Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

RECENT TRENDS IN STOCHASTIC GRADIENT DESCENT FOR MACHINE LEARNING AND BIG DATA

David Newton
Raghu Pasupathy

Department of Statistics
Purdue University
West Lafayette IN 47906

Farzad Yousefian

Dept. of Industrial Engineering and Management
Oklahoma State University
Stillwater, OK 74078

ABSTRACT

Stochastic Gradient Descent (SGD), also known as *stochastic approximation*, refers to certain simple iterative structures used for solving stochastic optimization and root finding problems. The identifying feature of SGD is that, much like in gradient descent for deterministic optimization, each successive iterate in the recursion is determined by adding an appropriately scaled gradient estimate to the prior iterate. Owing to several factors, SGD has become the leading method to solve optimization problems arising within large-scale machine learning and "big data" contexts such as classification and regression. This tutorial covers the basics of SGD with an emphasis on modern developments. The tutorial starts with examples where SGD is applicable, and then details important flavors of SGD and reported complexity calculations.

1 INTRODUCTION AND PRELIMINARIES

This tutorial considers the unconstrained smooth stochastic optimization problem having the form

$$\begin{aligned} \text{Problem } P: \min\ & f(x) = \mathbb{E}[F(x)] = \int_{\Xi} F(x,\xi)\, dP(\xi) \\ \text{s.t. } & x \in X \subset \mathbb{R}^n. \end{aligned} \tag{1}$$

The objective function f is assumed to be bounded from below, that is, $\inf_{x\in X} f(x) > -\infty$. The tutorial covers methods for smooth functions f. Recall that a function is said to be L-smooth if f is differentiable in $\mathbb{R}^n$ and for all $x, y \in \mathbb{R}^n$, $\|\nabla f(x) - \nabla f(y)\| \leq L\|x-y\|$. Unlike deterministic optimization (Nocedal and Wright 2006), the function f is not directly observable but we have access to a "first-order stochastic oracle" that can be "called" to obtain unbiased estimates $F(x,\xi)$ and $G(x,\xi)$ of $f(x)$ and $\nabla f(x)$, respectively, at any requested point $x \in X$. We assume that $F(x,\cdot)$ and $G(x,\cdot)$ are unbiased estimates of $f(x)$ and $\nabla f(x)$. The set X is assumed to be a convex subset of $\mathbb{R}^n$.

The problem statement that appears in (1) has recently generated renewed interest due to its direct applicability in parameter estimation problems arising within modern machine learning settings such as regression, classification, clustering, and anomaly detection (Hastie et al. 2015). As some observers have noted (Bottou et al. 2016), deep neural networks as a modeling paradigm, in concert with efficient stochastic optimization algorithms (mainly stochastic gradient descent to solve Problem P), have recently resulted in spectacular successes in diverse domains.

Especially because the importance of Problem P has been recorded at length elsewhere, we will not devote any more space to motivating Problem P. Instead, this tutorial will cover modern solution methods, particularly variants of *stochastic approximation* (Kushner and Yin 2003; Borkar 1997) for solving Problem P. Stochastic approximation, more recently called *stochastic gradient descent* (SGD), is understood here

as recursions having the form

$$X_{k+1} = \Pi_X \left(X_k - \alpha_k H_k^{-1} g(X_k, M(X_k)) \right). \tag{2}$$

The term α_k appearing in (2) is traditionally called the *step size*, the positive definite symmetric matrix H_k approximates the Hessian matrix of second derivatives $H(\cdot)$ at X_k, and $g(X_k, M(X_k))$ approximates the gradient $\nabla f(X_k)$ using sample size $M(X_k)$; also $\Pi_X(x) = \inf_{y \in X}\{\|x-y\|\}$ denotes the projection of the point $x \in \mathbb{R}^n$ on the convex set X.

1.1 Some Terminology and Notions

The following terminology and notions will be assumed throughout the tutorial.

