Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

SIMULATION OPTIMIZATION AND STOCHASTIC GRADIENTS: THEORY & PRACTICE

Michael C. Fu[1], Jiaqiao Hu[2], Ilya O. Ryzhov[3], and Enlu Zhou[4]

[1]Robert H. Smith School of Business & ISR, University of Maryland, College Park, MD, USA
[2]Department of Applied Mathematics & Statistics, State University of New York, Stony Brook, NY, USA
[3]Robert H. Smith School of Business, University of Maryland, College Park, MD, USA
[4]H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology,
Atlanta, GA, USA

ABSTRACT

This tutorial addresses the use of stochastic gradients in simulation optimization, including methodology and algorithms, theoretical convergence analysis, and applications. Specific topics include stochastic gradient estimation techniques – both direct unbiased and indirect finite-difference-based; stochastic approximation algorithms and their convergence rates; stochastic gradient descent (SGD) algorithms with momentum and reusing past samples via importance sampling; and a real-world application in geographical partitioning.

1 INTRODUCTION

The general problem considered in this tutorial is the optimization of an output performance function $h(\theta)$ – also referred to sometimes as the objective function or the loss function, depending on the context, where $\theta \in \Theta \subset \Re^d$ is the d-dimensional vector of decision variables (or input parameters). The setting is *continuous stochastic* optimization where $h(\theta)$ is estimated via stochastic simulation and gradient search is employed to find the optimum, which may only be locally optimal. The focus is *gradient-based* search methods for stochastic optimization problems arising in both traditional operations research/management science (OR/MS) settings and in more recent artificial intelligence/machine learning (AI/ML) contexts. The former refers to such algorithms as *stochastic approximation* (SA), dating back to its statistical roots and original root-finding origins, whereas the latter has adopted *stochastic gradient descent* (SGD) as its preferred terminology, due to its origination in minimizing loss functions.

We begin with two toy examples to illustrate the general problem setting and motivate the SA/SGD solution approach.

A Stochastic Modeling Perspective

Consider perhaps the simplest queueing system, a single-server queue, where we assume given fixed customer arrivals and varying service requirements. If a stochastic simulation model is used to estimate the expected time in queue and a cost is associated with the server speed $1/\theta$, the following objective (loss) function gives rise to an optimization problem of finding the optimal speed that minimizes this function:

$$h(\theta) = \bar{Q}_n + c/\theta, \quad \bar{Q}_n = \frac{1}{n}\sum_{i=1}^{n} Q_i,$$

where Q_i is the queueing time for the ith customer, n is the number of customers, and c is the server cost coefficient. One way to attempt to solve this optimization problem is to use gradient-based search, assuming the server speed is a continuous variable. Since the underlying model is stochastic, the gradient estimators will also be stochastic.

979-8-3315-8726-0/25/$31.00 ©2025 IEEE

An Optimization or AI/ML Perspective

Consider perhaps the simplest regression model, a single-input single-output linear model. Given N data points $\{(x_i, y_i), i = 1, \dots, N\}$, the best fit is obtained by finding the optimal slope θ (assume for simplicity the line passes through the origin) according to some objective, e.g., minimizing a least-squares loss function:

$$h(\theta) = \frac{1}{N} \sum_{i=1}^{N} [\hat{y}(x_i) - y_i]^2 = \frac{1}{N} \sum_{i=1}^{N} (\theta x_i - y_i)^2 .$$

This simple case can be solved analytically by setting $h'(\theta) = 0$ as a trivial deterministic optimization problem solved in high school calculus. If N is large as in big data settings and $h'(\theta) = 0$ could not be solved analytically (e.g., in a deep neural network), then one approach is to use a sample of size $n \ll N$ to do gradient-based search, where

$$h(\theta) = \frac{1}{n} \sum_{\tilde{i}=1}^{n} [\hat{y}(x_{\tilde{i}}) - y_{\tilde{i}}]^2 = \frac{1}{n} \sum_{\tilde{i}=1}^{n} (\theta x_{\tilde{i}} - y_{\tilde{i}})^2 , \text{ where } \tilde{i} \text{ denotes the } i\text{th element in the sample.}$$

The main intended takeaways from these two toy examples: (i) In one setting, the stochasticity is an intrinsic part of the underlying model, whereas in the other case it is externally imposed for computational reasons, but in both cases, one is sampling from a distribution. In the first case, it is often a theoretical distribution fitted to some data set, whereas in the second case, it is the empirical distribution sampled from a complete (and assumed very large) data set. (ii) Both settings involve gradient-based search using stochastic gradients but arising from different perspectives or directions.

This tutorial can be viewed as an update to the WSC tutorial (Chau et al. 2014), with two other related complementary WSC tutorials given in Newton et al. (2018), Ford et al. (2022). A WSC panel of historical interest discussing the early interplay between simulation and optimization can be found in Fu et al. (2000). A highly recommended (at least by the first two co-authors) very recent complementary "thought" piece on stochastic gradients with minimal technical details can be found in Fu, Hu, and Scheinberg (2025), and the two motivating examples are adapted from there.

The rest of this tutorial is organized as follows. Section 2 provides a brief overview of stochastic gradient estimation techniques, illustrated using the single-server queue example. Section 3 covers the basics of SA, the basis for gradient-based optimization in the stochastic setting. Section 4 discusses some recent SGD methodological/algorithmic and theoretical advances. Section 5 presents a stochastic gradient-based SA application in geographical partitioning.

2 STOCHASTIC GRADIENT ESTIMATION

We will briefly overview what we call the *direct* and *indirect* stochastic gradient (SG) estimators. Indirect SG estimators will refer to those estimators that are based on some form of finite differences (FD) of output performance estimates – thus requiring two or more sample paths or simulation replications – and are generally biased, where the estimator bias and variance is a function of the difference used; these estimators will be discussed in more detail in the next section. On the other hand, direct SG estimators are often unbiased and require just a single sample path or simulation replication (referred to as single-run estimators). Infinitesimal perturbation analysis (IPA) and the likelihood ratio method (LRM) (also known as the score function method) cleanly fit into this definition, because they provide both unbiased and single-run estimators. On the other hand, two of the most generally applicable methods, smoothed perturbation analysis (SPA) and measure-valued differentiation (also known as the method of weak derivatives), are direct in terms of providing unbiased estimators, but the estimators are not necessarily single run (in the latter case, most commonly not). We will use the single-server queue example to illustrate the direct IPA and LRM estimators. The reader is referred to Fu (2015) for more in-depth coverage and technical details.

Let A_i and X_i denote the interarrival times and service times of the ith customer. For a single-server first-come first-served (FCFS) queue, queueing time follows the well-known Lindley equation:

$$Q_i = (Q_{i-1} + X_{i-1} - A_i)^+, \tag{1}$$

which is differentiable almost everywhere (a.e.), the exception being at the "kink" point $Q_{i-1} + X_{i-1} = A_i$. Simple differentiation gives the (unbiased and single-run) IPA estimator for $dE[Q_i]/d\theta$:

$$\frac{dQ_i}{d\theta} = \left(\frac{dQ_{i-1}}{d\theta} + \frac{dX_{i-1}}{d\theta}\right) \mathbf{1}\{Q_{i-1} + X_{i-1} - A_i > 0\} \text{ almost everywhere (a.e.),} \quad \frac{dQ_0}{d\theta} \equiv 0. \tag{2}$$

If the interarrival times $\{A_i\}$ are deterministic constants and the service times $\{X_i\}$ are deterministic functions of θ, then the gradient (derivative) estimator given by (2) is well defined, assuming that $\frac{dX_i}{d\theta}$ is well defined. The simplest example of this is all service times being equal to θ, in which case $\frac{dX_i}{d\theta} = 1$ and $\frac{dQ_i}{d\theta}$ can be computed by applying the recursion (2) above. However, if either the interarrival times $\{A_i\}$ or service times $\{X_i\}$ are random variables, then (2) gives rise to a stochastic gradient. In the (more challenging) case where $\{X_i\}$ are random, the natural question arises as to the meaning of the derivative of a service time X_i w.r.t. θ. Parameters of distributions fall into one of three categories: location, (generalized) scale, or shape (all the rest). For the three cases of a location parameter, a scale parameter, or any parameter of a general continuous r.v. X, the stochastic derivative is given as follows:

$$\frac{dX}{d\theta} = 1 \quad \text{for } \theta \text{ a location parameter of the distribution of } X, \tag{3}$$

$$\frac{dX}{d\theta} = \frac{X}{\theta} \quad \text{for } \theta \text{ a scale parameter of the distribution of } X, \tag{4}$$

$$\frac{dX}{d\theta} = \frac{dF(X;\theta)/d\theta}{dF(X;\theta)/dX} \quad \text{for } X \text{ a continuous r.v. with c.d.f. } F \text{ parameterized by } \theta, \tag{5}$$

where for the righthand side of (5), $dF(\cdot;\theta)/dx$ is simply the probability density function (p.d.f.). A table for many commonly used distributions, including discrete distributions that are not covered by (5), is provided in Fu (2015).

