J. Appl. Prob. **49,** 795–805 (2012)
Printed in England
© *Applied Probability Trust* 2012

ON ε-OPTIMALITY OF THE PURSUIT LEARNING ALGORITHM

RYAN MARTIN,[*] *University of Illinois at Chicago*
OMKAR TILAK,[**] *Indiana University–Purdue University Indianapolis*

Abstract

Estimator algorithms in learning automata are useful tools for adaptive, real-time optimization in computer science and engineering applications. In this paper we investigate theoretical convergence properties for a special case of estimator algorithms—the pursuit learning algorithm. We identify and fill a gap in existing proofs of probabilistic convergence for pursuit learning. It is tradition to take the pursuit learning tuning parameter to be fixed in practical applications, but our proof sheds light on the importance of a vanishing sequence of tuning parameters in a theoretical convergence analysis.

Keywords: Convergence; indirect estimator algorithm; learning automaton

2010 Mathematics Subject Classification: Primary 68Q87
Secondary 68W27; 68W40

1. Introduction

A learning automaton consists of an adaptive learning agent operating in an unknown random environment [10]. In a nutshell, a learning automaton has a choice among a finite set of actions to take, with one such action being optimal in the sense that it has the highest probability of producing a reward from the environment. This optimal action is unknown and the automaton uses feedback from the environment to try to identify the optimal action. Applications of learning automata include game theory, pattern recognition, computer vision, and routeing in communications networks. Recently, learning automata have been used for call routeing in ATM networks [2], multiple access channel selection [24], congestion avoidance in wireless networks [9], channel selection in radio networks [23], modeling of students' behavior [11], clustering and backbone formation in ad-hoc wireless networks [21], [22], power system stabilizers [4], and spectrum allocation in cognitive networks [8]. The simplest type of learning automata applies a direct algorithm, such as the linear reward-inaction algorithm [10], which uses only the environmental feedback at iteration t to update the preference ordering of the actions. A drawback to using direct algorithms is their slow rate of convergence. Attention has recently focused on the faster indirect estimator algorithms. What sets indirect algorithms apart from their direct counterparts is that they use the entire history of environmental feedback, i.e. from iteration 1 to t, to update the action preference ordering at iteration t. It is this more efficient use of the environmental feedback which leads to faster convergence.

Received 6 May 2011; revision received 5 March 2012.
[*] Postal address: Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, 851 S. Morgan St., 322 Science and Engineering Offices, Chicago, Illinois 60607, USA.
Email address: rgmartin@math.uic.edu
[**] Postal address: Department of Computer and Information Sciences, Indiana University–Purdue University Indianapolis, 723 W. Michigan St., SL 280, Indianapolis, Indiana 46202, USA. Email address: otilak@cs.iupui.edu

Here we consider a special case of indirect estimator algorithms—the *pursuit learning algorithm*—and, in particular, the version in [15]. Starting with vacuous information about the unknown reward probabilities, pursuit learning adaptively samples actions and tracks the empirical reward probabilities for each action. As the algorithm progresses, the sampling probabilities for the set of actions are updated in a way consistent with the relative magnitudes of the empirical reward probabilities; see Section 2.1. Simulations demonstrate that the algorithm is fast to converge in a number of different estimation scenarios [7], [12], [17], [18]. Theoretically, the algorithm is said to converge if the sampling probability for the action with the highest reward probability becomes close to 1 as the number of iterations increases.

In the learning automata literature, ε-optimality is the gold standard for theoretical convergence. But, there seems to be two different notions of ε-optimality that appears in the estimator algorithm literature. The version that appears in the direct estimator context (e.g. the linear reward-inaction algorithms) is in some sense weaker than that which appears in the indirect algorithm context. The latter is essentially convergence in probability of the dominant action sampling probability to 1 as the number of iterations approaches ∞. In Section 2.2 we describe these two modes of stochastic convergence in more detail, but our focus is on the latter convergence-in-probability version.

The main goal of this paper is to identify and fill a gap in existing proofs of ε-optimality for pursuit learning. We believe that it is important to throw light on this gap because there are relatively recent papers proposing new algorithms that simply copy verbatim these incomplete arguments. Specifically, in many proofs, the weak law of large numbers is incorrectly interpreted as giving a bound on the probability that the sample path stays inside a fixed neighborhood of its target *forever* after some fixed iteration. It is true that any finite-dimensional properties of the sample path can be handled via the weak law of large numbers, but the word 'forever' implies that countably many time instances must be dealt with and, hence, more care must be taken. A detailed explanation of the gap in existing proofs is presented in Section 2.3. In Section 3 we give a new proof of convergence in probability for pursuit learning with some apparently new arguments. A further consequence of our analysis relates to the algorithm's tuning parameter. Indeed, it is standard to assume, in both theory and practice, that the algorithm's tuning parameter is a small but fixed quantity. However, our analysis suggests that it is necessary to consider a sequence of tuning parameters that vanish at a certain rate.

