Journal of Artificial Intelligence Research 15 (2001) 319-350 Submitted 9/00; published 11/01

# Infinite-Horizon Policy-Gradient Estimation

**Jonathan Baxter** JBAXTER@WHIZBANG.COM
*WhizBang! Labs.*
*4616 Henry Street Pittsburgh, PA 15213*
**Peter L. Bartlett** BARTLETT@BARNHILLTECHNOLOGIES.COM
*BIOwulf Technologies.*
*2030 Addison Street, Suite 102, Berkeley, CA 94704*

## Abstract

Gradient-based approaches to direct policy search in reinforcement learning have received much recent attention as a means to solve problems of partial observability and to avoid some of the problems associated with policy degradation in value-function methods. In this paper we introduce GPOMDP, a simulation-based algorithm for generating a *biased* estimate of the gradient of the *average reward* in Partially Observable Markov Decision Processes (POMDPs) controlled by parameterized stochastic policies. A similar algorithm was proposed by Kimura, Yamamura, and Kobayashi (1995). The algorithm's chief advantages are that it requires storage of only twice the number of policy parameters, uses one free parameter $\beta \in [0, 1)$ (which has a natural interpretation in terms of bias-variance trade-off), and requires no knowledge of the underlying state. We prove convergence of GPOMDP, and show how the correct choice of the parameter $\beta$ is related to the *mixing time* of the controlled POMDP. We briefly describe extensions of GPOMDP to controlled Markov chains, continuous state, observation and control spaces, multiple-agents, higher-order derivatives, and a version for training stochastic policies with internal states. In a companion paper (Baxter, Bartlett, & Weaver, 2001) we show how the gradient estimates generated by GPOMDP can be used in both a traditional stochastic gradient algorithm and a conjugate-gradient procedure to find local optima of the average reward.

## 1. Introduction

Dynamic Programming is the method of choice for solving problems of decision making under uncertainty (Bertsekas, 1995). However, the application of Dynamic Programming becomes problematic in large or infinite state-spaces, in situations where the system dynamics are unknown, or when the state is only partially observed. In such cases one looks for approximate techniques that rely on simulation, rather than an explicit model, and parametric representations of either the value-function or the policy, rather than exact representations.

Simulation-based methods that rely on a parametric form of the value function tend to go by the name "Reinforcement Learning," and have been extensively studied in the Machine Learning literature (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998). This approach has yielded some remarkable empirical successes in a number of different domains, including learning to play checkers (Samuel, 1959), backgammon (Tesauro, 1992, 1994), and chess (Baxter, Tridgell, & Weaver, 2000), job-shop scheduling (Zhang & Dietterich, 1995) and dynamic channel allocation (Singh & Bertsekas, 1997).

Despite this success, most algorithms for training approximate value functions suffer from the same theoretical flaw: the performance of the greedy policy derived from the approximate value-function is not guaranteed to improve on each iteration, and in fact can be worse than the old policy

©2001 AI Access Foundation and Morgan Kaufmann Publishers. All rights reserved.

by an amount equal to the *maximum* approximation error over all states. This can happen even when the parametric class contains a value function whose corresponding greedy policy is optimal. We illustrate this with a concrete and very simple example in Appendix A.

An alternative approach that circumvents this problem—the approach we pursue here—is to consider a class of *stochastic policies* parameterized by $\theta \in \mathbb{R}^K$, compute the gradient with respect to $\theta$ of the average reward, and then improve the policy by adjusting the parameters in the gradient direction. Note that the policy could be directly parameterized, or it could be generated indirectly from a value function. In the latter case the value-function parameters are the parameters of the policy, but instead of being adjusted to minimize error between the approximate and true value function, the parameters are adjusted to directly improve the performance of the policy generated by the value function.

These "policy-gradient" algorithms have a long history in Operations Research, Statistics, Control Theory, Discrete Event Systems and Machine Learning. Before describing the contribution of the present paper, it seems appropriate to introduce some background material explaining this approach. Readers already familiar with this material may want to skip directly to section 1.2, where the contributions of the present paper are described.

### 1.1 A Brief History of Policy-Gradient Algorithms

For large-scale problems or problems where the system dynamics are unknown, the performance gradient will not be computable in closed form[1]. Thus the challenging aspect of the policy-gradient approach is to find an algorithm for estimating the gradient via *simulation*. Naively, the gradient can be calculated numerically by adjusting each parameter in turn and estimating the effect on performance via simulation (the so-called *crude Monte-Carlo* technique), but that will be prohibitively inefficient for most problems. Somewhat surprisingly, under mild regularity conditions, it turns out that the full gradient can be estimated from a *single* simulation of the system. The technique is called the *score function* or *likelihood ratio* method and appears to have been first proposed in the sixties (Aleksandrov, Sysoyev, & Shemeneva, 1968; Rubinstein, 1969) for computing performance gradients in i.i.d. (independently and identically distributed) processes.

Specifically, suppose $r(X)$ is a performance function that depends on some random variable $X$, and $q(\theta, x)$ is the probability that $X = x$, parameterized by $\theta \in \mathbb{R}^K$. Under mild regularity conditions, the gradient with respect to $\theta$ of the expected performance,

$$\eta(\theta) = \mathbf{E}r(X), \tag{1}$$

may be written

$$\nabla\eta(\theta) = \mathbf{E}r(X)\frac{\nabla q(\theta, X)}{q(\theta, X)}. \tag{2}$$

To see this, rewrite (1) as a sum

$$\eta(\theta) = \sum_x r(x)q(\theta, x),$$

differentiate (one source of the requirement of "mild regularity conditions") to obtain

$$\nabla\eta(\theta) = \sum_x r(x)\nabla q(\theta, x),$$

1. See equation (17) for a closed-form expression for the performance gradient.

rewrite as

$$\nabla\eta(\theta) = \sum_x r(x)\frac{\nabla q(\theta,x)}{q(\theta,x)}q(\theta,x),$$

and observe that this formula is equivalent to (2).

If a simulator is available to generate samples $X$ distributed according to $q(\theta,x)$, then any sequence $X_1, X_2, \ldots, X_N$ generated i.i.d. according to $q(\theta,x)$ gives an unbiased estimate,

$$\hat{\nabla}\eta(\theta) = \frac{1}{N}\sum_{i=1}^{N} r(X_i)\frac{\nabla q(\theta,X_i)}{q(\theta,X_i)}, \tag{3}$$

of $\nabla\eta(\theta)$. By the law of large numbers, $\hat{\nabla}\eta(\theta) \to \nabla\eta(\theta)$ with probability one. The quantity $\nabla q(\theta,X)/q(\theta,X)$ is known as the *likelihood ratio* or *score function* in classical statistics. If the performance function $r(X)$ also depends on $\theta$, then $r(X)\nabla q(\theta,X)/q(\theta,X)$ is replaced by $\nabla r(\theta,X) + r(\theta,X)\nabla q(\theta,X)/q(\theta,X)$ in (2).

### 1.1.1 UNBIASED ESTIMATES OF THE PERFORMANCE GRADIENT FOR REGENERATIVE PROCESSES

Extensions of the likelihood-ratio method to *regenerative processes* (including Markov Decision Processes or MDPs) were given by Glynn (1986, 1990), Glynn and L'Ecuyer (1995) and Reiman and Weiss (1986, 1989), and independently for *episodic* Partially Observable Markov Decision Processes (POMDPs) by Williams (1992), who introduced the REINFORCE algorithm[2]. Here the i.i.d. samples $X$ of the previous section are *sequences* of states $X_0, \ldots, X_T$ (of random length) encountered between visits to some designated recurrent state $i^*$, or sequences of states from some start state to a goal state. In this case $\nabla q(\theta,X)/q(\theta,X)$ can be written as a sum

$$\frac{\nabla q(\theta,X)}{q(\theta,X)} = \sum_{t=0}^{T-1}\frac{\nabla p_{X_tX_{t+1}}(\theta)}{p_{X_tX_{t+1}}(\theta)}, \tag{4}$$

where $p_{X_tX_{t+1}}(\theta)$ is the transition probability from $X_t$ to $X_{t+1}$ given parameters $\theta$. Equation (4) admits a recursive computation over the course of a regenerative cycle of the form $z_0 = 0 \in \mathbb{R}^K$, and after each state transition $X_t \to X_{t+1}$,

$$z_{t+1} = z_t + \frac{\nabla p_{X_tX_{t+1}}(\theta)}{p_{X_tX_{t+1}}(\theta)}, \tag{5}$$

so that each term $r(X)\nabla q(\theta,X)/q(\theta,X)$ in the estimate (3) is of the form[3] $r(X_0,\ldots,X_T)z_T$. If, in addition, $r(X_0,\ldots,X_T)$ can be recursively computed by

$$r(X_0,\ldots,X_{t+1}) = \phi(r(X_0,\ldots,X_t),X_{t+1})$$

for some function $\phi$, then the estimate $r(X_0,\ldots,X_T)z_T$ for each cycle can be computed using storage of only $K+1$ parameters ($K$ for $z_t$ and 1 parameter to update the performance function $r$). Hence, the entire estimate (3) can be computed with storage of only $2K+1$ real parameters, as follows.

2. A *thresholded* version of these algorithms for neuron-like elements was described earlier in Barto, Sutton, and Anderson (1983).
3. The vector $z_T$ is known in reinforcement learning as an *eligibility trace*. This terminology is used in Barto et al. (1983).

**Algorithm 1.1: Policy-Gradient Algorithm for Regenerative Processes.**

1. Set $j = 0$, $r_0 = 0$, $z_0 = 0$, and $\Delta_0 = 0$ ($z_0, \Delta_0 \in \mathbb{R}^K$).
2. For each state transition $X_t \to X_{t+1}$:
   - If the episode is finished (that is, $X_{t+1} = i^*$), set
     $\Delta_{j+1} = \Delta_j + r_t z_t$,
     $j = j + 1$,
     $z_{t+1} = 0$,
     $r_{t+1} = 0$.
   - Otherwise, set
     $z_{t+1} = z_t + \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)}$,
     $r_{t+1} = \phi(r_t, X_{t+1})$.
3. If $j = N$ return $\Delta_N / N$, otherwise goto 2.

Examples of recursive performance functions include the sum of a scalar reward over a cycle, $r(X_0, \ldots, X_T) = \sum_{t=0}^{T} r(X_t)$ where $r(i)$ is a scalar reward associated with state $i$ (this corresponds to $\eta(\theta)$ being the *average reward* multiplied by the expected recurrence time $\mathbf{E}_\theta[T]$); the negative length of the cycle (which can be implemented by assigning a reward of $-1$ to each state, and is used when the task is to mimimize time taken to get to a goal state, since $\eta(\theta)$ in this case is just $-\mathbf{E}_\theta[T]$); the *discounted reward* from the start state, $r(X_0, \ldots, X_T) = \sum_{t=0}^{T} \alpha^t r(X_t)$, where $\alpha \in [0, 1)$ is the discount factor, and so on.

As Williams (1992) pointed out, a further simplification is possible in the case that $r_T = r(X_0, \ldots, X_T)$ is a sum of scalar rewards $r(X_t, t)$ depending on the state and possibly the time $t$ since the starting state (such as $r(X_t, t) = r(X_t)$, or $r(X_t, t) = \alpha^t r(X_t)$ as above). In that case, the update $\Delta$ from a single regenerative cycle may be written as

$$\Delta = \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \left[ \sum_{s=0}^{t} r(X_s, s) + \sum_{s=t+1}^{T} r(X_s, s) \right].$$

Because changes in $p_{X_t X_{t+1}}(\theta)$ have no influence on the rewards $r(X_s, s)$ associated with earlier states ($s \leq t$), we should be able to drop the first term in the parentheses on the right-hand-side and write

$$\Delta = \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \sum_{s=t+1}^{T} r(X_s, s). \tag{6}$$

Although the proof is not entirely trivial, this intuition can indeed be shown to be correct.

Equation (6) allows an even simpler recursive formula for estimating the performance gradient. Set $z_0 = \Delta_0 = 0$, and introduce a new variable $s = 0$. As before, set $z_{t+1} = z_t + \nabla p_{X_t X_{t+1}}(\theta) / p_{X_t X_{t+1}}(\theta)$ and $s = s + 1$ if $X_{t+1} \neq i^*$, or $s = 0$ and $z_{t+1} = 0$ otherwise. But now, on *each iteration*, set $\Delta_{t+1} = r(X_t, s) z_t + \Delta_t$. Then $\Delta_t / t$ is our estimate of $\nabla \eta(\theta)$. Since $\Delta_t$ is updated on every iteration, this suggests that we can do away with $\Delta_t$ altogether and simply update $\theta$ directly: $\theta_{t+1} = \theta_t + \gamma_t r(X_t, s) z_t$, where the $\gamma_t$ are suitable step-sizes[4]. Proving convergence

4. The usual requirements on $\gamma_t$ for convergence of a stochastic gradient algorithm are $\gamma_t > 0$, $\sum_{t=0}^{\infty} \gamma_t = \infty$, and $\sum_{t=0}^{\infty} \gamma_t^2 < \infty$.

of such an algorithm is not as straightforward as normal stochastic gradient algorithms because the updates $r(X_t)z_t$ are not in the gradient direction (in expectation), although the sum of these updates over a regenerative cycle are. Marbach and Tsitsiklis (1998) provide the only convergence proof that we know of, albeit for a slightly different update of the form $\theta_{t+1} = \theta_t + \gamma_t \left[r(X_t, s) - \hat{\eta}(\theta_t)\right] z_t$, where $\hat{\eta}(\theta_t)$ is a moving estimate of the expected performance, and is also updated on-line (this update was first suggested in the context of POMDPs by Jaakkola et al. (1995)).