For higher-order derivatives, one might "automatically" differentiate (2) a second time to obtain

$$\frac{d^2Q_i}{d\theta^2} = \left(\frac{d^2Q_{i-1}}{d\theta^2} + \frac{d^2X_{i-1}}{d\theta^2}\right) \mathbf{1}\{Q_{i-1} + X_{i-1} - A_i > 0\}, \tag{6}$$

which turns out to be a *biased* estimator. SPA can be used to derive an unbiased (and single-run) estimator, which results in an augmentation of the recursion (6) with a conditional term involving the square of the first-derivative estimator given by (2), thus remaining straightforward to implement and computationally efficient.

LRM is generally as easy to implement as IPA. For the queueing example, assuming i.i.d. service times with common p.d.f. $f(\cdot;\theta)$ parameterized by θ, the following LRM estimator for $dE[\bar{Q}_n]/d\theta$ can be derived by differentiating the density f rather than the sample performance $\bar{Q}_n$:

$$\bar{Q}_n \cdot \left(\sum_{i=1}^{n} \frac{d\ln f(X_i;\theta)}{d\theta}\right),$$

where the service time parameter θ is expressed in the service time *distribution* rather than in the service time *random variable* as in IPA, so no notion of a derivative of an r.v. is required! An advantage of LRM over IPA is that it is just as easy to implement for all higher-order derivatives. As mentioned previously, the other main advantage of LRM is that because it differentiates the distribution and not the sample performance (loss function), it works for *discontinuous* sample performances.

For the IPA and SPA estimators, the FCFS is critical; if the queue discipline changes, IPA and SPA estimators change form, whereas the form of the LR estimator is unchanged.

3 STOCHASTIC APPROXIMATION

Stochastic approximation (SA) was introduced by Robbins and Monro (1951) for solving the root-finding equation

$$g(\theta) = 0, \tag{7}$$

where g is a real-valued continuous function that is not known exactly and θ is as before the d-dimensional vector of decision variables (or input parameters). The setting assumes an estimate $\hat{g}$ is available, taking the general form $\hat{g}(\theta) = g(\theta) + \xi$, where ξ can be viewed as a (d-dimensional) "noisy" error term representing the combined effect of measurement bias and noise, which may depend on the underlying θ being evaluated, but we suppress the explicit dependency of ξ on θ for expositional simplicity.

In settings where $g(\theta)$ can be evaluated exactly (i.e., $\xi = 0$), a solution to (7) can be found numerically through the deterministic recursive procedure $\theta_{k+1} = \theta_k - \alpha_k g(\theta_k)$, where θ_k is an estimate of the solution at step k and $\alpha_k > 0$ is the step-size (multiplier). The Robbins-Monro (RM) SA algorithm is simply the stochastic analog for solving (7) given by

$$\theta_{k+1} = \theta_k - \alpha_k \hat{g}(\theta_k). \tag{8}$$

The intuition underlying the RMSA algorithm is that the noise terms at different evaluation values of θ_k are implicitly averaged out across successive iterations, so that the RMSA iteration (8) retains the same asymptotic behavior as its deterministic counterpart.

For the stochastic optimization setting, g represents the gradient of the function to be optimized, where we assume the objective takes the form of an expectation $h(\theta) = E[\hat{h}(\theta, \omega)]$, with $\theta \in \Theta \subseteq \Re^d$ being the vector of decision variables, ω representing a sample path, and $h(\theta)$ is estimated by the sample performance $\hat{h}(\theta, \omega)$. When h is differentiable, solving the optimization problem (at least locally under certain conditions) is equivalent to finding the solution to the root-finding problem $g(\theta) := \nabla h(\theta) = 0$. Thus, the RMSA iteration (8) solves a minimization problem, whereas changing the sign (from "$-$" to "$+$") would solve a maximization problem, with $\hat{g}(\theta)$ now denoting an estimate of the objective function gradient at θ, and ξ representing the gradient estimator error. In practical applications, a projection operation is often required to constrain the sequence $\{\theta_k\}$ within a user-defined feasible domain Θ, but for simplicity, we have not included such a projection operator, because it technically has the same effect of adding an extra correction term to the right-hand side of equation (8).

The effectiveness of the SA algorithm (8) relies on a statistically reliable gradient estimator $\hat{g}$. In the previous section, we summarized several direct stochastic gradient estimation techniques, which generally provide unbiased gradient estimators leading to more efficient algorithm performance (e.g., a faster convergence rate) when implemented in (8) as compared to biased estimators. In settings where a direct gradient is either unavailable or difficult to obtain, an indirect gradient technique based on finite differences (FD) requiring only output performance samples can be readily applied. In particular, a symmetric difference (SD) estimator for the ith component of the gradient is given by

$$\hat{g}_j(\theta, c) = \frac{\hat{h}(\theta + ce_j, \omega_j^+) - \hat{h}(\theta - ce_j, \omega_j^+)}{2c}, \quad j = 1, \ldots, d, \tag{9}$$

where $c > 0$ is the perturbation size, e_j is the unit vector in the jth direction, and $\omega_j^{\pm}$ represent two different sample paths. Using common random numbers (CRN) in simulation would imply that the same sample path (random numbers) is used for both output function estimates, i.e., $\omega_j^+ = \omega_j^-$. The SA algorithm (8) using a FD-based gradient estimate is called the Kiefer-Wolfowitz (KW) algorithm (Kiefer and Wolfowitz 1952), which we write as follows:

$$\theta_{k+1} = \theta_k - \alpha_k \hat{g}(\theta_k, c_k), \tag{10}$$

i.e., (8) using a FD-based $\hat{g}$ and thus also depending on the FD perturbation sequence $\{c_k\}$, in addition to the SA iteration itself depending on the step-size (multiplier) sequence $\{\alpha_k\}$.

The SD-based KW algorithm requires $2d$ function measurements at each step, meaning that the algorithm could be computationally demanding for high-dimensional problems. One way to work around this issue is to adopt a random direction (RD) approach that replaces the unit vectors e_j's by a single random vector Δ. The idea is to simultaneously vary all components of the parameter vector θ_k in random directions, so that the similar effect of the SD scheme (9) can be achieved with fewer function evaluations. For example, the SD version of the approach can generally be written as

$$\hat{g}(\theta, c) = \frac{\hat{h}(\theta + c\Delta, \omega^+) - \hat{h}(\theta - c\Delta, \omega^-)}{2c}\Delta, \tag{11}$$

which provides an estimator of the entire gradient at the expense of only two total output function measurements – only two sample paths ω^+ and ω^- at the randomly perturbed parameter vectors $\theta \pm c\Delta$, versus $2d$ sample paths for the SD gradient estimator given by (9). The simplest and most commonly adopted choices of Δ are the standard normal random vector and the symmetric Bernoulli random direction with independent components. Under the latter choice, each component of Δ takes values from $\{-1, 1\}$ with equal probability (commonly referred to as Rademacher distribution in the AI/ML community), so the multiplier Δ in (11) can equivalently be placed under the denominator (assuming elementwise division), in which case (11) used in (8) becomes the well-known simultaneous perturbation SA (SPSA) algorithm introduced by Spall (1992).