2. Pursuit learning algorithm

2.1. Notation and statement of the algorithm

Suppose that a learning automaton has a finite set of actions $A = \{a_1, \ldots, a_r\}$. If the automaton plays action a_i then it earns a reward with probability d_i; otherwise, it gets a penalty. An estimator algorithm tracks this reward/penalty information with the goal of identifying the optimal action—the one having the largest reward probability d. Pursuit learning, described below, is one such algorithm.

At iteration t, the automaton selects an action $\alpha(t) \in A$ with respective probabilities $\pi(t) = \{\pi_1(t), \ldots, \pi_r(t)\}$. When this action is played, the environment produces an outcome $X(t) \in \{0, 1\}$ that satisfies

$$d_i = \mathrm{E}\{X(t) \mid \alpha(t) = a_i\}, \qquad i = 1, \ldots, r.$$

As the algorithm proceeds and the various actions are tried, the automaton acquires more and more information about the ds indirectly through the Xs. In other words, estimates $\hat{d}(t)$ of d at time t can be used to update the sampling probabilities $\pi(t)$ in such a way that those actions with large $\hat{d}(t)$ are more likely to be chosen again in the next iteration. Algorithm 1 below gives the details.

Algorithm 1. (*Pursuit learning.*)

1. For $i = 1, \dots, r$, set
$$\pi_i(0) = \frac{1}{r} \quad \text{and} \quad N_i(0) = 0,$$
and initialize $\hat{d}_i(0)$ by playing action a_i a few times and recording the proportion of rewards. Set $t = 1$.

2. (a) Sample $\alpha(t)$ according to $\pi(t-1)$, and observe $X(t)$ drawn from its conditional distribution given $\alpha(t)$.

   (b) For $i = 1, \dots, r$, update
$$N_i(t) = \begin{cases} N_i(t-1) + 1 & \text{if } \alpha(t) = a_i, \\ N_i(t-1) & \text{if } \alpha(t) \neq a_i, \end{cases}$$
which denotes the number of times action a_i has been tried up to and including iteration t, and
$$\hat{d}_i(t) = \begin{cases} \hat{d}_i(t-1) + \dfrac{X(t) - \hat{d}_i(t-1)}{N_i(t)} & \text{if } \alpha(t) = a_i, \\ \hat{d}_i(t-1) & \text{if } \alpha(t) \neq a_i, \end{cases}$$
and then compute
$$m(t) = \arg\max\{\hat{d}_1(t), \dots, \hat{d}_r(t)\}.$$

   (c) Update
$$\pi(t) = (1-\lambda)\pi(t-1) + \lambda\delta_{m(t)},$$
where δ_j is an r-vector whose jth entry is 1 and whose other entries are 0.

3. Set $t \leftarrow t + 1$ and return to step 2.

For comparison, the direct linear reward-inaction algorithm updates $\pi(t)$ according to the following rule. If $\alpha(t) = a_i$ then
$$\pi_j(t) = \begin{cases} \pi_j(t-1) + \lambda X(t)[1 - \pi_j(t-1)] & \text{if } j = i, \\ \pi_j(t-1) - \lambda X(t)\pi_j(t-1) & \text{if } j \neq i. \end{cases}$$

This direct linear reward-inaction algorithm does not make efficient use of the full environmental history $X(1), \dots, X(t)$ up to and including iteration t. For this reason, it suffers from slower convergence than that of the indirect pursuit learning algorithm. In fact, simulations in [18] demonstrated that an indirect algorithm requires roughly 87% fewer iterations than a direct algorithm to achieve the same level of precision.

One might also notice that the pursuit learning algorithm is not unlike the popular stochastic approximation methods introduced in [16] and discussed in detail in [6]. But, a convergence analysis of pursuit learning using the powerful ordinary differential equation techniques seems particularly challenging due to the discontinuity of the $\delta_{m(t)}$ component in the step 2(c) update.

The internal parameter λ controls the size of the steps that can be made in moving from $\pi(t-1)$ to $\pi(t)$. In general, small values of λ correspond to slower rates of convergence, and vice versa. In our asymptotic results, we follow Tilak *et al.* [20] and actually take $\lambda = \lambda_t$ to change with t. They argued that a changing λ is consistent with the usual notion of convergence (see also Section 2.2), and does not necessarily conflict with the practical choice of small fixed λ. In what follows, we will assume that

$$\lambda_t = 1 - \theta^{1/t} \quad \text{for some fixed } \theta \in (\mathrm{e}^{-1}, 1), \tag{1}$$

although all that is necessary is that $\lambda_t \asymp 1 - \theta^{1/t}$ as $t \to \infty$.

2.2. Convergence and ε-optimality

Convergence of an estimator algorithm like pursuit learning implies that, eventually, the automaton will always play the optimal action. Assume without loss of generality that d_1 is the largest among the ds. Then convergence means that $\pi_1(t)$ gets close to 1, in some sense, as $t \to \infty$. This property is typically called ε-optimality, although the details vary in the literature. In the context of indirect estimator algorithms, the following is perhaps the most common definition of ε-optimality.

