Natural-Gradient Actor-Critic Algorithms

Shalabh Bhatnagar*, Richard S. Sutton†, Mohammad Ghavamzadeh†, and Mark Lee†

May 24, 2007

Abstract

We prove the convergence of four new reinforcement learning algorithms based on the actor-critic architecture, on function approximation, and on natural gradients. Reinforcement learning is a class of methods for solving Markov decision processes from sample trajectories under lack of model information. Actor-critic reinforcement learning methods are online approximations to policy iteration in which the value-function parameters are estimated using temporal difference learning and the policy parameters are updated by stochastic gradient ascent/descent. Reinforcement learning methods based on policy gradients in this way are of special interest because of their compatibility with function approximation methods, which are needed to handle infinite or large state spaces, and the use of temporal-difference learning in this way is of special interest because in many applications it dramatically reduces the variance of the estimates. Natural or Fisher-information versions of policy gradient algorithms are of interest because they can produce better conditioned parameterizations and have been shown to have further reduced variance in some cases. Our results extend prior two-timescale convergence results for actor-critic methods by Konda and Tsitsiklis by using temporal difference learning in the actor and by incorporating natural gradients. Our results extend prior empirical studies of natural-gradient actor-critic methods by Peters, Vijayakumar and Schaal by providing the first convergence proofs and the first fully incremental algorithms. We present empirical results verifying the convergence of our algorithms. A limitation of our results is that they do not address the use of least-squares methods and eligibility traces.

Key Words: Actor-critic reinforcement learning algorithms, policy gradient methods, approximate dynamic programming, bootstrapping, function approximation, two-timescale stochastic approximation, temporal-difference learning, natural-gradient.

1 Introduction

Many problems of scientific and economic importance are optimal sequential decision problems and as such can be formulated as Markov decision processes (MDPs) [15, 48, 60]. In some cases MDPs can be solved analytically, and in many cases they can be solved iteratively by dynamic programming or linear programming. However, in other cases these methods cannot be applied either because the state space is too large, a system model is available only as a simulator, or no

*Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India. E-Mail: shalabh@csa.iisc.ernet.in

†Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada T6G 2E8. E-Mail: {sutton,mgh,mlee}@cs.ualberta.ca

system model is available. It is in these cases that the techniques and algorithms of reinforcement learning may be helpful.

Reinforcement learning [18, 54] can be viewed as a broad class of sample-based methods for solving MDPs. In place of a model, these methods use sample trajectories of the system and the controller interacting, such as could be obtained from a simulation. It is not unusual in practical applications for such a simulator to be available when an explicit transition-probability model of the sort suitable for use by dynamic or linear programming is not (e.g., [56, 28]). Reinforcement learning methods can also be used with no model at all by obtaining sample trajectories by direct interaction with the system (e.g., [12, 36]).

One of the biggest challenges to solving MDPs with conventional methods is handling large state spaces. This is sometimes known as the "curse of dimensionality" because of the tendency of the size of a state space to grow exponentially with the number of its dimensions. The computational effort required to solve Bellman equation thus increases exponentially with the dimension and cardinality of the state space. A natural and venerable way of addressing the curse is to approximate the value function and policy parametrically with a number of parameters much smaller than the size of the state space (e.g., [13, 29, 27]). However a straightforward application of such function approximation methods to dynamic programming has not proved effective on very large problems. Some work with reinforcement learning and function approximation has also run into problems of convergence and instability [24, 7], but about a decade ago it was established that if trajectories were sampled according to their distribution under the target policy (the on-policy distribution) then convergence could be assured for linear feature-based function approximators [58, 52, 55]. Reinforcement learning's most impressive successes have in fact been on problems with extremely large state spaces that could not have been solved without function approximation (e.g., [56, 28, 45]). The ability of sample-based methods to use function approximation effectively is one of the most important reasons for interest in reinforcement learning within the engineering disciplines.

Policy gradient methods [62, 44, 53, 38, 10, 11] are some of the simplest reinforcement learning methods and provide both a good illustration of reinforcement learning and a foundation for the actor-critic methods that are the primary focus of this paper. In all these methods, the policy is taken to be an arbitrary differentiable function of a parameter vector $\theta \in \mathbb{R}^d$. Given some performance measure $J : \mathbb{R}^d \to \mathbb{R}$, we would like to update the policy parameter in the direction of the gradient:

$$\Delta\theta \propto \nabla_\theta J(\theta). \tag{1}$$

The gradient is not directly available of course, but sample trajectories can be used to construct consistent estimators of it, estimators that can be used in a stochastic approximation of the actual gradient. This is the basic idea behind policy-gradient reinforcement learning methods. Both theoretical analysis and empirical evaluations have highlighted a major shortcoming of these algorithms, namely, the high variance of the gradient estimates. This high variance results in the slowness and sample-inefficiency of these algorithms.

One possible solution to this problem, proposed by Kakade in 2002 [35] and then refined and extended by Bagnell and Schneider [8] and by Peters et al. [46], is based on the idea of *natural* gradients previously developed for supervised learning by Amari [4]. In the application to reinforcement learning, the policy gradient in (1) is replaced with a natural version. This is motivated by the intuition that the policy updates should be invariant to bijective transformations of the parametrization. Put more simply, a change in the way the policy is parametrized should not influence the result of the policy update. In terms of the policy update rule (1), the move to a

natural gradient rule amounts to linearly transforming the gradient using the inverse Fisher information matrix of the policy. In empirical evaluations, natural policy gradient has outperformed conventional policy gradient methods [35, 8, 46]. Moreover, the use of natural gradients can lead to simpler, and in some cases, more computationally efficient algorithms. Three of the four algorithms we introduce in this paper incorporate natural gradients.

In this paper we focus on a sub-class of policy-gradient methods known as actor-critic methods. These methods can be thought of as reinforcement learning analogs of dynamic programming's policy iteration method. Actor-critic methods are based on the simultaneous online estimation of the parameters of two structures, called the *actor* and the *critic*. The actor corresponds to a conventional action-selection policy, mapping states to actions in a probabilistic manner. The critic corresponds to a conventional state-value function, mapping states to expected cumulative future reward. Thus, the critic addresses a problem of prediction, whereas the actor is concerned with control. These problems are separable, but are solved simultaneously to find an optimal policy. A variety of methods can be used to solve the prediction problem, but the ones that have proved most effective in large applications are those based on some form of temporal difference (TD) learning [51], in which estimates are updated on the basis of other estimates much as they are in dynamic programming. Such "bootstrapping methods" [54] can be viewed as a way of accelerating learning by trading bias for variance [49].

Actor-critic methods were among the earliest to be investigated in reinforcement learning [9, 50]. They were largely supplanted in the 1990's by methods that estimate action-value functions (mappings from states and actions to the subsequent expected return) that are then used directly to select actions without constructing an explicit policy structure. The action-value approach was initially appealing because of its simplicity, but theoretical complications arose when it was combined with function approximation: these methods do not converge in the normal sense, but rather may "chatter" in the neighborhood of a good solution [32]. These complications lead to renewed interest in policy gradient methods. Policy gradient methods without bootstrapping can easily be proved convergent, but can suffer from high variance resulting in slow convergence as mentioned above, motivating their combination with bootstrapping TD methods as in actor-critic algorithms.

In this paper we introduce four novel actor-critic algorithms along these lines. For all four methods we prove convergence of the parameters of the policy and state-value function to a local maximum of a performance function that corresponds to the average reward plus a measure of the temporal difference error inherent in the function approximation. Our results are an extension of prior work on the convergence of two-timescale stochastic approximation recursions [1, 19, 37, 38]. That work had previously shown convergence to a locally optimal policy for several non-bootstrapping algorithms with or without function approximation. Konda and Tsitsiklis [38] have shown convergence for an actor-critic algorithm that uses bootstrapping in the critic, but our results are the first to prove convergence when the actor is bootstrapping as well. Our results also extend prior two-timescale results by incorporating natural gradients. Our results and algorithms differ in a number of other, smaller ways from those of Konda and Tsitsiklis; we detail these in Section 7 after the analysis has been presented.

Two other aspects of the theoretical results presented here should be mentioned at the outset. First, one of the issues that arises in policy gradient methods is the selection of a baseline reward level. In contrast to previous work we show that, in an actor-critic setting when compatible features are used, the baseline that minimizes the estimator variance for any given policy is in fact the state-

value function. Second, for the case of a fixed policy we use a recent result by Borkar and Meyn [22] to provide an alternative, simpler proof of convergence (cf. [58, 59]) in the Euclidean norm of temporal difference recursions.

In this paper we do not explicitly consider the treatment of eligibility traces ($\lambda > 0$ in TD(λ) [51]), which have been shown to improve performance in cases of function approximation or partial observability, but we believe the extension of all of our results to general λ would be straightforward. Less clear is how or whether our results could be extended to least-squares temporal-difference methods [25, 23, 41, 46, 31]. It is not clear how to satisfactorily incorporate these methods in a context in which the policy is changing. Our proof techniques do not immediately extend to this case and we leave it for future work. We do consider the use of approximate advantages as in the works of Baird [6] and Peters el al. [46].

The rest of the paper is organized as follows. In Section 2 we present our reinforcement learning framework and provide an overview of policy gradient methods. We motivate two-timescale stochastic approximation in Section 3 as this is the technique used by our algorithms. In Section 4 we discuss policy gradient methods with function approximation and present some preliminary results. We show here in particular that the minimum variance baseline corresponds to the state-value function and obtain a form of bias in gradient estimates that results from the use of function approximation. Our four actor-critic algorithms are presented in Section 5, and their convergence analysis is in Section 6. In Section 7 we discuss the relationship of our algorithms to the actor-critic algorithm of Konda and Tsitsiklis [38] and the natural actor-critic algorithm of Peters, Vijayakumar and Schaal [46]. Section 8 presents numerical experiments verifying the operation of our algorithms. Section 9 contains concluding remarks.

2 The Policy Gradient Framework

Consider the standard reinforcement learning framework in which a learning agent interacts with a stochastic environment. The overall model we consider is that of a discrete time Markov decision process (MDP) with finite numbers of states and actions, and bounded rewards. We allow $\mathcal{S}$ and $\mathcal{A}$ to respectively denote the state and action spaces of this MDP. For simplicity, we assume that $\mathcal{S}$ is the set $\mathcal{S} = \{1, \ldots, n\}$. We denote by s_t, a_t and r_t, the state, action and reward, respectively, at time t. We assume that reward is random, real-valued and uniformly bounded. For simplicity and ease of notation, we assume that all actions in $\mathcal{A}$ are feasible in each state. The state transition probabilities for the environment will be characterized by $P(s, a, s') = \Pr(s_{t+1} = s' | s_t = s, a_t = a)$, $\forall s, s' \in \mathcal{S}, a \in \mathcal{A}$. Further, the single-stage expected reward when action a is taken in state s will be denoted $R(s, a) = \mathbf{E}[r_{t+1} | s_t = s, a_t = a]$.

An admissible policy $\bar{\pi}$ is a decision rule that is described by a sequence of functions $\bar{\pi} = \{\mu_0, \mu_1, \ldots\}$ such that each $\mu_t : \mathcal{S} \to \mathcal{A}$, with action $\mu_t(s)$ taken in state s at instant $t \geq 0$. A stationary policy is a time invariant decision rule, i.e., one for which $\mu_t = \mu$, $\forall t \geq 0$, for some $\mu : \mathcal{S} \to \mathcal{A}$. Most often, one refers to the function μ itself as the stationary policy. A stationary randomized policy π that we refer to as simply a randomized policy is specified via a probability distribution $\pi(s, \cdot)$ over $\mathcal{A}$, for $s \in \mathcal{S}$. Under the long-run average reward setting considered in this paper, it can be shown that a stationary optimal policy exists. Note that any stationary policy is trivially a randomized policy as well. We motivate the following discussion from the viewpoint of randomized policies as we consider a parameterized class of these in this paper. From now on, for simplicity, we shall refer to a randomized policy as simply a policy.

Let $J(\pi)$ be the long-run average reward under policy π. Let $\{s_t,\ t = 0, 1, 2, \ldots\}$ denote the MDP. The following assumption will be considered throughout the paper.

(A1) Under any policy π, the Markov chain resulting from the MDP $\{s_t,\ t = 0, 1, 2, \ldots\}$ is irreducible and aperiodic.

Let $d^\pi(s)$ denote the stationary probability of the Markov chain under policy π being in state $s \in \mathcal{S}$ and let $d^\pi = (d^\pi(s), s \in \mathcal{S})$. Our aim is to find a policy π that maximizes the long-run average reward $J(\pi)$ given by

$$J(\pi) = \lim_{T\to\infty} \frac{1}{T}\mathbf{E}\left[\sum_{t=0}^{T-1} r_{t+1}|\pi\right] = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)R(s,a). \tag{2}$$

The limit in (2) is well defined by (A1). Let π^{opt} denote the optimal policy

$$\pi^{opt} = \arg\max_\pi J(\pi).$$

Further, we shall denote by $Q^\pi(s,a)$, the expected differential reward associated with a state-action pair (s,a), given policy π, that is defined by

$$Q^\pi(s,a) = \sum_{t=1}^{\infty} \mathbf{E}[r_{t+1} - J(\pi)|s_0 = s, a_0 = a, \pi], \quad \forall s \in \mathcal{S},\ a \in \mathcal{A}.$$

Likewise, we denote by $V^\pi(s)$, the expected differential reward associated with a state s when actions are selected according to policy π. Here

$$V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(s,a)Q^\pi(s,a).$$

The Poisson equation under policy π is given by

$$J(\pi) + V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(s,a)[R(s,a) + \sum_{s'\in\mathcal{S}} P(s,a,s')V^\pi(s')], \quad \forall s \in \mathcal{S}. \tag{3}$$

In policy gradient methods, we define a class of parameterized randomized policies $\{\pi^\theta(s,.), s \in \mathcal{S}, \theta \in \mathbb{R}^{d_1}\}$, estimate the gradient of the average reward with respect to the policy parameters θ from the observed states, actions, and rewards, and then improve the policy by adjusting its parameters in the direction of an estimate of the gradient of J with respect to θ. Since in this setting a policy π is represented by its parameters θ, J can be viewed as a function of θ and by abuse of notation, we let $J(\theta)$ denote the long-run average reward when the parameter is θ. In what follows, we shall interchangeably use $J(\pi)$ or $J(\theta)$ to denote the long-run average reward when the policy π or its associated parameter θ are to be emphasized. We also drop θ from π^θ, and simply denote this quantity as π. The optimum parameter can now be obtained as

$$\theta^{opt} = \arg\max_\theta J(\theta).$$

In order for the gradient of the policy to be well defined, we shall make the following assumption.

