2019 57th Annual Allerton Conference on Communication, Control, and Computing
(Allerton)
Allerton Park and Retreat Center
Monticello, IL, USA, September 24-27, 2019

On Matrix Momentum Stochastic Approximation and Applications to Q-learning

Adithya M. Devraj[1], Ana Bušić[2], and Sean Meyn[1]

***Abstract*— Stochastic approximation (SA) algorithms are recursive techniques used to obtain the roots of functions that can be expressed as expectations of a noisy parameterized family of functions. In this paper two new SA algorithms are introduced: 1) PolSA, an extension of Polyak's momentum technique with a specially designed matrix momentum, and 2) NeSA, which can either be regarded as a variant of Nesterov's acceleration method, or a simplification of PolSA. The rates of convergence of SA algorithms is well understood. Under special conditions, the mean square error of the parameter estimates is bounded by $\sigma^2/n+o(1/n)$, where $\sigma^2 \geq 0$ is an identifiable constant. If these conditions fail, the rate is typically sub-linear. There are two well known SA algorithms that ensure a linear rate, with minimal value of variance, σ^2: the Ruppert-Polyak averaging technique, and the stochastic Newton-Raphson (SNR) algorithm. It is demonstrated here that under mild technical assumptions, the PolSA algorithm also achieves this optimality criteria. This result is established via novel coupling arguments: It is shown that the parameter estimates obtained from the PolSA algorithm couple with those of the optimal variance (but computationally more expensive) SNR algorithm, at a rate $O(1/n^2)$. The newly proposed algorithms are extended to a reinforcement learning setting to obtain new Q-learning algorithms, and numerical results confirm the coupling of PolSA and SNR.**

I. Introduction

The general goal of *stochastic approximation* (SA) is the efficient computation of the root of a vector valued function: obtain the solution $\theta^* \in \mathbb{R}^d$ to the d-dimensional equation:

$$\bar{f}(\theta^*) = 0, \tag{1}$$

where the function $\bar{f}\colon \mathbb{R}^d \to \mathbb{R}^d$ is an expectation: $\bar{f}(\theta) = \mathsf{E}[f(\theta, \mathcal{X})]$, $f\colon \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$, and $\mathcal{X}$ is an $\mathbb{R}^m$-valued random variable. *The function $\bar{f}$ is not necessarily equal to a gradient, so the setting of this paper goes beyond optimization.* Specifically, the algorithms and analysis contained in this work admit application to both stochastic optimization and reinforcement learning (RL), among other areas of machine learning. As in much of the related literature, it is assumed in this paper that there is a sequence of random functions $\{f_n\}$, $f_n : \mathbb{R}^d \to \mathbb{R}^d$, satisfying for each $\theta \in \mathbb{R}^d$,

$$\bar{f}(\theta) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} f_k(\theta) \tag{2}$$

where the limit is in the a.s. sense.

The SA literature contains a large collection of tools to construct algorithms that solve (1), and obtain bounds on their convergence rate. In this paper we show how new algorithms with *optimal rate of convergence* can be constructed based on a synthesis of techniques from classical SA theory combined with variants of momentum algorithms pioneered by Polyak [27], [28].

Three general classes of algorithms are investigated in this work. Each is defined with respect to a non-negative scalar gain sequence $\{\alpha_n\}$, and two include $d \times d$ *matrix gain* sequences $\{G_n\}, \{M_n\}$. For each algorithm, with initialization $\theta_0 = \theta_{-1}$, the *difference sequence* is denoted: $\Delta\theta_n := \theta_n - \theta_{n-1}$, $n \geq 0$.

I. Matrix gain stochastic approximation:

$$\Delta\theta_{n+1} = \alpha_{n+1} G_{n+1} f_{n+1}(\theta_n) \tag{3}$$

II. Matrix momentum stochastic approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1} G_{n+1} f_{n+1}(\theta_n) \tag{4}$$

III. Nesterov stochastic approximation (NeSA): For a fixed scalar $\zeta > 0$,

$$\begin{aligned}\Delta\theta_{n+1} = \Delta\theta_n + \zeta[f_{n+1}(\theta_n) - f_{n+1}(\theta_{n-1})] \\ + \zeta\alpha_{n+1} f_{n+1}(\theta_n)\end{aligned} \tag{5}$$

A common assumption on the step-size sequence $\{\alpha_n\}$ is:

$$\textstyle\sum_{n=1}^{\infty} \alpha_n = \infty\,, \qquad \sum_{n=1}^{\infty} \alpha_n^2 < \infty \tag{6}$$

We take $\alpha_n \equiv 1/n$ throughout.

If $G_n \equiv I$, then (3) is the classical algorithm of Robbins and Monro [31]. In Stochastic Newton Raphson (SNR) and the more recent *Zap-SNR* algorithm [12], [13] (also see [32]), the matrix sequence $\{G_n\}$ is chosen to be an approximation of the Jacobian: $G_n \approx -[\partial\bar{f}(\theta_n)]^{-1}$. Stability of the algorithm has been demonstrated in application to Q-learning [12], [13], but general theory for such an algorithm is still open.

Funding from ARO grant W911NF1810334, National Science Foundation award EPCN 1609131, and French National Research Agency grant ANR-16-CE05-0008 is gratefully acknowledged.

[1]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611
[2]Inria and the Computer Science Department of École Normale Supérieure, 75005 Paris, France

The matrix momentum algorithm (4) coincides with the heavy-ball method of Polyak when $\{M_n\}$ is a sequence of scalars and $G_n \equiv I$ [27], [28], [21]. Justification for the special form (5) in NeSA is provided in the following section.