Marbach and Tsitsiklis (1998) also considered the case of $\theta$-dependent rewards (recall the discussion after (3)), as did Baird and Moore (1999) with their "VAPS" algorithm (*Value And Policy Search*). This last paper contains an interesting insight: through suitable choices of the performance function $r(X_0, \ldots, X_T, \theta)$, one can combine policy-gradient search with approximate value function methods. The resulting algorithms can be viewed as *actor-critic* techniques in the spirit of Barto et al. (1983); the policy is the *actor* and the value function is the *critic*. The primary motivation is to reduce variance in the policy-gradient estimates. Experimental evidence for this phenomenon has been presented by a number of authors, including Barto et al. (1983), Kimura and Kobayashi (1998a), and Baird and Moore (1999). More recent work on this subject includes that of Sutton et al. (2000) and Konda and Tsitsiklis (2000). We discuss the use of VAPS-style updates further in Section 6.2.

So far we have not addressed the question of how the parameterized state-transition probabilities $p_{X_t X_{t+1}}(\theta)$ arise. Of course, they could simply be generated by parameterizing the matrix of transition probabilities directly. Alternatively, in the case of MDPs or POMDPs, state transitions are typically generated by feeding an *observation* $Y_t$ that depends stochastically on the state $X_t$ into a parameterized *stochastic policy*, which selects a *control* $U_t$ at random from a set of available controls (approximate value-function based approaches that generate controls stochastically via some form of lookahead also fall into this category). The distribution over successor states $p_{X_t X_{t+1}}(U_t)$ is then a fixed function of the control. If we denote the probability of control $u_t$ given parameters $\theta$ and observation $y_t$ by $\mu_{u_t}(\theta, y_t)$, then all of the above discussion carries through with $\nabla p_{X_t X_{t+1}}(\theta)/p_{X_t X_{t+1}}(\theta)$ replaced by $\nabla \mu_{U_t}(\theta, Y_t)/\mu_{U_t}(\theta, Y_t)$. In that case, Algorithm 1.1 is precisely Williams' REINFORCE algorithm.

Algorithm 1.1 and the variants above have been extended to cover multiple agents (Peshkin et al., 2000), policies with internal state (Meuleau et al., 1999), and importance sampling methods (Meuleau et al., 2000). We also refer the reader to the work of Rubinstein and Shapiro (1993) and Rubinstein and Melamed (1998) for in-depth analysis of the application of the likelihood-ratio method to Discrete-Event Systems (DES), in particular networks of queues. Also worth mentioning is the large literature on Infinitesimal Perturbation Analysis (IPA), which seeks a similar goal of estimating performance gradients, but operates under more restrictive assumptions than the likelihood-ratio approach; see, for example, Ho and Cao (1991).

### 1.1.2 BIASED ESTIMATES OF THE PERFORMANCE GRADIENT

All the algorithms described in the previous section rely on an identifiable recurrent state $i^*$, either to update the gradient estimate, or in the case of the on-line algorithm, to zero the eligibility trace $z$. This reliance on a recurrent state can be problematic for two main reasons:

1. The *variance* of the algorithms is related to the recurrence time between visits to $i^*$, which will typically grow as the state space grows. Furthermore, the time between visits depends on

the parameters of the policy, and states that are frequently visited for the initial value of the parameters may become very rare as performance improves.

2. In situations of *partial observability* it may be difficult to estimate the underlying states, and therefore to determine when the gradient estimate should be updated, or the eligibility trace zeroed.

If the system is available only through simulation, it seems difficult (if not impossible) to obtain *unbiased* estimates of the gradient direction without access to a recurrent state. Thus, to solve 1 and 2, we must look to *biased* estimates. Two principle techniques for introducing bias have been proposed, both of which may be viewed as artificial truncations of the eligibility trace $z$. The first method takes as a starting point the formula[5] for the eligibility trace at time $t$:

$$z_t = \sum_{s=0}^{t-1} \frac{\nabla p_{X_s X_{s+1}}(\theta)}{p_{X_s X_{s+1}}(\theta)}$$

and simply truncates it at some (fixed, not random) number of terms $n$ looking backwards (Glynn, 1990; Rubinstein, 1991, 1992; Cao & Wan, 1998):

$$z_t(n) := \sum_{s=t-n}^{t-1} \frac{\nabla p_{X_s X_{s+1}}(\theta)}{p_{X_s X_{s+1}}(\theta)}. \tag{7}$$

The eligibility trace $z_t(n)$ is then updated after each transition $X_t \to X_{t+1}$ by

$$z_{t+1}(n) = z_t(n) + \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} - \frac{\nabla p_{X_{t-n} X_{t-n+1}}(\theta)}{p_{X_{t-n} X_{t-n+1}}(\theta)}, \tag{8}$$

and in the case of state-based rewards $r(X_t)$, the estimated gradient direction after $T$ steps is

$$\hat{\nabla}_n \eta(\theta) := \frac{1}{T-n+1} \sum_{t=n}^{T} z_t(n) r(X_t). \tag{9}$$

Unless $n$ exceeds the maximum recurrence time (which is infinite in an ergodic Markov chain), $\hat{\nabla}_n \eta(\theta)$ is a biased estimate of the gradient direction, although as $n \to \infty$, the bias approaches zero. However the *variance* of $\hat{\nabla}_n \eta(\theta)$ diverges in the limit of large $n$. This illustrates a natural trade-off in the selection of the parameter $n$: it should be large enough to ensure the bias is acceptable (the expectation of $\hat{\nabla}_n \eta(\theta)$ should at least be within $90^\circ$ of the true gradient direction), but not so large that the variance is prohibitive. Experimental results by Cao and Wan (1998) illustrate nicely this bias/variance trade-off.

One potential difficulty with this method is that the likelihood ratios $\nabla p_{X_s X_{s+1}}(\theta)/p_{X_s X_{s+1}}(\theta)$ must be remembered for the previous $n$ time steps, requiring storage of $Kn$ parameters. Thus, to obtain small bias, the memory may have to grow without bound. An alternative approach that requires a fixed amount of memory is to *discount* the eligibility trace, rather than truncating it:

$$z_{t+1}(\beta) := \beta z_t(\beta) + \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)}, \tag{10}$$

5. For ease of exposition, we have kept the expression for $z$ in terms of the likelihood ratios $\nabla p_{X_s X_{s+1}}(\theta)/p_{X_s X_{s+1}}(\theta)$ which rely on the availability of the underlying state $X_s$. If $X_s$ is not available, $\nabla p_{X_s X_{s+1}}(\theta)/p_{X_s X_{s+1}}(\theta)$ should be replaced with $\nabla \mu_{U_s}(\theta, Y_s)/\mu_{U_s}(\theta, Y_s)$.

where $z_0(\beta) = 0$ and $\beta \in [0, 1)$ is a discount factor. In this case the estimated gradient direction after $T$ steps is simply

$$\hat{\nabla}_\beta \eta(\theta) := \frac{1}{T} \sum_{t=0}^{T-1} r(X_t) z_t(\beta). \tag{11}$$

This is precisely the estimate we analyze in the present paper. A similar estimate with $r(X_t)z_t(\beta)$ replaced by $(r(X_t) - b)z_t(\beta)$ where $b$ is a *reward baseline* was proposed by Kimura et al. (1995, 1997) and for continuous control by Kimura and Kobayashi (1998b). In fact the use of $(r(X_t) - b)$ in place of $r(X_t)$ does not affect the expectation of the estimates of the algorithm (although judicious choice of the reward baseline $b$ can reduce the variance of the estimates). While the algorithm presented by Kimura et al. (1995) provides estimates of the expectation under the stationary distribution of the gradient of the discounted reward, we will show that these are in fact biased estimates of the gradient of the expected discounted reward. This arises because the stationary distribution itself depends on the parameters. A similar estimate to (11) was also proposed by Marbach and Tsitsiklis (1998), but this time with $r(X_t)z_t(\beta)$ replaced by $(r(X_t) - \hat{\eta}(\theta))z_t(\beta)$, where $\hat{\eta}(\theta)$ is an estimate of the average reward, and with $z_t$ zeroed on visits to an identifiable recurrent state.

As a final note, observe that the eligibility traces $z_t(\beta)$ and $z_t(n)$ defined by (10) and (8) are simply *filtered* versions of the sequence $\nabla p_{X_t X_{t+1}}(\theta)/p_{X_t X_{t+1}}(\theta)$, a first-order, infinite impulse response filter in the case of $z_t(\beta)$ and an $n$-th order, finite impulse response filter in the case of $z_t(n)$. This raises the question, not addressed in this paper, of whether there is an interesting theory of optimal filtering for policy-gradient estimators.

### 1.2 Our Contribution

We describe GPOMDP, a general algorithm based upon (11) for generating a *biased* estimate of the performance gradient $\nabla\eta(\theta)$ in general POMDPs controlled by parameterized stochastic policies. Here $\eta(\theta)$ denotes the *average* reward of the policy with parameters $\theta \in \mathbb{R}^K$. GPOMDP does not rely on access to an underlying recurrent state. Writing $\nabla_\beta\eta(\theta)$ for the expectation of the estimate produced by GPOMDP, we show that $\lim_{\beta\to 1} \nabla_\beta\eta(\theta) = \nabla\eta(\theta)$, and more quantitatively that $\nabla_\beta\eta(\theta)$ is close to the true gradient provided $1/(1-\beta)$ exceeds the *mixing time* of the Markov chain induced by the POMDP[6]. As with the truncated estimate above, the trade-off preventing the setting of $\beta$ arbitrarily close to 1 is that the variance of the algorithm's estimates increase as $\beta$ approaches 1. We prove convergence with probability 1 of GPOMDP for both discrete and continuous observation and control spaces. We present algorithms for both general parameterized Markov chains and POMDPs controlled by parameterized stochastic policies.

There are several extensions to GPOMDP that we have investigated since the first version of this paper was written. We outline these developments briefly in Section 7.

In a companion paper we show how the gradient estimates produced by GPOMDP can be used to perform gradient ascent on the average reward $\eta(\theta)$ (Baxter et al., 2001). We describe both traditional stochastic gradient algorithms, and a conjugate-gradient algorithm that utilizes gradient estimates in a novel way to perform line searches. Experimental results are presented illustrat-

6. The mixing-time result in this paper applies only to Markov chains with distinct eigenvalues. Better estimates of the bias and variance of GPOMDP may be found in Bartlett and Baxter (2001), for more general Markov chains than those treated here, and for more refined notions of the mixing time. Roughly speaking, the variance of GPOMDP grows with $1/(1-\beta)$, while the bias decreases as a function of $1/(1-\beta)$.

ing both the theoretical results of the present paper on a toy problem, and practical aspects of the algorithms on a number of more realistic problems.

## 2. The Reinforcement Learning Problem

We model reinforcement learning as a Markov decision process (MDP) with a finite state space $\mathcal{S} = \{1, \ldots, n\}$, and a stochastic matrix[7] $P = [p_{ij}]$ giving the probability of transition from state $i$ to state $j$. Each state $i$ has an associated reward[8] $r(i)$. The matrix $P$ belongs to a parameterized class of stochastic matrices, $\mathcal{P} := \{P(\theta) : \theta \in \mathbb{R}^K\}$. Denote the Markov chain corresponding to $P(\theta)$ by $M(\theta)$. We assume that these Markov chains and rewards satisfy the following assumptions:

**Assumption 1.** *Each $P(\theta) \in \mathcal{P}$ has a unique stationary distribution $\pi(\theta) := [\pi(\theta, 1), \ldots, \pi(\theta, n)]'$ satisfying the* balance equations

$$\pi'(\theta)P(\theta) = \pi'(\theta) \tag{12}$$

*(throughout $\pi'$ denotes the transpose of $\pi$).*

**Assumption 2.** *The magnitudes of the rewards, $|r(i)|$, are uniformly bounded by $R < \infty$ for all states $i$.*

Assumption 1 ensures that the Markov chain forms a single recurrent class for all parameters $\theta$. Since any finite-state Markov chain always ends up in a recurrent class, and it is the properties of this class that determine the long-term average reward, this assumption is mainly for convenience so that we do not have to include the recurrence class as a quantifier in our theorems. However, when we consider gradient-ascent algorithms Baxter et al. (2001), this assumption becomes more restrictive since it guarantees that the recurrence class cannot change as the parameters are adjusted.

