reinforcement learning to industrial problems is that reinforcement learning algorithms frequently fail to converge to a solution. This is particularly true for variants of temporal difference learning that use parametric function approximators (for example, linear combinations of feature vectors or neural networks) to represent the value function of the underlying Markov Decision Process (MDP). For a detailed discussion of this problem, as well as a list of exceptions, the interested reader is referred to [5, 31]. By adopting a non-parametric perspective on reinforcement learning, we suggest an algorithm that always converges to a unique solution. This algorithm assigns value function estimates to the states in a sample trajectory and updates these estimates iteratively. Each update is based on kernel-based averaging. An advantage of using kernel-based methods is that a wide body of statistical literature on local averaging methods can be brought to bear upon reinforcement learning. In particular, the close connection between kernel-based reinforcement learning and nonlinear regression using kernel smoothers allows us to demonstrate that our algorithm is consistent in the statistical sense. In other words, additional training data always improve the quality of the estimates and eventually lead to optimal performance. To the best of our knowledge, this is the first reinforcement learning algorithm for which such a global optimality property has been demonstrated in a continuous-space framework. By contrast, recently-advocated "direct" policy search or perturbation methods can, by construction, be optimal at most in a local sense [28, 33]. Besides establishing consistency we also derive the limiting distribution of the value function estimate. This is useful in studying the bias-variance tradeoff in reinforcement learning. We also find that all reinforcement learning algorithms, by their nature are subject to a bias. This bias is larger than in a comparable regression problem. We also provide an asymptotic formula for the bias increase which could help understand this issue in a more general framework.

In the context of reinforcement learning, local averaging has been suggested in work by Rust [23] and Gordon [11], making the assumption that the transition probabilities of the MDP are known and can be used for learning. Our approach is fundamentally different in that kernel-based reinforcement learning only relies on the sample trajectories of the MDP. Therefore it is more widely applicable. Other related ideas can be found in [35, 30, 6, 14, 19]. Baird and Klopf [2] apply the nearest neighbor algorithm to reinforcement learning, and Connell and Utgoff [8], Peng [20], and Atkeson, Moore, and Schaal [1] apply locally weighted regression to physical control problems. The temporal-difference learning algorithm is due to Sutton and the idea to directly approximate the "action-value function" that is also used in this work was first used by Watkins [27, 34]. With regard to theoretical results, Tsitsiklis and Van Roy [32] prove the convergence of a stochastic algorithm for the estimation of the value function in optimal stopping problems. For practical applications of simulation-based optimal control in Finance, see Longstaff and Schwartz's paper on American option pricing [15] and Brandt's work on Optimal Portfolio Choice [7]. While our method addresses both discounted- and average-cost problems, we focus on discounted-costs here and refer the reader interested in average-costs to other work [16, 17].

The remainder of this work is organized as follows. In Section 2, we review basic facts about Markov Decision Processes. In Sections 3 and 4, we introduce the kernel-based reinforcement learning operator and discuss algorithmic considerations relevant for its practical application. In Section 5 we present the main theoretical results of this paper, including theorems establishing the consistency of kernel-based reinforcement learning and the asymptotic distribution of the resulting estimate. In Section 6 we use these results to derive asymptotic bias formulas for the random Bellman operator. Section 7 concludes.

2. Markov Decision Processes

Consider a system which is observed at discrete time steps and where the state at time t, X_t, takes on values in $[0,1]^d$ for $t = 1, 2, \ldots, T$, $T \leq \infty$. At each time point, an agent has the option of taking one of a finitely many actions in the set $A = \{1, 2, \ldots, M\}$ based on the state of the system at that time, X_t, alone. The system reacts to the action taken at time t, denoted by a_t, stochastically in a manner that depends only on the current state of the system and the action taken (i.e., the dependence is Markovian). Assume that the transitions are homogeneous in time and the density of $X_{t+1} = y$ given $X_t = x$ and $a_t = a$ is given by $p(y|x, a)$. The corresponding probability measure is denoted by $P_a(x, \cdot)$. The reward obtained from this transition is given by the reward function $r(y, x, a)$. The goal is to identify "good" actions or policies that lead to large rewards.

The sequences $\{X_t\}$ and $\{a_t\}$ are stochastic processes defined on a probability space $(\Omega, \mathcal{F}, P)$. The knowledge of the agent at time t is represented by the σ-algebra $\mathcal{F}_t = \sigma(X_0, a_0, \ldots, X_t)$. Let E_t denote the expectation conditional on $\mathcal{F}_t$. Formal assumptions are summarized in Appendix A.1.

We consider both finite- and infinite-horizon versions of the scenario described above which correspond to the cases $T < \infty$ and $T = \infty$, respectively. We assume that at any given time t, the utility of all future actions is of the discounted additive form

$$u_t(a_t, \ldots, a_{T-1}) = \sum_{s=t}^{T-1} \alpha^{s-t} r(X_{s+1}, X_s, a_s) + \mathbb{1}(T < \infty)\alpha^{T-t} R(X_T),$$

where $1 \geq \alpha > 0$ is the discount factor.

In the finite-horizon case ($T < \infty$), there is a special terminal reward $R(X_T)$ at the last period T. The task is to identify actions so as to maximize the expected utility of all future actions given current knowledge, i.e. so as to maximize $E_t[u_t(a_t, \ldots, a_{T-1})]$. The *value* of being in a particular state x at any time t is defined as the expected utility given optimal actions for all future time points. It is expressed by the value function, $J_t^*(x)$, which may be found recursively: For the terminal time point, T, $J_T^*(x) = R(x)$ and

$$J_t^*(x) = \max_{a \in A} E[r(X_{t+1}, x, a) + \alpha J_{t+1}^*(X_{t+1}) | X_t = x, a_t = a] \tag{1}$$

for $t = T-1, \ldots, 1$. An action that maximizes the expression on the right of (1) is an optimal action. The recursive relation in (1) may be expressed in terms of the *Bellman operator* Λ that maps J_{t+1} to J_t. Thus (1) may be rewritten conveniently as $J_t^* = \Lambda J_{t+1}^*$.

In infinite-horizon problems ($T = \infty$), we will assume that $\alpha < 1$ to ensure the finiteness of the expected utility $u_t(a_t, \ldots, a_{T-1})$. The value of being in x is independent of t in this case because we are always facing an infinite planning horizon, and the value function can be defined as the unique solution of the fixed-point equation $J^* = \Lambda J^*$ [3]. Intuitively, $J^*(x)$ equals the expected utility of being in x at any time and choosing optimal actions at any future state as in the finite-horizon case. In the rest of the paper, when there is no danger of confusion, we will sometimes drop the time subscript and use the symbol J^* to denote either J_t^* or J^*.