Definition 1. The pursuit learning algorithm is ε-optimal if, for any $\varepsilon, \delta > 0$, there exist $T^\star = T^\star(\varepsilon, \delta)$ and $\lambda^\star = \lambda^\star(\varepsilon, \delta)$ such that

$$\mathrm{P}\{\pi_1(t) > 1 - \varepsilon\} > 1 - \delta \tag{2}$$

for all $t > T^\star$ and $\lambda < \lambda^\star$. Simply put, the algorithm has the ε-optimality property if $\pi_1(t) \to 1$ in probability as $(t, \lambda) \to (\infty, 0)$.

This is the definition of ε-optimality that appears in [1] and the references mentioned in Section 1; an arguably better adjective—*optimal in probability*—is used in [18]. However, a different notion of ε-optimality can be found in other contexts. This one says that the algorithm is ε-optimal if, for any $\varepsilon > 0$, there exists a fixed $\lambda > 0$ such that $\liminf_{t\to\infty} \pi_1(t) > 1 - \varepsilon$ with probability 1. Compared to Definition 1, this latter definition is, on the one hand, stronger because the condition is 'with probability 1', but, on the other hand, weaker because it does not even require $\pi_1(t)$ to converge. Since one will not, in general, imply the other, it is unclear which definition is to be preferred. Others have recognized the difference between the two [12], but apparently no explanation has been given for choosing one over the other.

Since both $T^\star$ and $\lambda^\star$ in Definition 1 are linked together through the choice of (ε, δ), it is intuitively clear that λ should decrease with t. In fact, allowing λ to change with t appears to be necessary in the proof presented in Section 3.2. So, throughout this paper, our notion of ε-optimality will be that (2) holds for all $t > T^\star$ with the particular (vanishing) sequence of tuning parameters $\{\lambda_t\}$ in (1).

2.3. Existing proofs of ε-optimality

Here we shall identify the gap in existing proofs of ε-optimality for pursuit learning. Focus will fall primarily on the proof in [15], but this is just for concreteness and not to single out these particular authors. In fact, essentially the same gap appears in [13]; there is a similar mis-step in other papers which we mention briefly below. The outline of these proofs goes roughly as follows.

Step 1. Show that $N_i(t) \to \infty$ in probability for each $i = 1, \dots, r$ as $t \to \infty$. That is, show that, for any large n and small δ, there exists $T^\star$ such that

$$\mathrm{P}\Big\{\min_{i=1,\dots,r} N_i(t) > n\Big\} > 1 - \delta \quad \text{for all } t > T^\star.$$

Step 2. Show that, for any small δ and ρ, there exists n such that

$$\mathrm{P}\Big\{\max_{i=1,\dots,r}|\hat{d}_i(t)-d_i|<\rho\;\Big|\;\min_{i=1,\dots,r}N_i(t)>n\Big\}>1-\delta. \tag{3}$$

Step 3. Reason from (3) that, for sufficiently small ρ and *all t larger than some $T^\star$*, $\hat{d}_1(t)$ will be the largest among the $\hat{d}_i(t)$s with probability at least $1-\delta$.

Step 4. Apply the monotonicity property [7] to show that $\pi_1(t)$ increases monotonically to 1 starting from some $t>T^\star$ and must, therefore, eventually cross the $1-\varepsilon$ threshold.

The trouble with this line of reasoning emerges in step 3, and is a consequence of an incorrect use of the law of large numbers. Roughly speaking, what is needed in step 3 is a control of the entire $\hat{d}(t)$ process over an infinite time horizon, $t>T^\star$, but the law of large numbers alone can provide control at only finitely many time instances. More precisely, from steps 1 and 2 and the law of large numbers, we can reason that

$$\mathrm{P}\{\hat{d}_1(t)\text{ is the largest of the }\hat{d}(t)\text{s}\}>1-\delta\quad\text{for all }t\geq T^\star. \tag{4}$$

Even though the left-hand side above is monotone increasing in t, we cannot conclude directly from this fact that

$$\mathrm{P}\{\hat{d}_1(t)\text{ is the largest of the }\hat{d}(t)\text{s}\textit{ for all }t\geq T^\star\}>1-\delta. \tag{5}$$

The proof in [15] implicitly assumes that (4) implies (5). A slightly different oversight is made in [7], [12], and [19]. They assumed that

$$\mathrm{P}\{\pi_1(t)>1-\varepsilon\mid\hat{d}_1(t)\text{ is the largest of the }\hat{d}(t)\text{s}\}$$

can be made arbitrarily close to 1 for large enough t. However, the knowledge that $\hat{d}_1(t)$ is the largest of the $\hat{d}(t)$s only at time t provides no control over how close $\pi_1(t)$ is to 1. The monotonicity property in step 4 requires that the $\hat{d}(t)$s be properly ordered *forever*, not just at a single point in time.