1. **Solving Problem *P*.** An iterative algorithm will have "solved" (1) if it generates a stochastic sequence $\{X_k\}$ that satisfies $\|\nabla f(X_k)\| \to 0$ as $k \to \infty$ with probability one (w.p.1.). Of course, this places no guarantees on the behavior of the sequence $\{X_k\}$ itself without further structural assumptions on the function f. Furthermore, the function f may have multiple minima (or none) and the guarantee $\|\nabla f(X_k)\| \to 0$ w.p.1 says little about which, if any, of the local minima of f are attained.
2. **Nature of the "stochastic oracle."** The notion of a "stochastic oracle" is deliberately left vague to subsume a variety of contexts. For the purposes of this tutorial, a "stochastic oracle" is either a Monte Carlo simulation (Nelson 2013; Asmussen and Glynn 2007; Glasserman 2004) or a large dataset of collected observations. Accordingly, "calling the stochastic oracle" at the point $x \in \mathbb{R}^n$ using the "seed" ξ_i results in observing $F(x, \xi_i)$ and $G(x, \xi_i)$.
3. **Algorithm assessment and work complexity.** The number of calls to the stochastic oracle is the sole unit of computational burden. Thus, the work complexity for the purposes of this tutorial relates to the quality of a solution retuned by an algorithm as a function of the number of calls to the stochastic oracle. As an example, when we say that an algorithm exhibits $\mathcal{O}(\varepsilon^{-2})$ complexity, we mean that the solution X_k returned after k calls to the stochastic oracle guarantees that $\mathbb{E}[\|\nabla f(X_k)\|^2] \leq \varepsilon$. Iteration complexity is not useful as a measure except when the sample size during each iteration is fixed, in which case the work complexity and the iteration complexity differ only by a constant.

1.2 Scope of the Tutorial

The tutorial is primarily aimed at early researchers and consumers of stochastic optimization. The content will accordingly be kept at an accessible level. Codes corresponding to best-performing algorithms included in the oral presentation of this tutorial will be made available upon request.

Since stochastic optimization has recently become so widespread, we find it especially necessary to list caveats and key topics that this tutorial will *not cover*. This is a tutorial on the recent variants of SGD, which for the purposes of this tutorial are understood to have the structure in (2). Numerous other paradigms such as stochastic trust region methods (Shashaani et al. 2016; Shashaani et al. 2018; Bandeira et al. 2014; Chen et al. 2016; Chang et al. 2013), sample average approximation (Shapiro et al. 2009), and retrospective approximation (Pasupathy 2010; Pasupathy and Schmeiser 2009), which have recently gained prominence, will not be discussed here.

As has been noted, SGD is the new nomenclature for stochastic approximation, which was first introduced through a seminal paper by Robbins and Monro (1951). Over the previous six decades, an enormous literature that includes comprehensive surveys (Lai 2003) on virtually all aspects of stochastic approximation has been written. Wisely, this tutorial will not undertake to supplant or add to any of these surveys on stochastic approximation. Instead, the tutorial explains trends in the last two decades appearing under the topic *stochastic gradient descent*. Such papers have mainly appeared in the recent literature on machine learning. While most of these more-recent methods have already appeared in some form in

the older stochastic approximation literature, they tend to have an increased focus on complexity results afforded through more-stringent structural assumptions on the function f.

Owing to space restrictions here and time restrictions on the oral presentation, solutions to several important variations of Problem P will not be considered in the tutorial. For example, the tutorial will discuss only algorithms that assume access to a first-order oracle, that is, the algorithms have access to "direct gradient" observations $G(x,\xi)$. A substantial number of simulation contexts are "derivative free" implying that even though the gradient $\nabla f(x)$ might exist at the point $x \in \mathbb{R}^n$, estimates of $\nabla f(x)$ can be constructed only using methods such as finite differencing (Asmussen and Glynn 2007). Similarly, optimization in Problem P is stated to be on finite-dimensional Euclidean spaces, excluding important stochastic optimization problems on non-Euclidean spaces (Bubeck 2015; Nemirovski et al. 2009).

A substantial fraction of methods that appear under the SGD banner are conditioned on a given dataset. One of the many important implications of this fact is that any mathematical expectations that appear in a claimed result on complexity are on the probability space on which the algorithm is defined, and *conditional on the given dataset*. Four prominent examples of algorithms in this category are Schmidt et al. (2013), Shalev-Shwartz and Zhang (2013), Johnson and Zhang (2013), and Defazio et al. (2014). The fact that algorithmic inference in such contexts is conditional on the given dataset is often missed or ignored, leading to misinterpretations about the effectiveness of an algorithm. Since our interest is on inference made at the population level, we have tended to de- emphasize algorithms that are customized to a specific dataset.

The SGD literature contains a substantial fraction of algorithms that apply to the *online learning* context. Such problems are very related to stochastic optimization problems (Duchi et al. 2011), but differ mainly in the way performance is measured. In the online learning setting, for example, algorithms seek to minimize *regret* measured as the deviation of the incumbent solution's quality from the optimal value, integrated over a specified time horizon. Again, due to space restrictions, online learning algorithms are not discussed in this tutorial.