Within the above framework, reinforcement learning is concerned with the identification of good actions or strategies in the absence of explicit knowledge of the transition density $p(y|x, a)$. For this purpose it is convenient to introduce the *action-value function*

$$Q_{t,a}^*(x) = E_t[r(X_{t+1}, x, a) + \alpha J_{t+1}^*(X_{t+1}) | X_t = x, a_t = a]. \tag{2}$$

$Q_{t,a}^*$ is the value of taking action a at time t and optimal actions in all future times. Definition (2) can be abbreviated by using an operator Γ_a that maps J_{t+1}^* to $Q_{t,a}^*$; i.e. $Q_{t,a}^* = \Gamma_a J_{t+1}^*(x) = E[r(X_{t+1}, x, a) + \alpha J_{t+1}^*(X_{t+1}) | X_t = x, a_t = a]$.

Estimates of $Q^*_{t,a}(x)$ may be computed by stochastic approximation of the parameters in a suitable model of the action-value function Q, for example, in a neural network architecture [22, 34]. Unfortunately, proofs for the convergence of the resulting algorithms can only be obtained in special cases [6, 14, 32]. Furthermore, these estimates are typically *inconsistent* in the sense that they do not necessarily converge to the true Q^* as the number of samples grows towards infinity.

We suggest an alternative approach to approximating the unknown Γ_a using a random operator $\hat{\Gamma}_a$ based on historic realizations of outcomes given action a. Define $Q_t = \{Q_{t,a} : a \in A\}$ and $Q = \{Q_a : a \in A\}$, i.e. the set of all action value functions. Let $\mathcal{T}$ be the maximum operator over all actions defined as $\mathcal{T}Q_a(x) = \max_a Q_a(x)$. Then $J_t = \mathcal{T}Q_t$ and $J = \mathcal{T}Q$. Using definition (2), the Bellman equations $J^*_t = \Lambda J^*_{t+1}$ and $J^* = \Lambda J^*$ can be written as $Q^*_{t,a} = \Gamma_a \mathcal{T} Q^*_{t+1,a}$ and $Q^* = \Gamma_a \mathcal{T} Q^*$, respectively. Formally, we can thus simply substitute the approximation $\hat{\Gamma}_a$ for Γ_a to obtain a new set of *approximate Bellman equations*, $\hat{Q}_t = \hat{\Gamma}_a \mathcal{T} \hat{Q}_{t+1}$ and $\hat{Q} = \hat{\Gamma}_a \mathcal{T} \hat{Q}$. We can now define a reinforcement learning algorithm as a method to solve these equations. Of course, the existence of such solutions depends on the definition of the approximate expectation operator $\hat{\Gamma}_a$. Below we will show that a suitable definition can be found by kernel-based averaging. This approximation guarantees a unique solution and is statistically consistent.

3. Kernel-Based Reinforcement Learning

In this section, we describe the construction of the approximate expectation operator using kernel-based averaging and we illustrate details of the resulting learning algorithm. To motivate the idea of local averaging, we will first consider a simple case. Suppose the reward and the value function are constant on a finite number of partitions $B_1, \ldots, B_N$ of $[0,1]^d$. If there are a sufficient number of historical realizations of the system available, then we can approximate the transition probabilities between partitions, $p_{i,j}(a) = P(X_{t+1} \in B_j | X_t \in B_i, a_t = a)$, arbitrarily well. We can now think of $p_{i,j}(a)$ the transition probability in a new, discrete MDP with the states $i = 1, \ldots, N$ whose transition probability matrix is known. Hence, the usual contraction arguments to demonstrate the convergence of dynamic programming apply to this approximation [21, 4]. In principle, we may thus recover the value function of an arbitrary MDP by using finer and finer partitions of the state space which corresponds to the construction of a sequence of piecewise constant approximations of J^*. Practically, it is well-known that piecewise constant function estimates are typically inferior to more elaborate "smoothing" methods [10]. Suppose S^a is a collection of m_a historical state transitions from x_s to y^a_s, generated using action a. To apply smoothing to reinforcement learning, we replace the events B_i with "fuzzy events" defined by a "membership function" or a *weighting kernel* $k_{S^a,b}(x_s, x)$, centered at x. Forming conditional probability estimates by analogy to the piecewise constant case and taking expectations leads us to define the following random approximation of Γ_a:

$$\hat{\Gamma}_a J(x) = \sum_{(x_s, y_s^a) \in S^a} k_{S^a,b}(x_s, x)[r(y_s^a, x_s, a) + \alpha J(y_s^a)]. \tag{3}$$

S^a=$\{(x_s, y_s^a)|s = 1, \ldots, m_a\}$ is a collection of historical transitions where action a was taken. The state of the system when action a was taken is given by the first component, x_s. The second component, y_s^a, is the state of the system after taking the action and is distributed according to $P_a(x_s, B)$. There is a separate training data set for each action. The weighting function $k_{S^a,b}(x_s, x)$ depends on the training set S^a, x and a "mother kernel" function ϕ^+ and is given by:

$$k_{S^a,b}(x_s, x) = \phi^+\left(\frac{||x_s - x||}{b}\right) \Big/ \sum_{(x_u, y_u^a) \in S^a} \phi^+\left(\frac{||x_u - x||}{b}\right). \tag{4}$$

This weighting scheme gives equal weight to equidistant points. The magnitude of the weight as a function of distance is governed by the univariate, non-negative mother kernel function ϕ^+. By construction, the weights are designed to be positive and add up to unity; this fact plays an important role in the convergence properties of the learning algorithm. Formal details of this construction and of the sample data generating process are described in the assumptions in Appendix A.1. For concreteness, the reader may wish to think of the mother kernel function $\phi^+(z)$ as a univariate normal density. The degree of "smoothing" can be controlled by the *bandwidth* parameter, b, corresponds the "standard deviation" of this normal density. Even though definition (4) depends on b and (implicitly) on the action a, we will omit the corresponding subscripts for simplicity.

Recalling our notation from the previous section, the expression for $\hat{\Gamma}_a J(x)$ in (3) can be interpreted as an approximation of the action-value function, $Q(a, x)$, that is associated with the value function $J(x)$. Intuitively, (4) assesses the value of applying action a in state x by looking at all times in the training data where a has been applied previously in a state similar to x, and by averaging the immediate rewards and the value estimates at the outcomes of these previous transitions. Because the weights $k(x_s, x)$ are related inversely to the distance $||x_s - x||$, transitions originating in the neighborhood of x are most influential in this averaging procedure. A more statistical interpretation of (4) would suggest that ideally we could simply generate a large number of independent samples from the conditional distribution $P_a(x, \cdot)$ and estimate (2) using Monte-Carlo approximation. Practically speaking, this approach is infeasible even if we can generate samples from $P_a(x, \cdot)$: First, we may not have sufficient control over the experiment to generate the samples using arbitrary starting points. Second, in order to assess the value of the simulated successor states we would need to sample recursively, thereby incurring exponentially increasing computational complexity. A more realistic alternative is to estimate $\hat{\Gamma}_a J(x)$ as a local average of the rewards that were generated in previous transitions originating in the neighborhood of x, where the membership of an observation x_s in the neighborhood of x is quantified using $k(x_s, x)$. For a smooth value function J, the value of $\Gamma_a J$ at location x should be similar to the value of $\Gamma_a J$ in the neighborhood of x, resulting in little loss of precision using this approximation.

More precisely, the amount of local averaging has to be controlled carefully using the bandwidth parameter. A discussion of the resulting bias-variance dilemma follows in Section 5.