It will be insightful to see what the problem is mathematically. First, the left-hand side of (5) is, in general, much smaller than the left-hand side of (4), so the claim '(4) implies (5)' immediately seems questionable. In fact, if E_t is the event that $\hat{d}_1(t)$ is the largest of the $\hat{d}(t)$s at time t, then from (4) we can conclude that

$$\liminf_{t\to\infty}\mathrm{P}\{E_t\}\geq 1-\delta. \tag{6}$$

But, the event inside $\mathrm{P}\{\cdot\}$ in (5) is

$$\bigcap_{t\geq T^\star}E_t\subset\bigcup_{T^\star\geq 1}\bigcap_{t\geq T^\star}E_t=:\liminf_{t\to\infty}E_t,$$

and it follows from Fatou's lemma that the

$$\text{left-hand side of (5)}\leq\mathrm{P}\Big\{\liminf_{t\to\infty}E_t\Big\}\leq\liminf_{t\to\infty}\mathrm{P}\{E_t\}. \tag{7}$$

So, from (6) and (7), we can conclude that only the left-hand side of (5) is bounded from above by something greater than $1-\delta$ and, hence, (4) need not imply (5). Therefore, some pursuit learning specific considerations are needed and, to the authors' knowledge, there is no obvious way to fill this gap. In the next section we give a proof of ε-optimality based on some apparently new arguments.

3. A refined analysis of pursuit learning

3.1. An infinite-series result

Here we state an infinite-series result which will be useful in our analysis in Section 3.2. For completeness, a proof is given in Appendix A.

Lemma 1. *Given* $a, b \in (0, 1)$, *let* $\zeta(t) = (1 - a/t^b)^t$, $t \geq 1$. *Then* $\sum_{t=1}^{\infty} \zeta(t) < \infty$.

It is easy to see that the condition $b \in (0, 1)$ is necessary. Indeed, if $b > 1$ then the sequence itself converges to e^{-a} and the series cannot hope to converge. In our pursuit learning application below, this condition will be taken care of in our choice of tuning parameter sequence λ_t.

3.2. Main results

We start by summarizing a few known results from [20] which will be needed in the proof of the main theorem. Recall the notation $N_i(t)$ used for the number of times, up to iteration t, that action i has been tried, $i = 1, \dots, r$. The first result is that all of the $N(t)$s are unbounded in probability as the number of iterations $t \to \infty$.

Lemma 2. *Suppose that* λ_t *satisfies (1) with* $\mathrm{e}^{-1} < \theta < 1$. *Then, for any small* $\delta > 0$ *and any* $K > 0$, *there exists* $T_1^\star$ *such that, for each* $i = 1, \dots, r$,

$$\mathrm{P}\{N_i(t) \leq K\} < \delta \quad \textit{for all } t > T_1^\star.$$

As the number of times each action is played is increasing to ∞, it is reasonable to think that the estimates, namely the $\hat{d}(t)$s, should be approaching their respective targets, the ds. It turns out that this intuition is indeed correct.

Lemma 3. *Suppose that* λ_t *satisfies (1) with* $\mathrm{e}^{-1} < \theta < 1$. *Then, for any small* $\delta > 0$ *and any small* $\eta > 0$, *there exists* $T_2^\star$ *such that, for each* $i = 1, \dots, r$,

$$\mathrm{P}\{|\hat{d}_i(t) - d_i| > \eta\} < \delta \quad \textit{for all } t > T_2^\star.$$

An alternative way to phrase the previous two lemmas is that, under the stated conditions, $N_i(t)$ and $\hat{d}_i(t)$ converge in probability to ∞ and d_i, respectively, as $t \to \infty$. Next is the main ε-optimality result.

Theorem 1. *Suppose that* λ_t *satisfies (1) with* $\mathrm{e}^{-1} < \theta < 1$. *Then, for any small* $\varepsilon, \delta > 0$, *there exists* $T^\star$ *such that*

$$\mathrm{P}\{\pi_1(t) > 1 - \varepsilon\} > 1 - \delta \quad \textit{for all } t > T^\star.$$

To prove Theorem 1, we shall initially follow the argument of [15]. To simplify notation, define the events

$$A_\varepsilon(t) = \{\pi_1(t) > 1 - \varepsilon\}, \qquad t \geq 1.$$

Next, we observe that, since the reward probabilities are fixed, there is a number $\eta > 0$ such that, if $|\hat{d}_1(t) - d_1| < \eta$ then $\hat{d}_1(t)$ must be the largest of the estimates $\hat{d}(t)$ at iteration t. For this η, define the two sequences of events

$$B(t) = \{|\hat{d}_1(t) - d_1| < \eta\}, \qquad t > 0,$$

$$\bar{B}(T) = \Big\{\sup_{t \geq T} |\hat{d}_1(t) - d_1| < \eta\Big\} = \bigcap_{t \geq T} B(t), \qquad T > 0.$$