(A2) For any state-action pair (s, a), $\pi(s, a)$ is continuously differentiable in the parameter θ.

Previous works [44, 53, 10] have shown that the gradient of the average reward for parameterized policies that satisfy (A1) and (A2) is given by[1]

$$\nabla J(\pi) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \nabla \pi(s, a) Q^\pi(s, a). \tag{4}$$

For the case of parameterized Markov processes, a similar expression can be found in [26]. Observe that if $b(s)$ is any given function of s (also called a *baseline*), then

$$\sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \nabla \pi(s, a) b(s) = \sum_{s \in \mathcal{S}} d^\pi(s) b(s) \nabla (\sum_{a \in \mathcal{A}} \pi(s, a)) = 0,$$

and thus the gradient of the average reward can be written as

$$\nabla J(\pi) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \nabla \pi(s, a) [Q^\pi(s, a) \pm b(s)], \tag{5}$$

for any $b(s)$. The baseline $b(s)$ can be chosen in a way that the variance of the gradient estimates $\nabla J(\pi)$ is minimized [33].

The *natural* gradient, denoted $\tilde{\nabla} J(\pi)$, can be calculated by linearly transforming the *regular* gradient, $\nabla J(\pi)$, using the inverse of the Fisher information matrix of the policy $\tilde{\nabla} J(\pi) = G^{-1}(\theta) \nabla J(\pi)$. The Fisher information matrix $G(\theta)$ is given by

$$G(\theta) = \mathbf{E}_{s \sim d^\pi, a \sim \pi} [\nabla \log \pi(s, a) \nabla \log \pi(s, a)^\top] = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi(s, a) \nabla \log \pi(s, a) \nabla \log \pi(s, a)^\top. \tag{6}$$

Matrix $G(\theta)$ plays an important role in the algorithms that use natural gradients [35, 46]. Here $\mathbf{E}_{s \sim d^\pi, a \sim \pi}[\cdot]$ denotes the expectation under the conditional joint distribution where states are first selected according to distribution d^π, and then given that a state s is selected, actions are selected according to distribution $\pi(s, \cdot)$. The Fisher information matrix is clearly positive definite [35]. In the next section, we shall develop certain multi-timescale actor-critic algorithms. While certain recursions in some of these algorithms use regular gradients, their counterparts that we also present use the more efficient *natural* gradients [46] and are based on the Fisher information matrix. In two of these algorithms, we obtain a least squares optimal parametric representation for the advantage function $A^\pi(s, a)$ along the faster timescale recursion, for any given θ. The parameter θ is then updated on the slower timescale. These ideas will become clear subsequently. We first present a few basic results for the case when the parameter θ in policy π is held fixed in Section 4. Our control algorithms will then be presented in Section 5.

A well-studied example of parameterized randomized policies, which we use in the experiments of this paper, is the Gibbs (or Boltzmann) distribution having the form

$$\pi(s, a) = \frac{e^{\theta^\top \phi_{sa}}}{\sum_{a' \in \mathcal{A}} e^{\theta^\top \phi_{sa'}}}, \quad \forall s \in \mathcal{S}, \ a \in \mathcal{A}, \tag{7}$$

[1] In the rest of the paper we use the notation ∇ to denote ∇_θ — the gradient with respect to the policy parameters.

where each ϕ_{sa} is a d_1-dimensional feature vector for the state-action pair (s, a). The Gibbs distribution has connections with statistical mechanics and is also used in other domains such as evolutionary algorithms [30] and the well-known simulated annealing search technique for multivariate optimization [3].

Before we proceed further, we first motivate two-timescale stochastic approximation [21] as our algorithms in the next section also use this technique.

3 Two-Timescale Stochastic Approximation Algorithms

These are typically characterized by coupled stochastic recursions that are driven by two different (decreasing) step-size schedules, of which one has a higher convergence rate to zero than the other. We present here more generally the setting of two-timescale stochastic approximations. Suppose X_t, Y_t, $t \geq 0$ be two parameter sequences that are governed according to

$$X_{t+1} = X_t + \alpha_t(f(X_t, Y_t) + N^1_{t+1}), \tag{8}$$

$$Y_{t+1} = Y_t + \beta_t(g(X_t, Y_t) + N^2_{t+1}), \tag{9}$$

where f, g are Lipschitz continuous functions and $\{N^1_t\}$, $\{N^2_t\}$ are martingale difference sequences w.r.t. the σ-fields $\bar{\mathcal{F}}_t = \sigma(X_n, Y_n, N^1_n, N^2_n, n \leq t)$, $t \geq 0$, satisfying

$$\mathbf{E}[\| N^i_{t+1} \|^2 | \bar{\mathcal{F}}_t] \leq D_1(1 + \| X_t \|^2 + \| Y_t \|^2), \ i = 1, 2, \ t \geq 0,$$

for some constant $D_1 < \infty$. Also, here $\{\alpha_t\}$ and $\{\beta_t\}$ are two step-size schedules that satisfy

$$\sum_t \alpha_t = \sum_t \beta_t = \infty, \ \sum_t \alpha_t^2, \sum_t \beta_t^2 < \infty, \tag{10}$$

$$\beta_t = o(\alpha_t). \tag{11}$$

As a consequence of (11), $\beta_t \to 0$ faster than $\{\alpha_t\}$. Hence (8) is a 'faster' recursion than (9) as beyond some t_0 (i.e., for $t \geq t_0$), (8) has uniformly higher increments as compared to (9). Consider the ODEs

$$\dot{X} = f(X(t), Y(t)), \tag{12}$$

$$\dot{Y} = 0. \tag{13}$$

Alternatively, as a consequence of (13), one can consider the ODE

$$\dot{X} = f(X(t), Y) \tag{14}$$

in place of (12), where because of (13), Y is a constant. We assume

(B1) $\sup_t \| X_t \|, \sup_t \| Y_t \| < \infty$.

(B2) The ODE (14) has a globally asymptotically stable equilibrium $\mu(Y)$ where $\mu(\cdot)$ is a Lipschitz continuous function.

Consider also the ODE

$$\dot{Y} = g(\mu(Y(t)), Y(t)). \tag{15}$$

We also assume

(B3) The ODE (15) has a globally asymptotically stable equilibrium $Y^\star$.

Define two real-valued sequences $\{r_t\}$ and $\{s_t\}$ as $r_t = \sum_{n=0}^{t-1} \alpha_n$ and $s_t = \sum_{n=0}^{t-1} \beta_n$, respectively. Note that $(r_t - r_{t-1}), (s_t - s_{t-1}) \to 0$ as $t \to \infty$. Define continuous time processes $\bar{X}(r)$, $\bar{Y}(r)$, $r \geq 0$ as $\bar{X}(r_t) = X_t$, $\bar{Y}(r_t) = Y_t$, respectively, with linear interpolations in between. For $s \geq 0$, let $X^s(r)$, $Y^s(r)$, $r \geq s$ denote the trajectories of (12)-(13) with $X^s(s) = \bar{X}(s)$ and $Y^s(s) = \bar{Y}(s)$. Note that because of (13), $Y^s(r) = \bar{Y}(s)$ $\forall r \geq s$. Now (8)-(9) can be viewed as 'noisy' Euler discretizations of the ODEs (12)-(13) when the time discretization corresponds to $\{r_t\}$. This is because (9) can be written as

$$Y_{t+1} = Y_t + \alpha_t \left(\frac{\beta_t}{\alpha_t} \left(g(X_t, Y_t) + N_{t+1}^2 \right) \right),$$

and (11) implies that the term multiplying α_t on the RHS above vanishes in the limit. One can now show (cf. [21]) using a sequence of approximations involving the Gronwall inequality that for any given $T > 0$, with probability one, $\sup_{r \in [s, s+T]} \| \bar{X}(r) - X^s(r) \| \to 0$ as $s \to \infty$. The same is also true for $\sup_{r \in [s, s+T]} \| \bar{Y}(r) - Y^s(r) \|$ as well. Further, using the time discretization $\{s_t\}$ for the ODE (15), a similar conclusion with regards to iteration (9) (and ODE (15)) can be drawn following a continuous time trajectory that is obtained with the iterates in (9) interpolated along the time line $\{s_t\}$. The following is the main two-timescale convergence result (cf. [21]).

Theorem 1 With probability one, $(X_t, Y_t) \to (\mu(Y^\star), Y^\star)$ as $t \to \infty$.

In the analysis of our algorithms in Section 6, we shall show that assumptions (B1)-(B3) above hold good. For this purpose, we shall show some key requirements in [22] that allow us to use the main convergence result there. The analysis in [22] is however only for the one-timescale case. Hence, we combine the same with the requirements for the two-timescale algorithms [21]. We shall skip details wherever (in Section 6) the analysis is along standard lines and appropriately point to suitable references.

4 Policy Gradient with Approximation

Now consider the case in which the state-action value function for a fixed policy π, Q^π, is approximated by a learned function approximator. If the approximation is sufficiently good, we might hope to use it in place of Q^π in Equations (4) and (5), and still point roughly in the direction of the true gradient. Sutton et al. [53] showed that if the approximation $\hat{Q}_w^\pi$ with parameter $w \in \mathbb{R}^{d_1}$ is *compatible*, i.e., $\nabla_w \hat{Q}_w^\pi(s,a) = \nabla \log \pi(s,a)$, and minimizes the mean squared error

$$\mathcal{E}^\pi(w) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi(s,a) [Q^\pi(s,a) - \hat{Q}_w^\pi(s,a)]^2, \tag{16}$$

for parameter value $w^\star$, then we can replace Q^π with $\hat{Q}^\pi_{w^\star}$ in Equations (4) and (5). Thus, we work with linear approximation $\hat{Q}^\pi_w(s,a) = w^\top \psi_{sa}$ in which the ψ_{sa}'s are *compatible* features defined according to $\psi_{sa} = \nabla \log \pi(s,a)$. Note that compatible features are well-defined under (A2). As an example, the compatible features for the Gibbs policy in Equation (7) are $\psi_{sa} = \phi_{sa} - \sum_{a' \in \mathcal{A}} \pi(s,a')\phi_{sa'}$.

The Fisher information matrix of Equation (6) can be written using the compatible features as

$$G(\theta) = \mathbf{E}_{s\sim d^\pi, a\sim\pi}[\psi_{sa}\psi_{sa}^\top] = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)\psi_{sa}\psi_{sa}^\top.$$

Suppose $\mathcal{E}^\pi(w)$ denotes the mean squared error

$$\mathcal{E}^\pi(w) = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)[Q^\pi(s,a) - w^\top\psi_{sa} - b(s)]^2, \tag{17}$$

of our compatible linear parameterized approximation $w^\top\psi_{sa}$ and an arbitrary baseline $b(s)$. Let $w^\star = \arg\min_w \mathcal{E}^\pi(w)$ denote the optimal weight parameter. We first show in Lemma 1 that the value of $w^\star$ does not depend on the given baseline $b(s)$ and as a result the mean squared error problems of Equations (16) and (17) have the same solutions. Next, in Lemma 2, we show that if the weight parameter is set to be equal to $w^\star$, the resulting mean squared error $\mathcal{E}^\pi(w^\star)$ (now treated as a function of the baseline $b(s)$) is further minimized when $b(s) = V^\pi(s)$. In other words, the variance in the state-action value function estimator is minimized if the baseline is chosen to be the value function itself.

Lemma 1 The optimum weight parameter $w^\star$ for any given θ satisfies[2]

$$w^\star = G(\theta)^{-1}\mathbf{E}_{s\sim d^\pi, a\sim\pi}[Q^\pi(s,a)\psi_{sa}].$$

Proof Note that

$$\nabla_w \mathcal{E}^\pi(w) = -\sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a) 2[Q^\pi(s,a) - w^\top\psi_{sa} - b(s)]\psi_{sa}. \tag{18}$$

Equating the above to zero, one obtains

$$\sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)\psi_{sa}\psi_{sa}^\top w^\star = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)Q^\pi(s,a)\psi_{sa} - \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)b(s)\psi_{sa}.$$

The last term on the right hand side equals zero since $\sum_{a\in\mathcal{A}} \pi(s,a)\psi_{sa} = 0$. Now from (18), the Hessian $\nabla_w^2 \mathcal{E}^\pi(w)$ of $\mathcal{E}^\pi(w)$ evaluated at $w^\star$ can be seen to be $2G(\theta)$. The claim follows since $G(\theta)$ is positive definite for any θ. □

Next (as stated above), given the optimum weight parameter $w^\star$, we obtain the minimum variance baseline corresponding to policy π. Thus we consider now $\mathcal{E}^\pi(w^\star)$ and obtain $b^\star(s) = \arg\min_{b=b(s)\ ,\ s\in\mathcal{S}} \mathcal{E}^\pi(w^\star)$.

[2]This lemma is similar to Theorem 1 in [35].

Lemma 2 For any given policy π, the minimum variance baseline $b^\star(s)$ corresponds to the value function $V^\pi(s)$.