Regardless of which perturbation scheme is used (elementwise or simultaneous), the step-size sequence $\{\alpha_k\}$ and the perturbation size sequence $\{c_k\}$ must be carefully chosen in practice to find the right balance between estimation bias and variance. Unlike direct gradients, bias is an inherent part of these estimators that cannot be easily eliminated (as opposed to noise) over the iterations. Thus, convergence to the optimal solution relies on the use of a diminishing perturbation size $c_k \to 0$ to eliminate the bias effect. This requirement, however, implies that estimation variance would grow without bound as $k \to \infty$. Consequently, in both KWSA and RDSA methods, the step-size α_k is required to decrease to zero sufficiently faster than c_k, e.g., satisfying the typical condition $\sum_k (\alpha_k / c_k)^2 < \infty$, to counteract the large gradient estimation variance in the long run.

3.1 Asynchronous SA

The basic root-finding SA method can be extended to estimate an unknown function $Q : \mathscr{S} \to \Re$ based on its noisy input-output measurements. The extension is essentially a pointwise application of (8) to solving functional equations, where θ_k becomes a function serving as an estimate for Q. In its general form, the method can be written as

$$\theta_{k+1}(s) = \theta_k(s) - \alpha_k(s)\big(\theta_k(s) - X_k(s)\big), \; s \in \mathscr{S}, \tag{12}$$

where $X_k(s)$ represents a noisy evaluation of $Q(s)$ at iteration k. In practice, it is often difficult to carry out the recursion simultaneously for all s even for a finite $\mathscr{S}$. For example, consider the (online) scenario where the argument s itself evolves over time according to a set of rules, and in each instance the measurement of Q can only be taken at the current s value. In such cases, the components of θ_k must be updated asynchronously at different times and frequencies. Additionally, the step-size $\alpha_k(s)$ in general becomes a random variable that may depend on how frequently each s value is visited.

The well-known Q-learning algorithm (Watkins 1989) can be viewed as an asynchronous SA method for solving Bellman's equation (Tsitsiklis 1994), a functional equation whose solution is called the Q-function, which measures the expected reward of an action taken by the decision-maker at a given state (Bertsekas 1995). In Q-learning, θ_k in (12) corresponds to an estimate of the Q-function at step k, with s being the state-action pair visited at the current step, whereas X_k represents the sum of the immediately observed reward and the estimated future reward. The algorithm allows function estimates to be constructed based on a single state-action trajectory and requires all state-action pairs to be visited infinitely often for convergence. Some variants of vanilla Q-learning include SARSA (Singh et al. 2000), which update function estimates based on the experience gained from executing a specified policy; and approximate stochastic annealing (Hu and Chang 2012), which combines Q-learning with a random search procedure to optimize the underlying learning policy. Asynchronous SA has also been integrated with function approximation techniques to develop generalizations of Q-learning for solving continuous-state reinforcement learning (RL) problems; see, e.g., Szepesvári and Smart (2004), Shah and Xie (2018), Hu et al. (2024), Yang et al. (2024).

3.2 Multi-Timescale Algorithms

In the basic root-finding SA (8), all components of θ_k are updated using a single step-size α_k. It is possible to consider using distinct step-sizes, allowing different components of the iteration to be carried out along different speeds. Borkar (1997) formalized this idea and introduced a two-timescale SA method:

$$\theta_{k+1}^{(1)} = \theta_k^{(1)} - \alpha_k\big[g^{(1)}(\theta_k^{(1)}, \theta_k^{(2)}) + \xi^{(1)}\big] \tag{13}$$

$$\theta_{k+1}^{(2)} = \theta_k^{(2)} - \beta_k\big[g^{(2)}(\theta_k^{(1)}, \theta_k^{(2)}) + \xi^{(2)}\big], \tag{14}$$

where α_k and β_k are two step-sizes satisfying $\alpha_k/\beta_k \to 0$ (or vice versa), and $\xi^{(1)}$ and $\xi^{(2)}$ are the measurement errors of $g^{(1)}$ and $g^{(2)}$. From a root-finding perspective, the method can be simply viewed as a variant of (8) for solving the augmented problem $g := (g^{(1)}, g^{(2)}) = 0$. However, the use of distinct step-sizes $\alpha_k \ll \beta_k$ implies that the increments in $\theta_k^{(1)}$ will become negligible compared to that of $\theta_k^{(2)}$, meaning that one can roughly treat $\theta_k^{(1)}$ as constant when executing the (faster) recursion (14). Conversely, when viewed from the (slower) recursion (13), the sequence $\{\theta_k^{(2)}\}$ generated from (14) would appear to have converged. This observation suggests that the method could be well suited for optimizing complex systems (e.g., hidden Markov models; see Bhatnagar and Borkar 1997; Bhatnagar et al. 2001), where our decision at each step relies on some unknown system dynamics or parameters that need to be simultaneously estimated for each decision.

Two-timescale SA has been extensively used to develop RL methods. A large class of such methods rely on gradient-based SA to find improved policies, where the gradient estimate is obtained at each step using an approximate value function of the current policy computed through a coupled SA recursion. These are called actor-critic methods (Barto et al. 1983; Konda and Tsitsiklis 2003; Bhatnagar and Kumar 2004; Borkar 2005; Bhatnagar et al. 2009; Bhatnagar and Lakshmanan 2015), where the actor corresponds to the approximate policy, which is often updated on a slower timescale than the critic (value function).

Another application of two-timescale SA is the gradient descent ascent method for solving minimax optimization problems of the form $\min_\theta \max_\psi H(\theta, \psi)$ (Heusel et al. 2017; Lin et al. 2020). The method consists of two nested SA recursions, where the faster recursion approximately solves the inner maximization problem $\max_\psi H(\theta, \psi)$ by holding θ constant, whereas the slower recursion searches for the next θ value by performing a gradient decent on the function $\max_\psi H(\cdot, \psi)$.

Recently, Hu et al. (2022) introduced a three-timescale SA method incorporating direct gradients for optimizing the quantile function of an unknown distribution. Two alternative algorithms based on indirect gradient techniques were subsequently proposed in Hu et al. (2024) for addressing differentiable quantile optimization problems under a general black-box setting. The basic idea underlying these approaches is to transform the quantile and its gradient estimations jointly into a stochastic root-finding problem, then approximate its solution by constructing separate but coupled SA iterations running at different timescales.

3.3 Convergence and Convergence Rates

Over the years, the convergence of SA has been well studied in the literature, with results obtained under varied conditions. Following the discussion in Spall (2003), we classify these analysis techniques/conditions into two broad types: "statistical" and "engineering". Statistical methods often require some knowledge about the (unique) limit θ^* to the sequence $\{\theta_k\}$ and focus on specifying conditions on g and the error term ξ that ensure the convergence (to zero) of the process $\{\theta_k - \theta^*\}$. The smoothness of the function g is typically not assumed, and in some studies (Evans and Weber 1986), there is no explicit condition on the step-size. This type of analysis is often useful in applications where one has a good sense of what an algorithm converges to (Spall and Cristion 1998; Hu and Hu 2011) and wishes to establish the "global" convergence of the algorithm.

The "engineering" approach, often referred to as the ordinary differential equation (ODE) method (Ljung 1977; Kushner and Clark 1978; Kushner and Yin 1997), provides more insight into the asymptotic behavior of SA. The method may be understood as the "reversed" process of numerically solving the ODE

$$\frac{d\theta(t)}{dt} = -g(\theta(t)), \tag{15}$$

where the idea is to construct a continuous-time representation of (8) by "stretching" the iterates $\{\theta_k\}$ continuously in time and then capture its long-run behavior using (15). Unlike the "statistical" conditions mentioned before, the validity of this approach relies on the continuity of g, a condition also needed to guarantee the existence of a solution to the ODE.