Problem Setting: As in many previous works in the context of high-dimensional optimization [21] and SA [19], [20], [9], parameter error analysis in this paper is restricted to a linear setting:

$$f_{n+1}(\theta_n) = A_{n+1}\theta_n - b_{n+1} = A(\tilde{\theta}_n) + \Delta_{n+1} \tag{7}$$

in which $\{A_n\}$ is a $d \times d$ matrix valued stochastic process, $\{b_n\}$ is d-dimensional vector valued stochastic processes, with respective means $A := \mathsf{E}[A_n]$, $b := \mathsf{E}[b_n]$, and for $n \geq 1$,

$$\Delta_{n+1} := \tilde{A}_{n+1}\tilde{\theta}_n + \Delta^*_{n+1} \tag{8}$$

where,

$$\Delta^*_{n+1} := f_{n+1}(\theta^*) = A_{n+1}\theta^* - b_{n+1}$$

and the tilde always denotes deviation: $\tilde{\theta}_n := \theta_n - \theta^*$, $\tilde{A}_{n+1} := A_{n+1} - A$.

Rates of Convergence: The main contribution of the paper is convergence analysis of the matrix momentum stochastic approximation algorithm (4), and the NeSA algorithm (5).

Rates of convergence are well understood for the SA recursion (3). It is known that the Central Limit Theorem (CLT) and Law of the Iterated Logarithm (LIL) hold under general conditions, and the *asymptotic covariance* appearing in these results can be expressed as the limit [5], [20], [9]:

$$\Sigma^\theta = \lim_{n\to\infty} \Sigma^\theta_n := \lim_{n\to\infty} n\mathsf{E}[\tilde{\theta}_n \tilde{\theta}_n^{\intercal}]\,. \tag{9}$$

The CLT implies: $\sqrt{n}\tilde{\theta}_n \xrightarrow{\text{dist.}} \mathcal{N}(0, \Sigma^\theta)$.

A *necessary* condition for quick convergence is that the CLT or LIL hold with small asymptotic covariance (9). Again, for the SA recursion (3), optimization of this quantity is well-understood [5], [20], [9]; Denote by Σ^G the asymptotic covariance for (3) with a fixed matrix gain $G_n \equiv G$. When Σ^G is finite, it is given by the solution to the following Lyapunov equation [5], [20], [9]:

$$\Big(\tfrac{1}{2}I + GA\Big)\Sigma^G + \Sigma^G\Big(\tfrac{1}{2}I + GA\Big)^{\intercal} + \Sigma^\Delta = 0 \tag{10}$$

where Σ^Δ is the asymptotic covariance of the noise sequence $\{\Delta^*_n\}$:

$$\Sigma^\Delta = \lim_{n\to\infty} \frac{1}{n}\mathsf{E}\Big[\Big(\sum_{k=1}^n \Delta^*_k\Big)\Big(\sum_{k=1}^n \Delta^*_k\Big)^{\intercal}\Big] \tag{11}$$

The choice $G = G^* := -A^{-1}$ results in the SNR algorithm, for which asymptotic covariance admits an explicit form:

$$\Sigma^* := A^{-1}\Sigma^\Delta (A^{-1})^{\intercal} \tag{12}$$

This is optimal: the difference $\Sigma^G - \Sigma^*$ is positive semi-definite for any G [5], [20], [9].

However, in general, the asymptotic covariance of a recursion of the form (3) *need not* be finite. A sufficient condition for finite asymptotic variance is that the real parts of all the eigenvalues of the matrix GA are *strictly* smaller than $-\frac{1}{2}$; The condition is necessary and sufficient if Σ^Δ is positive definite [13, Proposition A.1].

Infinite asymptotic variance implies a rate of convergence that is *slower than* $\mathcal{O}(1/\sqrt{n})$.

What about computational complexity? In realistic applications of SNR, the matrix gain sequence appearing in (3) will be of the form $G_n = -\widehat{A}_n^{-1}$, where $\{\widehat{A}_n\}$ are Monte-Carlo estimates of the mean A. The resulting computational complexity of inverting such a matrix at each iteration (which could be as bad as $O(d^3)$) is a barrier to application in higher dimension. Steps towards resolving this obstacle are presented in this paper:

(i) The hyper-parameters appearing in the *matrix momentum* SA algorithm (4) can be designed so that the error sequence enjoys all the attractive properties of SNR, but *without* the need for matrix inversion; Since the recursion in (4) only involves matrix-vector products, this would result in an overall per iteration complexity of $O(d^2)$.

(ii) NeSA defined in (5) is often simpler than the matrix momentum method in applications to RL. A formula for the asymptotic covariance of a variant of NeSA is obtained in this paper; While not equal to Σ^*, the reduced $O(d)$ complexity makes it a valuable option.

These conclusions are established in Propositions 3.2, 4.1, and 4.2 for a linear setting of the form (7), and illustrated in numerical examples for new Q-learning algorithms that are discussed in Section V. The assumptions of the main results are violated in application to Q-learning since the particular root finding problem is non-linear. Nevertheless, coupling is seen between PolSA and Zap SNR versions of Q-learning in all of the numerical experiments conducted.

Unifying algorithms: An additional contribution of the paper is establishing strong theoretical connections between existing popular algorithms. Nesterov's acceleration and the heavy-ball method both are known to have second order dynamics, but their relationship has not been very clear. In this paper we propose a new understanding of the relationship, which is only possible through introducing the concept of *matrix momentum*.

We show that the matrix momentum algorithm PolSA can be interpreted as a linearization of a particular formulation of Nesterov's method. We further show that PolSA approximates (stochastic) Newton Raphson, thus establishing connections between the three algorithms: Nesterov's accleration, PolSA, and SNR. This not only helps in explaining the success of Nesterov's acceleration, but may also lead to new algorithms in other application domains.

Literature survey: The present paper is built on a vast literature on optimization [25], [27], [28], [26] and stochastic approximation [19], [20], [9], [32], [29], [30]. The work of Polyak is central to both thrusts: the introduction of momentum, and techniques to minimize variance in SA algorithms. The reader is referred to [13] for a survey on SNR and the more recent Zap SNR algorithms, which are also designed to achieve minimum asymptotic variance.