2 MOTIVATING EXAMPLES

In what follows, we provide two example contexts where the solution methods we discuss in this tutorial are relevant. The first example context is typical of machine learning settings involving a large amount of data that facilitate the construction of the relevant estimators. In the second example, by contrast, the "data generation" results from using Monte Carlo simulation.

2.1 Example 1: Classification and Regression

Consider the context of classifying (or labeling) an object based on its observed features. An especially apt example of such an object classification is that of face recognition using a photograph that is represented as a large number of pixels, each of which has a color value. This task is to be accomplished algorithmically, by constructing a parametrized model that is trained over a given dataset of photographs to minimize a chosen loss function.

To abstract this setup, suppose $W \in \mathscr{W} \subset \mathbb{R}^{d_w}$ and $Y \in \mathscr{Y}$ are well-defined random objects in a probability space that represent the *feature* and the *label* in machine learning parlance. Also, let $x \in \mathbb{R}^d$ denote the decision variable representing a parameter vector of interest, $m(\cdot;x) : \mathscr{W} \to \mathscr{Y}$ a family of models parameterized by x, and $\ell : \mathscr{Y} \times \mathscr{Y} \to [0,\infty)$ a chosen "loss" function.

In the facial recognition scenario, the set $\mathscr{Y}$ could be a finite set $\{1,2,\ldots,p\}$ of integers representing a fixed group of people to be identified, and the set $\mathscr{W} = \{1,2,\ldots,N\} \times \{1,2,\ldots,N\} \times \{0,1,2,\ldots,255\}$ where a photograph has $N \times N$ pixels each taking a color value in the set $\{0,1,2,\ldots,255\}$. A popular choice for the model $m(\cdot;\cdot)$ is the *linear model*

$$m(w;x) = x^T \tilde{w},\ \tilde{w}^T = (w^T, 1).$$

Popular choices for the loss function $\ell(\cdot,\cdot)$ include the logistic loss $\ell(y',y) = \log(1+\exp(-yy'))$, the hinge loss $\ell(y',y) = \max(0,1-yy')$, and the zero-one loss $\ell(y',y) = \mathbb{1}_{\{y' \neq y\}}$.

The stochastic optimization problem is then to identify the parameters $x \in \mathbb{R}^d$ that minimize the expected loss, where the expectation is taken with respect to the random objects (W,Y). Formally, we would like to solve

$$\begin{aligned} &\min\ f(x) = \mathbb{E}\left[\ell\left(m(W;x),Y\right)\right] = \int_{\mathscr{W}\times\mathscr{Y}} \ell\left(m(w;x),y\right)\mathbb{P}\left(d(w,y)\right) \\ &\text{s.t. } x \in \mathbb{R}^d. \end{aligned} \tag{3}$$

Of course, the objective function f cannot be directly observed. Instead, it can be estimated by using observed data $(W_i,Y_i), i = 1,2,\ldots,n$ and an implicit or explicit model for the probability measure $\mathbb{P}$.

The difficulty of solving the optimization problem in (3) depends on the choices made for the model m and the loss ℓ. Specifically, it depends on whether these choices result in a convex optimization problem, and whether direct gradients or sub-gradients are easily available. For example, the use of a hinge loss ℓ results in a convex objective function f, but the use of a zero-one loss results in a non-convex objective. Similarly, deep neural networks, as noted by Bottou et al. (2016), are essentially highly non-linear and non-convex models m that lend themselves to the easy construction of direct gradients through *back propagation*.

The stochastic optimization problem formulated in (3) using a parametrized model and a loss function subsumes a large number of other contexts including regression, compressed sensing, and matrix completion in numerous real-world scenarios (see Bubeck 2011 for more on this issue).

2.2 Example 2: Portfolio Optimization

The classical Markowitz portfolio optimization problem (Markowitz et al. 2000) in finance seeks proportions $x = (x_1,x_2,\ldots,x_d)$ of a given budget to be allocated across d assets in a given financial portfolio to maximize a combination of the expected return and variance over a given time horizon $[0,T]$. The asset price movement over the time interval $[0,t]$ is assumed to be governed by a probability model, e.g., geometric Brownian motion (Glasserman 2004).

Formally, suppose $Z_t \in \mathbb{R}^d, t \in [0,T]$ represents the asset price process for d assets in a portfolio, and suppose $\eta > 0$ is a *risk aversion* parameter for a user. Then, the Markowitz portfolio stochastic optimization problem can be written as

$$\begin{aligned} &\min\ f(x) = \mu^T x - \eta\, x^T \Sigma x, \\ &\text{s.t. } \sum_{i=1}^{d} x_i = 1, x_i \geq 0. \end{aligned}$$

where $\mu = \mathbb{E}[Z_T]$ and $\Sigma = \mathrm{Var}(Z_T)$ are the mean and covariance of Z_T.

