LMS-2: Towards an algorithm that is as cheap as LMS and almost as efficient as RLS

Hengshuai Yao
Department of Computing Science
University of Alberta
T6G2E8 Edmonton, AB, Canada
hengshua@cs.ualberta.ca

Shalabh Bhatnagar
Department of Computer Science and Automation
Indian Institute of Science
560 012, Bangalore, India
shalabh@csa.iisc.ernet.in

Csaba Szepesvári
Department of Computing Science
University of Alberta
T6G2E8 Edmonton, AB, Canada
szepesva@cs.ualberta.ca

***Abstract*— We consider linear prediction problems in a stochastic environment. The least mean square (LMS) algorithm is a well-known, easy to implement and computationally cheap solution to this problem. However, as it is well known, the LMS algorithm, being a stochastic gradient descent rule, may converge slowly. The recursive least squares (RLS) algorithm overcomes this problem, but its computational cost is quadratic in the problem dimension. In this paper we propose a two timescale stochastic approximation algorithm which, as far as its slower timescale is considered, behaves the same way as the RLS algorithm, while it is as cheap as the LMS algorithm. In addition, the algorithm is easy to implement. The algorithm is shown to give estimates that converge to the best possible estimate with probability one. The performance of the algorithm is tested in two examples and it is found that it may indeed offer some performance gain over the LMS algorithm.**

I. INTRODUCTION

Consider the problem of estimating $\theta_* \in \mathbb{R}^d$ based on the i.i.d. sequence $(\phi_t, y_t) \in \mathbb{R}^d \times \mathbb{R}$, assuming that

$$y_t = \theta_*' \phi_t + \varepsilon_t, \quad t = 1, 2, \ldots, \tag{1}$$

where ε_t is a zero mean noise sequence with finite variance. This problem has been studied in various contexts, such as statistics, control, signal processing, system identification or machine learning [5].

Here, we are interested in algorithms that incrementally update an estimate θ_t of the parameter based on the most recent observations, such that $\theta_t \to \theta$ with probability one (w.p.1). More precisely, we are interested in algorithms that make θ_t converge to θ relatively fast, while they are simple to implement and computationally efficient. In particular, we are interested in algorithms whose per-update computational complexity is linear ($O(d)$) in the number of dimensions of the feature vectors. Such algorithms are important for application where computational resources are scarce, or when the number of features is so large that algorithms with higher computational complexity are infeasible.

One algorithm whose computational cost is linear is the Least-Mean-Square (LMS) algorithm. The LMS update rule is

$$\theta_{t+1} = \theta_t + \alpha_t (y_t - \theta_t' \phi_t)\, \phi_t,$$

where $(\alpha_t,\ t \geq 0)$, the so-called step-size sequence, is a sequence of non-negative numbers. In order to make sure that θ_t converges to θ w.p.1, the step-size sequence has to converge to zero, while satisfying

$$\sum_{t=1}^{\infty} \alpha_t = \infty, \quad \sum_{t=1}^{\infty} \alpha_t^2 < \infty.$$

The LMS algorithm is attractive due to its simplicity and low complexity. However, as is well known, the LMS algorithm is sensitive to the eigenvalue spread of the dispersion matrix $\Phi = \mathbb{E}\left[\phi_t \phi_t'\right]$ in that a large eigenvalue spread of Φ results in a slow rate of convergence of the error, $e_t = \mathbb{E}\left[\|\theta_t - \theta\|^2\right]$, to zero in the transient phase (see [5]). The asymptotic rate of convergence of a two time-scale version of LMS can be shown to be optimal. In particular, if the LMS algorithm is used with large step-sizes such as $\alpha_t \approx \mathrm{const}/t^p$ where p is close to $1/2$ and the final estimate is provided by a second update equation

$$\overline{\theta}_{t+1} = \overline{\theta}_t + \beta_t (\theta_t - \overline{\theta}_t),$$

where $\beta_t = 1/t$, then the iterates $\overline{\theta}_t$ can be shown to converge to θ at an asymptotically optimal rate (see, [10] and Chapter 11 of [8]). This idea works more generally and is called iterate-averaging and was developed independently by Polyak [9] and Ruppert [11].

One way to improve the transient performance is to let $\theta_t = \mathrm{argmin}_\theta \sum_{s=1}^{t} (y_s - \theta' \phi_s)^2$, i.e., the least-squares estimate of θ given all previous data. In this case the effect of the eigenvalue spread of Φ will be smaller on the rate

of convergence.[1] Using the Sherman-Morrison formula, one can derive an incremental update rule for θ_t as follows:

$$\begin{aligned} P_{t+1} &= \frac{1}{\lambda_t}\left[P_t - \frac{P_t\phi_t\phi_t'P_t'}{\lambda_t + \phi_t'P_t\phi_t}\right] \\ \theta_{t+1} &= \theta_t + \eta_t\left(y_t - \theta_t'\phi_t\right)P_t\phi_t. \end{aligned}$$

Here $\eta_t = 1/(\lambda_t + \phi_t'P_t\phi_t)$ is a step-size sequence and λ_t is a forgetting factor, which in stationary environments can be set to 1.[2] The price paid for the improved statistical performance is the increased computational complexity that is now $O(d^2)$.

The question studied in this paper is whether it is possible to design an algorithm that is as efficient in the transient phase as RLS and whose computational complexity is the same as that of LMS. In FIR filtering applications this question has been answered positively: In this case there exist algorithms which are as efficient as RLS, yet achieve $O(d)$ complexity [6], [3]. However, these algorithms exploit heavily the special structure of the task and are not easy to generalize. The normalized LMS (NLMS) algorithm is another candidate that aims to achieve this goal. However, for fixed schedules, the NLMS algorithm is known to trade off transient performance for asymptotic performance. Slock suggested an "optimal" step-size to be used with NLMS [12]. Following a heuristic analysis he concludes that NLMS performs the same as RLS as far as sensitivity to the eigenvalue spread is concerned. Our interpretation is that normalization decreases the eigenvalue spread and thus indeed helps. However, when the inputs have unit norm, NLMS is identical to LMS. Hence, if the eigenvalue spread is large when the features are normalized, then NLMS should be as sensitive to the eigenvalue spread as LMS is.