In the stochastic optimization literature, the goal is to minimize an expectation of a function. In connection to (2), each f_n can be viewed as an unbiased estimator of the gradient of the objective. The works [24], [15], [17] obtain conditions for optimal convergence rate of $\mathcal{O}(1/\sqrt{n})$ for various algorithms.

In ERM literature, the sample path limit in (2) is replaced by a finite average [2], [11], [17]: $\bar{f}_n(\theta) = n^{-1}\sum_{k=1}^{n} f_k(\theta)$. Denoting $\theta_n^* = \arg\min_\theta \bar{f}_n(\theta)$, under general conditions it can be shown that the sequence of ERM optimizers $\{\theta_n^*\}$ is convergent to θ^*, and has optimal asymptotic covariance (a survey and further discussion is presented in [17]).

The recent paper [17] is most closely related to the present work, considering the shared goal of optimizing the asymptotic covariance, along with rapidly vanishing transients through algorithm design. The paper restricts to "least squares regression" rather than the general root finding problems considered here, thus ruling out application to many RL algorithms such as TD- and Q-learning [36], [37], [19].

The algorithms presented in this work achieve the optimal asymptotic covariance, are not restricted to optimization, and we believe that in many applications they will be simpler to implement.

II. Motivation & Insights

Consider first the *deterministic* root-finding problem. The notation $f : \mathbb{R}^d \to \mathbb{R}^d$ is used in place of $\bar{f}$ in this deterministic setting, and the goal remains the same: find the vector $\theta^* \in \mathbb{R}^d$ such that $f(\theta^*) = 0$. The objective in this section is to bring insight into the relationship between the three algorithms (3–5) discussed in the introduction.

Deterministic variants of (3–5) commonly considered in the literature are, respectively,

Successive approximation:

$$\Delta\theta_{n+1} = \alpha f(\theta_n) \tag{13}$$

Polyak's heavy ball:

$$\Delta\theta_{n+1} = \mu\Delta\theta_n + \alpha f(\theta_n) \tag{14}$$

Nesterov's acceleration:

$$\Delta\theta_{n+1} = \mu\Delta\theta_n + \zeta[f(\theta_n) - f(\theta_{n-1})] + \alpha f(\theta_n) \tag{15}$$

where α, μ, ζ are positive scalars. Nesterov's algorithm was designed for extremal seeking, which is the special case $f = -\nabla J$ for a real-valued function $J : \mathbb{R}^d \to \mathbb{R}$. The recursion (15) is the natural extension to the root-finding problem considered here.

The questions asked in this paper are posed in a stochastic setting, but analogous questions are:

(a) Why restrict to a scalar momentum term μ, rather than a matrix M?

(b) Can online algorithms be designed to approximate the optimal matrix momentum? If so, we require tools to investigate the performance of a given matrix sequence $\{M_n\}$:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha f(\theta_n) \tag{16}$$

Potential answers are obtained by establishing relationships between (13–15). The heuristic relationships presented here are justified for the stochastic models considered later in the paper.

Consider the successive approximation algorithm (13) under the assumption of global convergence: $\theta_n \to \theta^*$ as $n \to \infty$. Assume moreover that $f \in C^1$ and Lipschitz, so that,

$$\begin{aligned}\Delta\theta_{n+1} - \Delta\theta_n &= \alpha[f(\theta_n) - f(\theta_{n-1})] \\ &\overset{(a)}{\approx} \alpha\partial f(\theta_n)\Delta\theta_n \\ &\overset{(b)}{=} \alpha^2\partial f(\theta_n) f(\theta_{n-1})\end{aligned} \tag{17}$$

where (a) follows from the assumption that f is Lipschitz and C^1, and (b) follows from (13). It follows that $\|\Delta\theta_{n+1} - \Delta\theta_n\| = \mathcal{O}(\min\{\alpha^2, \alpha\|\Delta\theta_n\|\})$. This suggests a *heuristic*: swap $\Delta\theta_{n+1}$ and $\Delta\theta_n$ in a given "convergent" algorithm to obtain a new algorithm that is hopefully simpler, and has desirable properties. Applying this heuristic to (16) (replacing $\Delta\theta_n$ with $\Delta\theta_{n+1}$ on the right hand side), we obtain

$$\Delta\theta_{n+1} \approx M_{n+1}\Delta\theta_{n+1} + \alpha f(\theta_n)$$

Assuming that an inverse exists, this becomes

$$\Delta\theta_{n+1} \approx \alpha[I - M_{n+1}]^{-1} f(\theta_n) \tag{18}$$

We thus arrive at a possible answer to the question of *optimal matrix momentum* in (16): For the matrix sequence $M_{n+1} = I + \alpha\partial f(\theta_n)$, the algorithm (16) can be expressed

$$\Delta\theta_{n+1} = [I + \alpha\partial f(\theta_n)]\Delta\theta_n + \alpha f(\theta_n) \tag{19}$$

The foregoing approximation in (18) suggest that this is an approximation of Newton-Raphson (which is (13), with the scalar α replaced by the Jacobian $-[\partial f(\theta_n)]^{-1}$): $\Delta\theta_{n+1} \approx -[\partial f(\theta_n)]^{-1} f(\theta_n)$.

Further approximations lead to different interpretations: A Taylor series argument shows that the recursion (19) is approximated by

$$\Delta\theta_{n+1} = \Delta\theta_n + \alpha[f(\theta_n) - f(\theta_{n-1})] + \alpha f(\theta_n) \quad (20)$$

This is the special case of Nesterov's algorithm (15) with $\mu = 1$ and $\zeta = \alpha$.