Since the returns Z_t are usually the result of detailed models of evolution of an asset over time (e.g., see Chapter 3 in Glasserman 2004), the quantities μ and Σ are not known in closed form but estimators $\hat{\mu}_n$, $\hat{\Sigma}_n$ of μ and Σ, respectively, can be constructed using Monte Carlo simulation. In fact, since the function f is quadratic and concave, estimators for its first derivative $\mu - 2\eta\Sigma x$ and second derivative $2\eta\Sigma$, are readily available.

The portfolio optimization problem exemplifies numerous contexts where the underlying objective function is very well behaved and lends itself to the easy construction of unbiased derivative estimators that are of immense value within algorithms for stochastic optimization. This is in contrast to many simulation settings where such derivative estimators are not available directly.

3 SGD AND VARIANTS FOR SMOOTH STOCHASTIC OPTIMIZATION

In what follows, we present modern variants of SGD that are effective at solving Problem P when the function f is smooth. Effectiveness of an algorithm is understood here in the sense of finite-time performance and in the sense of enjoying good complexity rates. For historical importance and perspective, we start

with basic SGD and its more recent variant mini-batch SGD. This is followed by dynamic and adaptive sampling SGD methods, arguably the most popular implementations.

3.1 Basic SGD

The basic SGD algorithm (Robbins and Monro 1951) proceeds by taking a step of size $\alpha_k \geq 0$ along the negative gradient estimate $-g(X_k,1)$. As the notation makes explicit, the gradient estimate $g(X_k,1)$ is obtained using a unit sample size. A formal listing of basic SGD appears in Figure 1.

1:	Initialize X_0
2:	Obtain stochastic gradient $g(X_k,1)$
3:	Set $X_{k+1} \leftarrow \Pi_X\left(X_k - \alpha_k g(X_k,1)\right)$
4:	Set $k \leftarrow k+1$
5:	Go to Step 1

Figure 1: Basic SGD algorithm (Algorithm 1).

The following result characterizes the convergence of the iterates generated by the basic SGD algorithm of Figure 1 under the assumption that the function $F(\cdot,\xi)$ is convex.

Theorem 1 In Problem P, let the set X be convex and closed, let the optimal set X^* of Problem P be non-empty, and let the function $F(\cdot,\xi)$ be convex on $\mathbb{R}^n$ for each ξ. Also, let the function f be L-smooth, that is, f is differentiable with gradient $\nabla f(\cdot)$ satisfying $\|\nabla f(x)-\nabla f(y)\| \leq L\|x-y\|$ for all $x,y \in \mathbb{R}^n$. Let α_k denote the step size at step k. Furthermore, let the following additional assumptions hold.

$$\alpha_k \geq 0 \ \forall k, \ \sum_{k=1}^{\infty} \alpha_k = \infty, \ \sum_{k=1}^{\infty} \alpha_k^2 < \infty, \tag{4}$$

and

$$\sum_k \alpha_k^2 \, \mathbb{E}[\|g(x_k,1)-\nabla f(x_k)\|^2 | \mathscr{F}_k] < \infty. \tag{5}$$

Then, $\lim_k \inf_{x \in X^*}\{\|x_k - x\|\} = 0$ *w.p.*1.

Theorem 1 appears in Yousefian et al. (2012); a proof of Theorem 1 follows using simple arguments that are standard in the analysis of optimization algorithms for convex smooth functions.

The above theorem guarantees convergence to the unique optimum x^* when, for instance, the step size is chosen as $\alpha_k = c/k$, $c > 0$ and the sampling of the gradient is performed in such a way that there is not too much accumulation of bias across iterations. The latter stipulation is encoded in the third assumption of the theorem.

Basic SGD is important for having laid down a simple algorithmic framework. However, it was quickly realized that, depending on the extent of "noise," even the optimal step size choice $\alpha_k = c/k$ might result in steps that are "too short." Perhaps more importantly, it was realized that SGD simply did not have a recipe for choosing an appropriate value of c.