Proof For any $s \in \mathcal{S}$, let $\mathcal{E}^{\pi,s}(w^\star)$ denote

$$\mathcal{E}^{\pi,s}(w^\star) = \sum_{a\in\mathcal{A}} \pi(s,a)[Q^\pi(s,a) - w^{\star\top}\psi_{sa} - b(s)]^2.$$

Then $\mathcal{E}^\pi(w^\star) = \sum_{s\in\mathcal{S}} d^\pi(s)\mathcal{E}^{\pi,s}(w^\star)$. Note that by Assumption (A1), the Markov chain corresponding to any policy π is positive recurrent since the number of states is finite. Hence, $d^\pi(s) > 0$ for all $s \in \mathcal{S}$. Thus, for each $s \in \mathcal{S}$, one needs to find the optimum $b(s)$ in $\mathcal{E}^{\pi,s}(w^\star)$. Now for any $s \in \mathcal{S}$,

$$\frac{\partial \mathcal{E}^{\pi,s}(w^\star)}{\partial b(s)} = -\sum_{a\in\mathcal{A}} \pi(s,a)2[Q^\pi(s,a) - w^{\star\top}\psi_{sa} - b(s)].$$

Equating the above to zero, we obtain

$$b^\star(s) = \sum_{a\in\mathcal{A}} \pi(s,a)Q^\pi(s,a) - \sum_{a\in\mathcal{A}} \pi(s,a)w^{\star\top}\psi_{sa}.$$

The last term again equals zero since $\sum_{a\in\mathcal{A}} \pi(s,a)\psi_{sa} = 0$. Hence $b^\star(s) = \sum_{a\in\mathcal{A}} \pi(s,a)Q^\pi(s,a) = V^\pi(s)$. The second derivative of $\mathcal{E}^{\pi,s}(w^\star)$ with respect to $b(s)$ is equal to 2. The claim follows. □

From Lemmas 1 and 2, $w^{\star\top}\psi_{sa}$ is a least squared optimal parametric representation for the *advantage* function $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ as well as the state-action value function $Q^\pi(s,a)$. However, since $\sum_{a\in\mathcal{A}} \pi(s,a)w^\top\psi_{sa} = 0,\ \forall s \in \mathcal{S}$, it is better to think of $w^\top\psi_{sa}$ as an approximation of the advantage function $A^\pi(s,a)$ rather than of the state-action value function $Q^\pi(s,a)$.

The temporal-difference (TD) error δ_t is a random quantity that is defined according to

$$\delta_t = r_{t+1} - \hat{J}_{t+1} + \hat{V}_{s_{t+1}} - \hat{V}_{s_t}, \tag{19}$$

where $\hat{V}_{s_i}$ is a consistent estimates of the differential reward in state s_i, $i = t, t+1$. Likewise, $\hat{J}_{t+1}$ is a consistent estimate of the average reward. Thus, in particular, these estimates satisfy $\mathbf{E}[\hat{V}_{s_t}|s_t,\pi] = V^\pi(s_t)$ and $\mathbf{E}[\hat{J}_{t+1}|s_t,\pi] = J(\pi)$, respectively, for any $t \geq 0$. We assume here that actions are chosen according to policy π. The next lemma shows that δ_t is a consistent estimate of the advantage function A^π.

Lemma 3 Under given policy π with actions chosen according to it, we have

$$\mathbf{E}[\delta_t|s_t,a_t] = A^\pi(s_t,a_t).$$

Proof Note that

$$\mathbf{E}[\delta_t|s_t,a_t] = \mathbf{E}[r_{t+1} - \hat{J}_{t+1} + \hat{V}_{s_{t+1}} - \hat{V}_{s_t}|s_t,a_t] = R(s_t,a_t) - J(\pi) + \mathbf{E}[\hat{V}_{s_{t+1}}|s_t,a_t] - V^\pi(s_t).$$

Now

$$\mathbf{E}[\hat{V}_{s_{t+1}}|s_t,a_t] = \mathbf{E}[\mathbf{E}[\hat{V}_{s_{t+1}}|s_{t+1}] \mid s_t,a_t] = \mathbf{E}[V^\pi(s_{t+1})|s_t,a_t] = \sum_{s_{t+1}\in\mathcal{S}} P(s_t,a_t,s_{t+1})V^\pi(s_{t+1}).$$

Also, $R(s_t,a_t) - J(\pi) + \sum_{s_{t+1}\in\mathcal{S}} P(s_t,a_t,s_{t+1})V^\pi(s_{t+1}) = Q^\pi(s_t,a_t)$. The claim follows. □

By setting the baseline $b(s)$ equal to the value function $V^\pi(s)$, Equation (5) can be written as $\nabla J(\pi) = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)\psi_{sa}A^\pi(s,a)$. From Lemma 3, $\hat{\nabla}J(\pi) = \delta_t\psi_{s_ta_t}$ is a consistent estimate of $\nabla J(\pi)$.

However, calculating δ_t requires having estimates of average reward $\hat{J}$ and value function $\hat{V}$. While an average reward estimate is simple enough to obtain given the single stage reward function, the same is not necessarily true for the value function. We use function approximation for the value functions as well. Suppose f_s is a d_2-dimensional feature vector for state s (for some $d_2 \geq 1$). We denote $f_s = (f_s(1),\ldots,f_s(d_2))^\top$. One may then approximate $V^\pi(s)$ with $v^\top f_s$, where v is a d_2-dimensional weight vector which can be tuned (for a fixed policy π) using a TD algorithm. In our algorithms, we then use

$$\delta_t = r_{t+1} - \hat{J}_{t+1} + v_t^\top f_{s_{t+1}} - v_t^\top f_{s_t} \tag{20}$$

as an estimate for the TD error, where v_t corresponds to the tth update of the value function weight parameter. From now on, unless explicitly mentioned, we shall consider δ_t to be defined according to (20). Let $\bar{V}^\pi(s)$ denote the quantity

$$\bar{V}^\pi(s) = \sum_{a\in\mathcal{A}} \pi(s,a)[R(s,a) - J(\pi) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi\top}f_{s'}], \tag{21}$$

where $v^{\pi\top}f_{s'}$ is an estimate of the differential value function $V^\pi(s')$ that is obtained upon convergence of a TD recursion (above) viz., $\lim_{t\to\infty} v_t = v^\pi$ with probability one. Let also δ_t^π denote the associated quantity

$$\delta_t^\pi = r_{t+1} - \hat{J}_{t+1} + v^{\pi\top}f_{s_{t+1}} - v^{\pi\top}f_{s_t}.$$

Here r_{t+1} and $\hat{J}_{t+1}$ are the same as before. Then δ_t^π corresponds to a stationary estimate of the TD error (with function approximation) under policy π. We have the following analog of Theorem 1 of [53].

Lemma 4 $\mathbf{E}[\delta_t^\pi\psi_{s_ta_t}|\theta] = \nabla J(\pi) + \sum_{s\in\mathcal{S}} d^\pi(s)[\nabla\bar{V}^\pi(s) - \nabla v^{\pi\top}f_s]$.

Proof A simple calculation shows that

$$\mathbf{E}[\delta_t^\pi\psi_{s_ta_t}|\theta] = \mathbf{E}[\mathbf{E}[\delta_t^\pi|s_t,a_t]\psi_{s_ta_t}|\theta]$$

$$= \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \nabla\pi(s,a)[R(s,a) - J(\pi) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi\top}f_{s'} - v^{\pi\top}f_s]. \tag{22}$$

Now from (21),

$$\nabla\bar{V}^\pi(s) = \sum_{a\in\mathcal{A}} \nabla\pi(s,a)[R(s,a) - J(\pi) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi\top}f'_s]$$

$$+\sum_{a\in\mathcal{A}}\pi(s,a)[-\nabla J(\pi)+\sum_{s'\in\mathcal{S}}P(s,a,s')\nabla v^{\pi^\top}f_{s'}].$$

Thus, from (22) and the above, we get

$$\sum_{s\in\mathcal{S}}d^\pi(s)\nabla\bar{V}^\pi(s)=\mathbf{E}[\delta_t^\pi\psi_{s_ta_t}|\theta]-\nabla J(\pi)+\sum_{s\in\mathcal{S}}d^\pi(s)\sum_{a\in\mathcal{A}}\pi(s,a)\sum_{s'\in\mathcal{S}}P(s,a,s')\nabla v^{\pi^\top}f_{s'}. \tag{23}$$

Now observe that $d^\pi(s)$ correspond to the stationary probabilities that satisfy

$$d^\pi(s)=\sum_{s''\in\mathcal{S}}d^\pi(s'')p^\pi(s'',s),\ \ s\in\mathcal{S},\ \text{with}\ \sum_{s''\in\mathcal{S}}d^\pi(s'')=1, \tag{24}$$

where $p^\pi(s'',s)=\sum_{a\in\mathcal{A}}\pi(s'',a)P(s'',a,s)$ are the transition probabilities of the resulting Markov chain under policy π. Hence,

$$\begin{aligned}\sum_{s\in\mathcal{S}}d^\pi(s)\sum_{a\in\mathcal{A}}\pi(s,a)\sum_{s'\in\mathcal{S}}P(s,a,s')\nabla v^{\pi^\top}f_{s'}&=\sum_{s\in\mathcal{S}}d^\pi(s)\sum_{s'\in\mathcal{S}}p^\pi(s,s')\nabla v^{\pi^\top}f_{s'}\\&=\sum_{s'\in\mathcal{S}}\sum_{s\in\mathcal{S}}d^\pi(s)p^\pi(s,s')\nabla v^{\pi^\top}f_{s'}=\sum_{s'\in\mathcal{S}}d^\pi(s')\nabla v^{\pi^\top}f_{s'}.\end{aligned} \tag{25}$$

The claim now follows from (23). □

Note that according to Theorem 1 of [53], $\mathbf{E}[\delta_t\psi_{s_ta_t}|\theta]=\nabla J(\pi)$, provided δ_t is defined according to (19). For the case with function approximation that we study, from Lemma 4, the quantity $\sum_{s\in\mathcal{S}}d^\pi(s)[\nabla\bar{V}^\pi(s)-\nabla v^{\pi^\top}f_s]$ may be viewed as the error or bias in the estimate of the gradient of average reward that results from the use of function approximation. It is interesting to observe that this does not depend on the differential reward $V^\pi(s)$ that is obtained as a solution to (3). We also have

Corollary 1 $\sum_{s\in\mathcal{S}}d^\pi(s)[\bar{V}^\pi(s)-v^{\pi^\top}f_s]=0.$

Proof This follows directly from the definition of $\bar{V}^\pi(s)$ in (21), the definition of $J(\pi)$ in (2), and an analogous equation as (25) with $v^{\pi^\top}f_{s'}$ in place of $\nabla v^{\pi^\top}f_{s'}$. □

5 Actor-Critic Algorithms

We present four new actor-critic algorithms in this section. These algorithms are in the general form shown in Table 1. They update the policy parameters along the direction of the average reward gradient. While estimates of the *regular* gradient are used for this purpose in Algorithm 1, *natural* gradient estimates are used in Algorithms 2-4. While critic updates in our algorithms can be easily extended to the case of TD(λ), $\lambda>0$, we restrict our attention to the case when $\lambda=0$. Let $\{s_t,\ t=0,1,2,\ldots\}$ denote the MDP. Also, let $\hat{V}(s,v)=v^\top f_s$ denote the parameterized approximation to the differential reward in state s. One can also denote the same as $\hat{V}(v)=\Phi v$, where Φ is an $n\times d_2$–dimensional matrix whose kth column ($k=1,\ldots,d_2$) is $f(k)=(f_s(k)\ ,\ s\in\mathcal{S})^\top$. We make the following assumption as in [58, 59].

(A3) The basis functions $\{f(k),\ k = 1, \ldots, d_2\}$ are linearly independent. In particular, $d_2 \leq n$ and Φ has full rank. Also, for every $v \in \mathbb{R}^{d_2}$, $\Phi v \neq e$, where e is the n-dimensional vector with all entries equal to one.

Let $\{\alpha_t\}$ and $\{\beta_t\}$ be two step-size schedules that satisfy (10)-(11). As a consequence of (10)-(11), $\beta_t \to 0$ faster than α_t. Hence as explained in the couple of lines following (11), actor is a faster recursion than critic. In addition, we assume that $\alpha_t < 1,\ \forall t \geq 0$. This is however required only for Algorithms 2-4. We set the average reward step-size $\xi_t = c\alpha_t$, for a positive scalar c. However, more general step-sizes may be chosen. For instance, it may be desirable in some cases to have the average reward update move on a faster timescale as compared to critic (in which case it will converge faster than critic does).

Table 1: A Template for AC Algorithms.

1:	**Input:**	
	• Randomized parameterized policy $\pi^{\theta}(\cdot, \cdot)$,	
	• Value function feature vector f_s.	
2:	**Initialization:**	
	• Policy parameters $\theta = \theta_0$,	
	• Value function weight vector $v = v_0$,	
	• Step sizes $\alpha = \alpha_0,\ \ \beta = \beta_0,\ \ \xi = c\alpha_0$.	
3:	**for** $t = 0, 1, 2, \ldots$ **do**	
4:	**Execution:**	
	• Draw action $a_t \sim \pi^{\theta_t}(s_t, a_t)$,	
	• Observe next state $s_{t+1} \sim P(s_t, a_t, s_{t+1})$,	
	• Observe reward r_{t+1}.	
5:	**Average Reward Update:**	$\hat{J}_{t+1} = (1 - \xi_t)\hat{J}_t + \xi_t r_{t+1}$
6:	**TD Error:**	$\delta_t = r_{t+1} - \hat{J}_{t+1} + v_t^{\top} f_{s_{t+1}} - v_t^{\top} f_{s_t}$
7:	**Critic Update:**	algorithm specific
8:	**Actor Update:**	algorithm specific
9 :	**endfor**	
10:	**return** Policy and value function parameters θ, v	

We now present the critic and the actor updates of our four actor-critic algorithms.

Algorithm 1:

$$\textbf{Critic Update:} \qquad v_{t+1} = v_t + \alpha_t \delta_t f_{s_t}, \tag{26}$$

$$\textbf{Actor Update:} \qquad \theta_{t+1} = \theta_t + \beta_t \delta_t \psi_{s_t a_t}. \tag{27}$$

This is the only actor-critic algorithm presented in the paper that is based on the regular gradient estimate.