We would like to mention that (4) is by no means the only way to specify the weighting function $k(x_s, x)$. In [17] the authors discuss several alternatives including nearest-neighbor regression, grid-based approximations, and trees. Yet another interesting possibility is to use locally weighted regression in place of the local averaging rule (4). Practical applications of this idea are described in [25]. Locally weighted regression can be shown to eliminate much of the bias at the boundaries of the state-space and it is sometimes believed to lead to superior performance in regression problems [12, 18]. From a mathematical perspective, it is well-known that locally weighted regression can be interpreted as a special case of local averaging using the notion of "equivalent kernels" [10]. However, local regression estimates need to be suitably constrained in practice to guarantee the positivity of the equivalent weights. One possibility to achieve this is by using regularization.

4. Approximate Dynamic Programming

We mentioned previously that the main application of the random operator $\hat{\Gamma}_a$ is to derive "plug-in" estimates of the functions $J^*(x)$ and $Q^*(a,x)$ by solving the approximate Bellman equations $\hat{Q}_t = \hat{\Gamma}_a \mathcal{T} \hat{Q}_{t+1}$ and $\hat{Q} = \hat{\Gamma}_a \mathcal{T} \hat{Q}$. These equation can be written alternatively in terms of the approximate value function as $\hat{J}_t = \mathcal{T} \hat{\Gamma}_a \hat{J}_{t+1}$ and $\hat{J} = \mathcal{T} \hat{\Gamma}_a \hat{J}$ where we use (3) as the approximate expectation operator. For finite-horizon problems, this approximate Bellman equation is readily solved by iterating backwards from the known terminal condition $\hat{J}_T(X_T) = R(X_T)$. This procedure is analogous to the value iteration algorithm in dynamic programming. With regard to a practical implementation note that, as J is only evaluated at the locations y_s^a on the right side of (3), backward iteration does not require storage of an explicit representation of the value function $\hat{J}_{t+1}(x)$ for *all* values of x; instead, only the values $\hat{J}_{t+1}(y_s^a)$ are needed. Similarly, in infinite-horizon problems we compute a solution to $\hat{J} = \mathcal{T} \hat{\Gamma}_a \hat{J}$ by using the infinite-horizon version of value iteration. It is helpful to note that $\hat{\Gamma}_a$ is "self-approximating" in the sense that in order to characterize $\hat{J}$ it suffices to find a set of function values at the locations y_s^a satisfying $\hat{J}(y_s^a) = (\mathcal{T} \hat{\Gamma}_a \hat{J})(y_s^a)$ (see also [23]). Then the value of $\hat{\Gamma}_a J(x)$ at new locations $x \neq y_s^a$ can be derived directly from the definition of $\hat{\Gamma}_a$ in (3). Below we assume for simplicity that we have an equal number of observations in S^a for all actions, i.e. $m_a = m$ for all $a \in A$. Hence the value iteration update rule – restricted to the locations y_s^a – can be written compactly using matrix notation:

$$\tilde{J}' := \tilde{\mathcal{T}}(\Theta[\tilde{R} + \alpha \tilde{J}]). \tag{5}$$

Here $\tilde{R}$ is a $m \times M$ matrix with entry $r(y_s^a, x_s, a)$ at location (s,a), Θ is a $m \times M \times m$ tensor with entry $k(x_s, y_{s'}^a)$ at location (s', a, s), and $\tilde{\mathcal{T}}$ is an operator that takes a $m \times M \times M$ tensor and maximizes over its second dimension. The old and new value function estimates , $\tilde{J}$ and $\tilde{J}'$, are matrices of dimensionality $m \times M$. Using the fact that the weights in Θ can essentially be interpreted as probabilities, it is easy to prove the following theorem:

THEOREM 1 *The approximate value iteration (5) converges to a unique fixed point.*

The proof of this theorem is provided in Appendix A.2. Computationally, a single application of (5) requires $O(m^2 \times M)$ operations in a "naive" matrix implementation. Substantial improvements can be obtained by defining the weighting function such that it is zero outside a fixed range. Using a nearest neighbor kernel, for example, the weight matrix Θ is sparse and the complexity of (5) reduces to $O(l \times m \times M)$ where l is the number of neighbors [9]. This computational requirement is sufficiently small to accommodate many real-world applications where the number of observations is fixed. For online problems, however, we require an algorithm whose computational complexity is independent of the number of observations m. In the case of temporal-difference learning, this is achieved typically by using a parametric approximation of the value function with a fixed number of parameters. A comparable strategy in the context of kernel-based reinforcement learning would be to discard old observations or to summarize clusters of data using "sufficient" statistics. Note that the convergence property in Theorem 1 remains unaffected by such an approximation.

5. Consistency and Optimal Bandwidth Selection

Above we described a reinforcement learning algorithm that uses training data to derive estimates of the value function $J^*(x)$ and of the action-value function $Q^*(a, x)$. From a statistical perspective, a minimum requirement for such an algorithm is that additional training data should always improve the quality of the approximation and eventually lead to optimal performance. More formally, reinforcement learning should be consistent in the sense that as the number of observed transitions m goes to infinity, the solutions of $Q_t = \hat{\Gamma}_a \mathcal{T} Q_{t+1}$ and $Q = \hat{\Gamma}_a \mathcal{T} Q$ should deviate arbitrarily little from the true Q_t^* and Q^*, and the actions generated by $\hat{\Gamma}_a$ should be optimal asymptotically. While consistency is highly desirable, it is hard to establish using parametric approximations of the value function such as neural networks. Partly, this is due to the previously mentioned convergence problems of temporal-difference learning which make a formal characterization of the solution difficult if not meaningless in many cases. As a consequence, previous results in this spirit are limited to restricted families of value functions, such as linear combinations of basis functions in optimal stopping problems [32]. In the case of kernel-based reinforcement learning, on the other hand, the value function estimate is uniquely defined as the solution of the approximate Bellman equation. Therefore, a rich body of convergence results for kernel smoothers can be brought to bear upon kernel-based reinforcement learning (for an overview, see [9]).

As the basis for the results in this section we make the following simplifying assumptions, formally listed in Appendix A.1: First, we mentioned above that the very reason for averaging over neighboring transitions is the belief that the functions r and J^* are smooth. Formally, this smoothness is expressed in terms of Lipschitz continuity in Assumption 1. Second, to obtain accurate approximations over the entire state space, we draw the starting points of the sample transitions (x_s, y_s^a) in S^a independently according to a uniform distribution (see Assumption 3). Of course, this assumption is unrealistic for many real systems. A proper account of weaker assumptions that would lead to identical results is beyond the scope of this paper and therefore left for future work. The d-dimensional weighting function$k(x_s, x)$ is assumed to be defined in terms of a univariate "mother kernel" as described in Section 3 (see also Assumption 4).

We first establish the asymptotic normality of the kernel-based approximation. Even though this property is used only indirectly in our consistency proof below, it will turn out to be convenient for the bias analysis of kernel-based reinforcement learning in Section 6.