Then, for any positive integers t and T, the law of total probability gives

$$\mathrm{P}\{A_\varepsilon(t+T)\} \geq \mathrm{P}\{A_\varepsilon(t+T) \mid \bar{B}(T)\}\,\mathrm{P}\{\bar{B}(T)\}.$$

Moreover, from the monotonicity property of pursuit learning, it follows that there exists $T_3^\star$ such that

$$\mathrm{P}\{A_\varepsilon(t+T) \mid \bar{B}(T)\} = 1 \quad \text{for all } t > T_3^\star.$$

Therefore, to complete the proof, it remains to show that there exists $T > 0$ such that $\mathrm{P}\{\bar{B}(T)\} > 1-\delta$. By DeMorgan's law,

$$\mathrm{P}\{\bar{B}(T)\} = \mathrm{P}\biggl\{\bigcap_{t\geq T} B(t)\biggr\} = 1 - \mathrm{P}\biggl\{\bigcup_{t\geq T} B(t)^{\mathrm{c}}\biggr\},$$

so we are done if we can find $T > 0$ such that

$$\mathrm{P}\biggl\{\bigcup_{t\geq T} B(t)^{\mathrm{c}}\biggr\} < \delta. \tag{8}$$

To this end, write $N(t) = N_1(t)$ and note that

$$\mathrm{P}\biggl\{\bigcup_{t\geq T} B(t)^{\mathrm{c}}\biggr\} \leq \sum_{t\geq T} \mathrm{P}\{B(t)^{\mathrm{c}}\} = \sum_{t\geq T}\biggl(\sum_{n=0}^{t} \mathrm{P}\{B(t)^{\mathrm{c}} \mid N(t)=n\}\,\mathrm{P}\{N(t)=n\}\biggr).$$

It follows easily from Hoeffding's inequality [3] that

$$\mathrm{P}\{B(t)^{\mathrm{c}} \mid N(t)=n\} = \mathrm{P}\{|\hat{d}_1(t)-d_1| \geq \eta \mid N(t)=n\} \leq \mathrm{e}^{-hn},$$

where $h = \eta^2/8 > 0$ is a constant independent of n. Therefore,

$$\begin{aligned}\mathrm{P}\biggl\{\bigcup_{t\geq T} B(t)^{\mathrm{c}}\biggr\} &\leq \sum_{t\geq T}\biggl(\sum_{n=0}^{t} \mathrm{P}\{B(t)^{\mathrm{c}} \mid N(t)=n\}\,\mathrm{P}\{N(t)=n\}\biggr)\\ &\leq \sum_{t\geq T}\biggl(\sum_{n=0}^{t} \mathrm{e}^{-hn}\,\mathrm{P}\{N(t)=n\}\biggr),\end{aligned}$$

and the innermost sum is easily seen to be the moment generating function, call it $\psi_t(u)$, of the random variable $N(t)$ evaluated at $u=-h$. To prove that this sum is finite, we must show that $\psi_t(-h)$ vanishes sufficiently fast in t.

Formulae for moment generating functions of standard random variables are readily available. But, $N(t)$ is not a standard random variable; it is like a Bernoulli convolution [5], [14], but the summands are only conditionally Bernoulli. In Lemma 4 below we show that $\psi_t(u)$ for $u \leq 0$ is bounded above by a certain binomial random variable's moment generating function.

Lemma 4. *Consider a binomial random variable with parameters* (t, ω_t), *where* $\omega_t = \pi_1(0)\theta^{\gamma(t)}$ *and* $\gamma(t) = \sum_{s=1}^{t} s^{-1}$. *If* φ_t *is the corresponding moment generating function then* $\psi_t(u) \leq \varphi_t(u)$ *for* $u \leq 0$.

Proof. Let $N(t) = \sum_{s=1}^{t} \xi(s)$, where $\xi(s) = 1$ if the optimal action is sampled at iteration s and 0 otherwise. If $\mathcal{A}_s$ denotes the σ-algebra generated by the history of the algorithm up to and including iteration s, then $\xi(s)$ satisfies

$$\mathrm{E}\{\xi(s) \mid \mathcal{A}_{s-1}\} = \pi(s-1).$$

For $u \leq 0$, the moment generating function $\psi_t(u)$ of $N(t)$ satisfies

$$\begin{aligned}\psi_t(u) &= \mathrm{E}\{\mathrm{e}^{uN(t)}\}\\ &= \mathrm{E}\{\mathrm{e}^{u\xi(1)+\cdots+u\xi(t)}\}\\ &= \mathrm{E}\Biggl\{\prod_{s=1}^{t} \mathrm{E}\{\mathrm{e}^{u\xi(s)} \mid \mathcal{A}_{s-1}\}\Biggr\}\\ &= \mathrm{E}\Biggl\{\prod_{s=1}^{t} [1-(1-\mathrm{e}^{u})\pi(s-1)]\Biggr\}.\end{aligned}$$

Rajaraman and Sastry [15] showed that $\omega_t \leq \min\{\pi(1), \dots, \pi(t)\}$, so

$$\psi_t(u) \leq \prod_{s=1}^{t} [1-(1-\mathrm{e}^{u})\omega_t] = [1-(1-\mathrm{e}^{u})\omega_t]^t.$$

The right-hand side above is exactly $\varphi_t(u)$, completing the proof.