The next algorithm is based on the natural gradient estimate $\tilde{\nabla} J(\theta_t) = G(\theta_t)^{-1}\delta_t \psi_{s_t a_t}$ in place of the regular gradient estimate in Algorithm 1. We derive a procedure below for recursively estimating $G(\theta)^{-1}$ on a faster timescale. The above estimation is done on a faster scale so that convergence of the associated iterates is achieved prior to a θ-update. Suppose G_t^{-1} denote the tth estimate of $G(\theta)^{-1}$. Our procedure is obtained in a similar manner as the method described on pp. 147-152 of [61]. The latter approach however considers the estimates as being obtained via a "fading memory" condition in which the most recent observation is given the highest weight. The weights themselves decrease geometrically over past observations. On the other hand, unlike [61], we consider stationary averages that depend on parameter θ, that in turn gets updated along the "slower timescale". This constitutes a natural setting for our algorithm. We show in Lemma 6 that $G_t^{-1} \to G(\theta)^{-1}$ as $t \to \infty$ with probability one. This is required for proving convergence of our algorithm. On the other hand, showing the same for the corresponding estimates in [61] does not seem possible as $G_t \not\to G(\theta)$ there.

Note that in the case where G_t, $t \geq 0$ are defined as (the sample averages)

$$G_t = \frac{1}{t+1} \sum_{l=0}^{t} \psi_{s_l a_l} \psi_{s_l a_l}^\top,$$

one may obtain recursively

$$G_t = \left(1 - \frac{1}{t+1}\right) G_{t-1} + \frac{1}{t+1} \psi_{s_t a_t} \psi_{s_t a_t}^\top. \tag{28}$$

More generally, one may consider the recursion

$$G_t = (1 - \alpha_t) G_{t-1} + \alpha_t \psi_{s_t a_t} \psi_{s_t a_t}^\top, \tag{29}$$

where the step-size α_t is as before. This would correspond to a case of weighted averages (with the weights corresponding to the step-sizes α_t), however, through a stochastic approximation argument, one can see that (29) would asymptotically converge to $G(\theta)$, almost surely, if θ is held fixed. In fact, with an appropriate choice of $\{\alpha_t\}$, one can obtain faster convergence of iterates in (29) over those in (28). Using Sherman-Morrison matrix inversion lemma, one obtains

$$G_t^{-1} = \frac{1}{1 - \alpha_t} \left[G_{t-1}^{-1} - \alpha_t \frac{(G_{t-1}^{-1} \psi_{s_t a_t})(G_{t-1}^{-1} \psi_{s_t a_t})^\top}{1 - \alpha_t + \alpha_t \psi_{s_t a_t}^\top G_{t-1}^{-1} \psi_{s_t a_t}} \right]. \tag{30}$$

We thus have the following algorithm:

Algorithm 2:

$$\textbf{Critic Update:} \qquad v_{t+1} = v_t + \alpha_t \delta_t f_{s_t}, \tag{31}$$

$$\textbf{Actor Update:} \qquad \theta_{t+1} = \theta_t + \beta_t G_t^{-1} \delta_t \psi_{s_t a_t}. \tag{32}$$

where the estimate of the inverse of the Fisher information matrix is updated according to Equation (30). As with [61], we let $G_0^{-1} = kI$, where I is a $d_1 \times d_1$-dimensional identity matrix and $k > 0$ is

a large constant. Thus G_0^{-1} and hence also G_0 are positive definite and symmetric matrices. From (29), $G_t,\ t \geq 1$ can be seen to be positive definite and symmetric as these are convex combinations of positive definite and symmetric matrices. Hence, $G_t^{-1},\ t \geq 1$ are positive definite and symmetric matrices as well.

As we mentioned in Section 4, it is better to think of the *compatible* approximation $w^\top \psi_{sa}$ as an approximation of the advantage function rather than of the state-action value function. In our next algorithm, we tune the weight parameters w in a way as to minimize an estimate of the least squared error $\mathcal{E}^\pi(w) = \mathbf{E}_{s\sim d_\pi, a\sim\pi}[(w^\top \psi_{sa} - A^\pi(s,a))^2]$. Note that the gradient of $\mathcal{E}^\pi(w)$ is

$$\nabla_w \mathcal{E}^\pi(w) = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a) 2[w^\top \psi_{sa} - A^\pi(s,a)]\psi_{sa}.$$

We use the following estimate of $\nabla_w \mathcal{E}^\pi(w)$.

$$\hat{\nabla}_w \mathcal{E}^\pi(w) = 2(\psi_{s_t a_t} \psi_{s_t a_t}^\top w - \delta_t \psi_{s_t a_t}). \tag{33}$$

Hence, we update parameters w along with value function parameters v in the critic update of this algorithm as

$$w_{t+1} = w_t - \alpha_t \hat{\nabla}_{w_t} \mathcal{E}^\pi(w_t) = w_t - \alpha_t(\psi_{s_t a_t} \psi_{s_t a_t}^\top w_t - \delta_t \psi_{s_t a_t}).$$

The factor 2 on the RHS of (33) does not play a role because of the diminishing step-size sequence $\alpha_t,\ t \geq 0$ and so has been dropped in the above recursion. We maximize the long-run average reward $J(\theta)$ along the slower timescale. We use the natural gradient estimate for this purpose. As with [46], we use the natural gradient estimate $\tilde{\nabla} J(\theta_t) = w_{t+1}$ in the actor update of Algorithm 3.

Algorithm 3:

Critic Update:

$$v_{t+1} = v_t + \alpha_t \delta_t f_{s_t}, \tag{34}$$

$$w_{t+1} = [I - \alpha_t \psi_{s_t a_t} \psi_{s_t a_t}^\top] w_t + \alpha_t \delta_t \psi_{s_t a_t} \tag{35}$$

Actor Update:

$$\theta_{t+1} = \theta_t + \beta_t w_{t+1}. \tag{36}$$

Although the estimates of $G(\theta)^{-1}$ are not explicitly computed and used in Algorithm 3, the convergence analysis of this algorithm in the next section shows that the overall scheme still moves in the direction of the natural gradient of average reward.

In Algorithm 4, however, we explicitly estimate $G(\theta)^{-1}$ (as in Algorithm 2), and use it in the critic update for w. The overall scheme is again seen to follow the direction of the natural gradient of average reward. Here, we let

$$\tilde{\nabla}_w \mathcal{E}^\pi(w) = G_t^{-1} 2(\psi_{s_t a_t} \psi_{s_t a_t}^\top w - \delta_t \psi_{s_t a_t}) \tag{37}$$

be the estimate of the natural gradient of $\mathcal{E}^\pi(w)$. This also results in a simplification of our faster timescale recursion (critic update). Further, we remove the factor 2 from the natural gradient estimate (37) because of diminishing $\alpha_t,\ t \geq 0$ as before.

Algorithm 4:

$$\textbf{Critic Update:} \qquad v_{t+1} = v_t + \alpha_t \delta_t f_{s_t}, \tag{38}$$

$$w_{t+1} = (1 - \alpha_t) w_t + \alpha_t G_t^{-1} \delta_t \psi_{s_t a_t}. \tag{39}$$

$$\textbf{Actor Update:} \qquad \theta_{t+1} = \theta_t + \beta_t w_{t+1}, \tag{40}$$

where the estimate of the inverse of the Fisher information matrix is updated according to Equation (30). As with Algorithm 2, we let $G_0^{-1} = kI$ with $k > 0$ is a large constant.

6 Convergence Analysis

We now present the convergence analysis of our algorithms. The analysis mainly follows the ordinary differential equation (ODE) approach. Note that the problem we consider is a maximization and not a minimization problem. For the purpose of analysis, we consider an associated problem with costs defined as negative rewards and our aim is to minimize the associated long-run average cost. The negative of the minimum cost thus obtained then corresponds to the maximum reward in the original problem. This is useful in pushing through certain stability arguments and showing convergence of iterates. Our algorithms use function approximation and aim at finding the local maxima of the average rewards. All our convergence results are in the Euclidean norm. Further, for any matrix A, we define its norm as the induced matrix norm $\| A \| = \max_{\{x | \|x\|=1\}} \| Ax \|$.

6.1 Convergence Analysis for Algorithm 1

We require Assumptions (A1)–(A3) here. As explained above, one may view $-r_{t+1}$ as the cost incurred at instant t in a transformed problem. Because of the above, a change occurs only in the actor recursion (27) due to this transformation, and it becomes

$$\theta_{t+1} = \theta_t - \beta_t \delta_t \psi_{s_t a_t}. \tag{41}$$

Recursions for average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (26) being fixed point recursions (see [59]) are left unchanged. Unfortunately, because of the use of function approximation, it appears difficult to show boundedness of $\{\theta_t\}$. We explain the reasons for this in Remark 1 below. In particular, the iterates in (41) can be seen to be almost surely bounded using the stochastic Lyapunov function approach described in [40], provided the look up table representation is used as it is easy to obtain in such a case the form of the Lyapunov function to use. An exact functional form of the Lyapunov function, however, appears difficult to obtain when function approximation is used. For the purposes of proving convergence of the proposed scheme, we replace (41) by

$$\theta_{t+1} = \Gamma(\theta_t - \beta_t \delta_t \psi_{s_t a_t}), \tag{42}$$

where $\Gamma(\cdot)$ is the projection onto a compact set $C = \{x \mid q_i(x) \leq 0,\ i = 1, \ldots, s\} \subset \mathbb{R}^{d_1}$, where $q_i(\cdot)$, $i = 1, \ldots, s$ are real-valued, continuously differentiable functions on $\mathbb{R}^{d_1}$ that represent the

constraints specifying the (above) compact region. Suppose also that for each x on the boundary of C, the gradients of the active constraints are linearly independent. This is the setting considered for projection based algorithms in Chapter 5 of [39]. For any $x \in \mathbb{R}^{d_1}$, $\Gamma(x) \in C$ and in particular for $x \in C$, $\Gamma(x) = x$ itself. As explained in Chapter 2 of [39], any compact hyperrectangle in $\mathbb{R}^{d_1}$ is a special case of C (above). The projection method is an often used technique to ensure boundedness of iterates in stochastic approximation algorithms, see for instance, [2] where it has been used in the context of a stochastic shortest path queue learning algorithm. Some discussion on this is also available in [57]. The other approach (that is also usually taken, which we do not follow) is to simply assume that the iterates (41) are bounded and show convergence of these under this assumption. In our experiments, however, we do not project the iterates to a constraint region as they are seen to remain bounded.

Note that recursions for average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (26) move on the faster timescale or step-size schedule, hence converge faster, while (42) moves slower [21], see the discussion in Section 3. For any given policy π (along the faster timescale), average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (26) recursions correspond to the TD(λ) recursions in [59] with $\lambda = 0$. In [59], the updates in these recursions are rewritten as

$$\mu_{t+1} = \mu_t + \alpha_t(A(X_t)\mu_t + b(X_t)),$$

where, $X_t = (s_t, s_{t+1}, f_{s_t})$ is another associated Markov chain under π, $\mu_t = (J_t, v_t)^\top$, and $A(X_t)$, $b(X_t)$ are suitably defined matrix and column vector respectively.

Let D denote the diagonal matrix with elements $d^\pi(s_1), \ldots, d^\pi(s_n)$ along its diagonal. Let P^π be the probability matrix with elements $p^\pi(s, s') = \sum_{a \in \mathcal{A}} \pi(s, a) P(s, a, s')$, $s, s' \in \mathcal{S}$. Let R^π be the column vector $(\sum_{a \in \mathcal{A}} \pi(s_1, a) R(s_1, a), \ldots, \sum_{a \in \mathcal{A}} \pi(s_n, a) R(s_n, a))^\top$. Also, let $T : \mathbb{R}^n \to \mathbb{R}^n$ be the operator given by

$$T(J) = R^\pi - J(\pi) + P^\pi J.$$

The proof of convergence of TD(λ) in [59] is based on a result from [14]. We provide in Lemma 5 an alternative simpler proof of convergence using a recently developed result in [22]. We consider $\lambda = 0$ to suit our algorithm. The proof however carries through quite easily for $\lambda > 0$ as well. We have

Lemma 5 For any given π and $\{\hat{J}_t\}$, $\{v_t\}$ as in the average reward recursion (Line 5 in Table 1) and the critic recursion (26), we have $\hat{J}_t \to J(\pi)$ and $v_t \to v^\pi$ with probability one, where

$$J(\pi) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi(s, a) R(s, a), \tag{43}$$

is the average cost under π and v^π is obtained as the solution to

$$\Phi' D \Phi v^\pi = \Phi' D T(\Phi v^\pi). \tag{44}$$

Proof First consider the average reward recursion (Line 5 in Table 1). The ODE describing the asymptotic behavior of this recursion corresponds to

$$\dot{\eta} = -\eta + \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{s \in \mathcal{A}} \pi(s, a) R(s, a). \tag{45}$$

Let $f(\eta)$ denote the RHS of (45). Then $f(\eta)$ is Lipschitz continuous in η. Let $f_\infty(\eta) = \lim_{r\to\infty} \frac{f(r\eta)}{r}$. The function $f_\infty(\eta)$ exists and is simply $f_\infty(\eta) = -\eta$. The origin is an asymptotically stable equilibrium for the ODE $\dot{\eta} = f_\infty(\eta)$, with $V_1(\eta) = \eta^2/2$ serving as an associated Lyapunov function.

Now consider recursions for TD-error (Line 6 in Table 1) and critic (26). Consider the following ODE associated with them.

$$\dot{v} = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)[R(s,a) - J(\pi) + v^\top \sum_{s'\in\mathcal{S}} P(s,a,s')f_{s'} - v^\top f_s]f_s. \tag{46}$$

Let $g^1(v)$ denote the RHS of (46). Then $g^1(v)$ is also Lipschitz continuous in v. Further, for $g^1_\infty(v) \stackrel{\triangle}{=} \lim_{r\to\infty} \frac{g^1(rv)}{r}$, it can be seen that $g^1_\infty(v)$ exists and equals

$$g^1_\infty(v) = \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi(s,a)[v^\top \sum_{s'\in\mathcal{S}} P(s,a,s')f_{s'} - v^\top f_s]f_s.$$

The origin can again be shown [20] to be an asymptotically stable equilibrium for the ODE $\dot{v} = g^1_\infty(v)$.