The ODE method allows the convergence of an SA algorithm to be studied by examining the limiting solution to an underlying ODE. This connection can be established under "minimal" conditions without imposing any assumptions on the ODE dynamics (Benaim 1996; Kushner and Yin 1997). Thus, compared to "statistical" approaches, the method offers more flexibility and can be adopted in a broader range of applications, including the analysis of multi-timescale SA algorithms (Borkar 1997; Bhatnagar et al. 2001; Borkar 2009; Bhatnagar and Lakshmanan 2015; Hu et al. 2022). However, in the absence of regularity assumptions on the function g, the limiting solutions to (15) are generally described using the notion of invariant sets (Benaim 1996), which may not correspond to the solution of the original root finding problem (7) (Absil and Kurdyka 2006). Consequently, the conclusions that can be drawn from such analysis could sometimes be vague or limited.

Suppose that $\{\theta_k\}$ converges to a limit point θ^*, the convergence rate analysis of SA is concerned with the error behavior of $\{\theta_k\}$ around θ^*. Early studies e.g., Sacks (1958), Fabian (1968), Kushner and Yin (1997), Spall (1992), focused on investigating the statistical properties of the (random) difference $\theta_k - \theta^*$, indicating that for large values of k, a properly scaled version of the difference can be closely approximated by a normal random vector. This type of result can generally be stated as follows:

$$k^{\gamma}(\theta_k - \theta^*) \xrightarrow{d} \mathscr{N}(0, \Sigma), \tag{16}$$

where $\gamma > 0$ is a constant, $\xrightarrow{d}$ denotes convergence in distribution, and $\mathscr{N}(0,\Sigma)$ stands for a normal random vector with mean vector 0 and covariance matrix Σ. In a stochastic sense, (16) implies that when k is large, the iterate θ_k must approach θ^* at a speed that is proportional to $k^{-\gamma}$ in order to keep the product $k^{\gamma}(\theta_k - \theta^*)$ constant in the limit. Under the standard step-size choice $\alpha_k = a/k^{\alpha}$ for constants $a > 0$ and $\alpha \in (0,1]$, it has been shown that the optimal convergence rate of RM algorithms (assuming no estimation bias) is of order $O(k^{-1/2})$ (Sacks 1958), whereas for KW algorithms, this rate reduces to $O(k^{-1/3})$ (Spall 1992) due to the bias in finite difference estimators. In both cases, the best rate is achieved when $\alpha_k = a/k$, i.e., with $\alpha = 1$.

Asymptotic results in this form are mainly of technical interest because they do not offer much insight into an algorithm's behavior when k is small. However, in modern SA applications, such as the training of complex ML/RL models, the primary concern is with the practical performance of an algorithm within a finite computational time/budget. This has led to recent work investigating the finite-time complexity of SA under the SGD context (Nemirovski et al. 2009; Ghadimi and Lan 2012; Duchi et al. 2015; Bottou et al. 2018; Karimi et al. 2019; Driggs et al. 2022; Demidovich et al. 2023), where results are often given in the form of bounds on, e.g., $E[h(\theta_k)-h(\theta^*)]$, $E[\|\theta_k-\theta^*\|^2]$, or $E[\|\nabla h(\theta_k)\|^2]$. The former two error measures are typically used in convex optimization, whereas a (vanishing) bound on the expected gradient $E[\|\nabla h(\theta_k)\|^2]$ can be used to address the algorithm's convergence rate to a stationary point in the stochastic non-convex setting. Demidovich et al. (2023) provide a comprehensive review of different assumptions adopted in SGD literature. In particular, many existing studies have been motivated by ML applications, where constant step-sizes are frequently used to train complex models such as neural networks. Therefore, these results are established based on assuming a constant step-size and that the gradient estimation error either vanishes or remains uniformly bounded at the optimum.

A complementary result that examines the diminishing step-size case is given in Hu and Fu (2025) for convex functions. They show that, through a fixed point argument introduced in Hu et al. (2024),

$$E[\|\theta_k-\theta^*\|] = \mathscr{C} e^{-\rho \sum_{i=1}^{k-1}\alpha_i} + O\big(\sqrt{E[\|b_k\|^2]}\big) + O\big(\alpha_k^{\frac{1}{2}}\sqrt{E[\|\varepsilon_k\|^2]}\big), \tag{17}$$

where $\mathscr{C}>0$ is a constant that depends on the initial solution θ_0, ρ is the smallest eigenvalue of the Hessian matrix of the objective function, and b_k and ε_k denote the respective bias and noise of the gradient estimator. As contrasted with previous studies, the result is based on an explicit bias-variance decomposition, where the variance of the gradient estimator is allowed to increase with the number of algorithm iterations, provided that its effect will eventually be damped out by the step-size in the sense that $\alpha_k E[\|\varepsilon_k\|^2]\to 0$. Note that with a step-size of the form $\alpha_k = a/k$, (17) suggests that the influence of the initial condition will vanish only polynomially in k. Therefore, care must be taken in choosing the constant a, because a too-small value of a can cause the term $e^{-\rho\sum_{i=1}^{k-1}\alpha_i}$ to decay at a rate that is actually slower than those of the bias and variance terms. This explains why SA algorithms with $\alpha_k = a/k$ may perform poorly in practice, even if such a step-size achieves the best possible asymptotic convergence rate.

Unlike single-timescale SA, research on analyzing the convergence rates for multi-timescale algorithms is sparse. Most of the results are in the linear setting, i.e., when $g^{(1)}$ and $g^{(2)}$ in (13) and (14) are both linear functions of their arguments. Notable examples include Konda and Tsitsiklis (2004), which establishes a central limit theorem-type result for two-timescale SA; Dalal et al. (2018), which provides a finite-time probability bound for linear two-timescale SA; and the work of Kaledin et al. (2020), where finite-time expected error bounds are developed under both martingale and Markovian noises.

An extension of the result of Konda and Tsitsiklis (2004) to the nonlinear case is obtained in Mokkadem and Pelletier (2006). Their analysis is based on a local linearization of a nonlinear system around the optimal solution, yielding an asymptotic normality result similar to that of Konda and Tsitsiklis (2004). In another contribution, Doan (2023) derived a finite-time mean squared error (MSE) bound for nonlinear two-timescale SA under the i.i.d. noise setting. He concluded an $O(k^{-2/3})$ convergence rate of the algorithm with appropriately chosen step sizes, assuming no estimation bias and constant covariance matrices for the estimation noise. More recently, Hu et al. (2024) introduced a fixed-point method for studying the finite-time convergence rate of a three-timescale SA algorithm. This method involves bounding an algorithm's estimation errors through the composition of a series of suitably constructed contraction mappings, so that the convergence rates of its iterates can be characterized in detail by inspecting the solutions to a set of fixed point equations. The approach does not require the estimator to be unbiased or to have bounded variance, and can be applied to study general multi-timescale SA algorithms.

4 STOCHASTIC GRADIENT DESCENT: RECENT ADVANCES

4.1 Momentum Stochastic Gradient Descent

Momentum Stochastic Gradient Descent (MSGD, Polyak (1964)), a widely used variant of SGD, takes the following form:

$$\theta_{k+1} = \theta_k - \eta \hat{g}(\theta_k) + \mu(\theta_k - \theta_{k-1}), \tag{18}$$

where $\alpha_k \equiv \eta > 0$ is the constant step size, and $\mu(\theta_k - \theta_{k-1})$ is the momentum parameterized by $\mu \in [0,1)$, with $\mu = 0$ corresponding to a constant step-size SA/SGD.

The ODE approach for showing convergence of ODE described in Section 3.3 cannot be directly applied to analyze MSGD due to the additional momentum term, so a different approach was provided in Liu et al. (2021), which involves first rigorously proving the weak convergence of the trajectory sequence, and then using martingale theory to find the ODE. More specifically, define the continuous-time interpolation $\theta^\eta(\cdot)$ of the solution trajectory of the algorithm as follows: for $t \geq 0$, set $\theta^\eta(t) = \theta_k^\eta$ on the time interval $[k\eta, k\eta + \eta)$. Under certain regularity conditions, for each subsequence of $\{\theta^\eta(\cdot)\}_{\eta>0}$, there exists a further subsequence and a process $\theta(\cdot)$ such that $\theta^\eta(\cdot) \Rightarrow \theta(\cdot)$ in the weak sense as $\eta \to 0$ through the convergent subsequence in the space $D^d[0,\infty)$ (the space of $\mathscr{R}^d$-valued operators which are right continuous and have left-hand limits for each dimension), where $\theta(\cdot)$ satisfies the following ODE:

$$\dot{\theta} = -\frac{1}{1-\mu} g(\theta), \quad \theta(0) = \theta_0. \tag{19}$$

Note that for any solution $\theta(t)$ to the ODE $\dot{\theta} = -g(\theta)$, $\theta(\frac{t}{1-\mu})$ is a solution to ODE (19), implying that asymptotically, MSGD is $\frac{1}{1-\mu}$ faster than SGD in converging to the neighborhood of a stationary point, given the same initialization. Intuitively, with the help of the momentum, the algorithm makes more progress along the descent direction, and therefore momentum can accelerate the algorithm asymptotically.