Back to our main discussion, we now have

$$\mathrm{P}\Biggl\{\bigcup_{t\geq T} B(t)^{\mathrm{c}}\Biggr\} \leq \sum_{t\geq T} \psi_t(-h) \leq \sum_{t\geq T} \varphi_t(-h),$$

and it is well known that the moment generating function φ_t for the binomial random variable satisfies

$$\varphi_t(-h) = [1-\omega_t(1-\mathrm{e}^{-h})]^t = [1-\pi_1(0)(1-\mathrm{e}^{-h})\theta^{\gamma(t)}]^t.$$

The sequence $\gamma(t)$ grows like $\ln(t)$, and $\theta^{\ln(t)} = t^{\ln(\theta)}$, so, for large t,

$$\varphi_t(-h) \sim \biggl[1 - \frac{\pi_1(0)(1-\mathrm{e}^{-h})}{t^{-\ln(\theta)}}\biggr]^t.$$

The right-hand side above is just the sequence $\zeta(t)$ defined in Lemma 1, with

$$a = \pi_1(0)(1-\mathrm{e}^{-h}) \quad \text{and} \quad b = -\ln(\theta).$$

Thus, the series $\sum_{t\geq 1} \varphi_t(-h)$ converges; so, for any δ, there exists T such that $\sum_{t\geq T} \varphi_t(-h) < \delta$, proving (8). To put everything together, let $T_4^\star$ be the smallest T with $\sum_{t\geq T} \varphi_t(-h) < \delta$. Then Theorem 1 follows by taking $T^\star = T_3^\star + T_4^\star$.

A natural question to ask is whether a deterministic bound for $T^\star$ in terms of the user specified ε, δ, and θ can be given. An affirmative answer to this question is given in Appendix B. We choose not to give much emphasis to this result, as the bound we obtain appears to be quite conservative. For example, for numerical experiments run under the setup in Simulation 1 of [18], we find that more than 95% of sample paths converge in roughly 25–250 iterations, while our conservative theoretical bounds are, for moderate θ, orders of magnitude greater.

4. Discussion

In this paper we have taken a closer look at convergence properties of pursuit learning. In particular, we have identified a gap in existing proofs of ε-optimality and provided a new argument to fill this gap. An important consequence of our theoretical analysis is that it seems necessary to explicitly specify the rate at which the tuning parameter sequence $\lambda = \lambda_t$ vanishes with t. In fact, if ω_t defined in Lemma 4 vanishes too quickly, which it would if $\lambda_t \equiv \lambda$, then $\sum_t \varphi_t(-h) = \infty$ and the proof fails. But we should also reiterate that a theoretical analysis that requires vanishing λ_t need not conflict with the tradition of running the algorithm with fixed small λ in practical applications. In fact, the particular λ_t vanishes relatively fast, so, for applications, we recommend running pursuit learning with, say,

$$\lambda_t = 1 - \theta^{[1+(t-t_0)^+]^{-1}},$$

where t_0 is some fixed cutoff and $x^+ = \max\{0, x\}$. This effectively keeps λ_t constant for a fixed finite period of time, after which it vanishes like that in (1). Alternatively, one might consider $\lambda_t = 1 - \theta^{v(t)}$, where $v(t)$ vanishes more slowly than t^{-1}. This choice of λ_t vanishes more slowly than that in (1), thus giving the algorithm more opportunities to adjust to the environment. We believe that an analysis similar to ours can be used to show that the corresponding pursuit learning converges in the sense of Definition 1.

It is also worth mentioning that the results of [20], for λ_t as in (1), can be applied to show that $\pi_1(t) \to 1$ with probability 1 as $(t, \lambda_t) \to (\infty, 0)$. This, of course, immediately implies ε-optimality in the sense of Definition 1. However, this indirect argument does not give any insight as to how to bound the number of iterations needed to be sufficiently close to convergence, as we do—albeit conservatively—in Appendix B. But the result proved in [20] that $\hat{d}_1(t)$ is the largest among the $\hat{d}(t)$s infinitely often with probability 1, together with the formula in (9) in Appendix B, can perhaps be used to reason towards an almost-sure rate of convergence for pursuit learning.