Now, from average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (26) recursions, define $N^1(t)$, $M^1(t)$, $t \geq 0$, according to

$$N^1(t) = r_{t+1} - \mathbf{E}[r_{t+1} \mid \mathcal{F}_1(t)], \qquad M^1(t) = \delta_t f_{s_t} - \mathbf{E}[\delta_t f_{s_t} \mid \mathcal{F}_1(t)],$$

respectively, where $\mathcal{F}_1(t) = \sigma(v_r, \hat{J}_r, M^1(r), N^1(r), r \leq t)$. It is easy to see that

$$\mathbf{E}[\| N^1(t+1) \|^2 | \mathcal{F}_1(t)] \leq C_1(1 + \| \hat{J}_t \|^2 + \| v_t \|^2), \ t \geq 0,$$

$$\mathbf{E}[\| M^1(t+1) \|^2 | \mathcal{F}_1(t)] \leq C_2(1 + \| v_t \|^2 + \| \hat{J}_t \|^2), \ t \geq 0,$$

for some $C_1, C_2 < \infty$. In fact, quantities $N^1(t)$ can be directly seen to be uniformly bounded almost surely. Thus Assumptions (A1) and (A2) of [22] can be seen to be satisfied in the case of the average reward (Line 5 in Table 1), the TD-error (Line 6 in Table 1), and the critic (26) recursions. From Theorem 2.1 of [22], average reward, TD-error, and critic iterates are uniformly bounded with probability one. Now note that (45) has $J(\pi)$ defined as in (43) as its unique globally asymptotically stable equilibrium with $V_2(\eta) = (\eta - J(\pi))^2$ serving as the associated Lyapunov function. For (46), v^π can be shown as in [20] to be a globally asymptotically stable equilibrium. The claim now follows from Theorem 2.2, pp. 450 of [22]. □

Note that from (44), by premultiplying both sides by $\Phi(\Phi' D\Phi)^{-1}$, one gets

$$\Phi v^\pi = \Phi(\Phi' D\Phi)^{-1}\Phi' DT(\Phi v^\pi) = \Pi T(\Phi v^\pi),$$

where $\Pi = \Phi(\Phi' \ D\Phi)^{-1}\Phi' D$ corresponds to the projection matrix that projects onto the subspace spanned by the basis functions and satisfies for any $J \in \mathbb{R}^n$,

$$\Pi J = \underset{\bar{J}\in\{\Phi r | r\in\mathbb{R}^{d_2}\}}{\arg\min} \ \| J - \bar{J} \|_D,$$

with respect to the weighted norm $\|\cdot\|_D$ (see [59]).

Consider now recursion (42) along the slower timescale corresponding to β_t. Let $v(\cdot)$ be a vector field on C. Define another vector field

$$\hat{\Gamma}(v(y)) = \lim_{0<\eta\to 0}\left(\frac{\Gamma(y+\eta v(y))-y}{\eta}\right).$$

In case the above limit is not unique, we let $\hat{\Gamma}(v(y))$ be the set of all possible limit points (see pp. 191 of [39]). Consider now the ODE

$$\dot{\theta} = \hat{\Gamma}\left(-\sum_s d^{\pi}(s)\sum_a \nabla\pi^{\theta}(s,a)(R(s,a)-J(\pi)+\sum_{s'}P(s,a,s')v^{\pi\top}f_{s'})\right). \tag{47}$$

Let $H(\theta)$ denote the RHS of (47). Also, let $\mathcal{Z}$ denote the set of asymptotically stable equilibria of (47) that is contained within the set $\{\theta \mid H(\theta)=0\}$. Note that because of the projection $\Gamma(\cdot)$, (47) may have spurious fixed points on the boundary of the constraint set C, see [40] for detailed discussions. We obtain

Theorem 2 For the parameter iterations given by Algorithm 1, we have $(\hat{J}_t, v_t, \theta_t) \to \{(J(\theta^\star), v^{\pi^\star}, \theta^\star) \mid \theta^\star \in \mathcal{Z}\}$ as $t\to\infty$ with probability one, where $\pi^\star$ is the policy corresponding to $\theta^\star$.

Proof Let $\mathcal{F}_2(t) = \sigma(\theta_r,\ r\le t)$ denote the sequence of σ-fields generated by $\theta_r,\ r\ge 0$. We have

$$\theta_{t+1} = \Gamma(\theta_t - \beta_t\mathbf{E}[\delta_t^{\pi_t}\psi_{s_ta_t}\mid\mathcal{F}_2(t)] - \beta_t(\delta_t\psi_{s_ta_t} - \mathbf{E}[\delta_t\psi_{s_ta_t}\mid\mathcal{F}_2(t)]) - \beta_t\mathbf{E}[(\delta_t-\delta_t^{\pi_t})\psi_{s_ta_t}\mid\mathcal{F}_2(t)]),$$

where π_t is the policy corresponding to θ_t. Since the critic converges along the faster timescale, from Lemma 5, it follows that $\mathbf{E}[(\delta_t-\delta_t^{\pi_t})\psi_{s_ta_t}\mid\mathcal{F}_2(t)] = o(1)$. Now let

$$M^2(t) = \sum_{r=0}^{t-1}\beta_r(\delta_r\psi_{s_ra_r} - \mathbf{E}[\delta_r\psi_{s_ra_r}\mid\mathcal{F}_2(t)]),\ \ t\ge 1.$$

The quantities δ_t can be seen to be uniformly bounded since from the proof in Lemma 5, $\{\hat{J}_{t+1}\}$ and $\{v_t\}$ are bounded sequences. It is now easy to see [19] using (10) that $\{M^2(t)\}$ is a convergent martingale sequence. This implies that $\beta_t(\delta_t\psi_{s_ta_t} - \mathbf{E}[\delta_t\psi_{s_ta_t}\mid\mathcal{F}_2(t)]) = o(1)$ as well. Next, it can be seen using similar arguments as before (see proof of Lemma 4) that

$$\mathbf{E}[\delta_t^{\pi_t}\psi_{s_ta_t}\mid\theta_t] = \sum_{s\in\mathcal{S}}d^{\pi_t}(s)\sum_{a\in\mathcal{A}}\nabla\pi_t(s,a)[R(s,a)-J(\pi_t)+\sum_{s'\in\mathcal{S}}P(s,a,s')v^{\pi_t\top}f_{s'}].$$

We now show that $h^1(\theta_t) \stackrel{\triangle}{=} -\sum_{s\in\mathcal{S}}d^{\pi_t}(s)\sum_{a\in\mathcal{A}}\nabla\pi_t(s,a)[R(s,a)-J(\pi_t)+\sum_{s'\in\mathcal{S}}P(s,a,s')v^{\pi_t\top}f_{s'}]$ is Lipschitz continuous. Here v^{π_t} corresponds to the weight vector to which the critic update converges along the faster timescale when the corresponding policy is π_t (see Lemma 5). A simple calculation shows that for $s\in\mathcal{S}$, $a\in\mathcal{A}$,

$$\nabla^2\pi_t(s,a) = \pi_t(s,a)[\psi_{sa}{}^\top\psi_{sa} - \sum_{a'\in\mathcal{A}}\pi_t(s,a')\psi_{sa'}{}^\top\phi_{sa'}].$$

Thus $\nabla^2\pi_t(s,a)$ exists and is bounded. Further, from (24), it can be seen that $d^{\pi_t}(s)$, $s \in \mathcal{S}$ are continuously differentiable in θ and have bounded derivatives. Also, $J(\pi_t)$ is continuously differentiable as well and has bounded derivative as can also be seen from (43). Further, v^{π_t} can be seen to be continuously differentiable with bounded derivatives. Thus $h^1(\theta)$ is a Lipschitz continuous function. The requirements in (B1)-(B3) of Section 3 can also be seen to hold. As described before, $J(\pi)$, v^π are asymptotically stable attractors of (43), (44), respectively, and are Lipschitz continuous functions of θ. The rest of the proof can be shown as explained in Section 3 (see [21]) using the arguments on pp. 191-196 of [39] leading to convergence in the case of projected algorithms (see [19]). In particular, we note that the hypotheses under which Theorem 5.3.1, pp. 191-196 of [39] is shown can easily be verified in our setting. This completes the proof. □

Remark 1 The set $\mathcal{Z}$ corresponds to the set of local maxima $\theta^\star$ of a performance function whose gradient is $\mathbf{E}[\delta_t^\pi \psi_{sa} \mid \theta]$ (cf. Lemma 4). The latter equals zero when $\theta = \theta^\star$ and for which $\pi = \pi^\star$. The algorithm thus converges to one of the points in $\mathcal{Z}$.

We discuss now the difficulties involved in proving boundedness of iterates in (41). Suppose we rewrite $h^1(\theta)$ as

$$h^1(\theta) = -\sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi^\theta(s,a)\psi^\theta_{sa}[R(s,a) - J(\pi) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi\top} f_{s'}].$$

Note here that we write ψ^θ_{sa} in place of ψ_{sa} in order to show explicit dependence of ψ_{sa} on θ. Then defining $h^1_\infty(\theta)$ as $h^1_\infty(\theta) = \lim_{r\to\infty} \dfrac{h^1(r\theta)}{r}$, one obtains

$$h^1_\infty(\theta) = -\lim_{r\to\infty} \frac{1}{r} \sum_{s\in\mathcal{S}} d^\pi(s) \sum_{a\in\mathcal{A}} \pi^{r\theta}(s,a)\psi^{r\theta}_{sa} \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi^{r\theta}\top} f_{s'}.$$

It is not clear whether the limit above exists because of the complex dependence of d^π and v^π on θ. Note that v^π is obtained as a solution to a linear system of equations (see Lemma 5) with the matrix D therein also depending on θ. Assumption (A1′), pp. 454 in [22] considers the case where the above limits may not exist. However, it requires that for $r \geq R$ and $t \geq T$, for some $R, T > 0$, the trajectories $\hat{\phi}(t)$ of the ODE $\dot{\theta}_t = \dfrac{h^1(r\theta_t)}{r}$ should lie within a ball of radius $1/2$ around the origin. This can be shown provided the origin is a unique asymptotically stable attractor for the above ODEs for all $r \geq R$. It is not clear if this is the case here. Next, note that the methods described in [2] and [57] for stability of iterates are for different classes of algorithms, largely of the Q-learning type, and are not directly applicable in our setting.

Finally, we discuss the use of the stochastic Lyapunov function method [40] for stability of iterates in (41). The prime requirement here is that there exists a real-valued nonnegative function $W(\cdot)$ that satisfies

$$\mathbf{E}[W(\theta_{t+1}) \mid \theta_t = \theta] - W(\theta) \leq -K(\theta)$$

for all $\theta \in Q_\lambda \stackrel{\triangle}{=} \{\theta \mid W(\theta) \leq \lambda\}$, where $K(\theta) \geq 0$ is continuous on Q_λ. Then by Theorem 4.1, pp. 80-81 of [40], the stability and convergence of iterates would follow. Hence consider the recursion (41). By a Taylor's expansion for "small" β_t assuming a smooth $W(\cdot)$, one gets

$$\mathbf{E}[W(\theta_{t+1})|\theta_t] \approx W(\theta_t) - \beta_t \mathbf{E}[\delta_t^\pi \psi_{s_t a_t}|\theta_t]' \nabla W(\theta_t) + o(\beta_t). \tag{48}$$

A natural choice for the Lyapunov function $W(\cdot)$ in (48) would be one for which $\nabla W(\theta_t) = \mathbf{E}[\delta_t^\pi \psi_{s_t a_t} | \theta_t]$ itself. This choice would then give

$$W(\theta) = J(\theta) - \int \sum_{s \in \mathcal{S}} \frac{d d^\pi(s)}{d\theta} [\bar{V}^\pi(s) - v^{\pi\top} f_s] d\theta.$$

It is difficult to obtain the exact dependence of $d^\pi(s)$ on θ and to solve the above integral precisely. On the other hand, if we use the look up table representation (viz., $d_2 = n$ in Assumption (A3) or that δ_t is as in (19)), then from Lemma 4 above, as also Theorem 1 of [53], one would get $\mathbf{E}[\delta_t \psi_{s_t a_t} \mid \theta] = \nabla_\theta J(\theta)$. Then $W(\theta) = J(\theta)$ would serve as a Lyapunov function and the iterates (41) can be seen to be bounded and almost surely convergent, in lieu of Theorem 4.1 of [40]. It is only because of the use of function approximation in the iterates that a Lyapunov function is hard to obtain. Because of the above arguments and as can also be seen from our experiments where we do not use projection at all (and yet convergence is achieved), we conjecture that the iterates in (41) shall remain bounded.

Note also that if function approximation is not used, $J(\theta)$ also serves as a Lyapunov function for the ODE associated with (41). When function approximation is used (as with our case), the above problem of finding a suitable Lyapunov function (now) for the associated ODE also carries over and it is difficult to suitably characterize the set of stable attractors.

Remark 1 and many of the arguments in the analysis of Algorithm 1 are also valid for the analysis of the other algorithms. We skip the details in such cases to avoid repetition.

6.2 Convergence Analysis for Algorithm 2

The analysis of recursions for average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (31) proceeds in the same manner as for Algorithm 1. We thus concentrate on showing convergence of the recursion for the inverse of the Fisher information matrix (30) and the actor recursion (32). As discussed previously, G_t, G_t^{-1}, $t \geq 1$ are positive definite and symmetric matrices. We require the following assumption in addition to (A1)–(A3).

(A4) The iterates G_t satisfy $\sup_{t,\theta,s,a} \| G_t \|, \sup_{t,\theta,s,a} \| G_t^{-1} \| < \infty$.