LEMMA 1 *For any Lipschitz continuous element J of $C([0,1]^d)$ and any fixed bandwidth $b > 0$, the sequence $\sqrt{m}(\hat{\Gamma}_a J(x) - E[\hat{\Gamma}_a J(x)])$ converges in distribution to a Gaussian process on $C([b, 1-b]^d)$.*

The proof of Lemma 1, which is analogous to Rust's proof of a corresponding theorem regarding density-based random operators [23], can be found in Appendix A.2. Note that we restrict ourselves to the interval $[b, 1-b]$ to avoid boundary effects of the weighting kernel. Below we argue that smaller and smaller bandwidths should be used as m increases, such that $[b, 1-b]$ converges to $[0, 1]$. Note also that throughout this section we explicitly require that the number of observations generated using *each* action goes to infinity.

We proceed in several steps to demonstrate the consistency of the value function estimates $\hat{J}_t$ and $\hat{J}$. In our first step, we establish the consistency of the random operator $\hat{\Gamma}_a J$ if applied to an arbitrary fixed function J. For this purpose, we decompose the approximation error $\hat{\Gamma}_a J - \Gamma_a J$ into a *bias* term, $E[\hat{\Gamma}_a J(x)] - \Gamma_a J(x)$ and a *variance* term, $\hat{\Gamma}_a J - E[\hat{\Gamma}_a J(x)]$. Note that these two terms depend on the bandwidth parameter b in opposite fashions: A small bandwidth reduces the size of the neighborhood used for averaging and hence increases the variance. Simultaneously, because the observations in the reduced neighborhood are closer to x, a small bandwidth also reduces the bias of the estimate. As a result, we face a bias-variance tradeoff in that b must be chosen such as to balance the contradictory tasks of minimizing the bias and minimizing the variance. In order for both of these terms to disappear asymptotically, we need to "shrink" the weighting kernel with increasing sample size at an appropriate rate. Lemma 2 establishes formal conditions on "shrinkage rates" that lead to a consistent estimate of $\Gamma_a J$.

LEMMA 2 *A shrinkage rate $b(m)$ is called "admissible" if for any Lipschitz continuous element J of $C([0,1]^d)$, the random operator $\hat{\Gamma}_a J$ satisfies:*

$$||\hat{\Gamma}_a J - \Gamma_a J||_\infty \xrightarrow{P} 0 \text{ as } m \to \infty.$$

b is admissible if it satisfies $b^{d+1}\sqrt{m} \to \infty$ and $b \to 0$ as $m \to \infty$.

The "admissibility" condition of Lemma 2 ensures that the bandwidth decreases to zero slowly enough so as not to cause an undesired increase in variance. Intuitively, more and more data need to be taken into account as they arrive by a smaller and smaller kernel. Given the finiteness of the action space A, it is simple to show that the consistency of $\hat{\Gamma}_a J$ implies the consistency of $\hat{\Lambda}J = \mathcal{T}\hat{\Gamma}_a J$ in our second step:

LEMMA 3 *For any admissible shrinkage rate $b(m)$ and any Lipschitz continuous element J of $C([0,1]^d)$, $\hat{\Lambda}J$ satisfies:*

$$||\hat{\Lambda}J - \Lambda J||_\infty \xrightarrow{P} 0 \text{ as } m \to \infty.$$

Next, we demonstrate that consistency carries over from a single application of $\hat{\Lambda}$ to a fixed J to the estimates $\hat{J}_t$ and $\hat{J}$, obtained as the result of the iteration procedure described in Section 3. It follows the main result of this section:

THEOREM 2 *Let $b(m)$ be an admissible shrinkage rate used to evaluate the random operator $\hat{\Gamma}_a$. In the finite-horizon case, let $\hat{J}_t$ denote the approximation of J_t^* that is obtained using the iteration $\hat{J}_t = \mathcal{T}\hat{\Gamma}_a \hat{J}_{t+1}$. Then we have*

$$||\hat{J}_t - J_t^*||_\infty \xrightarrow{P} 0 \text{ as } m \to \infty.$$

In the infinite-horizon case, let $\hat{J}$ be the fixed point of the approximate Bellman's equation $J' = \mathcal{T}\hat{\Gamma}_a J$. Then we have

$$||\hat{J} - J^*||_\infty \xrightarrow{P} 0 \text{ as } m \to \infty.$$

Theorem 2 confirms that kernel-based reinforcement learning consistently estimates the true value function if an admissible shrinkage rate is chosen. Given the relatively broad class of admissible shrinkage strategies identified in Lemma 2, it is natural to ask whether there exists any one particular strategy that is optimal in that it leads to the fastest possible convergence. Theorem 3 provides an affirmative answer to this question, and characterizes the optimal shrinkage rate as well as the corresponding convergence rate.

THEOREM 3 *The optimal convergence rate that may be obtained using a shrinking kernel is $O_P(m^{-\frac{1}{2(d+2)}})$. The optimal shrinkage rate is $O(m^{-\frac{1}{2(d+2)}})$.*

This result follows from the error bounds (A.1) and (A.2) in the Proof of Lemma 2. Besides guiding the practical choice of b, Theorem 3 provides insight into the question of whether simulation-based methods are suited to "break" the curse of dimensionality. This question is the main focus of Rust's analysis of density-based random approximations of the Bellman operator [23]. In particular, Rust concludes that if the transition density $p(y|x,a)$ is known, an algorithm similar to ours can be used to approximate the value function in a computation time that is only polynomial in d. By contrast, Theorem 3 suggests that for kernel-based reinforcement learning, where $p(y|x,a)$ is unknown, the number of observations grows exponentially in d even if b is chosen optimally. Maybe little surprisingly, knowledge of the transition density – if available – can thus be used to improve the approximation of the value function estimate dramatically in high-dimensional problems.

While our focus above was on the asymptotic identification of the value function, it is frequently more important in practice to ensure that the actual strategy derived using this estimate is near-optimal for sufficiently large samples. We conclude this section with a theorem that formally establishes this property:

THEOREM 4 *By using the random operator $\hat{\Gamma}_a \hat{J}(x)$ and an admissible shrinkage rate to approximate the true action-value function, $\Gamma_a J^*(x)$, the probability of choosing a suboptimal action converges to zero as the number of samples in each data set S^a goes to infinity.*

6. Bias-Variance Tradeoff in Reinforcement Learning

In Section 5 we proved the consistency of kernel-based reinforcement learning with regard to estimates of the value function and of the optimal strategy. In this section, we investigate more closely the bias-variance tradeoff for finite samples. As mentioned in Section 5, bias arises in small samples due to the nonzero bandwidth of the averaging kernel, b, which leads to a "blurring" of the function values in the neighborhood defined by b. This bias effect is typical for kernel-based methods and not specific to the reinforcement learning problem. By contrast to alternative applications of kernel smoothers, however, there is also a second bias term which arises from the fact that in the approximate Bellman's equation $\hat{J} = \mathcal{T}\hat{\Gamma}_a \hat{J}$ we apply a maximum operator to a random estimate of the true expectation Γ_a. This estimate is clearly biased upward because the expectation of the maximum of a random function generally exceeds the maximum of the expectation by the convexity of the maximum operation and Jensens's inequality. We formulate this important finding in the following lemma:

LEMMA 4 *If $\hat{\Gamma}_a J(x)$ is an unbiased estimate of $\Gamma_a J(x)$ then $E[\mathcal{T}\hat{\Gamma}_a J(x)] \geq \Lambda J(x)$.*

Note that this undesired artifact is not specific to the kernel-based method but it affects all reinforcement learning methods that involve estimates of the value function. In the case of kernel-based learning, however, we can exploit our knowledge of the asymptotic distribution of the random approximation in order to quantify the magnitude of the second bias term. Statistically speaking, the biased estimate can be characterized using "order statistics", and the distributions of these statistics lead to an analytic expression for the (asymptotic) bias:

THEOREM 5 *For any Lipschitz continuous element J of $C([0,1]^d)$, the asymptotic bias of the second type resulting from a single application of $\hat{\Lambda}$ to J, defined as $E[\hat{\Lambda} J(x)] - \mathcal{T} E[\hat{\Gamma}_a J(x)]$, equals*

$$-\sum_{i=1}^{M} (\bar{\mu} - \mu_i) P(U_i \geq \bar{U}_{-i}) + \sqrt{\frac{\sigma_i^2 + \sigma_j^2}{2\pi}} \sum_{j \neq i} e^{-\frac{1}{2}\tilde{\mu}_{i,j}} P(\tilde{U}_{i,j} \geq \bar{U}_{-i,j}), \tag{6}$$

where we use the definitions $\bar{\mu} = \max\{\mu_1, \ldots, \mu_n\}$,

$$\begin{array}{ll} \mu_i = E[\hat{\Gamma}_i J(x)], & \sigma_i^2 = Var[\hat{\Gamma}_i J(x)], \\ U_i = \hat{\Gamma}_i J(x), & \bar{U} = \max\{U_1, \ldots, U_n\}, \\ \bar{U}_{-i} = \max\{U_1, \ldots, U_n\} \setminus \{U_i\}, & \bar{U}_{-i,j} = \max\{U_1, \ldots, U_n\} \setminus \{U_i, U_j\}, \\ \tilde{U}_{i,j} = \frac{\sigma_j^2 U_i + \sigma_i^2 U_j}{\sigma_i^2 + \sigma_j^2}, & \tilde{\mu}_{i,j} = \frac{\sigma_i^2 \sigma_j^2 (\mu_i - \mu_j)^2}{(\sigma_i^2 + \sigma_j^2)^2}. \end{array}$$

Here the first term in (6) is non-positive and accounts for the possibility that sometimes the observed maximum may be generated from a suboptimal action. The second term depends on the "separability" of pairs of actions i and j. Intuitively, the closer the values μ_i and μ_j are to each other (relative to their respective variances σ_i^2 and σ_j^2), the greater is the bias that results from observing $U_i > U_j$ even though in reality $\mu_i < \mu_j$. Interestingly, formula (6) indicates that in reinforcement learning the bias-variance tradeoff with regard to the optimal kernel width is quite different from that in regression, because the variance term σ_i^2 shows up in the bias terms of both the first and second types. Based on an estimate of σ_i^2, (6) could also serve to construct a bias-correction, where a carefully chosen function of $U_1, \ldots, U_n$ – say a fraction of the bias estimate resulting from (6) – would be subtracted from $\hat{\Lambda} J(x)$. Note, however, that bias-adjustments of this sort frequently lead to new instabilities because they increase variance of the estimate to an undesired level. The exact nature of this tradeoff in reinforcement learning is unknown and would be an interesting subject for future research.

7. Conclusions

We have presented a new, kernel-based reinforcement learning method that overcomes important shortcomings of temporal-difference learning in continuous-state domains. Our method uses locally weighted averaging to assess the value of a particular state x using historic observations of transitions originating in the neighborhood of x. The resulting random operator is self-approximating in the sense that its fixed point may be characterized as the solution of a simple matrix equation. This equation may be solved using an iterative algorithm. Our main theorem established that kernel-based reinforcement learning consistently approximates the true value function of the MDP provided the kernel bandwidth is decreased appropriately with increasing sample size. We also provided an asymptotic bias formula that characterizes the bias-variance tradeoff in reinforcement learning. Our results were derived under the assumption that elements in the transition set are independent and that the first component was drawn uniformly from a unit hypercube. We believe these assumptions may be relaxed in future work.

Practically speaking, the performance of our approximation – and of any other method – is dictated by the amount of training data and by the available computational resources. Neither our method nor any other reinforcement learning algorithm that does not incorporate prior knowledge can "break" the "curse-of-dimensionality". This is because the lower bounds for the complexity of non-linear regression are exponential [26] and reinforcement learning is at least as hard as nonlinear regression (an arbitrary regression problem can be reduced to a trivial one-step MDP; see also [23]). However we note that this "curse" is accompanied by the "blessing" of not having to know the transition density of the underlying MDP. Moreover, for a number stochastic control problems the state-space may be satisfactorily described by using a low-dimensional set of features. We therefore believe that kernel-based reinforcement learning, because of its stability, may be a valuable tool to tackle many challenging problems.

8. Acknowledgments

The work of Dirk Ormoneit is supported by a grant of the Deutsche Forschungsgemeinschaft (DFG) as part of its post-doctoral program. Śaunak Sen would like to thank Jun Liu, and the NSF grant DMS-9803649 for support. Carrie Grimes did her best to make the reading of this paper more pleasant by pointing us to misleading formulations in earlier drafts. Any remaining mistakes are, of course, our own responsibility.

Appendix

A.1. Assumptions

1. *The reward function, $r(y,z,a)$ is a jointly Lipschitz continuous function of y and z for all $a \in A$, i.e. there exists a $K_r > 0$ such that*

$$|r(y',z',a) - r(y,z,a)| \leq K_r ||(y'-y, z'-z)||.$$

2. *The conditional distribution of X_{t+1} given $X_t{=}x$ and $a_t{=}a$, $P_a(x,\cdot)$, is homogeneous for all t and has a density function $p(y|x,a)$ with respect to the Lebesgue measure λ. The mean and the covariance matrix of $p(y|x,a)$ are finite.*

3. *For each action a, we define a sample data set S^a as $\{(x_s, y_s^a) | s = 1, \ldots, m_a\}$; for simplicity, let $m_a = m$ for all $a \in A$. Each element in this set, (x_s, y_s^a), is an independent draw from the distribution of (Z,Y). Here the first component, Z, is distributed uniformly on $[0,1]^d$ and the second component, Y, follows the conditional distribution $P_a(x_s,\cdot)$.*

4. *Given the sample set S^a, the weighting kernel $k(x_s,x) = k_{S^a,b}(x_s,x)$ is constructed using a Lipschitz continuous "mother kernel", $\phi : [0,1] \to I\!\!R^+$, satisfying $\int_0^1 \phi(z)dz = 1$. For each $x \in [0,1]^d$ and each $(x_s, y_s^a) \in S^a$, $k_{S^a,b}(x_s,x)$ is defined according to (4), where $\phi^+(z)$ is the completion of $\phi(z)$ on $I\!\!R$.*