However, since the noise of the stochastic gradient diminishes as $\eta \to 0$, such a deterministic ODE-based approach is insufficient to analyze the local behavior of MSGD around stationary points where the noise plays a dominant role over the vanishing gradient. Thus, a stochastic differential equation (SDE)-based approach is required for a more precise characterization. To characterize the local algorithmic behavior, define *normalized error* $\frac{\theta_k - \theta^*}{\sqrt{\eta}}$, where $\theta^* \in S$ is a stationary point of the ODE (19), and consider the algorithmic behavior of MSGD when it is around a local optimum θ^*. Define the normalized process $u_k^\eta = (\theta_k^\eta - \theta^*)/\sqrt{\eta}$, where $\lambda_{\min}(\nabla g(\theta^*)) > 0$. Consequently, $U^\eta(t) = (\theta^\eta(t) - \theta^*)/\sqrt{\eta}$. Then, as $\eta \to 0$, the limiting process $\{U^\eta(\cdot)\}$ converges weakly to the unique stationary solution of

$$dU = -\frac{1}{1-\mu} \nabla g(\theta^*) U dt + \frac{1}{1-\mu} dW_t, \tag{20}$$

where $\{W_t\}$ is a Wiener process with covariance matrix $\Sigma = E[\hat{g}(x^*)\hat{g}(x^*)^\top]$. The SDE (20) implies that the momentum essentially increases the variance of the normalized error by a factor of $\frac{1}{1-\mu}$ around the local optimum compared with the usual SGD (i.e., $\mu = 0$), making it more difficult for MSGD to converge. Analogous results obtained for saddle points implies that compared with the usual SGD, momentum accelerates escaping from saddle points by a factor of $1-\mu$. However, momentum can also hurt the final convergence due to the increased variance. To counter this, one suggestion is to decrease the step size by a factor $1-\mu$ in the later stage. The same approach has also been used to analyze asynchronous MSGD, focusing on the trade-off between momentum and asynchrony (Liu et al. 2018).

4.2 Stochastic Gradient Descent Reusing Past Samples via Importance Sampling

When the stochastic gradient estimator suffers from high variance, multiple replications – referred to as mini-batches in the AI/ML community – can be used to reduce variance. Another (complementary) alternative is to use the technique of importance sampling to reuse replications based on previous SA/SGD iterations to estimate the gradient at the current value of θ (Liu and Zhou 2020), i.e.,

$$\hat{g}(\theta_n) = \frac{1}{KB} \sum_{m=n-K+1}^{n} \sum_{i=1}^{B} l(\omega_m^i, \theta_n | \theta_m) \hat{h}(\theta_n), \tag{21}$$

where B is the mini-batch size, and K-1 is the number of past iterations of data used, $\{\omega_m^i, i = 1, \ldots, B\} \overset{\text{i.i.d}}{\sim} f(\cdot; \theta_m)$ for $m = n-K+1, ..., n$, and the likelihood ratio $l(\omega_m^i, \theta_n | \theta_m) = f(\omega_m^i; \theta_n)/f(\omega_m^i; \theta_m)$. Then SGD with reusing past replications (RSGD) is given by (8) using this stochastic gradient estimator.

Earlier work (Eckman and Henderson 2018; Eckman and Feng 2018) noticed that the dependence between iterations introduces bias into the stochastic gradient estimator (21), making theoretical analysis of RSGD difficult; however, empirical evidence indicates RSGD leads to improved performance over SGD. RSGD and SGD share the same limiting ODE for their solution trajectories, and the bias introduced by the dependence between iterations becomes asymptotically negligible (Liu and Zhou 2020); thus, RSGD has the same asymptotic convergence as SGD. Specifically, the continuous-time interpolations of both noise and bias can be shown to reach stationarity by establishing appropriate hidden Markov properties and applying the fixed-state chain method (Theorem 6.6.1 in Kushner and Yin (2003)); see Liu and Zhou (2020) for technical details.

4.3 Reusing Past Samples in Policy Gradient Algorithms

The general approach of RSGD can be used to accelerate policy optimization in reinforcement learning. Here, we describe one variant of the policy gradient method (Lin, Wang, and Zhou 2025). In the policy gradient algorithm, the policy parameter is updated according to the usual (projected version of) SA/SGD iteration:

$$\theta_{n+1} = \mathrm{Proj}_{\Theta}\left(\theta_n + \alpha_n \nabla\eta\left(\theta_n\right)\right),$$

where $\mathrm{Proj}_{\Theta}(\theta)$ is a projection operator that projects the iterate of θ to the feasible parameter space Θ, and $\nabla\eta\left(\theta_n\right)$ is the policy gradient (Sutton et al. 1999) given by

$$\nabla\eta(\theta) = \frac{1}{1-\gamma} \mathbb{E}_{(s,a)\sim d^{\pi_\theta}(s,a)}[A^{\pi_\theta}(s,a) \nabla \log \pi_\theta(a|s)],$$

where $d^{\pi_\theta}(s)$ is the discounted occupancy measure induced by policy π_θ and A^{π_θ} is the advantage function.

The natural policy gradient (NPG) algorithm (Kakade 2001) is the following variant of the policy gradient algorithm:

$$\theta_{n+1} = \mathrm{Proj}_{\Theta}\left(\theta_n + \alpha_n F^{-1}(\theta_n) \nabla\eta\left(\theta_n\right)\right),$$

where $F(\theta) = \mathbb{E}_{(s,a)\sim d^{\pi_\theta}(s,a)}[\nabla \log \pi_\theta(a|s) \left(\nabla \log \pi_\theta(a|s)\right)^T]$ is the Fisher information matrix (FIM) induced by π_θ. The gradient estimator $\widetilde{\nabla\eta}\left(\theta_n\right)$ and FIM estimator $\widetilde{F}\left(\theta_n\right)$ are given by

$$\widetilde{\nabla\eta}\left(\theta_n\right) = \frac{1}{B} \sum_{i=1}^{B} G(\omega_n^i, \theta_n), \quad \widetilde{F}\left(\theta_n\right) = \frac{1}{B} \sum_{i=1}^{B} S(\omega_n^i, \theta_n),$$

where $S(\omega, \theta) = \nabla \log \pi_\theta(a|s) (\nabla \log \pi_\theta(a|s))^T$ and $G(\omega, \theta) = \frac{1}{1-\gamma} A^{\pi_\theta}(s,a) \nabla \log \pi_\theta(a|s)$, and $\omega_n^i = (s_n^i, a_n^i)$ the i-th state-action pair sampled from the discounted occupancy measure $d^{\pi_{\theta_n}}(s,a)$ at iteration n. When reusing past trajectories, the gradient and FIM estimators are as follows:

$$\widehat{\nabla\eta}(\theta_n) = \frac{1}{K_1 B} \sum_{m=n-K_1+1}^{n} \sum_{i=1}^{B} l(\omega_m^i, \theta_n | \theta_m) G(\omega_m^i, \theta_n), \quad \widehat{F}\left(\theta_n\right) = \frac{1}{K_2 B} \sum_{m=n-K_2+1}^{n} \sum_{i=1}^{B} l(\omega_m^i, \theta_n | \theta_m) S(\omega_m^i, \theta_n),$$

where previous $K_1 - 1$ iterations' trajectories are used for estimating the gradient and $K_2 - 1$ iterations' trajectories for estimating FIM, with $l(\omega_m^i, \theta_n | \theta_m) = d^{\pi_{\theta_n}}(\omega_m^i)/d^{\pi_{\theta_m}}(\omega_m^i)$. The update of the natural policy gradient with reusing historical trajectories (RNPG) is then as follows:

$$\theta_{n+1} = \mathrm{Proj}_\Theta\left(\theta_n + \alpha_n \widehat{F}^{-1}(\theta_n)\widehat{\nabla\eta}(\theta_n)\right).$$

As in RSGD, RNPG introduces bias into the FIM and gradient estimators when reusing historical trajectories. By establishing hidden Markov properties similarly as for RSGD, the ODE and SDE methods can be used to prove RNPG converges w.p.1 to a local optimum θ^*, and characterize the error and improvement in convergence rate; see Lin, Wang, and Zhou (2025) for technical details.