II. THE NEW ALGORITHM

RLS is a stochastic version of Newton's method, as can be seen if it is written in the form

$$\theta_{t+1} = \theta_t + \alpha_t\Phi_t^{-1}(y_t - \theta_t'\phi_t)\phi_t, \tag{2}$$

where Φ_t is the empirical dispersion matrix:

$$\Phi_{t+1} = \Phi_t + \gamma_t(\phi_t\phi_t' - \Phi_t),$$

where α_t, γ_t, $t \geq 0$ are two step-size schedules. To understand why RLS is faster than LMS let us consider the ordinary differential equations (ODEs) underlying them. The ODE underlying the LMS algorithm takes the form $\dot\theta = b - \Phi\theta$, where $b = \mathbb{E}[y_t\phi_t]$. On the other hand, the ODE underlying the RLS algorithm takes the form, $\dot\theta = \Phi^{-1}b - \theta$. The solutions of these ODEs are exponentially converging, where the time constants depend on the eigenvalues of Φ. Now, we can think of the stochastic approximation methods as simulating the behavior of the respective ODEs with Euler's method, with decreasing step-sizes and in a "noisy manner". It is well known, that using too large step-sizes causes Euler's method to oscillate, or even diverge. In particular, if the i^{th} component's dynamics is $\dot\theta_i = a_i - \lambda_i\theta_i$, $\lambda_i > 0$, then the "safe zone" for the step-sizes is $(0, 1/\lambda_i)$. After a change of coordinate systems, one can obtain that the ODE takes the above form.

In particular, for the LMS method, the coefficients λ_i will correspond to the eigenvalues of matrix Φ. Thus, we can see that the LMS method can behave erratically until the step-size becomes smaller than the reciprocal value of the largest eigenvalue of Φ. This suggests to choose small step-sizes, but then the rate of convergence of the component corresponding to the smallest eigenvalue will be slowed down. The decreasing step-sizes make sure that the erratic behavior can only happen for a finite amount of time. However, the behavior in this transient phase will largely influence the rate of convergence. Hence, the ratio of the largest to the smallest eigenvalue of Φ will strongly (negatively) influence the behavior of LMS.

From the above reasoning, it is also clear why the RLS algorithm is less sensitive to the eigenvalue spread: In the ODE underlying RLS, the $-\Phi\theta$ term of the ODE of LMS is replaced by $-\theta$. Hence, the eigenvalues become all identical and in particular the eigenvalues of Φ do not influence the behavior of this ODE.

Now, the key difference of RLS and LMS is that the expected update direction of RLS, assuming that Φ_t converges to Φ, is

$$u^*(\theta) = \Phi^{-1}b - \theta.$$

The idea of the new algorithm is to use another stochastic approximation algorithm to approximate $u^*(\theta_t)$ for a fixed θ_t and then update θ_t with the resulting estimate. Clearly, $\Phi u^*(\theta) - (b - \Phi\theta) = 0$, or

$$\mathbb{E}\left[\phi_t(\phi_t'u^*(\theta) - \delta_t)\right] = 0,$$

where

$$\delta_t = y_t - \theta'\phi_t.$$

Hence, an appropriate update equation for estimating $u^*(\theta_t)$ is

$$u_{t+1} = u_t + \beta_t(\delta_t - u_t'\phi_t)\phi_t,$$

where β_t, $t \geq 0$ is a step-size sequence.

Thus, the proposed algorithm, which we call LMS-2, or the *second order LMS algorithm*, takes the form:

$$u_{t+1} = u_t + \beta_t(\delta_t - \phi_t'u_t)\phi_t, \tag{3}$$

$$\theta_{t+1} = \theta_t + \alpha_t u_{t+1}. \tag{4}$$

Here, u_t is updated on a faster time-scale than θ_t (i.e., $\alpha_t = o(\beta_t)$ viz., $\alpha_t/\beta_t \to 0$ as $t \to \infty$). Thus, from the point of view of θ_t, u_t can be thought of as having equilibrated to $u^*(\theta_t)$. (This will be formalized more precisely in the next section.) Hence, the update of θ_t approximately takes the form $\theta_{t+1} = \theta_t + \alpha_t u^*(\theta_t)$, i.e., the algorithm is expected to simulate the behavior of RLS. Note that the ODE underlying

[1] This is usually analyzed in the case of constant step-sizes, see, e.g., Chapter 5 of [14], but we also provide a heuristic explanation of this in the next section. We expect a similar conclusion to be true in the transient when diminishing step-sizes are used, but we were not able to find such an analysis. A similar difference between the performance of LMS and RLS exists when the inputs ϕ_t are correlated in time [4], again proven for the constant step-size case.

[2] This is the setting we used for our experiments.