Recall that we set $G_0^{-1} = aI$ with $a > 0$. A sufficient condition for both the requirements in (A4) is that (cf. pp. 35 of [16]) for some scalars $c_1, c_2 > 0$,

$$c_1 \| z \|^2 \leq z^\top \psi_{sa} \psi_{sa}^\top z \leq c_2 \| z \|^2,$$

for all $s \in \mathcal{S}$, $a \in \mathcal{A}$, $z \in \mathbb{R}^{d_2}$ and θ. It is then easy to see that

$$\bar{c}_1 \| z \|^2 \leq z^\top G_t z \leq \bar{c}_2 \| z \|^2,$$

for all $t \geq 0$, and the eigenvalues of G_t lie between $\bar{c}_1$ and $\bar{c}_2$. Here $\bar{c}_1 = \min(a, c_1)$ and $\bar{c}_2 = \max(a, c_2)$. Also, $\bar{c}_1$, $\bar{c}_2 > 0$. Hence, the procedure does not get stuck at a nonstationary point. Under the above sufficient condition, (A4) follows from Propositions A.9 and A.15 of [16]. We now have

Lemma 6 For any given parameter θ, G_t^{-1}, $t \geq 1$ in (30) satisfy $G_t^{-1} \to G(\theta)^{-1}$ as $t \to \infty$

with probability one.

Proof It is easy to see from recursion (29) that $G_t \rightarrow G(\theta)$ as $t \rightarrow \infty$ with probability one, for any given θ held fixed. Now for fixed θ, we have

$$\| G_t^{-1} - G(\theta)^{-1} \| = \| G(\theta)^{-1}(G(\theta)G_t^{-1} - I) \| = \| G(\theta)^{-1}(G(\theta) - G_t)G_t^{-1} \|$$
$$\leq \sup_{\theta} \| G(\theta)^{-1} \| \quad \sup_{t,s,a} \| G_t^{-1} \| \cdot \| G(\theta) - G_t \| \rightarrow 0 \qquad \text{as} \quad t \rightarrow \infty,$$

by (A4). In the above, I denotes the $d_2 \times d_2$–dimensional identity matrix. The inequality above follows from the property on induced matrix norms (see Proposition A.12 of [17]). The claim follows. □

As with Algorithm 1, we consider again the transformed problem with rewards replaced with costs (see above). This transformation, however, only affects the actor recursion (32). Further, as before, we use the projection $\Gamma(\cdot)$ to ensure boundedness of iterates. The transformed slower timescale recursion that we have is thus

$$\theta_{t+1} = \Gamma(\theta_t - \beta_t G_t^{-1} \delta_t \psi_{s_t a_t}). \tag{49}$$

We have

Theorem 3 For the parameter iterations given by Algorithm 2, we have $(G_t^{-1}, \hat{J}_t, v_t, \theta_t) \rightarrow \{(G(\theta^\star)^{-1},\ J(\theta^\star),\ v^{\pi^\star},\ \theta^\star) \mid \theta^\star \in \mathcal{Z}\}$ as $t \rightarrow \infty$ with probability one, where $\pi^\star$ is the policy corresponding to $\theta^\star$.

Proof As with the proof of Theorem 2, let $\mathcal{F}_3(t) = \sigma(\theta_r,\, r \leq t)$, $t \geq 0$. Note that

$$\theta_{t+1} = \Gamma(\theta_t - \beta_t \mathbf{E}[G(\theta_t)^{-1}\delta_t^{\pi_t}\psi_{s_t a_t} \mid \mathcal{F}_3(t)] - \beta_t(G(\theta_t)^{-1}\delta_t\psi_{s_t a_t} - \mathbf{E}[G(\theta_t)^{-1}\delta_t\psi_{s_t a_t} \mid \mathcal{F}_3(t)]) + \beta_t\xi_1(t)),$$

where in lieu of Lemmas 5 and 6, $\xi_1(t) = o(1)$. As before, the critic recursion (31) converges faster for given policy π_t corresponding to an actor update θ_t and converges to v^{π_t}. For $t \geq 1$, let

$$M^3(t) = \sum_{r=0}^{t-1} \beta_r (G(\theta_r)^{-1}\delta_r\psi_{s_r a_r} - \mathbf{E}[G(\theta_r)^{-1}\delta_r\psi_{s_r a_r} \mid \mathcal{F}_3(r)])$$
$$= \sum_{r=0}^{t-1} \beta_r G(\theta_r)^{-1}(\delta_r\psi_{s_r a_r} - \mathbf{E}[\delta_r\psi_{s_r a_r} \mid \mathcal{F}_3(r)]).$$

The quantities δ_t and $G(\theta_t)^{-1}$ are uniformly bounded from Lemmas 5 and 6, and (A4) respectively. Now using (10), it can be seen [19] that $\{M^3(t)\}$ is a convergent martingale sequence. Hence, $\beta_t(G(\theta_t)^{-1}\delta_t\psi_{s_t a_t} - \mathbf{E}[G(\theta_t)^{-1}\delta_t\psi_{s_t a_t} \mid \mathcal{F}_3(t)]) = o(1)$. As before, also note that

$$\mathbf{E}[G(\theta_t)^{-1}\delta_t^{\pi_t}\psi_{s_t a_t}|\theta_t] = G(\theta_t)^{-1}\left[\sum_{s\in\mathcal{S}} d^{\pi_t}(s) \sum_{a\in\mathcal{A}} \nabla\pi_t(s,a)[R(s,a) - J(\pi_t) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi_t\top} f_{s'}]\right]$$

Consider now the ODE

$$\dot{\theta} = \hat{\Gamma}(-G(\theta)^{-1} \sum_{s\in\mathcal{S}} d^{\pi}(s) \sum_{a\in\mathcal{A}} \nabla\pi^{\theta}(s,a)[R(s,a) - J(\pi) + \sum_{s'\in\mathcal{S}} P(s,a,s')v^{\pi\top} f_{s'}]), \tag{50}$$

associated with recursion (49). As with (47), $\mathcal{Z}$ can be seen to be the set of stable fixed points of (50) as well since $G(\theta)^{-1}$ is positive definite and symmetric for all θ. Recall that $J(\theta)$ and v^π are Lipschitz continuous functions. From (A4), $G(\theta)^{-1}$ can be seen to be Lipschitz continuous as well. The RHS of (50) is thus Lipschitz continuous. The rest follows in a similar manner as Theorem 2. □

6.3 Convergence Analysis for Algorithm 3

As stated previously, the main idea in this algorithm is to minimize the least squares error in estimating the advantage function via function approximation. The analysis of average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (34) recursions proceeds in the same manner as before. We now concentrate on recursion (35) and the actor recursion (36). Note that we only require Assumptions (A1)–(A3) here and not (A4). In the transformed problem (with costs in place of rewards), recursion (35) can be rewritten as

$$w_{t+1} = (I - \alpha_t \psi_{s_t a_t} \psi_{s_t a_t}^\top) w_t - \alpha_t \delta_t \psi_{s_t a_t}, \tag{51}$$

with the actor recursion (36) the same as before. Note that (51) moves on a faster timescale as compared to the actor recursion. Hence, on the timescale of the former recursion, one may consider the parameter θ_t to be fixed. We have the following result:

Lemma 7 Under a given parameter θ, w_t, $t \geq 1$ in (51) satisfy $w_t \to -G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}]$ as $t \to \infty$ with probability one, where π is the policy corresponding to θ.

Proof Consider the following ODE associated with (51) for given θ

$$\dot{w} = \mathbf{E}_{s_t \sim d^\pi, a_t \sim \pi}[-\psi_{s_t a_t} \psi_{s_t a_t}^T w - \delta_t^\pi \psi_{s_t a_t}]. \tag{52}$$

Let $g^2(w)$ correspond to the RHS of (52). Then $g^2(w)$ is Lipschitz continuous in w. For any $r \geq 1$, let $g_\infty^2(w) = \lim_{r \to \infty} \frac{g^2(rw)}{r}$. The function $g_\infty^2(w)$ exists and can be seen to satisfy $g_\infty^2(w) = -G(\theta)w$. For the ODE $\dot{w} = -G(\theta)w$, the origin is an asymptotically stable equilibrium with $V_4(w) = w'w/2$ as the associated Lyapunov function. Define now $\{M^4(t)\}$ as

$$M^4(t) = (-\psi_{s_t a_t} \psi_{s_t a_t}^T w_t - \delta_t \psi_{s_t a_t}) + \mathbf{E}[(\psi_{s_t a_t} \psi_{s_t a_t}^T w_t + \delta_t \psi_{s_t a_t}) \mid \mathcal{F}_4(t)],$$

where $\mathcal{F}_4(t) = \sigma(w_r, M^4(r), r \leq t)$. It is easy to see that there exists a constant $C_0 < \infty$ such that

$$\mathbf{E}[\| M^4(t+1) \|^2 | \mathcal{F}_4(t)] \leq C_0(1 + \| w_t \|^2),$$

for all $t \geq 0$. For the ODE (52), consider the function $V_5(w)$ defined by

$$V_5(w) = (w + G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}])'(w + G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}])/2.$$

Then

$$\begin{aligned} \frac{dV_5(w)}{dt} &= \nabla V_5(w)' \, \dot{w} = -(w + G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}])'(G(\theta)w + \mathbf{E}[\delta_t^\pi \psi_{s_t a_t}]) \\ &= -(w + G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}])' G(\theta)(w + G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}]) \end{aligned}$$

$$< 0 \text{ for all } w \neq -G(\theta)^{-1}\mathbf{E}[\delta_t^{\pi}\psi_{s_t a_t}],$$

since $G(\theta)^{-1}$ is a positive definite matrix. Thus (see [40]) $w^{\pi} = -G(\theta)^{-1}\mathbf{E}[\delta_t^{\pi}\psi_{s_t a_t}]$ is an asymptotically stable equilibrium for (52). Now from Theorem 2.2 of [22], recursion (51) converges with probability one to w^{π}. □

We now consider the actor recursion (36), which is the slower recursion. As before, we use the projection $\Gamma(\cdot)$ to ensure boundedness of the iterates. The projected recursion that we consider is thus

$$\theta_{t+1} = \Gamma(\theta_t + \beta_t w_t). \tag{53}$$

We have

Theorem 4 For the parameter iterations given by Algorithm 3, we have $(\hat{J}_t, v_t, w_t, \theta_t) \rightarrow \{(J(\theta^{\star}), v^{\pi^{\star}}, w^{\pi^{\star}}, \theta^{\star}) \mid \theta^{\star} \in \mathcal{Z}\}$ as $t \rightarrow \infty$ with probability one, where $\pi^{\star}$ is the policy corresponding to $\theta^{\star}$.

Proof Note that as a consequence of Lemma 7, recursion (53) can be replaced with

$$\theta_{t+1} = \Gamma(\theta_t - \beta_t G(\theta_t)^{-1}\mathbf{E}[\delta_t^{\pi_t}\psi_{s_t a_t} \mid \theta_t])$$

As with Theorem 3, w^{π}, the asymptotically stable equilibrium for (52) is also a Lipschitz continuous function of parameter θ, in addition to $J(\theta)$ and v^{π}. The rest can be shown in a similar manner as Theorems 2-3. □

6.4 Convergence Analysis for Algorithm 4

As with Algorithm 2, we require Assumption (A4) here as well in addition to (A1)–(A3). The result in Lemma 6 continues to hold here and we get for fixed θ, $G_t^{-1} \rightarrow G(\theta)^{-1}$ as $t \rightarrow \infty$ with probability one. Recursions for average reward (Line 5 in Table 1), TD-error (Line 6 in Table 1), and critic (38) are the same as before and have been analyzed earlier. We now concentrate on recursion (39) and the actor recursion (40). Under the transformed problem (with costs in place of rewards), recursion (39) can be rewritten as

$$w_{t+1} = (1 - \alpha_t)w_t - \alpha_t G_t^{-1}\delta_t\psi_{s_t a_t}, \tag{54}$$

with the actor recursion the same as before. An exactly similar result as Lemma 7 holds in this case as well (described as Lemma 8 below).

Lemma 8 Under a given parameter θ, w_t defined by (54) converge as $w_t \rightarrow -G(\theta)^{-1}\mathbf{E}[\delta_t^{\pi}\psi_{s_t a_t} \mid \theta]$ as $t \rightarrow \infty$ with probability one, with π being the policy corresponding to θ.

Proof Note that as a consequence of (A4) and Lemma 5, $\sup_{t,\theta,s_t,a_t} \| G_t^{-1}\delta_t\psi_{s_t a_t} \| < \infty$ with probability one. As a consequence of (10), there exists an integer $N_0 < \infty$, such that for all $t \geq N_0$, $\alpha_t \leq 1$. Hence for all $t \geq N_0$, w_{t+1} is a convex combination of w_t and a uniformly bounded quantity. Thus, starting from any initial value $w_0 \in \mathbb{R}^{d_2}$, the overall sequence w_t of iterates remains bounded with probability one. Now note that one can rewrite (54) as

$$w_{t+1} = (1 - \alpha_t)w_t - \alpha_t G(\theta)^{-1}\mathbf{E}[\delta_t^{\pi}\psi_{s_t a_t} \mid \theta] + M^5(t) + \xi_2(t) + \xi_3(t),$$

where $M^5(t) = \alpha_t G(\theta)^{-1}(\mathbf{E}[\delta_t \psi_{s_t a_t} \mid \theta] - \delta_t \psi_{s_t a_t})$, $\xi_2(t) = \alpha_t (G(\theta)^{-1} - G_t^{-1})\delta_t \psi_{s_t a_t}$, and $\xi_3(t) = \alpha_t$ $G(\theta)^{-1}\mathbf{E}[(\delta_t^\pi - \delta_t)\psi_{s_t a_t} \mid \theta]$, respectively. As before, one can see that $M^5(t)$, $\xi_2(t)$ and $\xi_3(t)$ are all $o(1)$. Consider the following ODE associated with (54).

$$\dot{w} = -w - G(\theta)^{-1}\mathbf{E}_{s_t \sim d^\pi, a_t \sim \pi}[\delta_t^\pi \psi_{s_t a_t}]. \tag{55}$$

It is easy to see that $h(w)$ defined as the RHS of (55) is Lipschitz continuous in w. One can show as in Lemma 7 that $w^\pi = -G(\theta)^{-1}\mathbf{E}[\delta_t^\pi \psi_{s_t a_t}]$ is an asymptotically stable attractor for the ODE (55). The rest follows as a consequence of Theorem 1, pp. 339 of [34], see for instance [19]. □

We now consider the actor recursion (40), which is the slower recursion. As before, we use the projection $\Gamma(\cdot)$ to ensure boundedness of the iterates. The projected recursion is the same as (53) in this case. We thus have the following result whose proof follows as in Theorems 3-4.