5 STOCHASTIC GRADIENT DESCENT: APPLICATION TO GEOGRAPHICAL PARTITIONING

An exciting recent application of stochastic gradient descent has emerged in the domain of optimal transport (Peyré and Cuturi 2019; Ryzhov et al. 2024). This literature focuses on what is essentially a generalization of the well-known "transportation problem" in linear programming (Ford and Fulkerson 1956), which matches two groups of entities (e.g., factories and distribution centers) in a way that minimizes cost. In its most general form, optimal transport aims to design a joint distribution for two random variables X and Y with prespecified marginals to optimize an expected cost function that depends on both X and Y simultaneously. Of particular interest is the *semidiscrete* setting (Hartmann and Schuhmacher 2020), where Y is a discrete random variable with finite support $\{1,...,K\}$ (as in the classical linear programming application), but X is a continuous random variable supported on some compact set $\mathscr{X} \subseteq \mathbb{R}^d$. Formally, we write

$$\inf_\pi \sum_{k=1}^K \int_{\mathscr{X}} c(x,k)\,\pi(x,k)\,dx \tag{22}$$

subject to

$$\int_{\mathscr{X}} \pi(x,k)\,dx = p_k, \qquad k = 1,...,K, \tag{23}$$

$$\sum_k \pi(x,k) = f(x), \qquad x \in \mathscr{X}, \tag{24}$$

$$\pi(x,k) \geq 0, \qquad x \in \mathscr{X}, \quad k = 1,...,K. \tag{25}$$

This is a functional optimization problem with an infinite-dimensional decision variable π representing the mixed joint likelihood $P(X \in dx, Y = k) = \pi(x,k)\,dx$. The vector p and the function f are known problem inputs representing the marginal pmf of Y and the marginal density of X, respectively. The objective (22) minimizes $\mathbb{E}_\pi(c(X,Y))$, while (23)-(24) ensure that the marginal distributions are preserved.

At first glance, there is nothing connecting this problem to stochastic gradient descent. However, (22)-(25) is an infinite-dimensional linear program, and thus has a Kantorovich dual given by

$$\sup_{\phi,\psi} \int_{\mathscr{X}} \phi(x) f(x)\,dx + \sum_k p_k \psi_k \tag{26}$$

subject to

$$\phi(x) + \psi_k \leq c(x,k), \qquad x \in \mathscr{X}, \quad k = 1,...,K. \tag{27}$$

This problem has an infinite-dimensional decision variable ϕ and a finite-dimensional decision variable ψ. However, since the dual objective (26) is separable in ϕ and ψ, we can eliminate (27) entirely by taking

$$\phi(x) = \min_k c(x,k) - \psi_k, \qquad x \in \mathscr{X}. \tag{28}$$

Substituting (28) into (27) yields the unconstrained and finite-dimensional optimization problem

$$\max_\psi \mathbb{E}\left(\min_k c(X,k) - \psi_k\right) + \sum_k p_k \psi_k. \tag{29}$$

It is straightforward (Carlsson, Carlsson, and Devulapalli 2016) to show that the optimal solution ψ^* of (29) induces an optimal solution to (22)-(25). One simply takes

$$Y^*(X) = \arg\min_k c(X,k) - \psi_k^*, \tag{30}$$

with ties broken arbitrarily. The joint distribution of $(X, Y^*(X))$ can be shown to optimally solve the original (primal) problem. This form of the solution has applications in geographical partitioning problems motivated by facility logistics. One may define $\mathscr{A}_k = \{x : Y^*(x) = k\}$ and observe that the sets $\mathscr{A}_1, ..., \mathscr{A}_K$ form a partition of $\mathscr{X}$. In a typical applied context, $\mathscr{X}$ is a planar region, and each k value is associated with a facility located at some $x_k \in \mathscr{X}$. The cost function $c(x,k)$ represents the cost incurred by a customer residing at location x when traveling to facility k, with the Euclidean distance $c(x,k) = \|x - x_k\|_2$ being a typical (but not the only possible) choice. Equation (30) assigns customers to facilities based on their locations; the values ψ_k^* can be viewed as bonuses and penalties that modify the travel costs in order to ensure that the kth facility receives exactly a proportion p_k of customers. In fact, by applying direct gradients, one can see that the optimality conditions of (29) are exactly

$$P\left(k = \arg\min_j c(X,j) - \psi_j\right) = p_k, \qquad k = 1, ..., K.$$

It remains only to solve (29). The quantity inside the expected value is a finite minimum of *linear* functions of ψ, and therefore concave. Expectations preserve concavity, so (29) is a concave maximization problem. If we have the ability to simulate i.i.d. samples $\{X^n\}_{n=1}^{\infty}$ from the density f, the problem turns into an almost trivial application of stochastic gradient descent (Genevay et al. 2016). We simply apply the update

$$\psi_k^{n+1} = \psi_k^n + \alpha_n \left(-1_{\left\{k=\arg\min_j c(X^{n+1},j)-\psi_j^n\right\}} + p_k\right).$$

In this way, a problem that originally started out as infinite-dimensional can be solved efficiently, and the optimal solution is completely characterized by a finite-dimensional vector ψ^*. With (30) used to make assignments, the optimal partition becomes easy to store in memory and visualize. Partitions based on (30) have been studied for many years under the name "additively weighted Voronoi diagrams" (Aurenhammer 1991), but traditional numerical methods focused on the geometry of (30) in $\mathscr{X}$ and were computationally quite cumbersome. By viewing (29) as a finite-dimensional simulation optimization problem, we are able to solve it using essentially a single line of code. In fact, although the optimal transport literature universally assumes that the marginal density f of X is known, strictly speaking we do not need to know it in order to solve (30). We only require the ability to observe samples from it.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (NSF) under Grants CMMI-2027527, CMMI-2146067, IIS-2123684 and ECCS-2419562; and by Air Force Office of Scientific Research (AFOSR) under Grant FA9550-22-1-0244.

REFERENCES

Absil, P. A., and K. Kurdyka. 2006. "On the Stable Equilibrium Points of Gradient Systems". *System & Control Letters* 55:573–577.

Aurenhammer, F. 1991. "Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure". *ACM Computing Surveys* 23(3):345–405.

Barto, A. G., R. S. Sutton, and C. W. Anderson. 1983. "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems". *IEEE Transactions on Systems, Man, and Cybernetics* 13:833–846.

Benaim, M. 1996. "A Dynamical System Approach to Stochastic Approximations". *SIAM Journal on Control and Optimization* 34(2):437–472.

Bertsekas, D. P. 1995. *Dynamic Programming and Optimal Control*, Volume 1. Athena Scientific.

Bhatnagar, S., and V. S. Borkar. 1997. "Multiscale Stochastic Approximation for Parametric Optimization of Hidden Markov Models". *Probability in the Engineering and Informational Sciences* 11:509–522.

Bhatnagar, S., M. C. Fu, S. I. Marcus, and S. Bhatnagar. 2001. “Two-Timescale Algorithms for Simulation Optimization of Hidden Markov Models”. *IIE Transactions* 33:245–258.

Bhatnagar, S., and S. Kumar. 2004. “A Simultaneous Perturbation Stochastic Approximation-based Actor-Critic Algorithm for Markov Decision Processes”. *IEEE Transactions on Automatic Control* 49(4):592–598.