Three aspects of the result in Theorem 1 are noteworthy. First, in order for the iterates generated by Algorithm 1 to converge, the step size α_k has to diminish to zero, and slow enough, as encoded in the assumption appearing in (4). This is a clear departure from the deterministic context where under similar conditions on f, the gradient algorithm will fixed step converge to the optimal set X^* if the step size is small enough. Second, as noted in Yousefian et al. (2012), the assumption appearing in (5) is satisfied if, for example, the error $\|g(x_k,1)-\nabla f(x_k)\|$ is uniformly bounded by a constant. However, a uniform bound on $\|g(x_k,1)-\nabla f(x_k)\|$ is often violated since the error $\|g(x_k,1)-\nabla f(x_k)\|$ is frequently proportional to $\|\nabla f(x_k)\|$. To cover such cases, a generalization of Theorem 1 using an assumption on the "spatial growth" of the error $\|g(x_k,1)-\nabla f(x_k)\|$ is obtainable using the method outlined in Polyak (1987).

Third, we emphasize that Theorem 1 assumes that the sample-path functions $F(\cdot,\xi)$ are convex for each ξ. Numerous variations that do not assume the convexity of $F(\cdot,\xi)$ are available through standard references on stochastic approximation, e.g., Kushner and Yin (2003).

We next provide a result that provides a path to proving the convergence rate of SGD methods. We list this result assuming that the function f is λ-strongly convex.

Theorem 2 In Problem P, let the set X be convex and closed, and let the function $F(\cdot,\xi)$ be convex on $\mathbb{R}^n$ for each ξ. Also, let the function f be L-smooth and λ-strongly convex with a unique minimum attained at $x_* \in X$. Let α_k denote the step size at step k. Furthermore, let the following additional assumptions hold.

$$\alpha_k \geq 0 \ \forall k, \ \sum_{k=1}^{\infty} \alpha_k = \infty, \ \sum_{k=1}^{\infty} \alpha_k^2 < \infty;$$

and

$$\mathbb{E}[\|g(x_k,1) - \nabla f(x_k)\|^2 | \mathscr{F}_k] < \nu^2 \ \forall k.$$

Then, $\lim_k \mathbb{E}\left[\|x_k - x_*\|^2\right] = 0$, and for every $\varepsilon > 0$,

$$\mathbb{P}\left(\|x_j - x_*\| \leq \varepsilon \text{ for all } j \geq k\right) \leq 1 - \frac{1}{\varepsilon}\left(\mathbb{E}\left[\|x_k - x_*\|^2\right] + \nu^2 \sum_{i=k}^{\infty} \alpha_i^2\right). \tag{6}$$

The condition $\sum_{k=1}^{\infty} \alpha_k = \infty$, $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$ on the step sizes that appears in Theorem 1 and Theorem 2 is satisfied by the choice $\alpha_k = c/k, c > 0$. In fact, it can be shown using (6) that the choice $\alpha_k = c/k, c > 0$ attains the fastest possible convergence rate to within a constant factor, as long as $2c\lambda > 1$. Basic SGD is important for having laid down a simple algorithmic framework. However, it was quickly realized that, depending on the extent of "noise," even the optimal step size choice $\alpha_k = c/k$ might result in steps that are "too short." Perhaps more importantly, it was realized that SGD simply did not have a recipe for choosing an appropriate value of c. Implementers thus executed SGD with a fixed step size $\alpha_k = \alpha$, and not so surprisingly, such an algorithm does not converge to a stationary point. Instead, the typical behavior involved a rapid descent to some vicinity of a first-order stationary point of f, followed a random walk around the first-order critical point. The following theorem, described and proved by Bottou et al. (2016), characterizes the expected behavior of SGD with fixed step size.

Theorem 3 Suppose the objective function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth and λ-strongly convex with unique minimum $x^* \in \mathbb{R}^p$. Also, let $\mathbb{E}[\|g(x_k,1)\|^2] \leq M + M_g\|\nabla f(x_k)\|^2$ and $\mathbb{E}[g(x_k,1)^T \nabla f(x_k)] \geq \mu\|\nabla f(x_k)\|^2$ for some $M, M_g, \mu > 0$. If SGD is executed with the fixed step size $\alpha_k = \alpha$ satisfying $0 < \alpha \leq \lambda/LM_g$, then SGD has the iteration complexity

$$\mathbb{E}\left[f(x_k) - f(x^*)\right] \leq \frac{\alpha LM}{2\lambda\mu} + (1 - \alpha\lambda\mu)^{k-1}\left(f(x_1) - f(x^*) - \frac{\alpha LM}{2\lambda\mu}\right).$$

Notice that Theorem 3 implies that the iterates approach x^* exponentially fast (in k) but do not make any further progress. This makes intuitive sense, as a large step size makes it impossible to approach the point x^* beyond a fixed distance in any systematic manner. The analogue of Theorem 3 for diminishing step sizes α_k is stated as follows. A "process convergence" version of the above theorems characterizes convergence to an Ornstein-Uhlenbeck process (Asmussen and Glynn 2007) under certain scaling.