Appendix A. Proof of Lemma 1

To start, write $\zeta(t)$ as

$$\zeta(t) = \left(1 - \frac{a}{t^b}\right)^t = \left[\left(1 - \frac{a}{t^b}\right)^{t^b}\right]^{t^{1-b}}.$$

If $f(t) = (1 - a/t)^t$ then ordinary calculus reveals that

$$\frac{\mathrm{d}}{\mathrm{d}t} \ln f(t) = \ln\left(1 - \frac{a}{t}\right) + \frac{a}{t-a} = -\ln\left(1 + \frac{a}{t-a}\right) + \frac{a}{t-a} > 0.$$

Therefore, we have shown that $\ln f(t)$ and, hence, $f(t)$ and, hence, $f(t^b)$ are monotone increasing. Moreover, $f(t^b) \uparrow \mathrm{e}^{-a} < 1$. Thus, $\zeta(t) \le \exp\{-at^{1-b}\}$. So to show that $\sum_{t=1}^{\infty} \zeta(t)$ is finite, it suffices to show that, for $c = 1 - b$,

$$\int_1^{\infty} \mathrm{e}^{-at^c}\, \mathrm{d}t < \infty.$$

Making the change of variable $x = t^c$, the integral becomes

$$\int_1^{\infty} \mathrm{e}^{-at^c}\, \mathrm{d}t = \int_1^{\infty} \frac{1}{cx^{1-1/c}} \mathrm{e}^{-ax}\, \mathrm{d}x = \frac{1}{c} \int_1^{\infty} x^{1/c-1} \mathrm{e}^{-ax}\, \mathrm{d}x.$$

Since $1/c > 1$ and $a > 0$, the integral is finite, completing the proof. Making one more change of variable, we find that the last integral above can be expressed as

$$\int_1^\infty \mathrm{e}^{-at^c}\,\mathrm{d}t = \frac{1}{ca^{1/c}}\int_a^\infty y^{1/c-1}\mathrm{e}^{-y}\,\mathrm{d}y = \frac{1}{ca^{1/c}}\Gamma(c^{-1}; a),$$

where $\Gamma(s,x) = \int_x^\infty u^{s-1}\mathrm{e}^{-u}\,\mathrm{d}u$ is the incomplete gamma function.

Appendix B. A bound on the number of iterations

As a follow-up to the proof of Theorem 1, we give a conservative upper bound on the number of iterations $T^\star$ needed to be sufficiently close to convergence.

Theorem 2. *For given* $\varepsilon, \delta \in (0,1)$ *and* $\theta \in (\mathrm{e}^{-1}, 1)$, *a deterministic bound on the necessary number of iterations* $T^\star$ *in Theorem 1 can be found numerically.*

In the proof that follows, we assume that h is a known constant, while it actually depends on the η used above which, in turn, depends on the unknown ds. The bounds obtained in [15] also depend on η, called the size of the problem. To use this bound in practice, users must estimate η by some other means.

Proof of Theorem 2. As stated above, the desired $T^\star$ is actually a sum $T_3^\star + T_4^\star$. Let us begin with $T_4^\star$, the smallest $T = T(\delta)$ such that $\sum_{t\ge T}\varphi_t(-h) < \delta$. From the proof of the classical integral test in calculus, it follows that

$$\sum_{t\ge T}\varphi_t(-h) \le \varphi_T(-h) + \int_T^\infty \varphi_t(-h)\,\mathrm{d}t.$$

A modification of the argument presented in Appendix A shows that

$$\int_T^\infty \varphi_t(-h)\,\mathrm{d}t \le \frac{b}{a^b}\Gamma(b; aT),$$

where $a = \pi_1(0)(1-\mathrm{e}^{-h})$, $b = (1+\ln\theta)^{-1}$, and $\Gamma(s,x)$ is the incomplete gamma function (defined in Appendix A). Since $\varphi_T(-h)$ and $\Gamma(b; aT)$ are both decreasing functions of T, it is possible to solve the equation

$$\varphi_T(-h) + \frac{b}{a^b}\Gamma(b; aT) = \delta$$

for T numerically to obtain the bound $T_4^\star$ in terms of the user-specified inputs.

Towards bounding $T_3^\star$, we note that $\hat{d}_1(t+T_4^\star)$ is the largest of the $\hat{d}$s for all $t \ge 1$ for sample paths in a set Ω of probability greater than $1-\delta$. For sample paths in this Ω, Tilak *et al.* [20] proved that

$$\pi_1(t+T_4^\star) = 1 - \theta^{\gamma(t+T_4^\star)}\{1-\pi_1(T_4^\star)\}, \qquad t \ge 1, \tag{9}$$

where $\gamma(t) = \sum_{s=1}^t s^{-1}$. Since $\pi_1(T_4^\star) \in (0,1)$, it easily follows that

$$1 - \pi_1(t+T_4^\star) \le \theta^{\gamma(t+T_4^\star)}, \qquad t \ge 1.$$

Given ε and $T_4^\star$, it is easy to calculate $T_3^\star$ such that $\theta^{\gamma(t+T_4^\star)} \le \varepsilon$ for all $t > T_3^\star$.