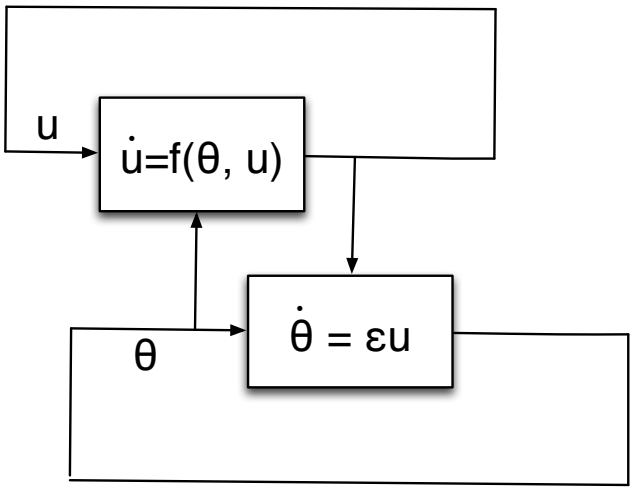

Fig. 1: The process of LMS-2.

the update equation of u_t takes the form $\dot{u} = (b - \Phi\theta) - \Phi u$ (assuming a fixed θ). Thus, the eigenvalue spread of Φ is expected to influence the transient behavior of u in the same way as it influenced the transient behavior of the LMS algorithm. However, the main recursion here is the one of θ_t and as indicated above, with respect to the timescale of this recursion, the recursion of u_t rapidly equilibrates. Thus the transient behavior of u_t in the analysis of the θ update does not matter as much as LMS. Figure 1 describes the general update rule for the LMS-2 algorithm involving two step-size schedules. As can be seen from (3)-(4), the computational complexity of LMS-2 is $O(d)$.

III. THEORY

We make the following assumptions.

Assumption 1 (ϕ_t, y_t), $t \geq 1$ are i.i.d. random variables and (ϕ_t) is uniformly bounded.[3]

Assumption 2 The matrix $\Phi = \mathbb{E}\left[\phi_t \phi_t^T\right]$ is nonsingular.

Assumption 3(i): Diminishing Step-Sizes The step-sizes α_t, β_t, $t \geq 0$ satisfy

$$\sum_t \alpha_t = \sum_t \beta_t = \infty, \ \sum_t (\alpha_t^2 + \beta_t^2) < \infty, \ \alpha_t = o(\beta_t). \tag{5}$$

Assumption 3(ii): Constant Step-Sizes The step-sizes α_t, β_t, $t \geq 0$ are in fact constants $\alpha_t = \bar{\alpha}$, $\beta_t = \bar{\beta}$, $t \geq 0$. Further, $\bar{\alpha} << \bar{\beta}$.

Remarks:

1) We shall assume either of Assumption 3(i) or 3(ii) alongwith Assumptions 1 and 2 in the convergence analysis.

[3]Hence, the setting considered here is slightly more general than the one mentioned in the introduction, where a particular relationship was assumed between y_t and ϕ_t.

2) From Assumption 1, it follows that the matrix Φ is uniformly bounded as well, i.e., $\| \Phi \| < \infty$. Here we define the matrix norm $\| \Phi \|$ as one obtained from the corresponding vector norm (which we assume is the Euclidean norm), also denoted $\| \cdot \|$ by abuse of notation and defined according to

$$\| \Phi \| = \max_{\{x \in \mathcal{R}^n | \|x\|=1\}} \| \Phi x \| .$$

3) The i.i.d. requirement in Assumption 1 can be relaxed and instead the sequence can be assumed to be a Markov process. The arguments in the following analysis can be modified to include this case. We leave it as one of the possible future directions.
4) Note that from its definition, the matrix Φ is a positive semi-definite matrix since

$$x^T \Phi x = x^T \mathbb{E}\left[\phi_t \phi_t^T\right] x = \mathbb{E}\left[(\phi_t^T x)^T (\phi_t^T x)\right] \geq 0,$$

for all $x \neq 0$. Hence, from Assumption 2, it follows that the matrix Φ does not contain the eigenvalue zero. Thus, Φ is in fact positive definite under Assumption 2.
5) From the above, under Assumptions 1 and 2, we also have that $\| \Phi^{-1} \| < \infty$.
6) Both Assumptions 3(i) and 3(ii) imply that we are in the two-timescale setting with either diminishing or constant step-size schedules where α_t, $t \geq 0$ corresponds to the slower step-size schedule and β_t, $t \geq 0$ to the faster one.

In what follows we shall use (ϕ, y) (without an index) to denote a pair of random variables whose joint distribution is the same as the one underlying (ϕ_t, y_t).

A. Convergence Analysis

Let $\theta_* = \Phi^{-1} \mathbb{E}[y\phi]$. We have the following convergence theorem.

Theorem 1 Let Assumptions 1, 2 and 3(i) hold. Then $\theta_t \to \theta_*$ as $t \to \infty$ with probability one.

Proof Before the proof, let us remind the reader that the ODE $\dot{\theta} = A\theta + b$ is globally asymptotically stable if and only if the real parts of the eigenvalues of A are all negative. In fact, in all the cases when we apply this result, A will be negative definite. The proof of convergence here follows in a straightforward manner from the results in [1], [2].

We will use Theorem 2 of [1]. Accordingly, we need to look at the faster recursion (3) and determine its behavior and limit point when θ_t is kept fixed. Then we investigate the behavior of the slower recursion, when this limit point is used in place of u_t.

Consider thus first the faster recursion (3). We verify Assumptions (A1) and (A2) of [2]. Let $\mathcal{G}_t = \sigma(u_s, \theta_s, s \leq t; \phi_s, s < t)$, $t \geq 0$ be an associated sequence of sigma fields. Note that (3) can now be written as

$$u_{t+1} = u_t + \beta_t(\mathbb{E}\left[\delta_t \phi_t | \theta_t\right] - \Phi u_t) + \beta_t M_{t+1}$$

where

$$M_{t+1} = (\delta_t - \phi_t^T u_t)\phi_t - \mathbb{E}\left[(\delta_t - \phi_t^T u_t)\phi_t \mid \mathcal{G}_t\right]$$

$$= (\delta_t - \phi_t^T u_t)\phi_t - (\mathbb{E}\left[\delta_t \phi_t | \theta_t\right] - \Phi u_t).$$

It is easy to see that $\mathbb{E}\left[M_{t+1} \mid \mathcal{G}_t\right] = 0$ for all $t \geq 0$. Further, in the light of Assumption 1,

$$\mathbb{E}\left[\| M_{t+1} \|^2 | \mathcal{G}_t\right] \leq K(1 + \| u_t \|^2 + \| \theta_t \|^2)$$

for some constant $K > 0$. Consider now the following system of ODEs:

$$\dot{\theta} = 0, \tag{6}$$

$$\dot{u} = \mathbb{E}\left[\delta_t \phi_t | \theta_t\right] - \Phi u. \tag{7}$$

Note that in the definition of δ_t in (7), $\theta_t \equiv \theta$ i.e., θ is constant as a result of (6), see arguments on pp.65 of [1].