A.2. Proofs of Theorems and Lemmata

Proof of Theorem 1:

In the finite-horizon case, the statement of the theorem is straightforward because the update (5) is deterministic and it is carried out exactly T times. In the infinite-horizon case, $\hat{\Lambda}$ defines a contraction mapping on the Banach space formed by $C([0,1]^d)$ and the supremum norm:

$$\begin{aligned}||\hat{\Lambda}J - \hat{\Lambda}J'||_\infty &= \sup_{x\in[0,1]^d}\left(\max_{a\in A}\hat{\Gamma}_a J(x) - \max_{a\in A}\hat{\Gamma}_a J'(x)\right)\\ &\leq \alpha \sup_{x\in[0,1]^d}\max_{a\in A}\left[\sum_s k(x_s,x)(J(y_s^a) - J'(y_s^a))\right]\\ &\leq \alpha||J-J'||_\infty.\end{aligned}$$

Hence we can appeal to the Banach Fixed-Point theorem to establish convergence and the uniqueness of the solution (see, for example, [21]). ■

Proof of Lemma 1: Let the "asymptotic weighting kernel" $\tilde{k}(Z,x)$ be defined as $\tilde{k}(z,x)=\frac{\phi^+(||\frac{z-x}{b}||)}{\int\phi^+(||\frac{u-x}{b}||)\lambda(du)}$. Note that $mk(z,x)$ converges to $\tilde{k}(z,x)$ uniformly almost surely as m goes to infinity. We will show that the random function $\tilde{\xi}_a(x) = \xi_a(x) - E[\xi_a(x)]$ satisfies a Central Limit Theorem by applying a Theorem 1 of [13], where

$$\xi_a(x) = \tilde{k}(Z,x)[r(Y,Z,a) + \alpha J(Y)].$$

Then it follows that, $1/\sqrt{m}\sum_{s=1}^m \tilde{\xi}_{a,s}(x) = \sqrt{m}(\hat{\Gamma}_a J(x) - E[\hat{\Gamma}_a J(x)])$ converges in distribution to a Gaussian measure on $C([b,1-b]^d)$ as $m\to\infty$. We now proceed to verify the assumptions in [13].

Since the expectation operator is linear, $E[F\tilde{\xi}_a] = 0$ for arbitrary linear functionals F. Since $\xi_a(x)$ is Lipschitz continuous and hence bounded, the unconditional expectation $E[\xi_a(x)^2]$ is bounded. Furthermore, realizations of $\tilde{\xi}_a$ are elements of $C([0,1]^d)$. By application of the triangular and Jensen's inequalities it can be seen that $\tilde{\xi}_a$ satisfies

$$|\tilde{\xi}_a(x,\omega) - \tilde{\xi}_a(x',\omega)| \leq M(\omega)||x-x'||$$

where

$$M(\omega) = K_{\tilde{k}}\left(|\Upsilon^a| + E[|\Upsilon^a|\,|a_t = a]\right).$$

Here $\Upsilon^a = r(Y,Z,a) + \alpha J(Y)$ and $K_{\tilde{k}}$ is the Lipschitz constant of the weighting kernel $\tilde{k}(x_s,\cdot)$. As furthermore the d-dimensional hypercube has finite metric entropy (see, for example, [23]) by Theorem 1 of Jain and Marcus $\tilde{\xi}_a$ satisfies a Central Limit Theorem [13]. ■

Proof of Lemma 2: First, we note that the approximation error of the random operator $\hat{\Gamma}_a$ depends on the "shape" of the mother kernel, the bandwidth parameter b, and the dimension d of the state space. In order to bound this error, we will need the following result regarding the "volume" of the derived weighting kernel $\phi^+(||z-x||/b)$ in d dimensions:

LEMMA 5 (KERNEL VOLUME LEMMA) *Let $C_{b,d}$ be defined according to $C_{b,d}=\int_{[0,1]^d} \phi^+\left(\left|\left|\frac{z-x}{b}\right|\right|\right)$ and let $\nu_i=\int_0^1 z^i\phi(z)dz$. Then for $x \in [b, 1-b]^d$:*

$$C_{b,d} = \frac{b^d 2\pi^{d/2}\nu_{d-1}}{G(d/2)}.$$

Proof: To prevent confusions with the operator Γ_a, we use the symbol $G(\cdot)$ to denote Euler's gamma function.

$$\begin{aligned}
\int_{[0,1]^d} \phi^+\left(\left|\left|\frac{z-x}{b}\right|\right|\right)\lambda(dz) &= b^d \int_{I\!R^d} \phi^+(||z||)\lambda(dz) \\
&= b^d \int_0^\infty \frac{2\pi^{d/2}r^{d-1}}{G(d/2)}\phi^+(r)dr \\
&= \frac{b^d 2\pi^{d/2}}{G(d/2)} \int_0^1 r^{d-1}\phi(r)dr
\end{aligned}$$

■

Second, to establish the consistency of $\hat{\Gamma}_a J$, let $\tilde{k}(z,x)$, m, and $\xi_{a,s}$ be defined as in the previous proof, and let $\bar{\xi}^m_{a,s}(x)=\frac{\xi_{a,s}(x)}{\sum_{s'=1}^m \tilde{k}(Z_{s'},x)}$. By the triangular inequality we have

$$\begin{aligned}
||\hat{\Gamma}_a J - \Gamma_a J||_\infty &\leq ||\hat{\Gamma}_a J - E[\bar{\xi}^m_{a,s}]||_\infty + ||E[\bar{\xi}^m_{a,s}] - \Gamma_a J||_\infty \\
&= V_m + B_m.
\end{aligned}$$

We use the symbol $||\cdot||_\infty$ for the supremum norm on $C([0,1]^d)$, as opposed to the Eucledian norm $||\cdot||$ on $I\!R^d$. Below we will refer to V_m as the "variance term" and to B_m as the "bias term". To establish that $P(||\hat{\Gamma}_a J - \Gamma_a J||_\infty > \delta) \to 0$ for all $\delta > 0$, it suffices to show that

$$V_m \xrightarrow{P} 0 \text{ and } B_m \to 0.$$

In particular, note that V_m is a random variable whereas B_m is a non-stochastic function of m. We discuss separately the convergence of V_m and B_m:

(i): $V_m \xrightarrow{P} 0$:

In [23], it is shown using an empirical process argument that the expectation of the variance term V_m may be approximated by (Theorem 3.3, Equation (3.19))

$$E\left[\sqrt{m}||\hat{\Gamma}_a J - E[\bar{\xi}^m_{a,s}]||_\infty\right] \leq \sqrt{\frac{\pi}{2}}\alpha[1 + d\sqrt{\pi}C]K_p||J||_\infty.$$