Ordinarily, a discussion of MDPs would not be complete without some mention of the actions available in each state and the space of policies available to the learner. In particular, the parameters $\theta$ would usually determine a policy (either directly or indirectly via a value function), which would then determine the transition probabilities $P(\theta)$. However, for our purposes we do not care *how* the dependence of $P$ on $\theta$ arises, just that it satisfies Assumption 1 (and some differentiability assumptions that we shall meet in the next section). Note also that it is easy to extend this setup to the case where the rewards also depend on the parameters $\theta$ or on the transitions $i \to j$. It is equally straightforward to extend our algorithms and results to these cases. See Section 6.1 for an illustration.

The goal is to find a $\theta \in \mathbb{R}^K$ maximizing the *average reward*:

$$\eta(\theta) := \lim_{T\to\infty} \mathbf{E}_\theta \left[ \frac{1}{T} \sum_{t=0}^{T-1} r(X_t) \middle| X_0 = i \right],$$

where $\mathbf{E}_\theta$ denotes the expectation over all sequences $X_0, X_1, \ldots$, with transitions generated according to $P(\theta)$. Under Assumption 1, $\eta(\theta)$ is independent of the starting state $i$ and is equal to

$$\eta(\theta) = \sum_{i=1}^{n} \pi(\theta, i) r(i) = \pi'(\theta) r, \tag{13}$$

where $r = [r(1), \ldots, r(n)]'$ (Bertsekas, 1995).

7. A *stochastic* matrix $P = [p_{ij}]$ has $p_{ij} \geq 0$ for all $i, j$ and $\sum_{j=1}^{n} p_{ij} = 1$ for all $i$.

8. All the results in the present paper apply to bounded stochastic rewards, in which case $r(i)$ is the expectation of the reward in state $i$.

## 3. Computing the Gradient of the Average Reward

For general MDPs little will be known about the average reward $\eta(\theta)$, hence finding its optimum will be problematic. However, in this section we will see that under general assumptions the gradient $\nabla\eta(\theta)$ exists, and so local optimization of $\eta(\theta)$ is possible.

To ensure the existence of suitable gradients (and the boundedness of certain random variables), we require that the parameterized class of stochastic matrices satisfies the following additional assumption.

**Assumption 3.** *The derivatives,*

$$\nabla P(\theta) := \left[\frac{\partial p_{ij}(\theta)}{\partial\theta_k}\right]_{i,j=1\ldots n;k=1\ldots K}$$

*exist for all $\theta \in \mathbb{R}^K$. The ratios*

$$\left[\frac{\left|\frac{\partial p_{ij}(\theta)}{\partial\theta_k}\right|}{p_{ij}(\theta)}\right]_{i,j=1\ldots n;k=1\ldots K}$$

*are uniformly bounded by $B < \infty$ for all $\theta \in \mathbb{R}^K$.*

The second part of this assumption allows zero-probability transitions $p_{ij}(\theta) = 0$ only if $\nabla p_{ij}(\theta)$ is also zero, in which case we set $0/0 = 0$. One example is if $i \to j$ is a forbidden transition, so that $p_{ij}(\theta) = 0$ for all $\theta \in \mathbb{R}^K$. Another example satisfying the assumption is

$$p_{ij}(\theta) = \frac{e^{\theta_{ij}}}{\sum_{j=1}^{n} e^{\theta_{ij}}},$$

where $\theta = [\theta_{11}, \ldots, \theta_{1n}, \ldots, \theta_{nn}] \in \mathbb{R}^{n^2}$ are the parameters of $P(\theta)$, for then

$$\frac{\partial p_{ij}(\theta)/\partial\theta_{ij}}{p_{ij}(\theta)} = 1 - p_{ij}(\theta), \quad \text{and}$$
$$\frac{\partial p_{ij}(\theta)/\partial\theta_{kl}}{p_{ij}(\theta)} = -p_{kl}(\theta).$$

Assuming for the moment that $\nabla\pi(\theta)$ exists (this will be justified shortly), then, suppressing $\theta$ dependencies,

$$\nabla\eta = \nabla\pi' r, \tag{14}$$

since the reward $r$ does not depend on $\theta$. Note that our convention for $\nabla$ in this paper is that it takes precedence over all other operations, so $\nabla g(\theta) f(\theta) = [\nabla g(\theta)]\, f(\theta)$. Equations like (14) should be regarded as shorthand notation for $K$ equations of the form

$$\frac{\partial\eta(\theta)}{\partial\theta_k} = \left[\frac{\partial\pi(\theta,1)}{\partial\theta_k}, \ldots, \frac{\partial\pi(\theta,n)}{\partial\theta_k}\right] [r(1), \ldots, r(n)]'$$

where $k = 1, \ldots, K$. To compute $\nabla\pi$, first differentiate the balance equations (12) to obtain

$$\nabla\pi' P + \pi'\nabla P = \nabla\pi',$$

and hence

$$\nabla\pi'(I-P) = \pi'\nabla P. \tag{15}$$

The system of equations defined by (15) is under-constrained because $I-P$ is not invertible (the balance equations show that $I-P$ has a left eigenvector with zero eigenvalue). However, let $e$ denote the $n$-dimensional column vector consisting of all 1s, so that $e\pi'$ is the $n\times n$ matrix with the stationary distribution $\pi'$ in each row. Since $\nabla\pi' e = \nabla(\pi' e) = \nabla(1) = 0$, we can rewrite (15) as

$$\nabla\pi'\left[I-(P-e\pi')\right] = \pi'\nabla P.$$

To see that the inverse $[I-(P-e\pi')]^{-1}$ exists, let $A$ be any matrix satisfying $\lim_{t\to\infty} A^t = 0$. Then we can write

$$\begin{aligned}\lim_{T\to\infty}\left[(I-A)\sum_{t=0}^{T}A^t\right] &= \lim_{T\to\infty}\left[\sum_{t=0}^{T}A^t - \sum_{t=1}^{T+1}A^t\right]\\ &= I - \lim_{T\to\infty}A^{T+1}\\ &= I.\end{aligned}$$

Thus,

$$(I-A)^{-1} = \sum_{t=0}^{\infty}A^t.$$

It is easy to prove by induction that $\left[P-e\pi'\right]^t = P^t - e\pi'$ which converges to 0 as $t\to\infty$ by Assumption 1. So $[I-(P-e\pi')]^{-1}$ exists and is equal to $\sum_{t=0}^{\infty}\left[P^t-e\pi'\right]$. Hence, we can write

$$\nabla\pi' = \pi'\nabla P\left[I-P+e\pi'\right]^{-1}, \tag{16}$$

and so[9]

$$\nabla\eta = \pi'\nabla P\left[I-P+e\pi'\right]^{-1} r. \tag{17}$$

For MDPs with a sufficiently small number of states, (17) could be solved exactly to yield the precise gradient direction. However, in general, if the state space is small enough that an exact solution of (17) is possible, then it will be small enough to derive the optimal policy using policy iteration and table-lookup, and there would be no point in pursuing a gradient based approach in the first place[10].

Thus, for problems of practical interest, (17) will be intractable and we will need to find some other way of computing the gradient. One approximate technique for doing this is presented in the next section.

9. The argument leading to (16) coupled with the fact that $\pi(\theta)$ is the unique solution to (12) can be used to justify the existence of $\nabla\pi$. Specifically, we can run through the same steps computing the value of $\pi(\theta+\delta)$ for small $\delta$ and show that the expression (16) for $\nabla\pi$ is the unique matrix satisfying $\pi(\theta+\delta) = \pi(\theta) + \delta\nabla\pi(\theta) + O(\|\delta\|^2)$.
10. Equation (17) may still be useful for POMDPs, since in that case there is no tractable dynamic programming algorithm.

## 4. Approximating the Gradient in Parameterized Markov Chains

In this section, we show that the gradient can be split into two components, one of which becomes negligible as a discount factor $\beta$ approaches 1.

For all $\beta \in [0, 1)$, let $J_\beta(\theta) = [J_\beta(\theta, 1), \ldots, J_\beta(\theta, n)]$ denote the vector of expected discounted rewards from each state $i$:

$$J_\beta(\theta, i) := \mathbf{E}_\theta \left[ \sum_{t=0}^{\infty} \beta^t r(X_t) \middle| X_0 = i \right]. \tag{18}$$

Where the $\theta$ dependence is obvious, we just write $J_\beta$.

**Proposition 1.** *For all $\theta \in \mathbb{R}^K$ and $\beta \in [0, 1)$,*

$$\nabla\eta = (1-\beta)\nabla\pi' J_\beta + \beta\pi'\nabla P J_\beta. \tag{19}$$

*Proof.* Observe that $J_\beta$ satisfies the *Bellman* equations:

$$J_\beta = r + \beta P J_\beta. \tag{20}$$

(Bertsekas, 1995). Hence,

$$\begin{aligned}
\nabla\eta &= \nabla\pi' r \\
&= \nabla\pi' \left[ J_\beta - \beta P J_\beta \right] \\
&= \nabla\pi' J_\beta - \beta\nabla\pi' J_\beta + \beta\pi'\nabla P J_\beta && \text{by (15)} \\
&= (1-\beta)\nabla\pi' J_\beta + \beta\pi'\nabla P J_\beta.
\end{aligned}$$

□

We shall see in the next section that the second term in (19) can be estimated from a single sample path of the Markov chain. In fact, Theorem 1 in (Kimura et al., 1997) shows that the gradient estimates of the algorithm presented in that paper converge to $(1-\beta)\pi'\nabla J_\beta$. By the Bellman equations (20), this is equal to $(1-\beta)\beta(\pi'\nabla P J_\beta + \pi'\nabla J_\beta)$, which implies $(1-\beta)\pi'\nabla J_\beta = \beta\pi'\nabla P J_\beta$. Thus the algorithm of Kimura et al. (1997) also estimates the second term in the expression for $\nabla\eta(\theta)$ given by (19). It is important to note that $\pi'\nabla J_\beta \neq \nabla\left[\pi' J_\beta\right]$—the two quantities disagree by the first term in (19). This arises because the the stationary distribution itself depends on the parameters. Hence, the algorithm of Kimura et al. (1997) does not estimate the gradient of the expected discounted reward. In fact, the expected discounted reward is simply $1/(1-\beta)$ times the average reward $\eta(\theta)$ (Singh et al., 1994, Fact 7), so the gradient of the expected discounted reward is proportional to the gradient of the average reward.

The following theorem shows that the first term in (19) becomes negligible as $\beta$ approaches 1. Notice that this is not immediate from Proposition 1, since $J_\beta$ can become arbitrarily large in the limit $\beta \to 1$.

**Theorem 2.** *For all $\theta \in \mathbb{R}^K$,*

$$\nabla\eta = \lim_{\beta \to 1} \nabla_\beta \eta, \tag{21}$$

*where*

$$\nabla_\beta \eta := \pi'\nabla P J_\beta. \tag{22}$$

*Proof.* Recalling equation (17) and the discussion preceeding it, we have[11]

$$\nabla\eta = \pi'\nabla P \sum_{t=0}^{\infty} \left[P^t - e\pi'\right] r. \tag{23}$$

But $\nabla Pe = \nabla(Pe) = \nabla(1) = 0$ since $P$ is a stochastic matrix, so (23) can be rewritten as

$$\nabla\eta = \pi' \left[\sum_{t=0}^{\infty} \nabla P P^t\right] r. \tag{24}$$

Now let $\beta \in [0, 1]$ be a discount factor and consider the expression

$$f(\beta) := \pi' \left[\sum_{t=0}^{\infty} \nabla P (\beta P)^t\right] r \tag{25}$$

Clearly $\nabla\eta = \lim_{\beta\to 1} f(\beta)$. To complete the proof we just need to show that $f(\beta) = \nabla_\beta\eta$.