A complete justification for the stochastic analog of (19) is provided through a coupling bound in Proposition 3.2. It is found that similar transformations lead to new algorithms for RL and other applications.

III. Optimal Matrix Momentum and PolSA

Returning to the stochastic setting, the PolSA algorithm considered in this paper is a special case of matrix momentum SA (4), and an analog of (19):

PolSA Algorithm:

$$\Delta\theta_{n+1} = [I + \zeta\widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n) \quad (21)$$

where $\zeta > 0$, and $\{\widehat{A}_n\}$ are estimates of $A(\theta_n)$, with $A(\theta) := \mathsf{E}[\partial f_n(\theta)]$ (assumed independent of n).

The choice $G_n \equiv \zeta I$ in (4) is imposed to simplify exposition; in numerical examples it is observed that a particular diagonal matrix gives much better performance in applications to Q-learning (details are contained in Section V).

The main technical results are obtained for a linear model defined in (7) - (8). In this case we have $A(\theta) \equiv A$, and its estimates are obtained recursively:

$$\widehat{A}_{n+1} = \widehat{A}_n + \frac{1}{n+1}(A_{n+1} - \widehat{A}_n) \quad (22)$$

The SNR algorithm is (3) in which $G_n = \widehat{A}_n^\dagger$ (the Moore–Penrose pseudo inverse):

$$\textbf{SNR:} \qquad \Delta\theta_{n+1} = -\alpha_{n+1}\widehat{A}_{n+1}^\dagger f_{n+1}(\theta_n) \quad (23)$$

Additional simplifying assumptions are imposed to ease analysis:

(A1) The stochastic process (A_n, b_n) is wide-sense stationary, with common mean (A, b).

(A2) $\{\tilde{A}_n, \tilde{b}_n\}$ are bounded martingale difference sequences, adapted to the filtration $\mathcal{F}_n := \sigma\{A_k, b_k : k \le n\}$

(A3) For any eigenvalue λ of A,

$$\text{Real}(\lambda) < 0 \quad \text{and} \quad |1 + \zeta\lambda| < 1 \quad (24)$$

It is assumed without loss of generality that $\zeta = 1$. Under Assumptions **A1** and **A2**, the covariance matrix in (11) can be expressed

$$\Sigma^\Delta = \mathsf{E}[\Delta^*_{n+1}(\Delta^*_{n+1})^T] \quad (25)$$

Even in the linear setting, full stability and coupling arguments are not yet available because the assumptions do not ensure that $\widehat{A}_n^{-1} \to A^{-1}$ in L_2. The main theoretical result of this section is therefore restricted to establishing the relationship between the following *idealized* versions of the SNR and PolSA algorithms, wherein we replace the estimates $\widehat{A}_n$ with the exact value A:

$$\textbf{SNR}^\star: \ \Delta\theta^*_{n+1} = -\alpha_{n+1}A^{-1}f_{n+1}(\theta^*_n) \quad (26)$$

$$\textbf{PolSA}^\star: \ \Delta\theta_{n+1} = [I + A]\Delta\theta_n + \alpha_{n+1}f_{n+1}(\theta_n) \quad (27)$$

Proposition 3.1 establishes optimality of the asymptotic variance of the SNR algorithm (23); the proof is contained in Appendix A of [1].

Proposition 3.1: Suppose assumptions **(A1)–(A3)** hold. Then, the following holds for the estimates $\{\theta^\bullet_n\}$ obtained using SNR algorithm (23) and $\{\theta^*_n\}$ obtained using ideal SNR algorithm (26):

(i) The following representations hold for the error sequences:

$$\tilde{\theta}^\bullet_n = -\widehat{A}_n^{-1}\frac{1}{n}\sum_{k=1}^n \Delta^*_k \quad \text{(whenever } \widehat{A}_n^{-1} \text{ exists)} \quad (28)$$

$$\tilde{\theta}^*_n = -A^{-1}\frac{1}{n}\sum_{k=1}^n \Delta_k \quad (29)$$

Consequently, each converges to zero with probability one.

(ii) The scaled covariances $\Sigma_n := n\mathsf{E}[\tilde{\theta}^*_n(\tilde{\theta}^*_n)^T]$ and $\Sigma^{22}_n := n^2\mathsf{E}[\Delta\theta^*_n(\Delta\theta^*_n)^T]$ satisfy

$$\lim_{n\to\infty}\Sigma_n = \lim_{n\to\infty}\Sigma^{22}_n = \Sigma^* \quad (30)$$

with Σ^* the optimal covariance defined in (12). □

A drawback with SNR is the matrix inversion. The PolSA algorithm is simpler and enjoys the same attractive properties. This is established through the following coupling result. The proof of Proposition 3.2, contained in Appendix B of [1], is a rigorous justification of the heuristic used to construct the deterministic recursion (19).

Proposition 3.2: Suppose assumptions **(A1)–(A3)** hold. Let $\{\theta^*_n\}$ denote the iterates obtained using (26) and $\{\theta_n\}$ the iterates obtained using (27), with identical initial conditions. Then,

$$\sup_{n\ge 0} n^2\mathsf{E}[\|\theta_n - \theta^*_n\|^2] < \infty \quad (31)$$

Consequently, the limits (30) hold for the PolSA algorithm (27):

$$\lim_{n\to\infty} n\mathsf{E}[\tilde{\theta}_n(\tilde{\theta}_n)^T] = \lim_{n\to\infty} n^2\mathsf{E}[\Delta\theta_n(\Delta\theta_n)^T] = \Sigma^*$$

□

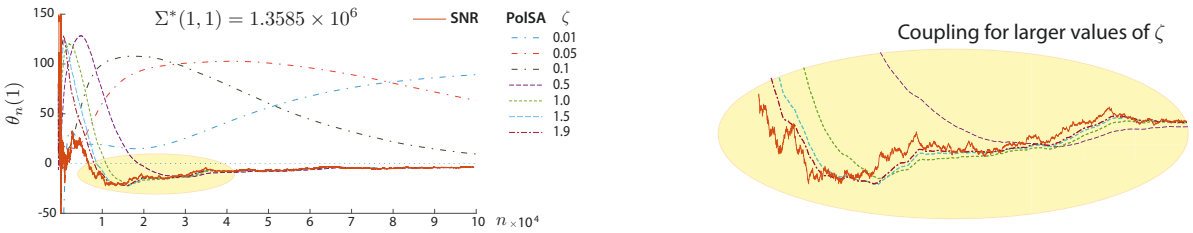

Fig. 1. Coupling between PolSA and SNR occurs quickly for $0.5 \le \zeta \le 1.9$.