Theorem 4 Suppose the objective function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth and λ-strongly convex with unique minimum $x^* \in \mathbb{R}^p$. Also, let $\mathbb{E}[\|g(x_k,1)\|^2] \leq M + M_g\|\nabla f(x_k)\|^2$ and $\mathbb{E}[g(x_k,1)^T \nabla f(x_k)] \geq \mu\|\nabla f(x_k)\|^2$ for some $M, M_g, \mu > 0$. Then, SGD executed with the step size $\alpha_k = \beta/(\gamma + k)$ where $\beta\,\lambda\,\mu > 1$, $\alpha_1 LM_g \leq \mu$, and $\gamma > 0$ has the iteration complexity

$$\mathbb{E}\left[f(x_k) - f(x^*)\right] \leq \frac{\max\left\{\frac{\beta^2 LM}{2(\beta\lambda\mu - 1)}, (\gamma+1)(f(x_0) - f(x^*))\right\}}{\gamma + k}.$$

As noted earlier, the complexity $\mathcal{O}(1/k)$ reported by Theorem 4 is optimal "to within a constant" (see Section 3.5). Unfortunately, however, such optimality does not guarantee good practical performance especially since its choice in basic SGD takes no cognizance of the initial optimality gap $f(x_0) - f(x^*)$. In other words, a poor starting point x_0 demands a large β while a good starting point will probably benefit more from a small β. Also notice from Theorem 4 that the ratio L/λ, called the condition number of the function f, decides the complexity in a crucial way, echoing the deterministic gradient method (Nesterov 2004). Large L/λ values imply poor conditioning and, therefore, more opportunity for SGD to take steps that are not productive in the sense of decreasing the objective function value.

3.2 Mini-batch SGD

Recall that the context of Problem P is such that we have access only to "noisy" gradients. This implies then, roughly speaking, that more noise in the observations should lead to more unproductive steps by SGD, leading one to wonder if there are benefits to using larger samples at each iteration. In other words, instead of just using $g(X_k, 1)$, we could employ a *mini-batch* SGD, where at each iteration we use a sample average of m noisy gradients. Specifically, mini-batch SGD is the basic SGD iteration that uses the gradient approximation $g(X_k; m)$. A basic version of the mini-batch algorithm appears as Algorithm **??**.

1: Initialize x_0
2: Choose batch size $m \in \mathbb{N}$
3: Generate $\xi_i, i = 1, 2, \ldots, m$ independently and construct $g(X_k, m)$
4: Set $X_{k+1} \leftarrow \Pi_X\left(X_k - \alpha_k g(X_k, m)\right)$
5: Go to Step 3

Figure 2: Mini-Batch SGD algorithm (Algorithm 2).

Given Theorem 4, the complexity of the mini-batch algorithm, easy to guess, is

$$\varepsilon_k \leq \frac{\max\left\{\frac{\beta^2 LM}{2m(\beta c\mu - 1)}, (\gamma+1)(f(x_0) - f(x^*))\right\}}{\gamma + k}.$$

It should be noted that while the mini-batch SGD is used widely, it is of little value over and above basic SGD when the mini-batch size m is not chosen based on any available information on the constant M. Practical implementations will usually involve a pilot run intended to obtain a rough sense of M, followed by an appropriate choice of m. In fact, most practical implementations of mini-batch SGD involve some sort of manual updating of the mini-batch size m. As noted by Bottou et al. (2016), the real advantages of mini-batch SGD are in a distributed or parallel computing context, where the massive parallelism of the mini-batch average computation can be exploited.

3.3 SGD with Dynamic and Adaptive Sampling

A key issue that governs efficiency of SGD is the extent of sampling to obtain gradient estimates. The rudimentary SGD, outlined in Section 3.2, leans on one extreme, where a gradient estimate is obtained using exactly one observation during each iteration. That is, the sample size during each iteration of SGD is set to unity. The mini-batch SGD, also outlined in Section 3.2, is an attempt to correct inefficiencies in SGD due to the choice of the sample size, sets the sample size during each iteration to $m > 0$, where m is some fixed positive integer that needs to be chosen.

A natural question to ask in response to the strategy adopted in mini-batch SGD pertains to *dynamic* and *adaptive* sampling. Specifically, why should the sample size be fixed across iterations? Why not choose sample sizes in response to the proximity to a critical point? Dynamic sampling methods implicitly ask

Require: Initial iterate w_0, initial sample $\mathscr{S}_0$, and constant $\theta \in (0,1)$.
  set $k = 0$
  repeat
    compute $d_k = \nabla J_{\mathscr{S}_k}(w_k)$
    set steplength α_k (using line search or as a fixed step)
    set $k = k+1$
    choose $\mathscr{S}_k$ such that $|\mathscr{S}_k| = |\mathscr{S}_{k-1}|$
    compute sample variance
  until a convergence condition is satisfied.