Let $h(u)$ denote the driving vector field of (7). Thus $h(u) = \mathbb{E}\left[\delta\phi\right] - \Phi u$. For a given $c \geq 1$, let $h_c(\cdot)$ be defined according to $h_c(u) = \dfrac{h(cu)}{c}$. Then $h_c \rightarrow h_\infty$ as $c \rightarrow \infty$ uniformly over compact sets where $h_\infty(u) = -\Phi u$. Now, by our remark, for the ODE

$$\dot{u} = h_\infty(u) = -\Phi u, \tag{8}$$

the origin is the unique globally asymptotically stable equilibrium since Φ is positive definite (as explained before). Similarly, for the ODE (7), $u^*(\theta) = \Phi^{-1}\mathbb{E}\left[\delta\phi\right]$ is a globally asymptotically stable equilibrium. Assumptions (A1)-(A2) of [2] are now satisfied and one obtains (i) $\sup_t \| u_t \| < \infty$ (from Theorem 2.1(i) of [2]) and (ii) $u_t \rightarrow u$ as $t \rightarrow \infty$ with probability one (from Theorem 2.2 of [2]).

Now consider the slower recursion (4) when viewed from its own (i.e., the slower timescale) schedule. In the light of the above and Theorem 2, pp.66 of [1], it suffices to study the ODE underlying it.

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t u^*(\theta_t) \\ &= \theta_t + \alpha_t(\Phi^{-1}\mathbb{E}\left[y_t \phi_t\right] - \theta_t). \end{aligned}$$

Thus, consider the ODE associated with (9):

$$\dot{\theta} = \Phi^{-1}\mathbb{E}\left[y\phi\right] - \theta. \tag{9}$$

Again let $G(\theta)$ denote the driving vector field for the ODE (9). Thus $G(\theta) = \Phi^{-1}\mathbb{E}\left[y\phi\right] - \theta$. Now define $G_c(\theta) = \dfrac{G(c\theta)}{c}$, $c \geq 1$. Let $G_\infty(\theta) = \lim_{c\rightarrow\infty} G_c(\theta) = -\theta$. Now consider the ODE

$$\dot{\theta} = -\theta. \tag{10}$$

Again, the origin is a unique globally asymptotically stable equilibrium for (10). Now for the ODE (9), $\theta = \Phi^{-1}\mathbb{E}\left[y\phi\right]$ is the unique globally asymptotically stable equilibrium. Now from Theorem 2.1(i) of [2], one gets $\sup_n \| \theta_n \| < \infty$ with probability one. Further, from Theorem 2.2 of [2], one obtains $\theta_t \rightarrow \theta_*$ as $t \rightarrow \infty$ with probability one. This completes the proof.

Theorem 2 Let Assumptions 1, 2 and 3(ii) hold. Then there exist $\alpha, \beta > 0$ such that for $0 < \bar{\alpha} < \alpha$ and $0 < \bar{\beta} < \beta$ and for any given $\epsilon > 0$, there exist $b_1(\epsilon), b_2(\epsilon) > 0$ such that

$$P(\| \theta_t - \theta \| > \epsilon) \leq b_1(\epsilon)\bar{\beta} + b_2(\epsilon)\left(\frac{\bar{\alpha}}{\bar{\beta}}\right).$$

Proof A detailed proof of this result will be provided in a longer version of this paper. We refer the reader to Chapter 9.3, pp.112 of [1] for ideas along these lines. Note however that the result in [1] assumes that the iterates u_t, θ_t satisfy $\sup_t \mathbb{E}\left[\| u_t \|^2\right], \sup_t \mathbb{E}\left[\| \theta_t \|^2\right] < \infty$, respectively, i.e., that the iterates remain bounded in a mean-square sense. We provide verifiable sufficient conditions in a general setting (which the current setting is a special case of) along the lines of [2] for these requirements.

B. A Discussion on the Convergence Rate

We now briefly discuss the (asymptotic) rate of convergence aspect. Consider first a one-timescale algorithm as below:

$$x_{t+1} = x_t + \beta_t(b - Ax_t + W_t).$$

Here W_t, $t \geq 0$ are certain zero-mean noise random variables. The linear stochastic iteration above will converge under suitable conditions to the solution of the linear system of equations

$$Ax = b.$$

Suppose x is a solution to the above system of equations. In [10], the average

$$\bar{\theta}_t = \frac{1}{t}\sum_{l=1}^{t} x_l$$

is suggested to be used as an estimate of the solution of the above system of equations instead of x_t. It is also shown that $t^{1/2}(\bar{\theta}_t - x)$ is asymptotically normally distributed with mean 0 and a certain covariance matrix that depends on both A and the covariance matrix of W. Their analysis shows that the performance of one-timescale algorithms can be enhanced (via the asymptotic rate of convergence) using the above iterate averaging. Iterate averaging such as suggested by [10] can in fact be easily obtained using two timescales. Konda and Tsitsiklis [7] follow this approach and obtain rate of convergence for general two-timescale algorithms in the case of linear systems under certain assumptions. They generalize the iterate averaging idea of [10] by considering more general (though linear) recursions. In the recursions in [7] along the faster timescale, one does not necessarily do iterate averaging as in [10] but these iterations can be more general. (Our two-timescale algorithm is also for a linear system of equations.) One of the main results in [7] (Theorem 4.1) says that for an algorithm such as (3)-(4) (under the assumptions in [7]), $\alpha_t^{-1/2}(\theta_t - \theta)$ converges in distribution to that of a Gaussian random variable with mean zero and a certain covariance matrix. Another important result in [7] (cf.Theorem 3.1) compares the single-timescale and two-timescale algorithms and says that the minimal possible covariance of $\alpha_t^{-1/2}(\theta_t - \theta)$ when the 'gain matrices' in each can be chosen freely is the same for both two-timescale and one-timescale recursions. The one-timescale case corresponds to the scenario where we let $\alpha_t = \eta\beta_t$ in our algorithm for some $\eta > 0$ (but not arbitrarily close to zero).