Here Rust's Lipschitz constant for the transition density K_p must be replaced by the Lipschitz constant $K_{\tilde{k}}$ for $\tilde{k}(z,x)$, if considered as a function of z only. In particular, it follows from the definition of the weighting kernel in Assumption 4:

$$\begin{aligned} |\tilde{k}(z,x) - \tilde{k}(z',x)| &\leq \frac{1}{C_{b,d}} \left| \phi\left(\left|\left|\frac{z-x}{b}\right|\right|\right) - \phi\left(\left|\left|\frac{z'-x}{b}\right|\right|\right) \right| \\ &\leq \frac{K_\phi}{C_{b,d}} \left| \frac{||z-x||}{b} - \frac{||z'-x||}{b} \right| \\ &\leq \frac{K_\phi}{bC_{b,d}} ||z-z'||. \end{aligned}$$

Thus, $K_{\tilde{k}} = \frac{K_\phi}{bC_{b,d}}$ where K_ϕ is the Lipschitz constant of the mother kernel and $C_{b,d}$ is the "kernel volume" derived in Lemma 5. Hence, using Markov's inequality:

$$\begin{aligned} P(||\hat{\Gamma}_a J - E[\bar{\xi}^m_{a,s}]||_\infty > \varepsilon) &< \sqrt{\frac{\pi}{2}} \frac{\alpha[1 + d\sqrt{\pi}C]K_{\tilde{k}}||J||_\infty}{\varepsilon\sqrt{m}} \\ &= \sqrt{\frac{\pi}{2}} \frac{\alpha[1 + d\sqrt{\pi}C]K_\phi G(d/2)||J||_\infty}{b^{d+1}\sqrt{m}\varepsilon 2\pi^{d/2}\nu_{d-1}}. \end{aligned} \tag{A.1}$$

Consequently, $V_m \to 0$ provided that $b^{d+1}\sqrt{m} \to \infty$.

(ii): $B_m \to 0$:
Let $h(z)$=$\Gamma_a J(z) = E[r(Y,z,a) + \alpha J(Y)|Z = z]$, and let K_h be the Lipschitz constant of $h(z)$. Then for all $x \in [b, 1-b]^d$:

$$\begin{aligned} ||E[\bar{\xi}^m_{a,s}] - \Gamma_a J||_\infty &= \sup_x \left| \int_{[0,1]^d} \tilde{k}(z,x)(h(z) - h(x))\lambda(dz) \right| \\ &\leq \sup_x \int_{[0,1]^d} \tilde{k}(z,x)|h(z) - h(x)|\lambda(dz) \\ &\leq \sup_x \int_{[0,1]^d} \tilde{k}(z,x)K_h||z-x||\lambda(dz) \\ &= \frac{K_h}{C_{b,d}} \sup_x \int_{[0,1]^d} \phi^+\left(\left|\left|\frac{z-x}{b}\right|\right|\right) ||z-x||\lambda(dz) \\ &= \frac{b^{d+1}K_h}{C_{b,d}} \sup_x \int_{\mathbb{R}^d} \phi^+(||z||)||z||\lambda(dz) \\ &= \frac{b^{d+1}K_h}{C_{b,d}} \frac{2\pi^{d/2}}{G(d/2)} \int_0^1 r^{d-1}\phi(r)rdr \\ &= K_h b \frac{\nu_d}{\nu_{d-1}}. \end{aligned} \tag{A.2}$$

Thus, $B_m \to 0$ provided that $b \to 0$. ∎

Proof of Lemma 3: (see also Lemma 3.1 in [23])

$$\begin{aligned} ||\hat{\Lambda}J - \Lambda J||_\infty &= \sup_{x\in[0,1]^d} \left(\max_{a\in A} \hat{\Gamma}_a J(x) - \max_{a\in A} \Gamma_a J(x) \right) \\ &\leq \sup_{x\in[0,1]^d} \max_{a\in A} \left(\hat{\Gamma}_a J(x) - \Gamma_a J(x) \right) \\ &= \max_{a\in A} ||\hat{\Gamma}_a J - \Gamma_a J||_\infty \end{aligned}$$

■

Proof of Theorem 2:
We need to show that, for any admissible shrinkage strategy, consistency carries over from a single application of $\hat{\Lambda}$ to a fixed J to the estimates $\hat{J}_t$ and $\hat{J}$. In the finite-horizon case, this is straightforward for if each application of $\hat{\Lambda}$ introduces a bounded error term, the error of the repeated application of $\hat{\Lambda}$ must be bounded by the sum of these terms. In the infinite-horizon case, $\hat{\Lambda}$ defines a contraction mapping according to the proof of Theorem 1:

$$||\hat{\Lambda}J - \hat{\Lambda}J'||_\infty \leq \alpha ||J - J'||_\infty.$$

As usual, let $\hat{J} = \hat{\Lambda}\hat{J}$ and $J^* = \Lambda J^*$. An application of the triangle inequality suffices to show that

$$||\hat{J} - J^*||_\infty \leq \alpha ||\hat{J} - J^*||_\infty + ||\hat{\Lambda}J^* - \Lambda J^*||_\infty$$

such that $P(||\hat{J} - J^*|| >_\infty \delta)$ is bounded by $P(||\hat{\Lambda}J - \Lambda J||_\infty > (1-\alpha)\delta)$. ■

Proof of Theorem 3: We consider shrinkage rates of the form $b = m^{-a}$ for $a > 0$. Thus, for a fixed d, the bias term (A.2) is of order $\frac{1}{b^{d+1}\sqrt{m}} = \frac{1}{m^{-a(d+1)+1/2}}$, and the variance term (A.1) grows like $b = \frac{1}{m^a}$. We need to choose a such that $\min\left\{\frac{1}{m^{-a(d+1)+1/2}}, \frac{1}{m^a}\right\}$ is maximal. Clearly, this is the case if $-a(d+1)+1/2 = a$ or $a = \frac{1}{2(d+2)}$. The resulting convergence rate is $\frac{1}{m^a} = m^{-\frac{1}{2(d+2)}}$. ■

Proof of Theorem 4: For simplicity, first consider the case where $\Gamma_a J^*(x)$ and $\hat{\Gamma}_a \hat{J}(x)$ take on different values for all $a \in A$ such that the functions $a^*(x) = \arg\max_{a\in A} \Gamma_a J^*(x)$ and $\hat{a}^m(x) = \arg\max_{a\in A} \hat{\Gamma}_a \hat{J}(x)$ are well-defined. For all $\varepsilon > 0$ we need to find a $N \in I\!N$ independent of x such that $P\left(\hat{a}(x)^m \neq a^*(x)\right) < \varepsilon$ for all $m > N$. For a particular x, let μ_i denote the ith-largest value of $\Gamma_a J^*(x)$, and let $V_i = \hat{\Gamma}_a \hat{J}(x)$ for the same action a. A necessary condition for $\hat{a}^m$ to differ from $a^*(m)$ is that either $|V_1 - \mu_1| \geq (\mu_1 - \mu_2)/2$ or $|V_i - \mu_i| \geq (\mu_1 - \mu_2)/2$ for some $i \neq 1$. Thus

$$P\left(\hat{a}(x)^m \neq a^*(x)\right) \leq \sum_i P\left(|V_i - \mu_i| \geq (\mu_1 - \mu_2)/2\right).$$

Then by Theorem 2 there exist constants N_i independent of x satisfying

$$P\left(|V_i - \mu_i| \geq (\mu_1 - \mu_2)/2\right) < \varepsilon/M$$

for all $m > N_i$. The maximum of these values is the desired constant N.