Bhatnagar, S., and K. Lakshmanan. 2015. “Multiscale Q-learning with Linear Function Approximation”. *Discrete Event Dynamic Systems* 26:477–509.

Bhatnagar, S., R. S. Sutton, M. Ghavamzadeh, and M. Lee. 2009. “Natural Actor-Critic Algorithms”. *Automatica* 45(11):2471–2482.

Borkar, V. S. 1997. “Stochastic Approximation with Two Time Scales”. *Systems & Control Letters* 29(5):291–294.

Borkar, v. S. 2005. “An Actor-Critic Algorithm for Constrained Markov Decision Processes”. *Systems & Control Letters* 54(3):207–213.

Borkar, V. S. 2009. *Stochastic Approximation: A Dynamical Systems Viewpoint*, Volume 48. Springer.

Bottou, L., F. E. Curtis, and J. Nocedal. 2018. “Optimization Methods for Large-scale Machine Learning”. *SIAM Review* 60(2):223–311.

Carlsson, J. G., E. Carlsson, and R. Devulapalli. 2016. “Shadow Prices in Territory Division”. *Networks and Spatial Economics* 16(3):893–931.

Chau, M., M. C. Fu, H. Qu, and I. O. Ryzhov. 2014. “Simulation Optimization: A Tutorial Overview and Recent Developments in Gradient-Based Methods”. In *2014 Winter Simulation Conference (WSC)*, 21–35.

Dalal, G., G. Thoppe, B. Szörényi, and S. Mannor. 2018, 06–09 Jul. “Finite Sample Analysis of Two-Timescale Stochastic Approximation with Applications to Reinforcement Learning”. In *Proceedings of the 31st Conference On Learning Theory*, edited by S. Bubeck, V. Perchet, and P. Rigollet, Volume 75 of *Proceedings of Machine Learning Research*, 1199–1233: PMLR.

Demidovich, Y., G. Malinovsky, I. Sokolov, and R. Richtárik. 2023. “A Guide Through the Zoo of Biased SGD”. *arXiv preprint arXiv:2305.16296*.

Doan, T. T. 2023. “Nonlinear Two-Time-Scale Stochastic Approximation: Convergence and Finite-Time Performance”. *IEEE Transactions on Automatic Control* 68(8):4695–4705.

Driggs, D., J. Liang, and C. Schönlieb. 2022. “On Biased Stochastic Gradient Estimation”. *Journal of Machine Learning Research* 23:1057–1099.

Duchi, J. C., M. I. Jordan, M. J. Wainwright, and A. Wibisono. 2015. “Optimal Rates for Zero-Order Convex Optimization: The Power of Two Function Evaluations”. *IEEE Transactions on Information Theory* 61(5):2788–2806.

Eckman, D. J., and M. Feng. 2018. “Green Simulation Optimization Using Likelihood Ratio Estimators”. In *2018 Winter Simulation Conference (WSC)*, 2049–2060.

Eckman, D. J., and S. G. Henderson. 2018, July. “Reusing Search Data in Ranking and Selection: What Could Possibly Go Wrong?”. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 28(3):1–15 https://doi.org/10.1145/3170503.

Evans, S. N., and N. C. Weber. 1986. “On the Almost Sure Convergence of a General Stochastic Approximation Procedure”. *Bulletin of the Australian Mathematical Society* 34(3):335–342.

Fabian, V. 1968. “On Asymptotic Normality in Stochastic Approximation”. *The Annals of Mathematical Statistics* 39:1327–1332.

Ford, L. R., and D. R. Fulkerson. 1956. “Solving the Transportation Problem”. *Management Science* 3(1):24–32.

Ford, M. T., D. J. Eckman, and S. G. Henderson. 2022. “Automatic Differentiation for Gradient Estimators in Simulation”. In *2022 Winter Simulation Conference (WSC)*, 1334–1345.

Fu, M. C. 2015. “Stochastic Gradient Estimation”. In *Handbook on Simulation Optimization*, edited by M. C. Fu, Chapter 5, 109–150. Springer.

Fu, M. C., S. Andradóttir, J. S. Carson, F. W. Glover, C. R. Harrell, Y. C. Ho, *et al*. 2000. “Integrating Optimization and Simulation: Research and Practice”. In *2000 Winter Simulation Conference (WSC)*, 610–616.

Fu, M. C., J. Hu, and K. Scheinberg. 2025. “Stochastic Gradients: Optimization, Simulation, Randomization, and Sensitivity Analysis”. *IISE Transactions* https://doi.org/10.1080/24725854.2025.2469839.

Genevay, A., M. Cuturi, G. Peyré, and F. Bach. 2016. “Stochastic Optimization for Large-scale Optimal Transport”. In *Advances in Neural Information Processing Systems*, edited by D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, Volume 29, 3440–3448: Curran Associates, Inc.

Ghadimi, S., and G. Lan. 2012. “Optimal Stochastic Approximation Algorithms for Strongly Convex Stochastic Composite Optimization, I: A Generic Algorithmic Framework”. *SIAM Journal on Optimization* 22:1469–1492.

Hartmann, V., and D. Schuhmacher. 2020. “Semi-discrete Optimal Transport: A Solution Procedure for the Unsquared Euclidean Distance Case”. *Mathematical Methods of Operations Research* 92(1):133–163.

Heusel, M., H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. 2017. “GANs Trained by a Two Time-scale Update Rule Converge to a Local Nash Equilibrium”. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 6629–6640.

Hu, J., and H. S. Chang. 2012. "Approximate Stochastic Annealing for Online Control of Infinite Horizon Markov Decision Processes". *Automatica* 48:2182–2188.

Hu, J., and M. C. Fu. 2025. "Technical Note: On the Convergence Rate of Stochastic Approximation for Gradient-Based Stochastic Optimization". *Operations Research* 73(2):1143–1150.

Hu, J., and P. Hu. 2011. "Annealing Adaptive Search, Cross-Entropy, and Stochastic Approximation in Global Optimization". *Naval Research Logistics* 58:457–477.

Hu, J., Y. Peng, G. Zhang, and Q. Zhang. 2022. "A Stochastic Approximation Method for Simulation-Based Quantile Optimization". *INFORMS Journal on Computing* 34(6):2889–2907.

Hu, J., M. Song, and M. C. Fu. 2024. "Quantile Optimization via Multiple-Timescale Local Search for Black-Box Functions". *Operations Research*:forthcoming https://doi.org/10.1287/opre.2022.0534.

Hu, J., X. Yang, J.-Q. Hu, and Y. Peng. 2024. "A Q-learning Algorithm for Markov Decision Processes with Continuous State Spaces". *Systems & Control Letters* 187(105782).

Kakade, S. M. 2001. "A Natural Policy Gradient". In *Advances in Neural Information Processing Systems*, 1531–1538.

Kaledin, M., E. Moulines, A. Naumov, V. Tadic, and H.-T. Wai. 2020, July. "Finite Time Analysis of Linear Two-timescale Stochastic Approximation with Markovian Noise". In *COLT 2020 - 33rd Conference on Learning Theory*.

Karimi, B., B. Miasojedow, E. Moulines, and H.-T. Wai. 2019. "Non-asymptotic Analysis of Biased Stochastic Approximation Scheme". In *Proceedings of Machine Learning Research*, Volume 99, 1–31.

Kiefer, J., and J. Wolfowitz. 1952. "Stochastic Estimation of the Maximum of a Regression Function". *The Annals of Mathematical Statistics* 23:462–266.

Konda, V. R., and J. N. Tsitsiklis. 2003. "On Actor-Critic Algorithms". *SIAM Journal on Control and Optimization* 42(4):1143–1166.

Konda, V. R., and J. N. Tsitsiklis. 2004. "Convergence Rate of Linear Two-time-scale Stochastic Approximation". *The Annals of Applied Probability* 14(2):796–819.

Kushner, H., and G. G. Yin. 2003. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer.

Kushner, H. J., and D. S. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag.

Kushner, H. J., and G. G. Yin. 1997. *Stochastic Approximation Algorithms and Applications*. Springer.