Since $(\beta P)^t = \beta^t P^t \to \beta^t e\pi' \to 0$, we can invoke the observation before (16) to write

$$\sum_{t=0}^{\infty} (\beta P)^t = [I - \beta P]^{-1}.$$

In particular, $\sum_{t=0}^{\infty}(\beta P)^t$ converges, so we can take $\nabla P$ back out of the sum in the right-hand-side of (25) and write[12]

$$f(\beta) = \pi'\nabla P \left[\sum_{t=0}^{\infty} \beta^t P^t\right] r. \tag{26}$$

But $\left[\sum_{t=0}^{\infty} \beta^t P^t\right] r = J_\beta$. Thus $f(\beta) = \pi'\nabla P J_\beta = \nabla_\beta\eta$. □

Theorem 2 shows that $\nabla_\beta\eta$ is a good approximation to the gradient as $\beta$ approaches 1, but it turns out that values of $\beta$ very close to 1 lead to large variance in the estimates of $\nabla_\beta\eta$ that we describe in the next section. However, the following theorem shows that $1 - \beta$ need not be too small, provided the transition probability matrix $P(\theta)$ has distinct eigenvalues, and the Markov chain has a short *mixing time*. From any initial state, the distribution over states of a Markov chain converges to the stationary distribution, provided the assumption (Assumption 1) about the existence and uniqueness of the stationary distribution is satisfied (see, for example, Lancaster & Tismenetsky, 1985, Theorem 15.8.1, p. 552). The spectral resolution theorem (Lancaster & Tismenetsky, 1985, Theorem 9.5.1, p. 314) implies that the distribution converges to stationarity at an exponential rate, and the time constant in this convergence rate (the mixing time) depends on the eigenvalues of the transition probability matrix. The existence of a unique stationary distribution implies that the

11. Since $e\pi' r = e\eta$, (23) motivates a different kind of algorithm for estimating $\nabla\eta$ based on *differential rewards* (Marbach & Tsitsiklis, 1998).

12. We cannot back $\nabla P$ out of the sum in the right-hand-side of (24) because $\sum_{t=0}^{\infty} P^t$ diverges ($P^t \to e\pi'$). The reason $\sum_{t=0}^{\infty} \nabla P P^t$ converges is that $P^t$ becomes orthogonal to $\nabla P$ in the limit of large $t$. Thus, we can view $\sum_{t=0}^{\infty} P^t$ as a sum of two orthogonal components: an infinite one in the direction $e$ and a finite one in the direction $e^{\perp}$. It is the finite component that we need to estimate. Approximating $\sum_{t=0}^{\infty} P^t$ with $\sum_{t=0}^{\infty}(\beta P)^t$ is a way of rendering the $e$-component finite while hopefully not altering the $e^{\perp}$-component too much. There should be other substitutions that lead to better approximations (in this context, see the final paragraph in Section 1.1).

largest magnitude eigenvalue is 1 and has multiplicity 1, and the corresponding left eigenvector is the stationary distribution. We sort the eigenvalues $\lambda_i$ in decreasing order of magnitude, so that $1 = \lambda_1 > |\lambda_2| > \cdots > |\lambda_s|$ for some $2 \leq s \leq n$. It turns out that $|\lambda_2|$ determines the mixing time of the chain.

The following theorem shows that if $1 - \beta$ is small compared to $1 - |\lambda_2|$, the gradient approximation described above is accurate. Since we will be using the estimate as a direction in which to update the parameters, the theorem compares the *directions* of the gradient and its estimate. In this theorem, $\kappa_2(A)$ denotes the *spectral condition number* of a nonsingular matrix $A$, which is defined as the product of the *spectral norms* of the matrices $A$ and $A^{-1}$,

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2,$$

where

$$\|A\|_2 = \max_{x:\|x\|=1} \|Ax\|,$$

and $\|x\|$ denotes the Euclidean norm of the vector $x$.

**Theorem 3.** *Suppose that the transition probability matrix $P(\theta)$ satisfies Assumption 1 with stationary distribution $\pi' = (\pi_1, \ldots, \pi_n)$, and has $n$ distinct eigenvalues. Let $S = (x_1 x_2 \cdots x_n)$ be the matrix of right eigenvectors of $P$ corresponding, in order, to the eigenvalues $1 = \lambda_1 > |\lambda_2| \geq \cdots \geq |\lambda_n|$. Then the normalized inner product between $\nabla\eta$ and $\beta\nabla_\beta\eta$ satisfies*

$$1 - \frac{\nabla\eta \cdot \beta\nabla_\beta\eta}{\|\nabla\eta\|^2} \leq \kappa_2\left(\Pi^{1/2}S\right) \frac{\|\nabla(\sqrt{\pi}_1, \ldots, \sqrt{\pi}_n)\|}{\|\nabla\eta\|} \sqrt{r'\Pi r} \frac{1-\beta}{1-\beta|\lambda_2|}, \tag{27}$$

*where $\Pi = \mathrm{diag}(\pi_1, \ldots, \pi_n)$.*

Notice that $r'\Pi r$ is the expectation under the stationary distribution of $r(X)^2$.

As well as the mixing time (via $|\lambda_2|$), the bound in the theorem depends on another parameter of the Markov chain: the spectral condition number of $\Pi^{1/2}S$. If the Markov chain is reversible (which implies that the eigenvectors $x_1, \ldots, x_n$ are orthogonal), this is equal to the ratio of the maximum to the minimum probability of states under the stationary distribution. However, the eigenvectors do not need to be nearly orthogonal. In fact, the condition that the transition probability matrix have $n$ distinct eigenvalues is not necessary; without it, the condition number is replaced by a more complicated expression involving spectral norms of matrices of the form $(P - \lambda_i I)$.

*Proof.* The existence of $n$ distinct eigenvalues implies that $P$ can be expressed as $S\Lambda S^{-1}$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ (Lancaster & Tismenetsky, 1985, Theorem 4.10.2, p 153). It follows that for any polynomial $f$, we can write $f(P) = Sf(\Lambda)S^{-1}$.

Now, Proposition 1 shows that $\nabla\eta - \beta\nabla_\beta\eta = \nabla\pi'(1-\beta)J_\beta$. But

$$\begin{aligned}
(1-\beta)J_\beta &= (1-\beta)\left(r + \beta Pr + \beta^2 P^2 r + \cdots\right) \\
&= (1-\beta)\left(I + \beta P + \beta^2 P^2 + \cdots\right) r \\
&= (1-\beta) S \left(\sum_{t=0}^{\infty} \beta^t \Lambda^t\right) S^{-1} r \\
&= (1-\beta) \sum_{j=1}^{n} x_j y_j' \left(\sum_{t=0}^{\infty} (\beta\lambda_j)^t\right) r,
\end{aligned}$$

where $S^{-1} = (y_1, \ldots, y_n)'$.

It is easy to verify that $y_i$ is the left eigenvector corresponding to $\lambda_i$, and that we can choose $y_1 = \pi$ and $x_1 = e$. Thus we can write

$$\begin{aligned}(1-\beta)J_\beta &= (1-\beta)e\pi' r + \sum_{j=2}^{n} x_j y_j' \left(\sum_{t=0}^{\infty}(1-\beta)(\beta\lambda_j)^t\right) r \\ &= (1-\beta)e\eta + \sum_{j=2}^{n} x_j y_j' \left(\frac{1-\beta}{1-\beta\lambda_j}\right) r \\ &= (1-\beta)e\eta + SMS^{-1}r,\end{aligned}$$

where

$$M = \operatorname{diag}\left(0, \frac{1-\beta}{1-\beta\lambda_2}, \ldots, \frac{1-\beta}{1-\beta\lambda_n}\right).$$

It follows from this and Proposition 1 that

$$\begin{aligned}1 - \frac{\nabla\eta \cdot \beta\nabla_\beta\eta}{\|\nabla\eta\|^2} &= 1 - \frac{\nabla\eta \cdot (\nabla\eta - \nabla\pi'(1-\beta)J_\beta)}{\|\nabla\eta\|^2} \\ &= \frac{\nabla\eta \cdot \nabla\pi'(1-\beta)J_\beta}{\|\nabla\eta\|^2} \\ &= \frac{\nabla\eta \cdot \nabla\pi' \left((1-\beta)e\eta + SMS^{-1}r\right)}{\|\nabla\eta\|^2} \\ &= \frac{\nabla\eta \cdot \nabla\pi' SMS^{-1}r}{\|\nabla\eta\|^2} \\ &\leq \frac{\left\|\nabla\pi' SMS^{-1}r\right\|}{\|\nabla\eta\|},\end{aligned}$$

by the Cauchy-Schwartz inequality. Since $\nabla\pi' = \nabla\left(\sqrt{\pi'}\right)\Pi^{1/2}$, we can apply the Cauchy-Schwartz inequality again to obtain

$$1 - \frac{\nabla\eta \cdot \beta\nabla_\beta\eta}{\|\nabla\eta\|^2} \leq \frac{\left\|\nabla\left(\sqrt{\pi'}\right)\right\| \left\|\Pi^{1/2}SMS^{-1}r\right\|}{\|\nabla\eta\|}. \tag{28}$$

We use spectral norms to bound the second factor in the numerator. It is clear from the definition that the spectral norm of a product of nonsingular matrices satisfies $\|AB\|_2 \leq \|A\|_2\|B\|_2$, and that the spectral norm of a diagonal matrix is given by $\|\operatorname{diag}(d_1, \ldots, d_n)\|_2 = \max_i |d_i|$. It follows that

$$\begin{aligned}\left\|\Pi^{1/2}SMS^{-1}r\right\| &= \left\|\Pi^{1/2}SMS^{-1}\Pi^{-1/2}\Pi^{1/2}r\right\| \\ &\leq \left\|\Pi^{1/2}S\right\|_2 \left\|S^{-1}\Pi^{-1/2}\right\|_2 \left\|\Pi^{1/2}r\right\| \|M\|_2 \\ &\leq \kappa_2\left(\Pi^{1/2}S\right)\sqrt{r'\Pi r}\frac{1-\beta}{1-\beta|\lambda_2|}.\end{aligned}$$

Combining with Equation (28) proves (27). □

## 5. Estimating the Gradient in Parameterized Markov Chains

Algorithm 1 introduces MCG (**M**arkov **C**hain **G**radient), an algorithm for estimating the approximate gradient $\nabla_\beta \eta$ from a single on-line sample path $X_0, X_1, \ldots$ from the Markov chain $M(\theta)$. MCG requires only $2K$ reals to be stored, where $K$ is the dimension of the parameter space: $K$ parameters for the eligibility trace $z_t$, and $K$ parameters for the gradient estimate $\Delta_t$. Note that after $T$ time steps $\Delta_T$ is the average so far of $r(X_t)z_t$,

$$\Delta_T = \frac{1}{T}\sum_{t=0}^{T-1} z_t r(X_t).$$

**Algorithm 1** The MCG (**M**arkov **C**hain **G**radient) algorithm

**Given:**

- Parameter $\theta \in \mathbb{R}^K$.
- Parameterized class of stochastic matrices $\mathcal{P} = \{P(\theta) \colon \theta \in \mathbb{R}^K\}$ satisfying Assumptions 3 and 1.
- $\beta \in [0, 1)$.
- Arbitrary starting state $X_0$.
- State sequence $X_0, X_1, \ldots$ generated by $M(\theta)$ (i.e. the Markov chain with transition probabilities $P(\theta)$).
- Reward sequence $r(X_0), r(X_1), \ldots$ satisfying Assumption 2.

Set $z_0 = 0$ and $\Delta_0 = 0$ ($z_0, \Delta_0 \in \mathbb{R}^K$).
**for** each state $X_{t+1}$ visited **do**
  $z_{t+1} = \beta z_t + \dfrac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)}$
  $\Delta_{t+1} = \Delta_t + \frac{1}{t+1}\left[r(X_{t+1}) z_{t+1} - \Delta_t\right]$
**end for**

**Theorem 4.** *Under Assumptions 1, 2 and 3, the* MCG *algorithm starting from any initial state $X_0$ will generate a sequence $\Delta_0, \Delta_1, \ldots, \Delta_t, \ldots$ satisfying*

$$\lim_{t\to\infty} \Delta_t = \nabla_\beta \eta \quad \text{w.p.1.} \tag{29}$$

*Proof.* Let $\{X_t\} = \{X_0, X_1, \ldots\}$ denote the random process corresponding to $M(\theta)$. If $X_0 \sim \pi$ then the entire process is stationary. The proof can easily be generalized to arbitrary initial distributions using the fact that under Assumption 1, $\{X_t\}$ is asymptotically stationary. When $\{X_t\}$ is

stationary, we can write

$$\begin{aligned}\pi'\nabla P J_\beta &= \sum_{i,j} \pi(i)\nabla p_{ij}(\theta) J_\beta(j) \\ &= \sum_{i,j} \pi(i) p_{ij}(\theta) \frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)} J_\beta(j) \\ &= \sum_{i,j} \Pr(X_t = i) \Pr(X_{t+1} = j | X_t = i) \frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)} \mathbf{E}(J(t+1) | X_{t+1} = j), \end{aligned} \tag{30}$$

where the first probability is with respect to the stationary distribution and $J(t+1)$ is the process

$$J(t+1) = \sum_{s=t+1}^{\infty} \beta^{s-t-1} r(X_s).$$

The fact that $\mathbf{E}(J(t+1)|X_{t+1}) = J_\beta(X_{t+1})$ for all $X_{t+1}$ follows from the boundedness of the magnitudes of the rewards (Assumption 2) and Lebesgue's dominated convergence theorem. We can rewrite Equation (30) as

$$\pi'\nabla P J_\beta = \sum_{i,j} \mathbf{E}\left[\chi_i(X_t)\chi_j(X_{t+1}) \frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)} J(t+1)\right],$$

where $\chi_i(\cdot)$ denotes the indicator function for state $i$,

$$\chi_i(X_t) := \begin{cases} 1 & \text{if } X_t = i, \\ 0 & \text{otherwise,} \end{cases}$$

and the expectation is again with respect to the stationary distribution. When $X_t$ is chosen according to the stationary distribution, the process $\{X_t\}$ is ergodic. Since the process $\{Z_t\}$ defined by

$$Z_t := \chi_i(X_t)\chi_j(X_{t+1}) \frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)} J(t+1)$$

is obtained by taking a fixed function of $\{X_t\}$, $\{Z_t\}$ is also stationary and ergodic (Breiman, 1966, Proposition 6.31). Since $\left|\frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)}\right|$ is bounded by Assumption 3, from the ergodic theorem we have (almost surely):

$$\begin{aligned}\pi'\nabla P J_\beta &= \sum_{i,j} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \chi_i(X_t)\chi_j(X_{t+1}) \frac{\nabla p_{ij}(\theta)}{p_{ij}(\theta)} J(t+1) \\ &= \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} J(t+1) \\ &= \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \left[\sum_{s=t+1}^{T} \beta^{s-t-1} r(X_s) + \sum_{s=T+1}^{\infty} \beta^{s-t-1} r(X_s)\right]. \end{aligned} \tag{31}$$