IV. EXPERIMENTAL RESULTS

A. A Low-dimensional Problem

We first consider a low-dimensional problem, in which we sample from a set comprising of two samples – $x^{(1)}$ and $x^{(2)}$, for which the targets are $y^{(1)} = 1$ and $y^{(2)} = -1$, respectively. In the experiment, $x^{(1)}$ was sampled 95% of the time, and $x^{(2)}$ was sampled 5% of the time. At each time step t, after sampling $x_t \in \{x^{(1)}, x^{(2)}\}$, we can observe the target, y_t, with some noise. For our experiments, we let $\tilde{y}_t$=$y_t + \epsilon_t$ be the noise-corrupted target for x_t, where ϵ_t is obtained as an independent sample from $N(0, 1)$ (i.e., ϵ_l, $l = 1, 2, \ldots$ are independent, $N(0, 1)$-distributed random variables).

We used the following correlated features:

$$x_1 = [1, 1]'; \quad x_2 = [0, 1]'.$$

We evaluate the performance of the algorithms by using the 2-norm error

$$e_t = ||\theta_t - \theta_*||_2.$$

For our example, θ_* can be seen to be $\theta_* = [2, -1]^T$. An algorithm is "better" if it exhibits a "lower" value of e_t. The initial value of θ_0 in all the algorithms was set to 0. The initial value of u_0 in algorithm LMS-2 was set to 0 as well. In the case of RLS, the matrix P_0 (the initial value) was set to $100I$ (I being the identity matrix).[4] In the following, we show plots of convergence of e_t for all algorithms. The curves in the figures from Figures 2 to 6 are averaged over 30 independent runs.

We tested the performance of LMS extensively using the following schedule of learning rates:

$$\alpha_t = \alpha_0(1 + N_0)/(t^p + N_0). \tag{11}$$

Here $\alpha_0, N_0 > 0$ and $p \in (0.5, 1]$ are setting parameters that we varied over a range of values in order to find the best set of these. Towards this end, we first considered the following form for the step-sizes α_t:

$$\alpha_t = \frac{1}{t^p}. \tag{12}$$

We varied the values of p over a range of eleven different values in the set $\{0.51, 0.55, 0.60, \ldots, 0.95, 1.0\}$. We observed that the value of $p = 0.51$ gives the best performance results here. Figure 2 shows the performance of LMS using three different values, $p = 0.51, 0.8$ and 1 respectively.[5] The plot for $p = 0.51$ is the best amongst the three. The performance plots of LMS using the other values of p lie in between those for $p = 0.51$ and $p = 1.0$. In Figure 2, we also show the plot of RLS in order to facilitate a comparison with the plots of LMS.

Since we observed that $p = 0.51$ shows the best performance in the range of step-sizes (12), in our subsequent

[4]We tested I, $10I$ and $100I$, out of which $100I$ was found to give the best performance. However, the performance differences were not large.

[5]We tried iterate averaging on top of the LMS algorithm, but during the transient that we investigate here, iterate averaging (naturally) did not help.

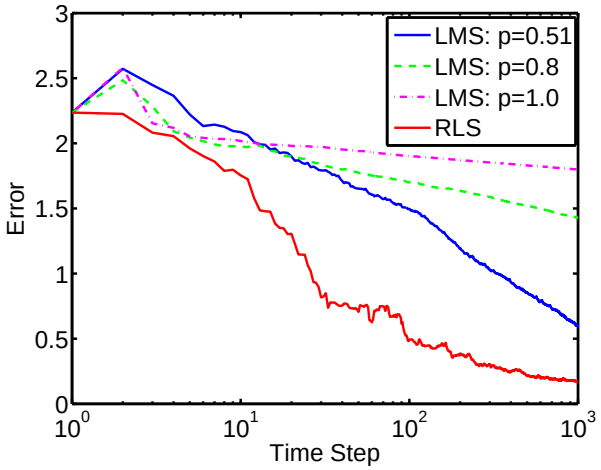

Fig. 2: Performance comparisons of LMS with learning rates (12) for different values of p with RLS.

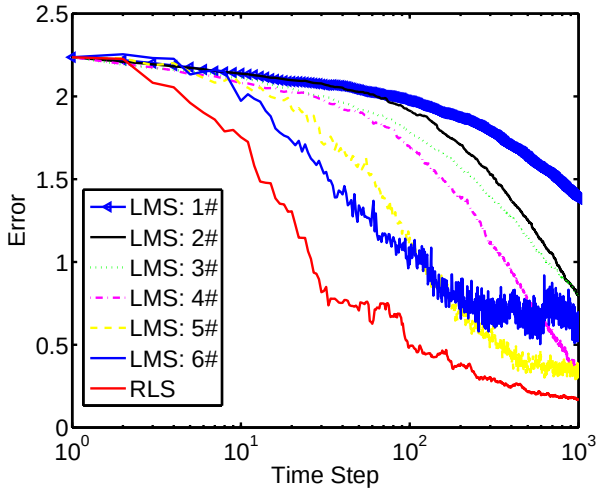

Fig. 3: Performance comparisons of LMS using learning rates (11) for different values of α_0 and N_0 with RLS.