Other than the classical algorithms SNR and Polyak-Ruppert averaging technique, PolSA is the only other known algorithm that achieves *optimal asymptotic variance*. The fact that it is a momentum based technique, and that it does not fit into the class of standard SA algorithms makes the result quite special.

An illustration of the coupling is provided in Fig. 1 for the linear model $f_n(\theta) = A\theta + \Delta_n$ in which $-A$ is symmetric and positive definite with $\lambda_{\max}(-A) = 1$, and $\{\Delta_n\}$ is i.i.d. and Gaussian. Shown are the trajectories of $\{\theta_n(1) : n \le 10^5\}$ (note that $\Sigma^*(1,1)$ is over one million).

Optimality of PolSA: It is conjectured that Proposition 3.2 on coupling of parameter estimates obtained from (26) and (27) can be extended to show that the coupling result will hold even when we replace A with any other $d \times d$ matrix M in both these recursions, as long as each eigenvalue λ of M satisfies (24).

The extended coupling result would then imply that the parameter estimates obtained using a matrix momentum algorithm of the form (27), with momentum M, will have asymptotic variance Σ^G that is obtained as a solution to the Lyapunov equation (10), with $G = -M^{-1}$, assuming invertibility.

Recall that Σ^* defined in (12) is the optimal solution to this Lyapunov equation, and is achieved when $G = -A^{-1}$. Assuming that the conjectures hold, the arugments above conclude that the matrix momentum $[I + A]$ in (27) is optimal: No other matrix momentum can achieve lower asymptotic variance.

Applications:

• **Reinforcement learning:** Section V describes application to Q-learning, and includes numerical examples. Appendix D of [1] contains a full account of TD-learning.

• **Stochastic optimization:** A common application of SA is convex optimization. In this setting, $\bar{f}(\theta) = \nabla \mathsf{E}[J_n(\theta)]$ for a sequence of smooth functions $\{J_n\}$, and then $f_n = -\nabla J_n$. The theory developed in this paper is directly applicable to this class of problems, except in degenerate cases. For comparison, consider the quadratic optimization problem in which $f_n(\theta) = A\theta - b + \Delta_n$, with $-A > 0$. The stability condition (24) holds provided we choose $\zeta < 1/\lambda_{\max}(-A)$: a condition familiar in the optimization literature.

IV. Variance Analysis of NeSA

The NeSA algorithm (5) has a finite asymptotic covariance that can be expressed as the solution to a Lyapunov equation. We again restrict to the linear model, so that the recursion (5) (with $\zeta = 1$, without loss of generality) becomes

$$\Delta\theta_{n+1} = [I + A_{n+1}]\Delta\theta_n + \alpha_{n+1}[A_{n+1}\theta_n - b_{n+1}] \quad (32)$$

Stability of the recursion requires a strengthening of assumption (24). Define the linear operator $\mathcal{L} : \mathbb{R}^{d\times d} \to \mathbb{R}^{d\times d}$ as follows: For any matrix $Q \in \mathbb{R}^{d\times d}$,

$$\mathcal{L}(Q) := \mathsf{E}[(I + A_n)Q(I + A_n)^{\intercal}] \quad (33)$$

Define the $2d$-dimensional vector processes $\Phi_n := (\sqrt{n}\tilde{\theta}_n, n\Delta\theta_n)^{\intercal}$, and

$$\Sigma_n := \mathsf{E}[\Phi_n \Phi_n^{\intercal}] = \begin{bmatrix} \Sigma_n^{11} & \Sigma_n^{12} \\ \Sigma_n^{21} & \Sigma_n^{22} \end{bmatrix} \quad (34)$$

The following assumptions are imposed throughout this section:

(N1) $\{\tilde{A}_n, \tilde{b}_n\}$ are bounded martingale difference sequences, adapted to the filtration $\mathcal{F}_n := \sigma\{A_k, b_k : k \le n\}$. Moreover, for any matrix Q,

$$\mathsf{E}[(I + A_n)Q(I + A_n)^{\intercal} \mid \mathcal{F}_{n-1}] = \mathcal{L}(Q)$$

(N2) The bounds in (24) hold for all eigenvalues λ of A, and the spectral radius of $\mathcal{L}$ is strictly bounded by unity.

(N3) The covariance sequence $\{\Sigma_n\}$ defined in (34) is bounded.

In of [1, Appendix C] we discuss how **(N3)** can be relaxed.

Proposition 4.1: Suppose that **(N1)** and **(N2)** hold. Then,

$$\lim_{n\to\infty} \Sigma_n = \begin{bmatrix} \Sigma_\infty^{11} & 0 \\ 0 & \Sigma_\infty^{22} \end{bmatrix} \quad (35)$$

in which the second limit is the solution to the Lyapunov equation

$$\Sigma_\infty^{22} = \mathcal{L}(\Sigma_\infty^{22}) + \Sigma^{\Delta} \quad (36)$$

(see [1, Appendix C.3] for an explicit expression), and

$$\Sigma^{11}_\infty = -\Sigma^{22}_\infty - A^{-1}\Sigma^{22}_\infty - \Sigma^{22}_\infty A^{-1} \tag{37}$$

The following result is a corollary to Proposition 4.1, with an independent proof provided in [1, Appendix C].