Theorem 5 For the parameter iterations given by Algorithm 4, we have $(G_t^{-1}, \hat{J}_t, v_t, w_t, \theta_t) \rightarrow \{(G(\theta^\star)^{-1},\ J(\theta^\star),\ v^{\pi^\star},\ w^{\pi^\star},\ \theta^\star) \mid \theta^\star \in \mathcal{Z}\}$ as $t \rightarrow \infty$ with probability one, where $\pi^\star$ is the policy corresponding to $\theta^\star$. □

7 Relation to Previous Algorithms

As we mentioned in Section 1, the actor-critic algorithms presented in this paper extend prior actor-critic methods, especially those of Konda and Tsitsiklis [38] and of Peters, Vijayakumar and Schaal [46]. In this section, we discuss these relationships further.

Actor-Critic Algorithm of Konda and Tsitsiklis [38]: Contrary to Algorithms 2-4, this algorithm does not use estimates of natural gradient in its actor's update. It is somewhat similar to our Algorithm 1, but with some key differences. **1)** Konda's algorithm uses the Markov process of state-action pairs and thus its critic update is based on an action-value function. Algorithm 1 uses the state process and therefore its critic update is based on a value function. **2)** While Algorithm 1 uses TD error in both critic and actor recursions, Konda's algorithm uses TD error only in its critic update. The actor recursion in Konda's algorithm uses a Q-value estimate instead. Because the TD error is a consistent estimate of the advantage function (Lemma 3), the actor recursion in Algorithm 1 uses estimates of advantages instead of Q-values, which may result in lower variances. **3)** The convergence analysis of Konda's algorithm is based on the martingale approach and aims at bounding error terms and directly showing convergence. Convergence to a local optimum is shown when TD(1) critic is used. For the case when $\lambda < 1$, they show that given $\epsilon > 0$, there exists λ close enough to one such that when a TD(λ) critic is used, one gets $\liminf_t |\nabla J(\boldsymbol{\theta}_t)| < \epsilon$ with probability 1. Unlike Konda and Tsitsiklis, we primarily use the ordinary differential equation (ODE) based approach for our convergence analysis. Even though we also use martingale arguments in our analysis, these are restricted to showing that the noise terms asymptotically diminish and the resulting scheme can be viewed as a Euler-discretization of the associated ODE.

Natural Actor-Critic Algorithm of Peters et al. [46]: Algorithms 2-4 extend this algorithm, by being fully incremental and providing convergence proofs. Peters's algorithm uses a least-squares TD method in its critic's update, while our algorithms are all fully incremental. It is not entirely

clear how to satisfactorily incorporate least-squares TD methods in a context in which the policy is changing. Our proof techniques do not immediately extend to this case. However, we use estimates of advantage function in Algorithms 3 and 4 as in Peters's algorithm.

8 Empirical Results

In this section we report empirical results applying the algorithms presented in the paper to a set of abstract randomly constructed MDPs which we call Garnet problems. We present results with our algorithms exactly as described in Section 5, illustrating the convergence proved in Section 6. We also report results for the most closely related algorithm in the prior literature, that by Konda and Tsitsiklis [38].[3] In all our experiments, we observed that the average rewards obtained by Konda's algorithm were much smaller than those obtained by our algorithms. Thus, we do not plot them in Figure 1 and only report their means and standard errors (STSs) in Table 2. The C++ code for all the experiments conducted in this section is available at [43].

Garnet problems are a class of randomly constructed finite MDPs serving as environments for reinforcement learning algorithms optimizing average reward. Garnet problems do not correspond to any particular application, but are meant to be totally abstract or generic while remaining representative of the kind of MDPs that might be encountered in practice (cf. [5]). The name "Garnet" is an acronym for Generic Average Reward Non-stationary Environment Testbed. The process for generating an instance of a Garnet problem is characterized by 5 parameters and written as Garnet(n, m, b, σ, τ). The parameters n and m are the number of states and actions respectively, and b is a branching factor specifying the number of possible next states for each state-action pair. The possible next states are chosen at random from the state set without replacement. The probability of going to each next state is generated by partitioning the unit interval at $b-1$ cut points selected randomly between 0 and 1. The expected reward for each such transition is a normally distributed random variable with mean 0 and unit variance. The actual reward is selected randomly according to a normal distribution with mean equal to the expected reward and standard deviation σ. Finally, the parameter $\tau,\ 0 \leq \tau \leq 1/n$ determines the degree of non-stationarity in the problem. If $\tau = 0$, the Garnet problem is stationary. If $\tau > 0$, states of the MDP are occasionally selected randomly for deletion and replacement with newly constructed expected rewards and transition probabilities. At each time step, with probability $n * \tau$, one of the states is selected at random and reconstructed as described above. We use stationary Garnet problems ($\tau = 0$) in the experiments of this paper. From the above definition, it is clear that Garnet(n, m, b, σ, τ) represents a family of Garnet problems with the same structure.

In our experiments, we used linear function approximation for state value functions $V(s, v) = v^\top f_s$, and parameterized Gibbs distribution for policies (7). State feature vectors f_s and state-action feature vectors ϕ_{sa} were binary and were randomly generated using two parameters d and l. The parameter d is the dimensionality of the state feature vectors $f_s \in \{0,1\}^d$ (i.e., $d_2 = d$). The parameter l is the number of components of the state feature vectors that were 1 (the others were 0). The locations of the 1's were chosen randomly with equal probability such that no two states had the same feature vector. The state-action feature vectors had dimension $d \times m,\ \phi_{sa} \in \{0,1\}^{d \times m}$

[3] From now on in the paper we call this algorithm Konda's algorithm.

(i.e., $d_1 = d \times m$) and were constructed using state feature vectors as follows:

$$\phi_{sa_i} = (\underbrace{0, \ldots, 0}_{d\times(i-1)}, f_s, \underbrace{0, \ldots, 0}_{d\times(m-i)})^\top \tag{56}$$

We chose d such that $d \times m$ are $\ll n$. Therefore, an exact solution is usually not possible and approximate value functions are required. We also chose l substantially less than d as this case has proven powerful in many applications of reinforcement learning and is computationally efficient.

We set the initial values for policy parameters θ_0, state value function weights v_0, and weights w_0 to 0.0. We used the following step-size schedules for the critic $\{\alpha_t\}$ and the actor $\{\beta_t\}$:

$$\alpha_t = \frac{\alpha_0 \cdot \alpha_c}{\alpha_c + t^{2/3}} \qquad , \qquad \beta_t = \frac{\beta_0 \cdot \beta_c}{\beta_c + t} \ .$$

Note that these step-size schedules satisfy (10) and (11). We set the constant c used for the average reward step-size in our algorithms to 0.95. In Algorithms 2 and 4, we initialized the inverse of the Fisher information matrix to $G_0^{-1} = 1.5I$ and $G_0^{-1} = 2.5I$ respectively. We also used step-size $0.001\alpha_t$ in place of α_t to update G_t^{-1} for numerical stability in these algorithms.

Figure 1 shows the average rewards obtained by the four actor-critic algorithms presented in the paper in two families of stationary Garnet problems, Garnet(30,4,2,0.1,0) (top row) and Garnet(100,10,3,0.1,0) (bottom row). The function approximation parameters were set to $d = 8$ and $l = 3$ in Garnet(30,4,2,0.1,0), and to $d = 20$ and $l = 5$ in Garnet(100,10,3,0.1,0). All the graphs in the top row are averaged over 100 independent runs of a fixed Garnet(30,4,2,0.1,0) problem (top left) and 100 different randomly and independently generated Garnet(30,4,2,0.1,0) problems (top right). All the graphs in the bottom row are averaged over 20 independent runs of a fixed Garnet(100,10,3,0.1,0) problem (bottom left) and 20 different randomly and independently generated Garnet(100,10,3,0.1,0) problems (bottom right). Table 2 contains the means and the standard errors (STEs) of the average rewards obtained by the four actor-critic algorithms presented in the paper, plus the Konda's algorithm, for 100 runs of a fixed Garnet(30,4,2,0.1,0) problem (2nd column), 100 different Garnet(30,4,2,0.1,0) problems (3rd column), 20 runs of a fixed Garnet(100,10,3,0.1,0) problem (4th column), and 20 different Garnet(100,10,3,0.1,0) problems (5th column).

Algorithm	**Mean ± STE**	**Mean ± STE**	**Mean ± STE**	**Mean ± STE**
Algorithm 1	1.592 ± 0.004	0.780 ± 0.025	0.764 ± 0.003	0.816 ± 0.018
Algorithm 2	1.582 ± 0.002	0.787 ± 0.024	0.872 ± 0.002	0.948 ± 0.022
Algorithm 3	1.597 ± 0.001	0.835 ± 0.025	0.918 ± 0.001	0.992 ± 0.014
Algorithm 4	1.570 ± 0.002	0.786 ± 0.024	0.871 ± 0.002	0.933 ± 0.021
Konda's Algorithm	0.607 ± 0.005	0.444 ± 0.017	0.144 ± 0.001	0.230 ± 0.012

Table 2: Means and standard errors (STEs) of the average rewards obtained by the algorithms on 100 runs of a fixed Garnet(30,4,2,0.1,0) problem (2nd column), 100 different Garnet(30,4,2,0.1,0) problems (3rd column), 20 runs of a fixed Garnet(100,10,3,0.1,0) problem (4th column), and 20 different Garnet(100,10,3,0.1,0) problems (5th column).

Table 3 contains the values of the step-size schedule parameters used by the algorithms in the experiments with Garnet(30,4,2,0.1,0) (2nd column) and Garnet(100,10,3,0.1,0) (3rd column)

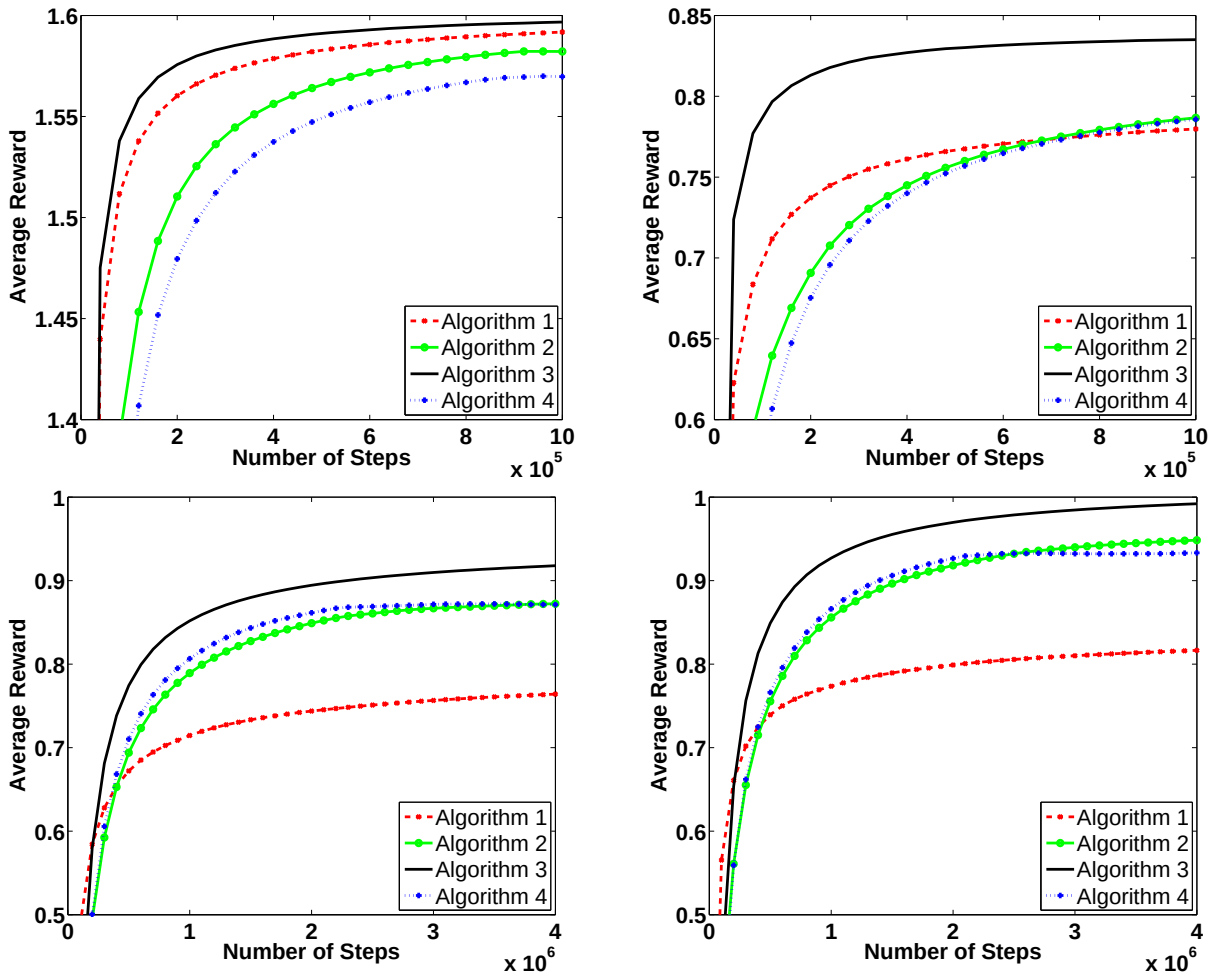

Figure 1: This figure shows the average rewards obtained by the four actor-critic algorithms presented in the paper in two families of stationary Garnet problems, Garnet(30,4,2,0.1,0) (top row) and Garnet(100,10,3,0.1,0) (bottom row). All the graphs in the top/bottom left figure are averaged over 100/20 independent runs of a fixed Garnet(30,4,2,0.1,0)/Garnet(100,10,3,0.1,0) problem, while the graphs in the top/bottom right figure are averaged over 100/20 different randomly and independently generated Garnet(30,4,2,0.1,0)/Garnet(100,10,3,0.1,0) problems.

problems. We tried many values for these parameters in the experiments with the fixed Garnet problems (left column in Figure 1) and those in the table yielded the best performance. We then used the same parameters in the experiments with different Garnet problems (right column in Figure 1).