experiments for obtaining the best setting parameters for LMS, we fixed the value of p at 0.51 and considered the following sets of values for the setting parameters α_0 and N_0,

respectively, in order to find the best combination thereof:

$$\begin{aligned}
&1. \quad \alpha_0 = 0.05, \quad N_0 = 10;\\
&2. \quad \alpha_0 = 0.05, \quad N_0 = 100;\\
&3. \quad \alpha_0 = 0.1, \quad N_0 = 10;\\
&4. \quad \alpha_0 = 0.1, \quad N_0 = 100;\\
&5. \quad \alpha_0 = 0.5, \quad N_0 = 10;\\
&6. \quad \alpha_0 = 0.5, \quad N_0 = 100,\\
&7. \quad \alpha_0 = 1, \quad N_0 = 100,\\
&8. \quad \alpha_0 = 1, \quad N_0 = 100. \qquad (13)
\end{aligned}$$

Figure 3 shows the performance of LMS using the first six schedules depicted above. The last two schedules did not show good performance, hence we do not plot in the figure the performance using these. The results show that even though there is a variation in performance of LMS when different setting parameters α_0 and N_0 in the step-sizes (11) are used, however still, there is a "performance gap" between LMS and RLS.

Next, we performed experiments with LMS-2. Note that LMS-2 requires two step-size schedules α_t and β_t. We set these according to:

$$\alpha_t = \alpha_0(1 + N_0)/(t + N_0), \qquad (14)$$

$$\beta_t = \beta_0(1 + M_0)/(t^p + M_0), \qquad (15)$$

respectively, where $0.5 < p < 1$ (in order to obtain a timescale difference between the step-sizes).

It appears that tuning the parameters in LMS-2 is harder than LMS because of the additional step-size schedule. For simplicity, we let $N_0 = M_0$ in the two step-size schedules (14)-(15). Also, we let $\alpha_0 \leq \beta_0$ in general. We first set β_0 and then heuristically set N_0 and α_0 smaller than β_0.

We experimented with three groups of parameters. In each group of experiments, we first fix β_0 and p, then select the value of α_0 and N_0 from the combinations in (13). In particular, we have

(G1) fix $\beta_0 = 0.1$, $p = 0.51$, and use one of the following:
- 1.1: α_0 and N_0 from 1. of (13)
- 1.2: α_0 and N_0 from 2. of (13)
- 1.3: α_0 and N_0 from 3. of (13)
- 1.4: α_0 and N_0 from 4. of (13)

(G2) fix $\beta_0 = 0.5$, $p = 0.51$, and use one of the following:
- 2.1: α_0 and N_0 from 1. of (13)
- 2.2: α_0 and N_0 from 2. of (13)
- 2.3: α_0 and N_0 from 3. of (13)
- 2.4: α_0 and N_0 from 4. of (13)
- 2.5: α_0 and N_0 from 5. of (13)
- 2.6: α_0 and N_0 from 6. of (13)

(G3) fix $\beta_0 = 1$, $p = 0.51$, and use one of the following:
- 3.1: α_0 and N_0 from 1. of (13)
- 3.2: α_0 and N_0 from 2. of (13)
- 3.3: α_0 and N_0 from 3. of (13)
- 3.4: α_0 and N_0 from 4. of (13)
- 3.5: α_0 and N_0 from 5. of (13)
- 3.6: α_0 and N_0 from 6. of (13)
- 3.7: α_0 and N_0 from 7. of (13)
- 3.8: α_0 and N_0 from 8. of (13)

The results of the three groups are shown in Figures 4, 5 and 6 respectively. The schedules showing "better" results are only shown in the figures. The results suggest that 3. of (13) gives the best performance for all the three values of β_0 chosen. Thus, for our next set of experiments, we fix α_0 and N_0 to their values prescribed by 3. of (13), and vary β over a range of values from 0.1 to 1.0. We study the performance in terms of both the mean and standard deviation of the error e_t upon termination of each algorithm. We also vary the values of the parameter α_0 of LMS from 0.1 to 1.2 and study the variation in the values of the error e_t obtained after convergence of the algorithms. We did not choose values of α_0 and β_0 higher than 1.2 and 1 respectively as such values result in a large blow up in the initial stages of the algorithms, thereby having a bearing on the overall algorithm performance. For both LMS and LMS-2, we chose $p = 0.51$ and $N_0 = 10$, which is a nearly optimal setting for both algorithms (as suggested by the experiments). The performance of both algorithms with these settings is compared after 100, 500 and 1000 steps, respectively. We obtained the mean and standard deviations from 500 independent runs in both algorithms in all the three graphs. For the sake of completeness and comparison, we also show the plots of RLS that however is a flat line parallel to x-axis (in each plot) as it does not depend on the quantities α_0 or β_0. These results are summarized in Figure 8 and suggest that the larger β_0 is for the same schedule of α_t, the faster LMS-2 becomes. It can be seen from the figure that LMS-2 shows better performance than LMS and its performance is very close to RLS.