Acknowledgements

A portion of this work was completed while the first author was affiliated with the Department of Mathematical Sciences at Indiana University–Purdue University Indianapolis. The authors thank Professors Snehasis Mukhopadhyay and Chuanhai Liu for sharing their insight, and the Editor and two anonymous referees for some helpful suggestions.

References

[1] Agache, M. and Oommen, B. J. (2002). Generalized pursuit learning schemes: new families of continuous and discretized learning automata. *IEEE Trans. Systems Man Cybernet.* **32,** 738–749.

[2] Atlasis, A. F., Loukas, A. N. H. and Vasilakos, A. V. (2000). The use of learning algorithms in ATM networks call admission control problem: a methodology. *Comput. Networks* **34,** 341–353.

[3] Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.* **58,** 13–30.

[4] Kashki, M., Abido, M. A. and Abdel-Magid, Y. L. (2010). Pole placement approach for robust optimum design of PSS and TCSC-based stabilizers using reinforcement learning automata. *Electrical Eng.* **91,** 383–394.

[5] Klenke, A. and Mattner, L. (2010). Stochastic ordering of classical discrete distributions. *Adv. Appl. Prob.* **42,** 392–410.

[6] Kushner, H. J. and Yin, G. G. (2003). *Stochastic Approximation and Recursive Algorithms and Applications*, 2nd edn. Springer, New York.

[7] Lanctôt, J. K. and Oommen, B. J. (1992). Discretized estimator learning automata. *IEEE Trans. Systems Man Cybernet.* **22,** 1473–1483.

[8] Lixia, L., Gang, H., Ming, X. and Yuxing, P. (2010). Learning automata based spectrum allocation in cognitive networks. In *IEEE Internat. Conf. Wireless Communications, Networking, and Information Security*, pp. 503–508.

[9] Misra, S., Tiwari, V. and Obaidat, M. S. (2009). Lacas: learning automata-based congestion avoidance scheme for healthcare wireless sensor networks. *IEEE J. Selected Areas Commun.* **27,** 466–479.

[10] Narendra, K. S. and Thathachar, M. A. L. (1989). *Learning Automata: An Introduction*. Prentice Hall, Englewood Cliffs, NJ.

[11] Oommen, B. J. and Hashem, M. K. (2010). Modeling a student's behavior in a tutorial-like system using learning automata. *IEEE Trans. Systems Man Cybernet. B* **40,** 481–492.

[12] Oommen, B. J. and Lanctôt, J. K. (1990). Discretized pursuit learning automata. *IEEE Trans. Systems Man Cybernet.* **20,** 931–938.

[13] Papadimitriou, G. I., Sklira, M. and Pomportsis, A. S. (2004). A new class of ϵ-optimal learning automata. *IEEE Trans. Systems Man Cybernet. B* **34,** 246–254.

[14] Proschan, F. and Sethuraman, J. (1976). Stochastic comparisons of order statistics from heterogeneous populations, with applications in reliability. *J. Multivariate Anal.* **6,** 608–616.

[15] Rajaraman, K. and Sastry, P. S. (1996). Finite time analysis of the pursuit algorithm for learning automata. *IEEE Trans. Systems Man Cybernet. B* **26,** 590–598.

[16] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Ann. Math. Statist.* **22,** 400–407.

[17] Sastry, P. S. (1985). Systems of learning automata: estimator algorithms and applications. Doctoral Thesis, Indian Institute of Science.

[18] Thathachar, M. A. L. and Sastry, P. S. (1985). A new approach to the design of reinforcement schemes for learning automata. *IEEE Trans. Systems Man Cybernet.* **15,** 168–175.

[19] Thathachar, M. A. L. and Sastry, P. S. (1987). Learning optimal discriminant functions through a cooperative game of automata. *IEEE Trans. Systems Man Cybernet.* **17,** 73–85.

[20] Tilak, O., Martin, R. and Mukhopadhyay, S. (2011). Decentralized, indirect methods for learning automata games. *IEEE Trans. Systems Man Cybernet. B* **41,** 1213–1223.

[21] Torkestania, J. A. and Meybodi, M. R. (2010). Clustering the wireless ad hoc networks: a distributed learning automata approach. *J. Parallel Distributed Computing* **70,** 394–405.

[22] Torkestania, J. A. and Meybodi, M. R. (2010). An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata. *Comput. Networks* **54,** 826–843.

[23] Tuan, T. A., Tong, L. C. and Premkumar, A. B. (2010). An adaptive learning automata algorithm for channel selection in cognitive radio network. In *2010 Internat. Conf. Communications and Mobile Computing*, Vol. 2, IEEE Computer Society, Washington, DC, pp. 159–163.

[24] Zhong, W., Xu, Y. and Tao, M. (2010). Precoding strategy selection for cognitive MIMO multiple access channels using learning automata. In *2010 IEEE Internat. Conf. Communications*, pp. 23–27.