Algorithm 3 showed reliably good performance in both small and large size problems. We found it easier to find good parameter settings for Algorithm 3 than for the other natural gradient algorithms and, perhaps because of this, it converged more rapidly than them and than Konda's algorithm. However, these empirical observations should be taken only as suggestive; our experiments were not extensive enough to be taken as showing anything comparative about the relative

Algorithm	α_0	α_c	β_0	β_c	α_0	α_c	β_0	β_c
Algorithm 1	0.1	1000	0.01	100000	0.1	1000000	0.01	100000000
Algorithm 2	0.1	1000	0.01	1000	0.1	1000	0.01	1000
Algorithm 3	0.1	10000	0.001	10000	0.1	10000	0.001	100000
Algorithm 4	0.1	1000	0.001	10000	0.1	1000	0.001	10000
Konda's Algorithm	0.1	10000	0.01	10000	0.1	10000	0.01	10000

Table 3: Values of the step-size schedule parameters in the Garnet(30,4,2,0.1,0) (second column) and Garnet(100,10,3,0.1,0) (third column) experiments.

rate of convergence of any of the algorithms.

We used relative value iteration algorithm [18] and separately computed the best average rewards if there were no constraints due to the function approximator, for the fixed Garnet problems. The unconstrained optimal rewards are 1.618 and 1.170 for the fixed Garnet(30,4,2,0.1,0) and Garnet(100,10,3,0.1,0) problems respectively. On the smaller Garnet problem, our four actor-critic algorithms converged to the unconstrained optimal average reward 1.618 (see Figure 1 top-left and the second column of Table 2). On the larger problem function approximation plays a larger role and the unconstrained optimum is not reached and presumably cannot be reached.

9 Conclusions and Future Work

We have introduced and analyzed four actor-critic reinforcement learning algorithms utilizing linear function approximation. All the algorithms are based on existing ideas such as temporal difference learning, natural policy gradients, and two-timescale convergence analysis, but we combine them in new ways. The main contribution of this paper is the proof of convergence of the four algorithms to a local maximum in the space of policy and value function parameters. Our work extends that by Konda and Tsitsiklis [38] and others [1, 19, 37] by incorporating a bootstrapping ($\lambda < 1$) form of temporal difference learning. Our four algorithms are the first actor-critic algorithms to be shown convergent that utilize both function approximation and bootstrapping, a combination which seems essential to large-scale applications of reinforcement learning.

Our Algorithms 2-4 are explorations of the use of natural gradients within an actor-critic policy gradient architecture. The way we use natural gradients is distinctive in that it is totally incremental: the policy is changed on every time step yet we never reset the gradient computation as is done in the algorithm of Peters et al. [46]. Algorithm 3 is perhaps the most interesting of the three natural gradient algorithms. It never explicitly stores an estimate of the inverse of the Fisher information matrix and, as a result, it requires less computation. In our empirical experiments we found it easier to find good parameter settings for Algorithm 3 than for the other natural gradient algorithms and, perhaps because of this, it converged more rapidly than them and than Konda and Tsitsiklis's algorithm. These empirical observations should be taken only as suggestive; more experiments to properly assess the relative performance of these algorithms must be carried out.

The most important potential extension of our results would be to characterize the quality of the converged solution. It may be possible to bound the performance loss due to bootstrapping and approximation error in a way similar to how it was bounded by Tsitsiklis and Van Roy [58]. There are a number of other ways in which our results are limited and suggest for future work.

First, there is the issue of rate of convergence. Ideally one would like analytic results but, short of that, it would be useful to conduct a thorough empirical study, varying parameters and schedules in a more extensive and sophisticated way than we have done here. Second, the algorithms could be extended to incorporate eligibility traces and least-squares methods. As discussed earlier, the former seems straightforward whereas the latter seems to require more fundamental extensions. Finally, application of these ideas and algorithms to a real-world problem is needed to assess their ultimate utility.

References

[1] Abdulla, M.S. and Bhatnagar, S. (2007) "Reinforcement learning based algorithms for average cost Markov decision processes", *Discrete Event Dynamic Systems: Theory and Applications*, 17(1):23-52.

[2] Abounadi, J., Bertsekas, D. and Borkar, V.S. (2001) "Learning Algorithms for Markov Decision Processes", *SIAM Journal on Control and Optimization*, 40:681-698.

[3] Alrefaei, M.H. and Andradóttir, S. (1999) "A simulated annealing algorithm with constant temperature for discrete stochastic optimization", *Management Science*, 45(5):748-764.

[4] Amari, S. (1998) "Natural gradient works efficiently in learning", *Neural Computation*, 10(2):251-276.

[5] Archibald, T., McKinnon, K., and Thomas, L. (1995) "On the Generation of Markov Decision Processes", Journal of the Operational Research Society, 46:354-361.

[6] Baird, L.C. (1993) "Advantage Updating", Technical Report WL-TR-93-1146, Wright laboratory, Wright-Patterson Air Force Base, OH 45433-7301.

[7] Baird, L. C. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 30–37. Morgan Kaufmann, San Francisco.

[8] Bagnell, J. and Schneider, J. (2003) "Covariant policy search", *Proceedings of International Joint Conference on Artificial Intelligence*.

[9] Barto, A., Sutton, R. and Anderson, C. (1983) "Neuron-like elements that can solve difficult learning control problems", *IEEE Transactions on Systems, Man and Cybernetics*, 13:835-846.

[10] Baxter, J. and Bartlett, P.L. (2001) "Infinite-horizon policy-gradient estimation", *Journal of Artificial Intelligence Research*, 15:319-350.

[11] Baxter, J., Bartlett, P.L. and Weaver, L. (2001) "Experiments with infinite-horizon, policy-gradient estimation", *Journal of Artificial Intelligence Research*, 15:351-381.

[12] Baxter, J., Tridgell, A., and Weaver, L. (1998) "KnightCap: A Chess Program that Learns by Combining TD(λ) with Game-Tree Search", *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 28–36.

[13] Bellman, R. E., and Dreyfus, S. E. (1959). Functional approximations and dynamic programming. *Mathematical Tables and Other Aids to Computation*, 13:247–251.

[14] Benveniste, A., Metivier, M. and Priouret, P. (1990) *Adaptive Algorithms and Stochastic Approximations*, Springer, Berlin.

[15] Bertsekas, D.P. (1995) *Dynamic Programming and Optimal Control*, Athena Scientific, Belmont, MA.

[16] Bertsekas, D.P. (1999) *Nonlinear Programming*, Athena Scientific, Belmont, MA.

[17] Bertsekas, D.P. and Tsitsiklis J.N. (1989) *Parallel and Distributed Computation*, Prentice Hall, New Jersey.

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

[19] Bhatnagar, S. and Kumar, S. (2004) "A simultaneous perturbation stochastic approximation based actor-critic algorithm for Markov decision processes", *IEEE Transactions on Automatic Control*, 49(4):592-598.

[20] Borkar, V.S. (1996) "Recursive self-tuning control of finite Markov chains", *Appl. Math.*, 24:169-188.

[21] Borkar, V.S. (1997) "Stochastic approximation with two timescales", *Systems and Control Letters*, 29:291-294.

[22] Borkar, V.S. and Meyn, S.P. (2000) "The O.D.E. method for convergence of stochastic approximation and reinforcement learning", *SIAM Journal of Control and Optimization*, 38(2):447-469.

[23] Boyan, J. A. (1999). Least-squares temporal difference learning. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 49–56. Morgan Kaufmann, San Francisco, CA.

[24] Boyan, J. A., and Moore, A. W. (1995). Generalization in reinforcement learning: Safely approximating the value function. In G. Tesauro, D. S. Touretzky, and T. Leen (eds.), *Advances in Neural Information Processing Systems: Proceedings of the 1994 Conference*, pp. 369–376. MIT Press, Cambridge, MA.

[25] Bradtke, S.J. and Barto, A.G. (1996) "Linear least-squares algorithms for temporal difference learning", *Machine Learning*, 22:33-57.

[26] Cao, X.-R. and Chen, H.F. (1997) "Perturbation realization, potentials and sensitivity analysis of Markov processes", *IEEE Transactions on Automatic Control*, 42:1382-1393.

[27] Chow, C.-S., and Tsitsiklis, J. N. (1991). An optimal one-way multigrid algorithm for discrete-time stochastic control. *IEEE Transactions on Automatic Control*, 36:898–914.

[28] Crites, R. H., and Barto, A. G. (1998). Elevator Group Control using Multiple Reinforcement Learning Agents. *Machine Learning*, 33:235–262.

[29] Daniel, J. W. (1976). Splines and efficiency in dynamic programming. *Journal of Mathematical Analysis and Applications*, 54:402–407.

[30] Dukkipati, A., Murty, M.N., and Bhatnagar, S. (2005) "Information theoretic justification of Boltzmann selection and its generalization to Tsallis case", *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 1667-1674, Vol.2, Edinburgh, U.K.

[31] Geramifard, A., Bowling, M., and Sutton, R. S. (2006). Incremental Least-Squares Temporal Difference Learning. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pp. 356-361.

[32] Gordon, G. J. (1995). Stable function approximation in dynamic programming. In A. Prieditis and S. Russell (eds.), *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 261–268. Morgan Kaufmann, San Francisco. An expanded version was published as Technical Report CMU-CS-95-103. Carnegie Mellon University, Pittsburgh, PA, 1995.

[33] Greensmith, E., Bartlett, P.L. and Baxter, J. (2004) "Variance reduction techniques for gradient estimates in reinforcement learning", *Journal of Machine Learning Research*, 5:1471-1530.

[34] Hirsch, M.W. (1989) "Convergent activation dynamics in continuous time networks", *Neural Networks*, 2:331-349.

[35] Kakade, S. (2002) "A Natural Policy Gradient", *Advances in Neural Information Processing Systems*, 14.

[36] Kohl, N., Stone, P. (2004). Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2004)* (pp. 2619–2624).

[37] Konda, V.R. and Borkar, V.S. (1999) "Actor-critic like learning algorithms for Markov decision processes", *SIAM Journal on Control and Optimization*, 38(1):94-123.

[38] Konda, V.R. and Tsitsiklis, J.N. (2003) "On actor-critic algorithms", *SIAM Journal on Control and Optimization*, 42(4):1143-1166.

[39] Kushner, H.J. and Clark, D.S. (1978) *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer Verlag, New York.

[40] Kushner, H.J. and Yin, G.G. (1997) *Stochastic Approximation Algorithms and Applications*, Springer Verlag, New York.

[41] Lagoudakis, M.G. and Parr, R. (2001) "Model-free least squares policy iteration", Technical Report CS-2001-05, Department of Computer Science, Duke University, Durham, North Carolina (*shorter version of paper available in Proceedings of NIPS, 2001*).

[42] Lasalle, J.P. and Lefschetz, S. (1961) *Stability by Lyapunov's Direct Method with Applications*, Academic Press, New York.

[43] Lee, M., Sutton, R. and Ghavamzadeh, M. (2006) "Garnet Natural Actor-Critic Project", *University of Alberta Reinforcement Learning Library.*

[44] Marbach, P. and Tsitsiklis, J.N. (2001) "Simulation-based optimization of Markov reward processes" *IEEE Transactions on Automatic Control*, 46:191-209.

[45] Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., Liang, E. (2004). Inverted autonomous helicopter flight via reinforcement learning. *International Symposium on Experimental Robotics*.

[46] Peters, J., Vijayakumar, S. and Schaal, S. (2005) "Natural Actor-Critic", *Proceedings of the 16th European Conference on Machine Learning*, pp. 280-291.

[47] Rummery, G. and Niranjan, M. (1994) "On-line Q-learning using Connectionist Systems", *Technical Report CUED/F-INFENG/TR 166, Engineering Department, Cambridge University*.

[48] Rust, J. (1996). Numerical dynamic programming in economics. In H. Amman, D. Kendrick, and J. Rust (eds.), *Handbook of Computational Economics*, pp. 614–722. Elsevier, Amsterdam.

[49] Singh, S., and Dayan, P. (1998) Analytical Mean Squared Error Curves for Temporal Difference Learning. *Machine Learning*, 32:5–40.

[50] Sutton, R. (1984). *Temporal credit assignment in reinforcement learning*. Doctoral dissertation, University of Massachusetts Amherst.

[51] Sutton, R. S. (1988). Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44.

[52] Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. In D. S. Touretzky, M. C. Mozer and M. E. Hasselmo (eds.), *Advances in Neural Information Processing Systems: Proceedings of the 1995 Conference*, pp. 1038–1044. MIT Press, Cambridge, MA.

[53] Sutton, R., McAllester, D., Singh, S. and Mansour, Y. (2000) "Policy gradient methods for reinforcement learning with function approximation", *Advances in Neural Information Processing Systems*, 12:1057-1063.

[54] Sutton, R. and Barto, A. (1998) *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA.

[55] Tadic, V. (2001). On the Convergence of Temporal-Difference Learning with Linear Function Approximation. *Machine Learning 42(3)*:241–267.

[56] Tesauro, G. J. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38:58–68.

[57] Tsitsikis, J. (1994) "Asynchronous Stochastic Approximation and Q-learning", *Machine Learning*, 16:185-202.

[58] Tsitsiklis, J. and Van Roy, B. (1997) "An analysis of temporal-difference learning with function approximation", *IEEE Transactions on Automatic Control*, 42(5):674-690.

[59] Tsitsikis, J. and Van Roy, B. (1999) "Average cost temporal-difference learning", *Automatica*, 35:1799-1808.

[60] White, D. J. (1993). A survey of applications of Markov decision processes. *Journal of the Operational Research Society*, 44:1073–1096.

[61] Widrow, B. and Stearns, S.D. (1985) *Adaptive Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ.

[62] Williams, R.J. (1992) "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Learning*, 8:229-256.