Concentrating on the second term in the right-hand-side of (31), observe that:

$$\begin{aligned}
&\left| \frac{1}{T} \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \sum_{s=T+1}^{\infty} \beta^{s-t-1} r(X_s) \right| \\
&\qquad \leq \frac{1}{T} \sum_{t=0}^{T-1} \left| \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \right| \sum_{s=T+1}^{\infty} \beta^{s-t-1} \left| r(X_s) \right| \\
&\qquad \leq \frac{BR}{T} \sum_{t=0}^{T-1} \sum_{s=T+1}^{\infty} \beta^{s-t-1} \\
&\qquad = \frac{BR}{T} \sum_{t=0}^{T-1} \frac{\beta^{T-t}}{1-\beta} \\
&\qquad = \frac{BR\beta \left(1-\beta^T\right)}{T \left(1-\beta\right)^2} \\
&\qquad \to 0 \text{ as } T \to \infty,
\end{aligned}$$

where $R$ and $B$ are the bounds on the magnitudes of the rewards and $\frac{|\nabla p_{ij}|}{p_{ij}}$ from Assumptions 2 and 3. Hence,

$$\pi' \nabla P J_\beta = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \sum_{s=t+1}^{T} \beta^{s-t-1} r(X_s). \tag{32}$$

Unrolling the equation for $\Delta_T$ in the MCG algorithm shows it is equal to

$$\frac{1}{T} \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \sum_{s=t+1}^{T} \beta^{s-t-1} r(i_s),$$

hence $\Delta_T \to \pi' \nabla P J_\beta$ w.p.1 as required. □

## 6. Estimating the Gradient in Partially Observable Markov Decision Processes

Algorithm 1 applies to any parameterized class of stochastic matrices $P(\theta)$ for which we can compute the gradients $\nabla p_{ij}(\theta)$. In this section we consider the special case of $P(\theta)$ that arise from a parameterized class of randomized policies controlling a partially observable Markov decision process (POMDP). The 'partially observable' qualification means we assume that these policies have access to an observation process that depends on the state, but in general they may not see the state.

Specifically, assume that there are $N$ controls $\mathcal{U} = \{1, \ldots, N\}$ and $M$ observations $\mathcal{Y} = \{1, \ldots, M\}$. Each $u \in \mathcal{U}$ determines a stochastic matrix $P(u)$ which does not depend on the parameters $\theta$. For each state $i \in \mathcal{S}$, an observation $Y \in \mathcal{Y}$ is generated independently according to a probability distribution $\nu(i)$ over observations in $\mathcal{Y}$. We denote the probability of observation $y$ by $\nu_y(i)$. A *randomized policy* is simply a function $\mu$ mapping observations $y \in \mathcal{Y}$ into probability distributions over the controls $\mathcal{U}$. That is, for each observation $y$, $\mu(y)$ is a distribution over the controls in $\mathcal{U}$. Denote the probability under $\mu$ of control $u$ given observation $y$ by $\mu_u(y)$.

To each randomized policy $\mu(\cdot)$ and observation distribution $\nu(\cdot)$ there corresponds a Markov chain in which state transitions are generated by first selecting an observation $y$ in state $i$ according

to the distribution $\nu(i)$, then selecting a control $u$ according to the distribution $\mu(y)$, and then generating a transition to state $j$ according to the probability $p_{ij}(u)$. To parameterize these chains we parameterize the policies, so that $\mu$ now becomes a function $\mu(\theta, y)$ of a set of parameters $\theta \in \mathbb{R}^K$ as well as the observation $y$. The Markov chain corresponding to $\theta$ has state transition matrix $[p_{ij}(\theta)]$ given by

$$p_{ij}(\theta) = \mathbf{E}_{Y\sim\nu(i)}\mathbf{E}_{U\sim\mu(\theta,Y)}p_{ij}(U). \tag{33}$$

Equation (33) implies

$$\nabla p_{ij}(\theta) = \sum_{u,y} \nu_y(i)p_{ij}(u)\nabla\mu_u(\theta, y). \tag{34}$$

Algorithm 2 introduces the GPOMDP algorithm (for **G**radient of a **P**artially **O**bservable **M**arkov **D**ecision **P**rocess), a modified form of Algorithm 1 in which updates of $z_t$ are based on $\mu_{U_t}(\theta, Y_t)$, rather than $p_{X_tX_{t+1}}(\theta)$. Note that Algorithm 2 does not require knowledge of the transition probability matrix $P$, nor of the observation process $\nu$; it only requires knowledge of the randomized policy $\mu$. GPOMDP is essentially the algorithm proposed by Kimura et al. (1997) without the reward baseline.

The algorithm GPOMDP assumes that the policy $\mu$ is a function only of the current observation. It is immediate that the same algorithm works for any finite history of observations. In general, an optimal policy needs to be a function of the entire observation history. GPOMDP can be extended to apply to policies with internal state (Aberdeen & Baxter, 2001).

**Algorithm 2** The GPOMDP algorithm.

**Given:**

- Parameterized class of randomized policies $\{\mu(\theta, \cdot) : \theta \in \mathbb{R}^K\}$ satisfying Assumption 4.
- Partially observable Markov decision process which when controlled by the randomized policies $\mu(\theta, \cdot)$ corresponds to a parameterized class of Markov chains satisfying Assumption 1.
- $\beta \in [0, 1)$.
- Arbitrary (unknown) starting state $X_0$.
- Observation sequence $Y_0, Y_1, \ldots$ generated by the POMDP with controls $U_0, U_1, \ldots$ generated randomly according to $\mu(\theta, Y_t)$.
- Reward sequence $r(X_0), r(X_1), \ldots$ satisfying Assumption 2, where $X_0, X_1, \ldots$ is the (hidden) sequence of states of the Markov decision process.

Set $z_0 = 0$ and $\Delta_0 = 0$ ($z_0, \Delta_0 \in \mathbb{R}^K$).
**for** each observation $Y_t$, control $U_t$, and subsequent reward $r(X_{t+1})$ **do**
    $z_{t+1} = \beta z_t + \dfrac{\nabla\mu_{U_t}(\theta, Y_t)}{\mu_{U_t}(\theta, Y_t)}$
    $\Delta_{t+1} = \Delta_t + \frac{1}{t+1}\left[r(X_{t+1})z_{t+1} - \Delta_t\right]$
**end for**

For convergence of Algorithm 2 we need to replace Assumption 3 with a similar bound on the gradient of $\mu$:

**Assumption 4.** *The derivatives,*

$$\frac{\partial \mu_u(\theta, y)}{\partial \theta_k}$$

*exist for all $u \in \mathcal{U}$, $y \in \mathcal{Y}$ and $\theta \in \mathbb{R}^K$. The ratios*

$$\left[ \frac{\left| \frac{\partial \mu_u(\theta,y)}{\partial \theta_k} \right|}{\mu_u(\theta, y)} \right]_{y=1\ldots M; u=1\ldots N; k=1\ldots K}$$

*are uniformly bounded by $B_\mu < \infty$ for all $\theta \in \mathbb{R}^K$.*

**Theorem 5.** *Under Assumptions 1, 2 and 4, Algorithm 2 starting from any initial state $X_0$ will generate a sequence $\Delta_0, \Delta_1, \ldots, \Delta_t, \ldots$ satisfying*

$$\lim_{t\to\infty} \Delta_t = \nabla_\beta \eta \quad \text{w.p.1.} \tag{35}$$

*Proof.* The proof follows the same lines as the proof of Theorem 4. In this case,

$$\begin{aligned}
\pi' \nabla P J_\beta &= \sum_{i,j} \pi(i) \nabla p_{ij}(\theta) J_\beta(j) \\
&= \sum_{i,j,y,u} \pi(i) p_{ij}(u) \nu_y(i) \nabla \mu_u(\theta, y) J_\beta(j) \text{ from (34)} \\
&= \sum_{i,j,y,u} \pi(i) p_{ij}(u) \nu_y(i) \frac{\nabla \mu_u(\theta, y)}{\mu_u(\theta, y)} \mu_u(\theta, y) J_\beta(j), \\
&= \sum_{i,j,y,u} \mathbf{E} Z'_t,
\end{aligned}$$

where the expectation is with respect to the stationary distribution of $\{X_t\}$, and the process $\{Z'_t\}$ is defined by

$$Z'_t := \chi_i(X_t)\chi_j(X_{t+1})\chi_u(U_t)\chi_y(Y_t) \frac{\nabla \mu_u(\theta, y)}{\mu_u(\theta, y)} J(t+1),$$

where $U_t$ is the control process and $Y_t$ is the observation process. The result follows from the same arguments used in the proof of Theorem 4. □

### 6.1 Control dependent rewards

There are many circumstances in which the rewards may themselves depend on the controls $u$. For example, some controls may consume more energy than others and so we may wish to add a penalty term to the reward function in order to conserve energy. The simplest way to deal with this is to define for each state $i$ the expected reward $\bar{r}(i)$ by

$$\bar{r}(i) = \mathbf{E}_{Y\sim\nu(i)} \mathbf{E}_{U\sim\mu(\theta,Y)} r(U, i), \tag{36}$$

and then redefine $J_\beta$ in terms of $\bar{r}$:

$$\bar{J}_\beta(\theta, i) := \lim_{N\to\infty} \mathbf{E}_\theta \left[ \sum_{t=0}^{N} \beta^t \bar{r}(X_t) \middle| X_0 = i \right], \tag{37}$$

where the expectation is over all trajectories $X_0, X_1, \ldots$. The performance gradient then becomes

$$\nabla\eta = \nabla\pi' \bar{r} + \pi' \nabla\bar{r},$$

which can be approximated by

$$\nabla_\beta \eta = \pi' \left[ \nabla P \bar{J}_\beta + \nabla \bar{r} \right],$$

due to the fact that $\bar{J}_\beta$ satisfies the Bellman equations (20) with $\bar{r}$ replaced by $r$.

For GPOMDP to take account of the dependence of $r$ on the controls, its fifth line should be replaced by

$$\Delta_{t+1} = \Delta_t + \frac{1}{t+1} \left[ r(U_{t+1}, X_{t+1}) \left( z_{t+1} + \frac{\nabla \mu_{U_{t+1}}(\theta, Y_{t+1})}{\mu_{U_{t+1}}(\theta, Y_{t+1})} \right) - \Delta_t \right].$$

It is straightforward to extend the proofs of Theorems 2, 3 and 5 to this setting.