Proposition 4.2: Under **(N1)–(N3)** the limit (35) of Proposition 4.1 holds for the PolSA recursion (27). In this case the solution to the Lyapunov equation is the optimal covariance:

$$\Sigma^{11}_\infty = \Sigma^* := A^{-1}\Sigma^\Delta A^{-1^T} \tag{38}$$

and $\Sigma^{22}_\infty \geq 0$ is the unique solution to the Lyapunov equation

$$\Sigma^{22}_\infty = (I+A)\Sigma^{22}_\infty (I+A)^T + \Sigma^\Delta \tag{39}$$

The main step in the proof of Proposition 4.1 involves a finer look at the off-diagonal blocks of the covariance matrix Σ_n. The proofs of the following are contained in Appendix C of [1].

Lemma 4.3: The following approximations hold, with $\psi_n := \sqrt{n}\Sigma^{21}_n$: For $n \geq 1$,

$$\begin{aligned} \Sigma^{22}_{n+1} &= \mathcal{L}(\Sigma^{22}_n) + \Sigma^\Delta + o(1) \\ \psi_n &= -\Sigma^{11}_n - A^{-1}\Sigma^{22}_\infty + o(1) \end{aligned} \tag{40}$$

The second iteration is used together with the following result to obtain (37).

Lemma 4.4: The following approximation holds:

$$\begin{aligned} \Sigma^{11}_{n+1} = \Sigma^{11}_n + \alpha_{n+1}\Big(&\Sigma^{11}_n + A\Sigma^{11}_n + \Sigma^{11}_n A^T + \Sigma^\Delta \\ &+ \psi^T_n (I+A)^T + (I+A)\psi_n + \mathcal{L}(\Sigma^{22}_\infty) + o(1)\Big) \end{aligned} \tag{41}$$

Proof of Proposition 4.1:
The first approximation in (40) combined with (N2) implies that the sequence $\{\Sigma^{22}_n\}$ is convergent, and the limit is the solution to the fixed point equation (36) (details are provided in Appendix C.2 of [1]).

Substituting the approximation (40) for ψ_n into (41) and simplifying gives

$$\begin{aligned} \Sigma^{11}_{n+1} = \Sigma^{11}_n + \alpha_{n+1}\Big(&- \Sigma^{11}_n \\ &- \Sigma^{22}_\infty - A^{-1}\Sigma^{22}_\infty - \Sigma^{22}_\infty A^{-1} + o(1)\Big) \end{aligned}$$

This can be regarded as a Euler approximation to the ODE:

$$\frac{d}{dt}x_t = -x_t - \Sigma^{22}_\infty - A^{-1}\Sigma^{22}_\infty - \Sigma^{22}_\infty A^{-1}$$

SA theory can then be applied to establish that the limits of $\{\Sigma^{11}_n\}$ and $\{x_t\}$ coincide with the stationary point, which is (37) [9]. □

V. Application to Q-learning

Consider a discounted cost MDP model with state space X, action space U, cost function $c\colon \mathsf{X}\times\mathsf{U}\to\mathbb{R}$, and discount factor $\beta \in (0,1)$. It is assumed that the state and action space are finite: denote $\ell = |\mathsf{X}|$, $\ell_u = |\mathsf{U}|$, and P_u the $\ell \times \ell$ controlled transition probability matrix.

The Q-function is the solution to the Bellman equation:

$$Q^*(x,u) = c(x,u) + \beta \mathsf{E}[\underline{Q}^*(X_{n+1}) \,|\, X_n = x, U_n = u] \tag{42}$$

where, for any function $Q : \mathsf{X}\times\mathsf{U}\to\mathbb{R}$ $\underline{Q}(x) := \min_u Q(x,u)$. The goal of Q-learning is to learn an approximation to Q^*. Given d basis functions $\{\phi_i : 1 \leq i \leq d\}$, with each $\phi_i : \mathsf{X}\times\mathsf{U}\to\mathbb{R}$, and a parameter vector $\theta \in \mathbb{R}^d$, the Q-function estimate is denoted $Q^\theta(x,u) = \theta^T\phi(x,u)$.

Watkins' Q-learning algorithm is designed to compute the exact Q-function that solves the Bellman equation (42) ([38], [39]). In this setting, the basis is taken to be the set of indicator functions: $\phi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$, $1 \leq i \leq d$, with $d = |\mathsf{X}\times\mathsf{U}|$. The goal is to find $\theta^* \in \mathbb{R}^d$ such that $\bar{f}(\theta^*) = 0$, where, for any $\theta \in \mathbb{R}^d$,

$$\begin{aligned} \bar{f}(\theta) = \mathsf{E}\big[\phi(X_n,U_n)\big(c(X_n,U_n) + \beta\underline{Q}^\theta(X_{n+1}) \\ - Q^\theta(X_n,U_n)\big)\big] \end{aligned}$$

and the expectation is with respect to the steady state distribution of the Markov chain.