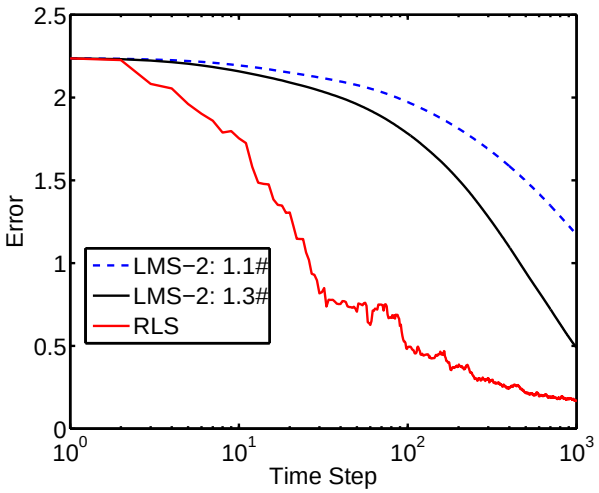

Fig. 4: Performance comparisons of LMS-2 with setting parameters (G1) with RLS

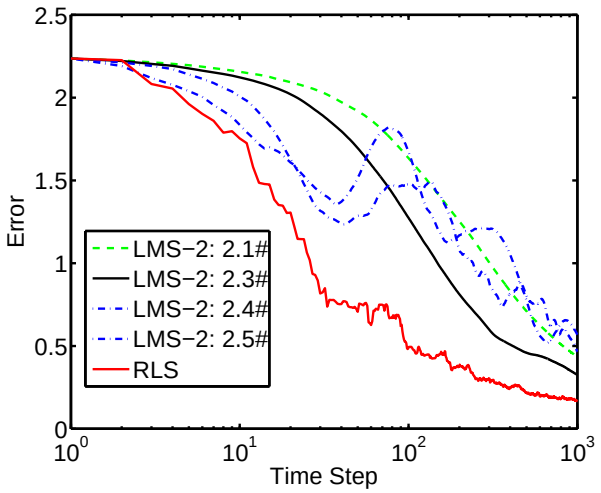

Fig. 5: Performance comparisons of LMS-2 with setting parameters (G2) with RLS

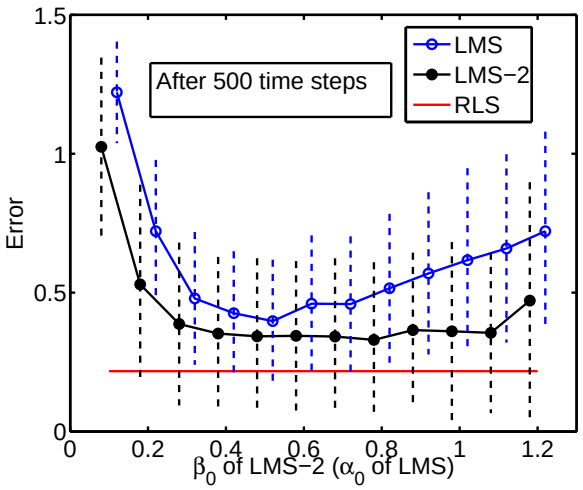

Fig. 7: Performance comparisons of LMS, LMS-2 and RLS after 500 time steps

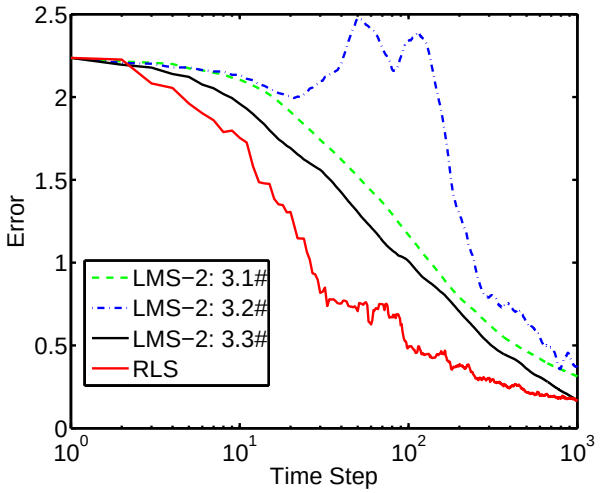

Fig. 6: Performance comparisons of LMS-2 with setting parameters (G3) with RLS

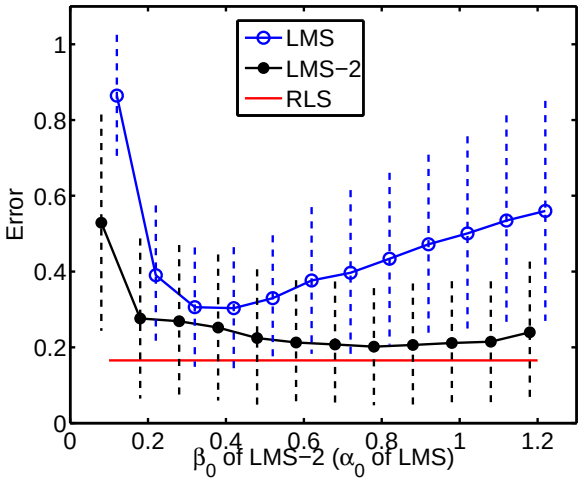

Fig. 8: Performance comparisons of LMS, LMS-2 and RLS after 1000 time steps.

B. A Higher-Dimensional Problem

We consider now the case when the parameter dimension is 10 and $y_t = \theta_*'\phi_t + \varepsilon_t$, where $\theta = [1, 1, \ldots, 1]^T$, $\varepsilon_t \sim N(0, 1)$ and ϕ_t is generated from a 10-dimensional Gaussian: $\phi_t \sim N(0, \Sigma)$. The covariance matrix Σ was selected in a way that its eigenvalues are loosely scattered. In particular, we let $\Sigma = U^T D U$, where U is an orthogonal matrix and D is a diagonal matrix. We first set the ten diagonal elements of D to be $1, 2, \ldots, 9$ and 100, respectively. We then generate a random matrix and use the Gram-Schmidt orthogonalization technique to convert the random matrix to an orthogonal matrix, U. Each curve in the following is averaged over 50 independent simulation runs.