### 6.2 Parameter dependent rewards

It is possible to modify GPOMDP when the rewards themselves depend directly on $\theta$. In this case, the fifth line of GPOMDP is replaced with

$$\Delta_{t+1} = \Delta_t + \frac{1}{t+1} \left[ r(\theta, X_{t+1}) z_{t+1} + \nabla r(\theta, X_{t+1}) - \Delta_t \right]. \tag{38}$$

Again, the convergence and approximation theorems will carry through, provided $\nabla r(\theta, i)$ is uniformly bounded. Parameter-dependent rewards have been considered by Glynn (1990), Marbach and Tsitsiklis (1998), and Baird and Moore (1999). In particular, Baird and Moore (1999) showed how suitable choices of $r(\theta, i)$ lead to a combination of value and policy search, or "VAPS". For example, if $\tilde{J}(\theta, i)$ is an approximate value-function, then setting[13]

$$r(\theta, X_t, X_{t-1}) = -\frac{1}{2} \left[ r(X_t) + \alpha \tilde{J}(\theta, X_t) - \tilde{J}(\theta, X_{t-1}) \right]^2,$$

where $r(X_t)$ is the usual reward and $\alpha \in [0, 1)$ is a discount factor, gives an update that seeks to minimize the expected Bellman error

$$\sum_{i=1}^{n} \pi(\theta, i) \left[ r(i) + \alpha \sum_{j=1}^{n} p_{ij}(\theta) \tilde{J}(\theta, j) - \tilde{J}(\theta, i) \right]^2. \tag{39}$$

This will have the effect of both minimizing the Bellman error in $\tilde{J}(\theta, i)$, and driving the system (via the policy) to states with small Bellman error. The motivation behind such an approach can be understood if one considers a $\tilde{J}$ that has *zero* Bellman error for all states. In that case a greedy policy derived from $\tilde{J}$ will be optimal, and regardless of how the actual policy is parameterized, the expectation of $z_t r(\theta, X_t, X_{t-1})$ will be zero and so will be the gradient computed by GPOMDP. This kind of update is known as an *actor-critic* algorithm (Barto et al., 1983), with the policy playing the role of the actor, and the value function playing the role of the critic.

13. The use of rewards $r(\theta, X_t, X_{t-1})$ that depend on the current and previous state does not substantially alter the analysis.

### 6.3 Extensions to infinite state, observation, and control spaces

The convergence proof for Algorithm 2 relied on finite state ($\mathcal{S}$), observation ($\mathcal{Y}$) and control ($\mathcal{U}$) spaces. However, it should be clear that with no modification Algorithm 2 can be applied immediately to POMDPs with countably or uncountably infinite $\mathcal{S}$ and $\mathcal{Y}$, and countable $\mathcal{U}$. All that changes is that $p_{ij}(u)$ becomes a *kernel* $p(x, x', u)$ and $\nu(i)$ becomes a density on observations. In addition, with the appropriate interpretation of $\nabla\mu/\mu$, it can be applied to uncountable $\mathcal{U}$. Specifically, if $\mathcal{U}$ is a subset of $\mathbb{R}^N$ then $\mu(y, \theta)$ will be a probability *density* function on $\mathcal{U}$ with $\mu_u(y, \theta)$ the density at $u$. If $\mathcal{U}$ and $\mathcal{Y}$ are subsets of Euclidean space (but $\mathcal{S}$ is a finite set), Theorem 5 can be extended to show that the estimates produced by this algorithm converge almost surely to $\nabla_\beta \eta$. In fact, we can prove a more general result that implies both this case of densities on subsets of $\mathbb{R}^N$ as well as the finite case of Theorem 5. We allow $\mathcal{U}$ and $\mathcal{Y}$ to be general spaces satisfying the following topological assumption. (For definitions see, for example, (Dudley, 1989).)

**Assumption 5.** *The control space $\mathcal{U}$ has an associated topology that is separable, Hausdorff, and first-countable. For the corresponding Borel $\sigma$-algebra $\mathcal{B}$ generated by this topology, there is a $\sigma$-finite measure $\lambda$ defined on the measurable space $(\mathcal{U}, \mathcal{B})$. We say that $\lambda$ is the* reference measure *for $\mathcal{U}$.*

*Similarly, the observation space $\mathcal{Y}$ has a topology, Borel $\sigma$-algebra, and reference measure satisfying the same conditions.*

In the case of Theorem 5, where $\mathcal{U}$ and $\mathcal{Y}$ are finite, the associated reference measure is the counting measure. For $\mathcal{U} = \mathbb{R}^N$ and $\mathcal{Y} = \mathbb{R}^M$, the reference measure is Lebesgue measure. We assume that the distributions $\nu(i)$ and $\mu(\theta, y)$ are absolutely continuous with respect to the reference measures, and the corresponding Radon-Nikodym derivatives (probability masses in the finite case, densities in the Euclidean case) satisfy the following assumption.

**Assumption 6.** *For every $y \in \mathcal{Y}$ and $\theta \in \mathbb{R}^K$, the probability measure $\mu(\theta, y)$ is absolutely continuous with respect to the reference measure for $\mathcal{U}$. For every $i \in \mathcal{S}$, the probability measure $\nu(i)$ is absolutely continuous with respect to the reference measure for $\mathcal{Y}$.*

*Let $\lambda$ be the reference measure for $\mathcal{U}$. For all $u \in \mathcal{U}$, $y \in \mathcal{Y}$, $\theta \in \mathbb{R}^K$, and $k \in \{1, \ldots, K\}$, the derivatives*

$$\frac{\partial}{\partial \theta_k} \frac{d\mu(\theta, y)}{d\lambda}(u)$$

*exist and the ratios*

$$\frac{\left| \frac{\partial}{\partial \theta_k} \frac{d\mu_u(\theta, y)}{d\lambda}(u) \right|}{\frac{d\mu_u(\theta, y)}{d\lambda}(u)}$$

*are bounded by $B_\mu < \infty$.*

With these assumptions, we can replace $\mu$ in Algorithm 2 with the Radon-Nikodym derivative of $\mu$ with respect to the reference measure on $\mathcal{U}$. In this case, we have the following convergence result. This generalizes Theorem 5, and also applies to densities $\mu$ on a Euclidean space $\mathcal{U}$.

**Theorem 6.** *Suppose the control space $\mathcal{U}$ and the observation space $\mathcal{Y}$ satisfy Assumption 5 and let $\lambda$ be the reference measure on the control space $\mathcal{U}$. Consider Algorithm 2 with*

$$\frac{\nabla \mu_{U_t}(\theta, Y_t)}{\mu_{U_t}(\theta, Y_t)}$$

*replaced by*

$$\frac{\nabla \frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}{\frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}.$$

*Under Assumptions 1, 2 and 6, this algorithm, starting from any initial state $X_0$ will generate a sequence $\Delta_0, \Delta_1, \ldots, \Delta_t, \ldots$ satisfying*

$$\lim_{t\to\infty} \Delta_t = \nabla_\beta \eta \quad \text{w.p.1.}$$

*Proof.* See Appendix B □

## 7. New Results

Since the first version of this paper, we have extended GPOMDP to several new settings, and also proved some new properties of the algorithm. In this section we briefly outline these results.

### 7.1 Multiple Agents

Instead of a single agent generating actions according to $\mu(\theta, y)$, suppose we have multiple agents $i = 1, \ldots, n_a$, each with their own parameter set $\theta^i$ and distinct observation of the environment $y^i$, and that generate their own actions $u^i$ according to a policy $\mu_{u^i}(\theta^i, y^i)$. If the agents all receive the same reward signal $r(X_t)$ (they may be cooperating to solve the same task, for example), then GPOMDP can be applied to the collective POMDP obtained by concatenating the observations, controls, and parameters into single vectors $y = \left[y^1, \ldots, y^{n_a}\right]$, $u = \left[u^1, \ldots, u^{n_a}\right]$, and $\theta = \left[\theta^1, \ldots, \theta^{n_a}\right]$ respectively. An easy calculation shows that the gradient estimate $\Delta$ generated by GPOMDP in the collective case is precisely the same as that obtained by applying GPOMDP to each agent independently, and then concatenating the results. That is, $\Delta = \left[\Delta^1, \ldots, \Delta^{n_a}\right]$, where $\Delta^i$ is the estimate produced by GPOMDP applied to agent $i$. This leads to an on-line algorithm in which the agents adjust their parameters independently and without any explicit communication, yet collectively the adjustments are maximizing the global average reward. For similar observations in the context of REINFORCE and VAPS, see Peshkin et al. (2000). This algorithm gives a biologically plausible synaptic weight-update rule when applied to networks of spiking neurons in which the neurons are regarded as independent agents (Bartlett & Baxter, 1999), and has shown some promise in a network routing application (Tao, Baxter, & Weaver, 2001).

### 7.2 Policies with internal states

So far we have only considered purely *reactive* or *memoryless* policies in which the chosen control is a function of only the current observation. GPOMDP is easily extended to cover the case of policies that depend on finite histories of observations $Y_t, Y_{t-1}, \ldots, Y_{t-k}$, but in general, for *optimal* control of POMDPs, the policy must be a function of the *entire* observation history. Fortunately, the observation history may be summarized in the form of a *belief state* (the current distribution over states), which is itself updated based only upon the current observation, and knowledge of which is sufficient for optimal behaviour (Smallwood & Sondik, 1973; Sondik, 1978). An extension of GPOMDP to policies with parameterized internal belief states is described by Aberdeen and Baxter (2001), similar in spirit to the extension of VAPS and REINFORCE described by Meuleau et al. (1999).

### 7.3 Higher-Order Derivatives

GPOMDP can be generalized to compute estimates of second and higher-order derivatives of the average reward (assuming they exist), still from a single sample path of the underlying POMDP. To see this for second-order derivatives, observe that if $\eta(\theta) = \int q(\theta, x) r(x)\, dx$ for some twice-differentiable density $q(\theta, x)$ and performance measure $r(x)$, then

$$\nabla^2 \eta(\theta) = \int r(x) \frac{\nabla^2 q(\theta, x)}{q(\theta, x)} q(\theta, x)\, dx$$

where $\nabla^2$ denotes the matrix of second derivatives (Hessian). It can be verified that

$$\frac{\nabla^2 q(\theta, x)}{q(\theta, x)} = \nabla^2 \log q(\theta, x) + [\nabla \log q(\theta, x)]^2 \tag{40}$$

where the second term on the right-hand-side is the *outer product* between $\nabla \log q(\theta, x)$ and itself (that is, the matrix with entries $\partial/\partial\theta_i \log q(\theta, x) \partial/\partial\theta_j \log q(\theta, x)$). Taking $x$ to be a sequence of states $X_0, X_1, \ldots, X_T$ between visits to a recurrent state $i^*$ in a parameterized Markov chain (recall Section 1.1.1), we have $q(\theta, X) = \Pi_{t=0}^{T-1} p_{X_t X_{t+1}}(\theta)$, which combined with (40) yields

$$\frac{\nabla^2 q(\theta, X)}{q(\theta, X)} = \sum_{t=0}^{T-1} \frac{\nabla^2 p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} - \sum_{t=0}^{T-1} \left[ \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \right]^2 + \left[ \sum_{t=0}^{T-1} \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \right]^2$$

(the squared terms in this expression are also outer products). From this expression we can derive a GPOMDP-like algorithm for computing a biased estimate of the Hessian $\nabla^2 \eta(\theta)$, which involves maintaining—in addition to the usual eligibility trace $z_t$—a second *matrix* trace updated as follows:

$$Z_{t+1} = \beta Z_t + \frac{\nabla^2 p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} - \left[ \frac{\nabla p_{X_t X_{t+1}}(\theta)}{p_{X_t X_{t+1}}(\theta)} \right]^2 .$$

After $T$ time steps the algorithm returns the average so far of $r(X_t) \left[ Z_t + z_t^2 \right]$ where the second term is again an outer product. Computation of higher-order derivatives could be used in second-order gradient methods for optimization of policy parameters.

### 7.4 Bias and Variance Bounds

Theorem 3 provides a bound on the *bias* of $\nabla_\beta \eta(\theta)$ relative to $\nabla \eta(\theta)$ that applies when the underlying Markov chain has distinct eigenvalues. We have extended this result to arbitrary Markov chains (Bartlett & Baxter, 2001). However, the extra generality comes at a price, since the latter bound involves the number of states in the chain, whereas Theorem 3 does not. The same paper also supplies a proof that the variance of GPOMDP scales as $1/(1-\beta)^2$, providing a formal justification for the interpretation of $\beta$ in terms of bias/variance trade-off.

## 8. Conclusion

We have presented a general algorithm (MCG) for computing arbitrarily accurate approximations to the gradient of the average reward in a parameterized Markov chain. When the chain's transition matrix has distinct eigenvalues, the accuracy of the approximation was shown to be controlled by the

size of the subdominant eigenvalue $|\lambda_2|$. We showed how the algorithm could be modified to apply to partially observable Markov decision processes controlled by parameterized stochastic policies, with both discrete and continuous control, observation and state spaces (GPOMDP). For the finite state case, we proved convergence with probability 1 of both algorithms.

We briefly described extensions to multi-agent problems, policies with internal state, estimating higher-order derivatives, generalizations of the bias result to chains with non-distinct eigenvalues, and a new variance result. There are many avenues for further research. Continuous time results should follow as extensions of the results presented here. The MCG and GPOMDP algorithms can be applied to countably or uncountably infinite state spaces; convergence results are also needed in these cases.