Lin, T., C. Jin, and M. Jordan. 2020, 13–18 Jul. "On Gradient Descent Ascent for Nonconvex-Concave Minimax Problems". In *Proceedings of the 37th International Conference on Machine Learning*, edited by H. D. III and A. Singh, Volume 119 of *Proceedings of Machine Learning Research*, 6083–6093: PMLR.

Lin, Y., Y. Wang, and E. Zhou. 2025. "Reusing Historical Trajectories in Natural Policy Gradient via Importance Sampling: Convergence and Convergence Rate". *Operations Research*. forthcoming.

Liu, T., Z. Chen, E. Zhou, and T. Zhao. 2021. "A Diffusion Approximation Theory of Momentum Stochastic Gradient Descent in Nonconvex Optimization". *Stochastic Systems* 11(4):307–323.

Liu, T., S. Li, J. Shi, E. Zhou, and T. Zhao. 2018. "Towards Understanding Acceleration Tradeoff between Momentum and Asynchrony in Nonconvex Stochastic Optimization". In *Advances in Neural Information Processing Systems*, 3682–3692.

Liu, T., and E. Zhou. 2020. "Simulation Optimization by Reusing Past Replications: Don't Be Afraid of Dependence". In *2020 Winter Simulation Conference (WSC)*, 2923–2934 https://doi.org/10.1109/WSC48552.2020.9384032.

Ljung, L. 1977. "Analysis of Recursive Stochastic Algorithms". *IEEE Transactions on Automatic Control* 22:551–575.

Mokkadem, A., and M. Pelletier. 2006. "Convergence Rate and Averaging of Nonlinear Two-time-scale Stochastic Approximation Algorithms". *The Annals of Applied Probability* 16(3):1671–1702.

Nemirovski, A., A. Juditsky, G. Lan, and A. Shapiro. 2009. "Robust Stochastic Approximation Approach to Stochastic Programming". *SIAM Journal on Optimization* 19(4):1574–1609.

Newton, D., R. Pasupathy, and F. Yousefian. 2018. "Recent Trends in Stochastic Gradient Descent for Machine Learning and Big Data". In *2018 Winter Simulation Conference (WSC)*, 366–380.

Peyré, G., and M. Cuturi. 2019. "Computational Optimal Transport: With Applications to Data Science". *Foundations and Trends in Machine Learning* 11(5-6):355–607.

Polyak, B. T. 1964. "Some Methods of Speeding Up the Convergence of Iteration Methods". *USSR Computational Mathematics and Mathematical Physics* 4(5):1–17.

Robbins, H., and S. Monro. 1951. "A Stochastic Approximation Method". *The Annals of Mathematical Statistics* 22:400–407.

Ryzhov, I., R. Pasupathy, and H. Honnappa. 2024. "Introduction to Optimal Transport". In *2024 Winter Simulation Conference (WSC)*, 101–115.

Sacks, J. 1958. "Asymptotic Distribution of Stochastic Approximation Procedures". *The Annals of Mathematical Statistics* 29:373–405.

Shah, D., and Q. Xie. 2018. "Q-learning with Nearest Neighbors". In *Advances in Neural Information Processing Systems*, 3111–3121.

Singh, S., T. Jaakkola, M. L. Littman, and C. Szepesvári. 2000. “Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms”. *Machine Learning* 38:287–308.

Spall, J., and J. Cristion. 1998. “Model-free Control of Nonlinear Stochastic Systems with Discrete-time Measurements”. *IEEE Transactions on Automatic Control* 43(9):1198–1210.

Spall, J. C. 1992. “Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation”. *IEEE Transactions on Automatic Control* 37(3):332–341.

Spall, J. C. 2003. *Introduction to Stochastic Search and Optimization*. John Wiley & Sons.

Sutton, R. S., D. McAllester, S. Singh, and Y. Mansour. 1999. “Policy Gradient Methods for Reinforcement Learning with Function Approximation”. In *Advances in Neural Information Processing Systems*, 1057–1063.

Szepesvári, C., and W. D. Smart. 2004. “Interpolation-based Q-learning”. In *Proceedings of the 21st International Conference on Machine Learning*, 791–798.

Tsitsiklis, J. N. 1994. “Asynchronous Stochastic Approximation and Q-learning”. In *Proceedings of the 32nd IEEE Conference on Decision and Control*.

Watkins, C. J. C. H. 1989. *Learning from Delayed Rewards*. Ph.D. thesis, Cambridge University.

Yang, X., J. Hu, and J.-Q. Hu. 2024. “Relative Q-Learning for Average Reward Markov Decision Processes with Continuous States”. *IEEE Transactions on Automatic Control* 69(10):6546–6560.

AUTHOR BIOGRAPHIES

MICHAEL C. FU holds the Smith Chair of Management Science at the Robert H. Smith School of Business, with a joint appointment in the Institute for Systems Research and affiliate faculty appointment in the Department of Electrical and Computer Engineering, A. James Clark School of Engineering, all at the University of Maryland, College Park, where he has been since 1989. His research interests include stochastic gradient estimation, simulation optimization, and applied probability, with applications to manufacturing, supply chain management, and healthcare. He has a Ph.D. in applied math from Harvard and degrees in math and EECS from MIT. He is the co-author of the books, *Conditional Monte Carlo: Gradient Estimation and Optimization Applications*, which received the INFORMS Simulation Society’s 1998 Outstanding Publication Award, and *Simulation-Based Algorithms for Markov Decision Processes*. He attended his first WSC in 1988 and served as Program Chair for the 2011 WSC. He is a Fellow of INFORMS and IEEE. His e-mail address is mfu@umd.edu, and his website is https://www.rhsmith.umd.edu/directory/michael-fu.

JIAQIAO HU is a professor in the Department of Applied Mathematics and Statistics at the State University of New York, Stony Brook. His research interests include Markov decision processes, simulation-based optimization, stochastic modeling and analysis, and computational learning theory. He received his undergraduate degree from Shanghai Jiao Tong University, an M.S. in mathematics & statistics from the University of Maryland, Baltimore County, and a PhD degree in Electrical and Computer Engineering from the University of Maryland, College Park. He is the co-author of the book *Simulation-Based Algorithms for Markov Decision Processes*. He served as an Associate Editor for *Operations Research* from 2014–2023, and has been on the editorial board of *IISE Transactions* since 2013, serving as a Department Editor since 2018. His email address is jiaqiao.hu.1@stonybrook.edu, and his website is https://www.stonybrook.edu/commcms/ams/people/_faculty_profiles/hu.

ILYA O. RYZHOV is the Dean’s Professor of Decision Sciences at the Robert H. Smith School of Business, University of Maryland. His research interests include stochastic optimization, statistics, and applications in public sector operations research. He received a B.S. in computer science from Cornell University, master’s degrees from Cornell, Stanford, and Princeton, and a Ph.D. in operations research and financial engineering from Princeton University. He currently serves as Associate Editor at *Operations Research* and *INFORMS Journal on Computing*. He is the co-author of the book *Optimal Learning*. He won I-SIM’s Outstanding Paper Award in 2017, and was recognized by WSC’s Best Theoretical Paper award competition on three occasions (winner in 2012, finalist in 2009 and 2016). His email address is iryzhov@rhsmith.umd.edu, and his website is https://sites.google.com/umd.edu/iryzhov/home.

ENLU ZHOU is a Fouts Family Professor in the H. Milton Stewart School of Industrial and Systems Engineering at the Georgia Institute of Technology. Her research interests include simulation optimization, stochastic optimization, and stochastic control. She received the B.S. degree with highest honors in electrical engineering from Zhejiang University, China, in 2004, and received the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2009. She is a recipient of the Best Theoretical Paper award at the Winter Simulation Conference, AFOSR Young Investigator award, NSF CAREER award, and INFORMS Outstanding Simulation Publication Award. She has served on the editorial boards of *Operations Research*, *IEEE Transactions on Automatic Control*, and *SIAM Journal on Optimization*, and is currently co-Editor-in-Chief for *Journal of Simulation*. She serves as the President of the INFORMS Simulation Society from 2024–2026. Her email address is enlu.zhou@isye.gatech.edu, and her website is https://www.enluzhou.gatech.edu/.