Here we wanted to explore sensitivity of the algorithms to the value of p in the step-size sequences. For reasonable behavior (particularly in the initial stages), we observed that for a large α_0, N_0 must be small and vice versa. The best parameters were found to be $\alpha_0 = 0.01$ and $N_0 = 10$ in the case of LMS while for LMS-2, these were found to be $\alpha_0 = 0.005$, $\beta_0 = 0.005$ and $N_0 = 1000$ (these parameter values made the algorithms perform reasonably well for all

values of p). We then tested the sensitivity of LMS, LMS-2 to the parameter p. In particular, we varied p from 0.51 to 1.0. The plots of mean and standard deviation of e_t after $1,000$ and $10,000$ time steps are shown in Figures 9 and 10 respectively. Again from these plots, it can be seen that the performance of LMS-2 is better than that of LMS even though RLS is the best of the three algorithms.

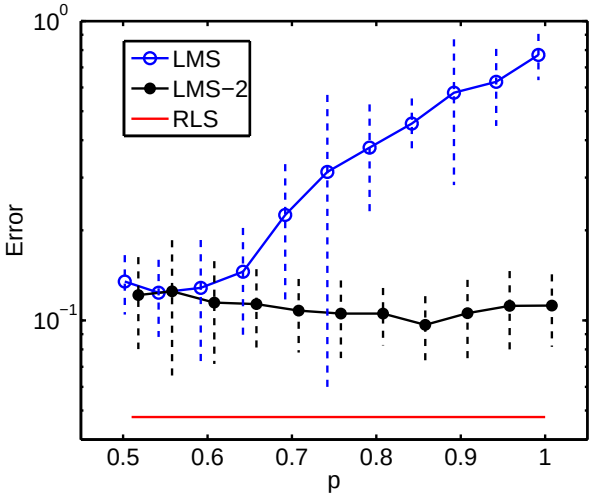

Fig. 9: The effect of p on performance of LMS and LMS-2 after 1000 time steps. LMS-2 is not sensitive to the choice of p parameter in the β_t step-size.

V. CONCLUSIONS

LMS-2 attempts to achieve the performance of RLS, while maintaining the computational complexity of LMS. The price is that now two step-size sequences must be tuned. Our experiments show that the algorithm is promising: In an example with $d = 2$, the performance improvement over LMS was considerable and the performance of LMS-2 got close to that of RLS, while in another example with $d = 10$, we observed that the performance of LMS-2 for almost all settings of the tested learning rate parameters that we considered was better than that of LMS. Our future work will include further empirical studies and the study of adaptive step-size rules. As mentioned in Remark 3, we would replace Assumption 1 with the weaker requirement that the process (ϕ_t, y_t), $t \geq 1$ is Markov. We find the LMS-2 approach promising as we think that the proposed rule has the potential of being competitive with RLS (when appropriately tuned), while the tuning problem only involves scalar quantities i.e., the step-size sequences (now matrix gains are involved). Another direction to explore is the tracking performance of the algorithms, and extensions to other similar learning problems, such as reinforcement learning with linear function approximation [13]. In connection to this, it is important to

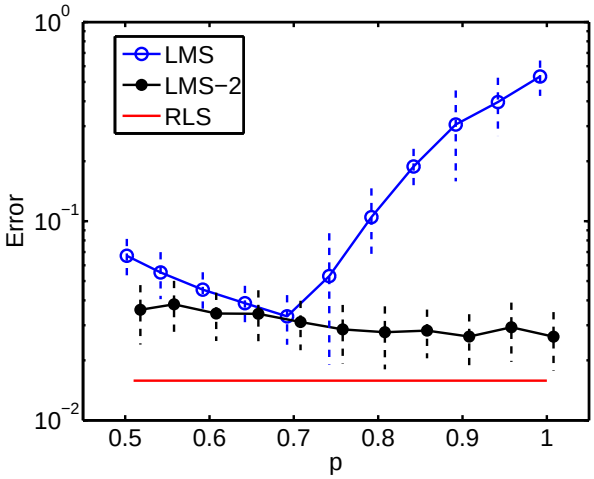

Fig. 10: The effect of p on performance of LMS and LMS-2 after $10,000$ time steps. LMS-2 is not sensitive to the choice of p parameter in the β_t step-size.

note that the idea behind the algorithm (to use an auxiliary parameter to learn the "optimal" update direction on a faster timescale) generalizes to all kinds of domains.

REFERENCES

[1] V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Hindustan Book Agency, New Delhi, 2008.

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

[3] J.M. Cioffi and T. Kailath. Fast, recursive-least-squares transversal filters for adaptive filtering. *IEEE Transactions on Acoustics*, 32(2):304–337, 1984.

[4] E. Eweda. Transient performance degradation of the LMS, RLS, sign, signed regressor, and sign-sign algorithms with data correlation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 46(8):1055–1063, 1999.

[5] S. Haykin. *Adaptive Filter Theory*. Prentice-Hall, Englewood Cliffs, NJ, 2001.

[6] N. Kalouptsidis, G. Carayannis, and D. Manolakis. A fast covariance type algorithm for sequential least-squares filtering and prediction. *IEEE Transactions on Automatic Control*, 29(8):752–755, 1984.

[7] V. R. Konda and J. N. Tsitsiklis. Convergence rate of linear two-timescale stochastic approximation. *The Annals of Applied Probability*, 14(2):796–819, 2004.

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

[9] B.T. Polyak. New method of stochastic approximation type. *Automation and Remote Control*, 51:937–946, 1990.

[10] B.T. Polyak and A.B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30:838–855, 1992.

[11] D. Ruppert. Stochastic approximation. In B.K. Ghosh and P.K. Sen, editors, *Handbook in Sequential Analysis*. Marcel Dekker, New York, 1991.

[12] D. Slock. On the convergence behavior of the LMS and the normalized LMS algorithms and the normalized LMS algorithms. *IEEE Transactions on Signal Processing*, 41:2811–2825, 1993.

[13] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. Bradford Book. MIT Press, 1998.

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