If either $\Gamma_a J^*(x)$ or $\hat{\Gamma}_a \hat{J}(x)$ takes on identical values for different actions, a similar argument may be used to show that $P(\hat{a}^m(x) \notin \arg\max \Gamma_a J^*(x)) \stackrel{m\to\infty}{\longrightarrow} 0$ where $\arg\max \Gamma_a J^*(x)$ denotes the set of maximizing a-values. ■

Proof of Theorem 5: Because the random variables U_i are independent according to Assumption 3 and asymptotically normal distributed according to Lemma 1, we have

$$P(\bar{U} \leq u) \stackrel{a}{=} P(U_1 \leq u, \ldots, U_n \leq u) = \prod_{i=1}^{n} \Phi\left(\frac{u-\mu_i}{\sigma_i}\right).$$

By the product rule, the density of $\bar{U}$ is of the form

$$p(z) \stackrel{a}{=} \sum_{i=1}^{n} \phi_{\mu_i,\sigma_i^2}(z) \prod_{j\neq i} \Phi\left(\frac{z-\mu_j}{\sigma_j}\right).$$

Here $\phi_{\mu_i,\sigma_i^2}(z)$ denotes the normal density function with mean μ_i and variance σ_i^2 and Φ denotes the standard normal distribution function.

Furthermore,

$$\begin{aligned}
E[\bar{U}] &\stackrel{a}{=} \sum_{i=1}^{n} \int u\phi_{\mu_i,\sigma_i^2}(u) \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du \\
&= \sum_{i=1}^{n} \mu_i \int \phi_{\mu_i,\sigma_i^2}(u) \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du - \sum_{i=1}^{n} \int \phi'_{\mu_i,\sigma_i^2}(u) \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du.
\end{aligned}$$

Here $\int \phi_{\mu_i,\sigma_i^2}(u) \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du = P(U_i \geq \bar{U}_{-i})$. Partial integration gives that

$$\begin{aligned}
\int \phi'_{\mu_i,\sigma_i^2}(u) \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du &= -\int \phi_{\mu_i,\sigma_i^2}(u) \frac{d}{du} \prod_{j\neq i} \Phi\left(\frac{u-\mu_j}{\sigma_j}\right) du \\
&= \sqrt{\frac{\sigma_i^2+\sigma_j^2}{2\pi}} \sum_{j\neq i} e^{-\frac{\sigma_i^2\sigma_j^2(\mu_i-\mu_j)^2}{(\sigma_i^2+\sigma_j^2)^2}} P(\tilde{U}_{i,j} \geq \bar{U}_{-i,j}).
\end{aligned}$$

Substituting back into the formula for $E[\bar{U}]$ gives (6). ■

References

1. C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted regression for control. *Artificial Intelligence Review*, 11(1-5):75–113, 1997.
2. L. C. Baird and A. H. Klopf. Reinforcement learning with high-dimensional, continuous actions. Technical Report WL-TR-93-1147, Wright Laboratory, Wright-Patterson Air Force Base Ohio, 1993.
3. R. E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.
4. D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1 and 2. Athena Scientific, 1995.
5. J. A. Boyan and A. W. Moore. Generalization in reinforcement learning: Safely approximating the value function. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 369–376. The MIT Press, 1995.
6. S. J. Bradtke. Reinforcement learning applied to linear quadratic regulation. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 295–302. Morgan Kaufmann, San Mateo, CA, 1993.
7. M. W. Brandt. Estimating portfolio and consumption choice: A conditional Euler equations approach. *Journal of Finance*, 54(5):1609–1645, 1999.
8. M. E. Connell and P. E. Utgoff. Learning to control a dynamic physical system. In *Sixth National Conference on Artificial Intelligence*, pages 456–460. Morgan Kaufmann, 1987.
9. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, 1996.
10. J. Fan and I. Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman & Hall, 1996.
11. G. Gordon. *Approximate Solutions to Markov Decision Processes*. PhD thesis, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 1999.
12. T. Hastie and C. Loader. Local regression: Autmatic kernel carpentry. *Statistical Science*, 8(2):120–143, 1993.
13. N. C. Jain and M. B. Marcus. Central Limit Theorems for C(S)-valued random variables. *Journal of Functional Analysis*, 19:216–231, 1975.
14. T. Landelius. *Reinforcement Learning and Distributed Local Model Synthesis*. PhD thesis, Linköping University, 1997.
15. F. A. Longstaff and E. S. Schwartz. Valuing American options by simulation: A simple least-squares approach. Technical Report 25-98, Department of Finance, UCLA, 1998.
16. D. Ormoneit and P. Glynn. Kernel-based reinforcement learning in average-cost problems: An application to optimal portfolio choice, 2000. Working Paper, Department of Statistics, Stanford University.
17. D. Ormoneit and P. Glynn. Kernel-based reinforcement learning in average-reward problems. Technical report, Department of Statistics, Stanford University, 2000. In preparation.
18. D. Ormoneit and T. Hastie. Optimal kernel shapes for local linear regression. In S. A. Solla, T. K. Leen, and K-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 540–546. The MIT Press, 2000.
19. V. Papavassiliou and S. Russell. Convergence of reinforcement learning with general function approximators. In *Proceedings IJCAI*, page 974, 1999.
20. J. Peng and R. J. Williams. Efficient learning and planning within the Dyna framework. In *Twelfth International Conference on Machine Learning*, pages 438–446. Morgan Kaufmann, 1995.
21. M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.
22. H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 20:400–407, 1951.
23. J. Rust. Using randomization to break the curse of dimensionality. *Econometrica*, 65(3):487–516, 1997.
24. S. Singh and D. Bertsekas. Reinforcement learning for dynamic channel allocation in cellular telephone systems. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 974. The MIT Press, 1997.

25. W. D. Smart and L. P. Kaebling. Practical reinforcement learning in continuous spaces. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 903–910. Morgan Kaufmann, San Francisco, CA, 2000.
26. C. J. Stone. Optimal global rates of convergence for nonparametric regression. *Annals of Statistics*, 10(4):1040–1053, 1982.
27. R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.
28. R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. A. Solla, T. K. Leen, and K.-R Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. The MIT Press, 2000.
29. G. Tesauro. Neurogammon wins computer olympiad. *Neural Computation*, 1(3):321–323, 1989.
30. S. B. Thrun and K. Möller. Active exploration in dynamic environments. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 531–538. Morgan Kaufmann Publishers, Inc., 1992.
31. J. N. Tsitsiklis and B. Van Roy. Feature-based methods for large-scale dynamic programming. *Machine Learning*, 22:59–94, 1996.
32. J. N. Tsitsiklis and B. Van Roy. Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Transactions on Automatic Control*, 44(10):1840–1851, 1999.
33. J. N. Tsitsiklis V. R. Konda. Actor-critic algorithms. In S. A. Solla, T. K. Leen, and K.-R Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. The MIT Press, 2000.
34. C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.
35. P. J. Werbos. Consistency of HDP applied to a simple reinforcement learning problem. *Neural Networks*, 3:179–189, 1990.