In the companion paper (Baxter et al., 2001), we present experimental results showing rapid convergence of the estimates generated by GPOMDP to the true gradient $\nabla\eta$. We give on-line variants of the algorithms of the present paper, and also variants of gradient ascent that make use of the estimates of $\nabla_\beta\eta$. We present experimental results showing the effectiveness of these algorithms in a variety of problems, including a three-state MDP, a nonlinear physical control problem, and a call-admission problem.

**Acknowledgements**

This work was supported by the Australian Research Council, and benefited from the comments of several anonymous referees. Most of this research was performed while the authors were with the Research School of Information Sciences and Engineering, Australian National University.

## Appendix A. A Simple Example of Policy Degradation in Value-Function Learning

Approximate value-function approaches to reinforcement work by minimizing some form of error between the approximate value function and the true value function. It has long been known that this may not necessarily lead to improved policy performance from the new value function. We include this appendix because it illustrates that this phenomenon can occur in the simplest possible system, a two-state MDP, and also provides some geometric intuition for why the phenomenon arises.

Consider the two-state Markov decision process (MDP) in Figure 1. There are two controls $u_1, u_2$ with corresponding transition probability matrices

$$P(u_1) = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} \end{bmatrix}, \quad P(u_2) = \begin{bmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & \frac{1}{3} \end{bmatrix},$$

so that $u_1$ always takes the system to state 2 with probability $2/3$, regardless of the starting state (and therefore to state 1 with probability $1/3$), and $u_2$ does the opposite. Since state 2 has a reward of 1, while state 1 has a reward of 0, the optimal policy is to always select action $u_1$. Under this policy the stationary distribution on states is $[\pi_1, \pi_2] = [1/3, 2/3]$, while the infinite-horizon discounted value of each state $i = 1, 2$ with discount value $\alpha \in [0, 1)$ is

$$J_\alpha(i) = \mathbf{E}\left( \sum_{t=0}^{\infty} \alpha^t r(X_t) \middle| X_0 = i \right),$$

where the expectation is over all state sequences $X_0, X_1, X_2, \ldots$ with state transitions generated according to $P(u_1)$. Solving *Bellman's equations:* $J_\alpha = r + \alpha P(u_1) J_\alpha$, where $J_\alpha = [J_\alpha(1), J_\alpha(2)]'$ and $r = [r(1), r(2)]'$ yields $J_\alpha(1) = \frac{2\alpha}{3(1-\alpha)}$ and $J_\alpha(2) = 1 + \frac{2\alpha}{3(1-\alpha)}$.

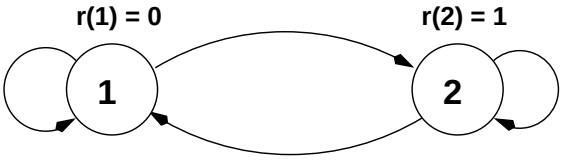


Figure 1: Two-state Markov Decsision Process

Now, suppose we are trying to learn an approximate value function $\tilde{J}$ for this MDP, *i.e.* , $\tilde{J}(i) = w\phi(i)$ for each state $i = 1, 2$ and some scalar feature $\phi$ ($\phi$ must have dimensionality 1 to ensure that $\tilde{J}$ really is *approximate*). Here $w \in \mathbb{R}$ is the parameter being learnt. For the greedy policy obtained from $\tilde{J}$ to be optimal, $\tilde{J}$ must value state 2 above state 1. For the purposes of this illustration choose $\phi(1) = 2, \phi(2) = 1$, so that for $\tilde{J}(2) > \tilde{J}(1)$, $w$ must be negative.

*Temporal Difference* learning (or $\mathrm{TD}(\lambda)$) is one of the most popular techniques for training approximate value functions (Sutton & Barto, 1998). It has been shown that for linear functions, $\mathrm{TD}(1)$ converges to a parameter $w^*$ minimizing the expected squared loss under the stationary distribution (Tsitsikilis & Van-Roy, 1997):

$$w^* = \mathrm{argmin}_w \sum_{i=1}^{2} \pi_i \left[ w\phi(i) - J_\alpha(i) \right]^2 . \tag{41}$$

Substituting the previous expressions for $\pi_1, \pi_2, \phi$ and $J_\alpha$ under the optimal policy and solving for $w^*$, yields $w^* = \frac{3+\alpha}{9(1-\alpha)}$. Hence $w^* > 0$ for all values of $\alpha \in [0, 1)$, which is the wrong sign. So we have a situation where the optimal policy is implementable as a greedy policy based on an approximate value function in the class (just choose any $w < 0$), yet $\mathrm{TD}(1)$ observing the optimal policy will converge to a value function whose corresponding greedy policy implements the suboptimal policy.

A geometrical illustration of why this occurs is shown in Figure 2. In this figure, points on the graph represent the values of the states. The scales of the state 1 and state 2 axes are weighted by $\sqrt{\pi(1)}$ and $\sqrt{\pi(2)}$ respectively. In this way, the squared euclidean distance on the graph between two points $J$ and $\tilde{J}$ corresponds to the expectation under the stationary distribution of the squared difference between values:

$$\left\| \left[ \sqrt{\pi(1)} J(1), \sqrt{\pi(2)} J(2) \right] - \left[ \sqrt{\pi(1)} \tilde{J}(1), \sqrt{\pi(2)} \tilde{J}(2) \right] \right\|^2 = \mathbf{E}_\pi \left( J(X) - \tilde{J}(X) \right)^2 .$$

For any value function in the shaded region, the corresponding greedy policy is optimal, since those value functions rank state 2 above state 1. The bold line represents the set of all realizable approximate value functions $(w\phi(1), w\phi(2))$. The solution to (41) is then the approximate value function found by projecting the point corresponding to the true value function $[(J_\alpha(1), J_\alpha(2)]$ onto this line. This is illustrated in the figure for $\alpha = 3/5$. The projection is suboptimal because weighted mean-squared distance in value-function space does not take account of the policy boundary.

## Appendix B. Proof of Theorem 6

The proof needs the following topological lemma. For definitions see, for example, (Dudley, 1989, pp. 24–25).

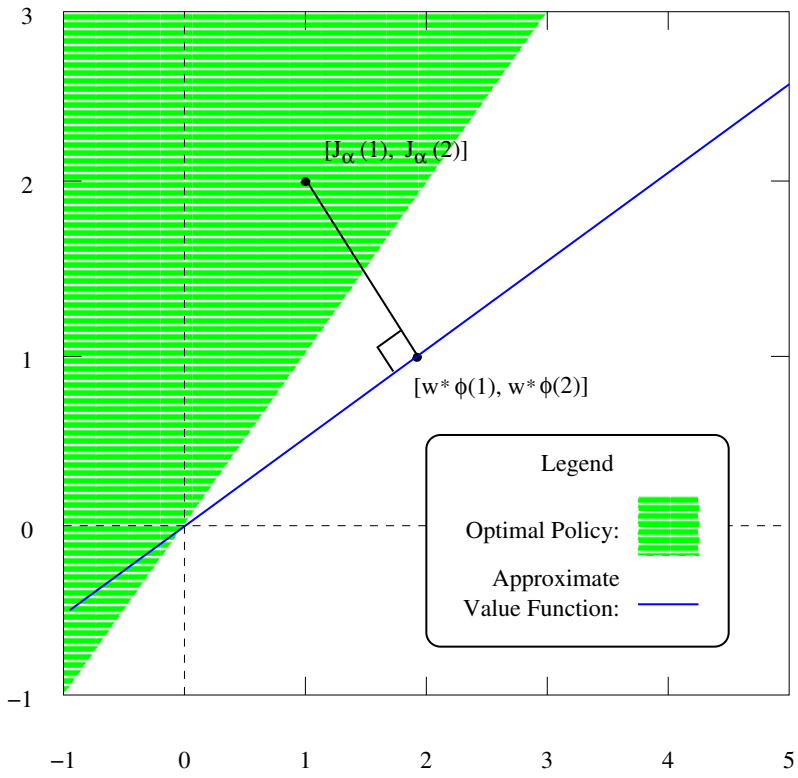


Figure 2: Plot of value-function space for the two-state system. Note that the scale of each axis has been weighted by the square root of the stationary probability of the corresponding state under the optimal policy. The solution found by TD(1) is simply the projection of the true value function onto the set of approximate value functions.

**Lemma 7.** *Let $(X, \mathcal{T})$ be a topological space that is Hausdorff, separable, and first-countable. Let $\mathcal{B}$ be the Borel $\sigma$-algebra generated by $\mathcal{T}$. Then the measurable space $(X, \mathcal{B})$ has a sequence $\mathcal{S}_1, \mathcal{S}_2, \ldots \subseteq \mathcal{B}$ of sets that satisfies the following conditions:*

1. *Each $\mathcal{S}_i$ is a partition of $X$ (that is, $X = \bigcup\{S\colon S \in \mathcal{S}_i\}$ and any two distinct elements of $\mathcal{S}_i$ have empty intersection).*
2. *For all $x \in X$, $\{x\} \in \mathcal{B}$ and*

$$\bigcap_{i=1}^{\infty} \{S \in \mathcal{S}_i : x \in S\} = \{x\}.$$

*Proof.* Since $X$ is separable, it has a countable dense subset $S = \{x_1, x_2, \ldots\}$. Since $X$ is first-countable, each of these $x_i$ has a countable neighbourhood base, $N_i$. Now, construct the partitions $\mathcal{S}_i$ using the countable set $N = \bigcup_{i=1}^{\infty} N_i$ as follows. Let $\mathcal{S}_0 = X$ and, for $i = 1, 2, \ldots$, define

$$\mathcal{S}_i = \{S \cap N_i : S \in \mathcal{S}_{i-1}\} \cup \{S \cap (X - N_i) : S \in \mathcal{S}_{i-1}\}.$$

Clearly, each $\mathcal{S}_i$ is a measurable partition of $X$. Since $X$ is Hausdorff, for each pair $x, x'$ of distinct points from $X$, there is a pair of disjoint open sets $A$ and $A'$ such that $x \in A$ and $x' \in A'$. Since $S$ is dense, there is a pair $s, s'$ from $S$ with $s \in A$ and $s' \in A'$. Also, $N$ contains neighbourhoods $N_s$ and $N_{s'}$ with $N_s \subseteq A$ and $N_{s'} \subseteq A'$. So $N_s$ and $N_{s'}$ are disjoint. Thus, for sufficiently large $i$, $x$ and $x'$ fall in distinct elements of the partition $\mathcal{S}_i$. Since this is true for any pair $x, x'$, it follows that

$$\bigcap_{i=1}^{\infty} \{S \in \mathcal{S}_i : x \in S\} \subseteq \{x\}.$$

The reverse inclusion is trivial. The measurability of all singletons $\{x\}$ follows from the measurability of $S_x := \bigcup_i \{S \in \mathcal{S}_i \colon S \cap \{x\} = \phi\}$ and the fact that $\{x\} = X - S_x$. □

We shall use Lemma 7 together with the following result to show that we can approximate expectations of certain random variables using a single sample path of the Markov chain.

**Lemma 8.** *Let $(X, \mathcal{B})$ be a measurable space satisfying the conditions of Lemma 7, and let $\mathcal{S}_1, \mathcal{S}_2, \ldots$ be a suitable sequence of partitions as in that lemma. Let $\mu$ be a probability measure defined on this space. Let $f$ be an absolutely integrable function on $X$. For an event $S$, define*

$$f(S) = \frac{\int_S f \, d\mu}{\mu(S)}.$$

*For each $x \in X$ and $k = 1, 2, \ldots$, let $S_k(x)$ be the unique element of $\mathcal{S}_k$ containing $x$. Then for almost all $x$ in $X$,*

$$\lim_{k \to \infty} f(S_k(x)) = f(x).$$

*Proof.* Clearly, the signed finite measure $\phi$ defined by

$$\phi(E) = \int_E f \, d\mu \tag{42}$$

is absolutely continuous with respect to $\mu$, and Equation (42) defines $f$ as the Radon-Nikodym derivative of $\phi$ with respect to $\mu$. This derivative can also be defined as

$$\frac{d\phi}{d\mu}(x) = \lim_{k \to \infty} \frac{\phi(S_k(x))}{\mu(S_k(x))}.$$

See, for example, (Shilov & Gurevich, 1966, Section 10.2). By the Radon-Nikodym Theorem (Dudley, 1989, Theorem 5.5.4, p. 134), these two expressions are equal a.e. ($\mu$). □

*Proof. (Theorem 6.)* From the definitions,

$$\begin{aligned} \nabla_\beta \eta &= \pi' \nabla P J_\beta \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} \pi(i) \nabla p_{ij}(\theta) J_\beta(j). \end{aligned} \tag{43}$$

For every $y$, $\mu$ is absolutely continuous with respect to the reference measure $\lambda$, hence for any $i$ and $j$ we can write

$$p_{ij}(\theta) = \int_{\mathcal{Y}} \int_{\mathcal{U}} p_{ij}(u) \, \frac{d\mu(\theta, y)}{d\lambda}(u) \, d\lambda(u) \, d\nu(i)(y).$$