Figure 3: Dynamic Sampling Gradient Algorithm (Algorithm 3).

these questions and use a pre-specified sequence of sample sizes $\{m_k\}$ across SGD iterations. The sample sizes generally diverge, that is, $m_k \to \infty$ as $k \to \infty$, to reflect the need for better estimation as the iterates tend closer to a critical point.

Remark 1 The phrases *adaptive sampling* (Hashemi et al. 2018; Hashemi et al. 2014; Bollapragada et al. 2018), *dynamic sampling* (Byrd et al. 2012), *variable sample size* (Deng and Ferris 2009; Homem-de-Mello 2003), and *retrospective approximation* (Chen and Schmeiser 2001; Pasupathy 2010; Pasupathy and Schmeiser 2009) have all been used in the literature to reflect similar if not identical ideas. In this paper, we have been careful to use only the two phrases *dynamic sampling* and *adaptive sampling*. The former phrase is used to describe a sampling strategy in SGD iterations that uses a fixed sequence of sample sizes that is a function of the iteration number. Adaptive sample sizes, by contrast, do not use a pre-determined sequence; the sample sizes are instead determined "on the fly" and are a function of the observed algorithm trajectory. Adaptive sample sizes are thus random while dynamic samples sizes are fixed.

Discounting the ideas of retrospective approximation (Pasupathy 2010; Kim et al. 2015) and variable sample sizing (Deng and Ferris 2009; Homem-de-Mello 2003; Bayraksan and Morton 2009) that have been detailed in other contexts, the earliest mention of dynamic sampling appears to be in Bertsekas and Tsitsiklis (1996). This idea is developed to a fuller extent in Byrd et al. (2012) and Friedlander and Schmidt (2012). In Byrd et al. (2012), for instance, the Dynamic Sampling Gradient Algorithm (see Figure 3) is presented where the sample size m_k during the kth iteration of the SGD iteration is obtained as the smallest sample size such that the estimated standard error of the estimated gradient is no more than the product of a constant θ and the norm of the estimated gradient. This sample sizing rule is formally stated as

$$m_k = \arg\inf \left\{ |\mathscr{S}| : \frac{\sqrt{\|\widehat{\mathrm{Var}}(\hat{\nabla}\ell(w_k))\|_1}}{\sqrt{|\mathscr{S}|}} \leq \theta \|\nabla J_{\mathscr{S}}\|_2 \right\}, \tag{7}$$

where $\mathscr{S}$ is the set of (gradient) observations sampled, the constant $\theta \in (0,1)$, and $\widehat{\mathrm{Var}}(\hat{\nabla}\ell(w_k))$ is the sample variance of the gradient estimate $\hat{\nabla}\ell(w_k)$ constructed from the set $\mathscr{S}$ of gradient samples. The variance estimate $\widehat{\mathrm{Var}}(\hat{\nabla}\ell(w_k))$ will differ depending on whether sampling is iid and whether the underlying population is assumed to be finite; hence, we have deliberately chosen to not provide an explicit expression for $\widehat{\mathrm{Var}}(\hat{\nabla}\ell(w_k))$. To reiterate, the salient feature of Algorithm 3 is the sample sizing rule (7) devised explicitly to keep the sampling error in the gradient estimate and a measure of proximity to a critical point in a fixed proportion.

The convergence and work complexity results in Byrd et al. (2012) correspond to an idealized version of Algorithm 3 that uses the simplified sampling rule

$$m_k = \lceil a^k \rceil \text{ for some } a > 1. \tag{8}$$

instead of the adaptive sampling rule in (7). The main convergence and complexity result in Byrd et al. (2012) is as follows.

Theorem 5 Suppose the function f is L-smooth and λ-strongly convex, and attains its minimum $w^* = \arg\inf_{x\in R^d}$. Then, the sequence $\{w_k\}$ generated by Algorithm 3 implemented with the sample size rule in (8) satisfies the following.

$$\begin{gathered}\mathbb{E}[f(w_k) - f(w^*)] \leq C\rho^k \text{for all } k \geq 1, \\ \text{where } \rho = \max\{1-\lambda/4L, a^{-1}\} < 1 \text{ and } C = \max\{f(w_0) - f(w^*), 2\sigma^2/\lambda\}.\end{gathered} \tag{9}$$

Also, the total number of gradient evaluations needed to obtain an ε-optimal solution, that is, a solution w_k satisfying $f(w_k) - f(w^*) \leq \varepsilon$, is $\mathcal{O}\left(\frac{L}{\lambda\varepsilon}\max\{f(w_0) - f(w^*), 2\sigma^2/\lambda\}\right)$.