The basic algorithm of Watkins can be written as [13]

$$\Delta\theta_{n+1} = \alpha_{n+1}\widehat{D}_{n+1}\big[A_{n+1}\theta_n - b_{n+1}\big] \tag{43}$$

in which the matrix gain $\widehat{D}_{n+1}$ is diagonal, with

$$\widehat{D}_n(i,i)^{-1} = \frac{1}{n}\sum_{k=0}^{n-1}\mathbb{I}\{(X_k,U_k) = (x^i,u^i)\},$$

and with $\pi_n(x) := \arg\min_u Q^{\theta_n}(x,u)$,

$$A_{n+1} = \phi(X_n,U_n)\big[\beta\phi(X_{n+1},\pi_n(X_{n+1})) - \phi(X_n,U_n)\big]^T$$

$$b_{n+1} = c(X_n,U_n)\phi(X_n,U_n)$$

Among the other algorithms compared are

$$\begin{aligned} &\textbf{SNR:} && \Delta\theta_{n+1} = -\alpha_{n+1}\widehat{A}^{-1}_{n+1}\big[A_{n+1}\theta_n - b_{n+1}\big] \\ &\textbf{PolSA:} && \Delta\theta_{n+1} = [I + \widehat{A}_{n+1}]\Delta\theta_n \\ &&& \qquad + \alpha_{n+1}\big[A_{n+1}\theta_n - b_{n+1}\big] \\ &\textbf{PolSA-D:} && \Delta\theta_{n+1} = [I + \widehat{D}_{n+1}\widehat{A}_{n+1}]\Delta\theta_n \\ &&& \qquad + \alpha_{n+1}\widehat{D}_{n+1}\big[A_{n+1}\theta_n - b_{n+1}\big] \\ &\textbf{NeSA:} && \Delta\theta_{n+1} = [I + A_{n+1}]\Delta\theta_n \\ &&& \qquad + \alpha_{n+1}\big[A_{n+1}\theta_n - b_{n+1}\big] \end{aligned}$$

In each of these algorithms, (22) is used to recursively estimate $\widehat{A}_n$ with A_{n+1} defined above. We have taken

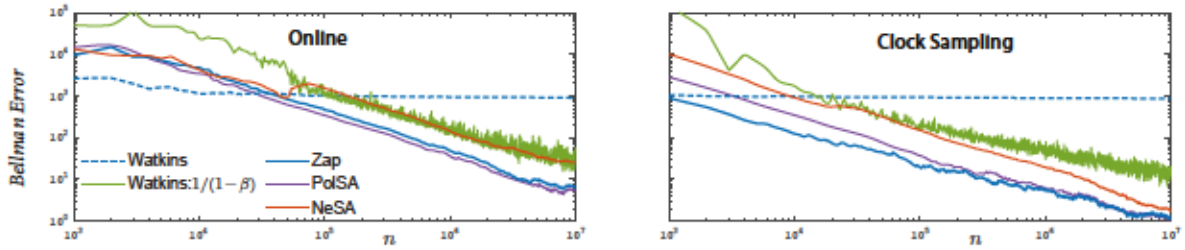

Fig. 2. Bellman error resulting from application of different Q-learning algorithms to a shortest path problem with number of state-action pairs $d = 117$.

$\zeta = 1$ in PolSA. The variant PolSA-D is (4) with $G_{n+1} = \widehat{D}_{n+1}$, and M_{n+1} chosen so that coupling with SNR can be expected.

The SNR algorithm considered coincides with the *Zap Q-learning* algorithm of [12], [13]. A simple 6-state MDP model was considered in this prior work, with the objective of finding the stochastic shortest path. Fig. 3 contains histograms of $\{\sqrt{n}\tilde{\theta}_n\}$ obtained from 1000 parallel simulations of PolSA-D, SNR and NeSA algorithms for this problem. It is observed that the histograms of PolSA-D and SNR nearly coincide after $n = 10^6$ iterations (performance for PolSA is similar). The histogram for NeSA shows a much higher variance, but the algorithm requires by-far the least computation per iteration. This is specifically true for Watkins' Q-learning since A_{n+1} is a sparse matrix, with just 2 non-zero entries.

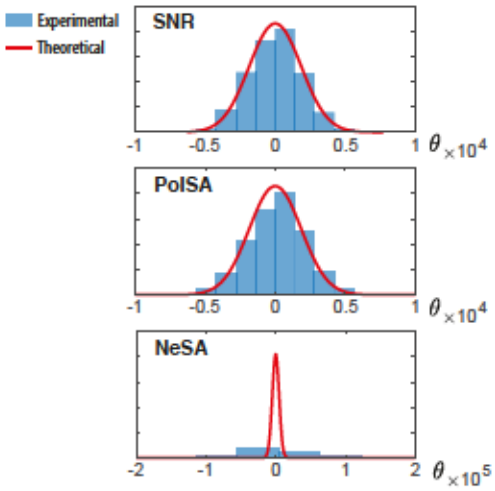

Fig. 3. Histograms for entry 18 of $\{\sqrt{n}\tilde{\theta}_n\}$ for three algorithms at iteration 10^6.

Experiments were also performed for larger examples. Results from two such experiments are shown in Fig. 2. The MDP model is once again a stochastic shortest path problem. The model construction was based on the creation of a graph with N nodes, in which the probability of an edge between a pair of nodes is i.i.d. with probability p. Additional edges $(i, i+1)$ are added, for each $i < N$, to ensure the resulting graph is strongly connected.

The transition law is similar to that used in the finite state-action example of [12]: with probability 0.8 the agent moves in the desired direction, and with remaining probability it ends up in one of the neighboring nodes, chosen uniformly. Two exploration rules were considered: the "online" version wherein at each iteration the agent randomly selects a feasible action (also known as asynchronous Q-learning), and the offline "clock sampling" approach in which state-action pairs (x^t, u^t) are chosen sequentially (also known as synchronous Q-learning). In the latter, at stage n, if (x, u) is the current state-action pair, a random variable X'_{n+1} is chosen according to the distribution $P_u(x, \cdot)$, and the (x, u) entry of the Q-function is updated according to the particular algorithm using the triple (x, u, X'_{n+1}). A significant change to Watkins' iteration (43) in the synchronous setting is that $\widehat{D}_n$ is replaced by $d^{-1}I$ (since each state is visited the same number of times after each cycle). This combined with deterministic sampling is observed to result in significant variance reduction. The *synchronous speedy Q-learning* recursion of [3] appears similar to the NeSA algorithm with clock sampling.