Since $\lambda$ and $\nu$ do not depend on $\theta$ and $d\mu(\theta, y)/d\lambda$ is absolutely integrable, we can differentiate under the integral to obtain

$$\nabla p_{ij}(\theta) = \int_{\mathcal{Y}} \int_{\mathcal{U}} p_{ij}(u)\, \nabla \frac{d\mu(\theta, y)}{d\lambda}(u)\, d\lambda(u)\, d\nu(i)(y).$$

To avoid cluttering the notation, we shall use $\mu$ to denote the distribution $\mu(\theta, y)$ on $\mathcal{U}$, and $\nu$ to denote the distribution $\nu(i)$ on $\mathcal{Y}$. With this notation, we have

$$\nabla p_{ij}(\theta) = \int_{\mathcal{Y}} \int_{\mathcal{U}} p_{ij}\, \frac{\nabla \frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\, d\mu\, d\nu.$$

Now, let $\rho$ be the probability measure on $\mathcal{Y} \times \mathcal{U}$ generated by $\mu$ and $\nu$. We can write (43) as

$$\nabla_\beta \eta = \sum_{i,j} \pi(i) J_\beta(j) \int_{\mathcal{Y}\times\mathcal{U}} p_{ij} \frac{\nabla \frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\, d\rho.$$

Using the notation of Lemma 8, we define

$$p_{ij}(S) = \frac{\int_S p_{ij}\, d\rho}{\rho(S)},$$
$$\nabla(S) = \frac{1}{\rho(S)} \int_S \frac{\nabla \frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\, d\rho,$$

for a measurable set $S \subseteq \mathcal{Y} \times \mathcal{U}$. Notice that, for a given $i$, $j$, and $S$,

$$p_{ij}(S) = \Pr\left(X_{t+1} = j \,|\, X_t = i,\ (y, u) \in S\right)$$
$$\nabla(S) = \mathbf{E}\left( \frac{\nabla \frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}} \,\middle|\, X_t = i,\ (Y_t, U_t) \in S \right).$$

Let $\mathcal{S}_1, \mathcal{S}_2, \ldots$ be a sequence of partitions of $\mathcal{Y} \times \mathcal{U}$ as in Lemma 7, and let $S_k(y, u)$ denote the element of $\mathcal{S}_k$ containing $(y, u)$. Using Lemma 8, we have

$$\begin{aligned}\int_{\mathcal{Y}\times\mathcal{U}} p_{ij}\, \frac{\nabla \frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\, d\rho &= \int_{\mathcal{Y}\times\mathcal{U}} \lim_{k\to\infty} p_{ij}\left(S_k(y,u)\right)\, \nabla\left(S_k(y,u)\right)\, d\rho(y,u) \\ &= \lim_{k\to\infty} \sum_{S\in\mathcal{S}_k} \int_S p_{ij}(S)\, \nabla(S)\, d\rho,\end{aligned}$$

where we have used Assumption 6 and the Lebesgue dominated convergence theorem to interchange the integral and the limit. Hence,

$$\begin{aligned}
\nabla_\beta \eta &= \lim_{k\to\infty} \sum_{i,j} \sum_{S\in\mathcal{S}_k} \pi(i)\rho(S)p_{ij}(S)J_\beta(j)\nabla(S) \\
&= \lim_{k\to\infty} \sum_{i,j,S} \Pr(X_t = i)\Pr((Y_t,U_t)\in S)\Pr\left(X_{t+1}=j \,|X_t=i,\ (Y_t,U_t)\in S\right) \\
&\qquad \mathbf{E}\left(J(t+1)|X_{t+1}=j\right)\mathbf{E}\left(\left.\frac{\nabla\frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\right| X_t=i,\ (Y_t,U_t)\in S\right) \\
&= \lim_{k\to\infty} \sum_{i,j,S} \mathbf{E}\left[\chi_i(X_t)\chi_S(Y_t,U_t)\chi_j(X_{t+1})J(t+1)\frac{\nabla\frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}\right],
\end{aligned}$$

where probabilities and expectations are with respect to the stationary distribution $\pi$ of $X_t$, and the distributions on $Y_t, U_t$. Now, the random process inside the expectation is asymptotically stationary and ergodic. From the ergodic theorem, we have (almost surely)

$$\nabla_\beta \eta = \lim_{k\to\infty}\lim_{T\to\infty} \frac{1}{T}\sum_{i,j,S}\sum_{t=0}^{T-1} \chi_i(X_t)\chi_S(Y_t,U_t)\chi_j(X_{t+1})J(t+1)\frac{\nabla\frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}}.$$

It is easy to see that the double limit also exists when the order is reversed, so

$$\begin{aligned}
\nabla_\beta \eta &= \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1}\lim_{k\to\infty} \sum_{i,j,S} \chi_i(X_t)\chi_S(Y_t,U_t)\chi_j(X_{t+1})J(t+1)\frac{\nabla\frac{d\mu}{d\lambda}}{\frac{d\mu}{d\lambda}} \\
&= \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \frac{\nabla\frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}{\frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}J(t+1).
\end{aligned}$$

The same argument as in the proof of Theorem 4 shows that the tails of $J(t+1)$ can be ignored when

$$\left|\frac{\nabla\frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}{\frac{d\mu(\theta,Y_t)}{d\lambda}(U_t)}\right|$$

and $|r(X_t)|$ are uniformly bounded. It follows that $\Delta_T \to \pi'\nabla P J_\beta$ w.p.1, as required. □

## References


Aberdeen, D., & Baxter, J. (2001). Policy-gradient learning of controllers with internal state. Tech. rep., Australian National University.

Aleksandrov, V. M., Sysoyev, V. I., & Shemeneva, V. V. (1968). Stochastic optimaization. *Engineering Cybernetics*, *5*, 11–16.

Baird, L., & Moore, A. (1999). Gradient descent for general reinforcement learning. In *Advances in Neural Information Processing Systems 11*. MIT Press.

Bartlett, P. L., & Baxter, J. (1999). Hebbian synaptic modifications in spiking neurons that learn. Tech. rep., Research School of Information Sciences and Engineering, Australian National University. http://csl.anu.edu.au/∼bartlett/papers/BartlettBaxter-Nov99.ps.gz.

Bartlett, P. L., & Baxter, J. (2001). Estimation and approximation bounds for gradient-based reinforcement learning. *Journal of Computer and Systems Sciences*, *62*. Invited Paper: Special Issue on COLT 2000.

Barto, A. G., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13*, 834–846.

Baxter, J., Bartlett, P. L., & Weaver, L. (2001). Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*. To appear.

Baxter, J., Tridgell, A., & Weaver, L. (2000). Learning to play chess using temporal-differences. *Machine Learning*, *40*(3), 243–263.

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control, Vol II*. Athena Scientific.

Breiman, L. (1966). *Probability*. Addison-Wesley.

Cao, X.-R., & Wan, Y.-W. (1998). Algorithms for Sensitivity Analysis of Markov Chains Through Potentials and Perturbation Realization. *IEEE Transactions on Control Systems Technology*, *6*, 482–492.

Dudley, R. M. (1989). *Real Analysis and Probability*. Wadsworth & Brooks/Cole, Belmont, California.

Glynn, P. W. (1986). Stochastic approximation for monte-carlo optimization. In *Proceedings of the 1986 Winter Simulation Conference*, pp. 356–365.

Glynn, P. W. (1990). Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, *33*, 75–84.

Glynn, P. W., & L'Ecuyer, P. (1995). Likelihood ratio gradient estimation for regenerative stochastic recursions. *Advances in Applied Probability, 27, 4 (1995)*, *27*, 1019–1053.

Ho, Y.-C., & Cao, X.-R. (1991). *Perturbation Analysis of Discrete Event Dynamic Systems*. Kluwer Academic, Boston.

Jaakkola, T., Singh, S. P., & Jordan, M. I. (1995). Reinforcement Learning Algorithm for Partially Observable Markov Decision Problems. In Tesauro, G., Touretzky, D., & Leen, T. (Eds.), *Advances in Neural Information Processing Systems*, Vol. 7. MIT Press, Cambridge, MA.

Kimura, H., & Kobayashi, S. (1998a). An analysis of actor/critic algorithms using eligibility traces: Reinforcement learning with imperfect value functions. In *Fifteenth International Conference on Machine Learning*, pp. 278–286.

Kimura, H., & Kobayashi, S. (1998b). Reinforcement learning for continuous action using stochastic gradient ascent. In *Intelligent Autonomous Systems (IAS-5)*, pp. 288–295.

Kimura, H., Miyazaki, K., & Kobayashi, S. (1997). Reinforcement learning in POMDPs with function approximation. In Fisher, D. H. (Ed.), *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, pp. 152–160.

Kimura, H., Yamamura, M., & Kobayashi, S. (1995). Reinforcement learning by stochastic hill climbing on discounted reward. In *Proceedings of the Twelfth International Conference on Machine Learning (ICML'95)*, pp. 295–303.

Konda, V. R., & Tsitsiklis, J. N. (2000). Actor-Critic Algorithms. In *Neural Information Processing Systems 1999*. MIT Press.

Lancaster, P., & Tismenetsky, M. (1985). *The Theory of Matrices*. Academic Press, San Diego, CA.

Marbach, P., & Tsitsiklis, J. N. (1998). Simulation-Based Optimization of Markov Reward Processes. Tech. rep., MIT.

Meuleau, N., Peshkin, L., Kaelbling, L. P., & Kim, K.-E. (2000). Off-policy policy search. Tech. rep., MIT Artificical Intelligence Laboratory.

Meuleau, N., Peshkin, L., Kim, K.-E., & Kaelbling, L. P. (1999). Learning finite-state controllers for partially observable environments. In *Proceedings of the Fifteenth International Conference on Uncertainty in Artificial Intelligence*.

Peshkin, L., Kim, K.-E., Meuleau, N., & Kaelbling, L. P. (2000). Learning to cooperate via policy search. In *Proceedings of the Sixteenth International Conference on Uncertainty in Artificial Intelligence*.

Reiman, M. I., & Weiss, A. (1986). Sensitivity analysis via likelihood ratios. In *Proceedings of the 1986 Winter Simulation Conference*.

Reiman, M. I., & Weiss, A. (1989). Sensitivity analysis for simulations via likelihood ratios. *Operations Research*, *37*.

Rubinstein, R. Y. (1969). *Some Problems in Monte Carlo Optimization*. Ph.D. thesis.

Rubinstein, R. Y. (1991). How to optimize complex stochastic systems from a single sample path by the score function method. *Annals of Operations Research*, *27*, 175–211.

Rubinstein, R. Y. (1992). Decomposable score function estimators for sensitivity analysis and optimization of queueing networks. *Annals of Operations Research*, *39*, 195–229.

Rubinstein, R. Y., & Melamed, B. (1998). *Modern Simulation and Modeling*. Wiley, New York.

Rubinstein, R. Y., & Shapiro, A. (1993). *Discrete Event Systems*. Wiley, New York.

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*, 210–229.

Shilov, G. E., & Gurevich, B. L. (1966). *Integral, Measure and Derivative: A Unified Approach*. Prentice-Hall, Englewood Cliffs, N.J.

Singh, S. P., Jaakkola, T., & Jordan, M. I. (1994). Learning Without State-Estimation in Partially Observable Markovian Decision Processes. In *Proceedings of the Eleventh International Conference on Machine Learning*.

Singh, S., & Bertsekas, D. (1997). Reinforcement learning for dynamic channel allocation in cellular telephone systems. In *Advances in Neural Information Processing Systems: Proceedings of the 1996 Conference*, pp. 974–980. MIT Press.

Smallwood, R. D., & Sondik, E. J. (1973). The optimal control of partially observable Markov decision processes over a finite horizon. *Operations Research*, *21*, 1071–1098.

Sondik, E. J. (1978). The optimal control of partially observable Markov decision processes over the infinite horizon: Discounted costs. *Operations Research*, *26*.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge MA. ISBN 0-262-19398-1.

Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (2000). Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Neural Information Processing Systems 1999*. MIT Press.

Tao, N., Baxter, J., & Weaver, L. (2001). A multi-agent, policy-gradient approach to network routing. Tech. rep., Australian National University.

Tesauro, G. (1992). Practical Issues in Temporal Difference Learning. *Machine Learning*, *8*, 257–278.

Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, *6*, 215–219.

Tsitsikilis, J. N., & Van-Roy, B. (1997). An Analysis of Temporal Difference Learning with Function Approximation. *IEEE Transactions on Automatic Control*, *42*(5), 674–690.

Williams, R. J. (1992). Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, *8*, 229–256.

Zhang, W., & Dietterich, T. (1995). A reinforcement learning approach to job-shop scheduling. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pp. 1114–1120. Morgan Kaufmann.