Using the above described method, a random graph was generated resulting in an MDP with $d = 117$ state-action pairs. The plots in Fig. 2 show Bellman error as a function of iteration n (for definitions see [7], [13]). Comparison of the performance of algorithms in a deterministic exploration setting versus the online setting is also shown. The coupling of PolSA and the Zap algorithms are easily observed in the clock sampling case.

VI. Conclusions

It is exciting to see how the intuitive transformation from SNR to PolSA and NeSA can be justified theoretically and in simulations. In Fig. 2, it is particularly exciting to see the coupling of PolSA and Zap (SNR) algorithms, even in the non-linear setting of Q-learning. While the covariance of NeSA is not optimal, it is the simplest of the three algorithms and

is observed to perform well in applications.

An important next step is to create adaptive techniques to ensure fast coupling or other ways to ensure fast forgetting of the initial condition. It is possible that techniques in [17] may be adapted to achieve this. The work can be extended in several ways:

(i) It will be of great interest to pursue analysis of the proposed algorithms in the special case of nonlinear optimization. It is possible that the structure of the problem such as convexity of the objective and smoothness of the gradients could help us derive bounds on the transients.

(ii) In [12] we suggest that the SNR algorithm can be extended to obtain theory for the convergence of Q-learning with function approximation. It would be interesting to see how the PolSA and NeSA algorithms can be extended to this setting. Applications to TD-learning with function approximation is discussed in [1, Appendix B].

REFERENCES

[1] A. M. Devraj, A. Bušić, and S. Meyn, "Optimal matrix momentum stochastic approximation and applications to Q-learning," *arXiv e-prints*, p. arXiv:1809.06277, Sep 2018.

[2] Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *ArXiv e-prints*, Mar. 2016.

[3] M. G. Azar, R. Munos, M. Ghavamzadeh, and H. Kappen. Speedy Q-learning. In *Advances in Neural Information Processing Systems*, 2011.

[4] F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $o(1/n)$. In *Advances in Neural Information Processing Systems 26*, pages 773–781. Curran Associates, Inc., 2013.

[5] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990. Translated from the French by Stephen S. Wilson.

[6] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*. Springer, 2012.

[7] D. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Atena Scientific, Cambridge, Mass, 1996.

[8] V. S. Borkar. Average cost dynamic programming equations for controlled Markov chains with partial observations. *SIAM J. Control Optim.*, 39(3):673–681 (electronic), 2000.

[9] V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Hindustan Book Agency and Cambridge University Press (jointly), Delhi, India and Cambridge, UK, 2008.

[10] J. A. Boyan. Technical update: Least-squares temporal difference learning. *Mach. Learn.*, 49(2-3):233–246, 2002.

[11] A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

[12] A. M. Devraj and S. Meyn. Zap Q-learning. In *Advances in Neural Information Processing Systems*, pages 2235–2244, 2017.

[13] A. M. Devraj and S. P. Meyn. Fastest convergence for Q-learning. *ArXiv e-prints*, July 2017.

[14] J. Duchi. Introductory lectures on stochastic optimization. *Stanford Lecture Series*, 2016.

[16] P. W. Glynn and S. P. Meyn. A Liapounov bound for solutions of the Poisson equation. *Ann. Probab.*, 24(2):916–931, 1996.

[15] S. Gadat, F. Panloup, and S. Saadane. Stochastic heavy ball. *Electron. J. Statist.*, 12(1):461–529, 2018.

[17] P. Jain, S. M. Kakade, R. Kidambi, P. Netrapalli, and A. Sidford. Accelerating Stochastic Gradient Descent. *ArXiv e-prints (and to appear, COLT 2018)*, Apr. 2017.

[18] T. Kailath. *Linear systems*, volume 156. Prentice-Hall Englewood Cliffs, NJ, 1980.

[19] V. R. Konda and J. N. Tsitsiklis. Convergence rate of linear two-time-scale stochastic approximation. *Ann. Appl. Probab.*, 14(2):796–819, 2004.

[20] H. J. Kushner and G. G. Yin. *Stochastic approximation algorithms and applications*, volume 35 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1997.

[21] N. Loizou and P. Richtárik. Momentum and Stochastic Momentum for Stochastic Gradient, Newton, Proximal Point and Subspace Descent Methods. *ArXiv e-prints*, Dec. 2017.

[22] M. Metivier and P. Priouret. Applications of a Kushner and Clark lemma to general classes of stochastic algorithms. *IEEE Transactions on Information Theory*, 30(2):140–151, March 1984.

[23] S. P. Meyn and R. L. Tweedie. *Markov chains and stochastic stability*. Cambridge University Press, Cambridge, second edition, 2009. Published in the Cambridge Mathematical Library. 1993 edition online.

[24] E. Moulines and F. R. Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems 24*, pages 451–459. Curran Associates, Inc., 2011.

[25] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. In *Soviet Mathematics Doklady*, 1983.

[26] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[27] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[28] B. T. Polyak. *Introduction to Optimization*. Optimization Software Inc, New York, 1987.

[29] B. T. Polyak. A new method of stochastic approximation type. *Avtomatika i telemekhanika (in Russian). translated in Automat. Remote Control, 51 (1991)*, pages 98–107, 1990.

[30] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM J. Control Optim.*, 30(4):838–855, 1992.

[31] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[32] D. Ruppert. A Newton-Raphson version of the multivariate Robbins-Monro procedure. *The Annals of Statistics*, 13(1):236–245, 1985.

[33] R. S. Sutton. Learning to predict by the methods of temporal differences. *Mach. Learn.*, 3(1):9–44, 1988.

[34] C. Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

[35] C. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[36] J. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16:185–202, 1994.

[37] J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Trans. Automat. Control*, 42(5):674–690, 1997.

[38] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, Cambridge, UK, 1989.

[39] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.