Three aspects of Theorem 5 are important. First, the theorem is proved for smooth and strongly convex functions. As we shall see, the corresponding complexity will be higher without the guaranteed presence of strong convexity. Second, afforded by structural conditions on f, the result holds for all $k \geq 1$ unlike typical results that one tends to see in the literature on stochastic approximation (Kushner and Yin 2003). Third, Theorem 5 assumes that a fixed sample-size sequence $m_k = \lceil a^k \rceil$ for some $a > 1$ is in effect.

As noted earlier, Theorem 5 pertains to an idealized version of Algorithm 3 that uses a fixed sample-size sequence. For effective implementation, Byrd et al. (2012) suggest using Algorithm 3 with the adaptive sampling rule in (7) but with the direction-finding Step 3 of Algorithm 3 augmented with a Hessian through the Newton-CG method, and the steplength α_k in Step 4 of Algorithm 3 obtained using a line search that satisfies the Wolfe conditions (Nocedal and Wright 2006). Two issues are noted as especially important when incorporating the Hessian into the direction finding step of Algorithm 3. First, to avoid excessive sampling, the sample size used for constructing the Hessian is such that it is retained at a constant factor R of the sample size obtained from the sampling rule in (7). Second, the Hessian should not be computed explicitly; instead, using what is called the Hessian-free *conjugate gradient* method (Nocedal and Wright 2006), a Hessian vector product is formed and incorporated directly. The "implementable" version of Algorithm 3 is listed as Algorithm 5.2 and Algorithm 5.1 in Byrd et al. (2012).

Along the same lines as the idealized variant of Algorithm 3, Friedlander and Schmidt (2012) propose what is called an *incremental gradient* method. There, the growth condition on the sample sizes m_k is specified indirectly, through an appropriate condition on the expected rate at which the error in the gradient estimate decays to zero. For example, Friedlander and Schmidt (2012) prove the result on the behavior of SGD's iterates presented in the following theorem.

Theorem 6 Suppose the objective function f in Problem (P) is L-smooth and λ-strongly convex. Let

$$x_{k+1} = x_k - \alpha_k\, g(x_k, m_k), \qquad k = 0, 1, 2, \ldots \tag{10}$$

where m_k is such that $B_k := \mathbb{E}[\|g(x_k, m_k) - \nabla f(x_k)\|^2]$ satisfies

$$\lim_{k\to\infty} \frac{B_{k+1}}{B_k} \leq 1.$$

Then, for any $\varepsilon > 0$,

$$\mathbb{E}[f(x_k) - f(x^*)] \leq (1-\lambda/L)^k\,[f(x_0) - f(x^*)] + \mathcal{O}(C_k), \tag{11}$$

where $C_k = \max\{B_k, (1-\lambda/L+\varepsilon)^k\}$.

A particular construction of the sequence $\{B_k\}$ (albeit one that depends on a few unknown constants) is also presented in Friedlander and Schmidt (2012) to ensure that the upper bound in (11) converges at a linear rate. An implementable version of the recursion in (10) that incorporates a Hessian approximation via limited-memory BFGS (Nocedal and Wright 2006), and a step-length obtained on the basis on a line search with the Armijo condition (Nocedal and Wright 2006) is recommended in Friedlander and Schmidt (2012).

3.4 Iterate-Averaging Methods

To motivate iterate-averaging as a general idea, as was noted in Section 3.2, the basic SGD is optimal for L-smooth, c-strongly convex functions f with the step size choice $\alpha_k = \theta/k$, whenever $\theta > (2c)^{-1}$. Hence, as lucidly illustrated by Nemirovski et al. (2009), if we execute SGD on the L-smooth, c-strongly convex function $f(x) = x^2/10$ with $\alpha_k = \theta/k$ and $\theta > (2c)^{-1} = 2.5$, the resulting iterations will exhibit the optimal complexity rate $\mathscr{O}(k^{-1})$. However, if $\theta = 1$, it can be shown using simple algebra that the resulting complexity rate deteriorates to $\mathscr{O}(k^{-0.2})$. In other words, shorter steps $\alpha_k = k^{-1}$ resulting from any mis-estimation of the strong convexity constant c causes significant degradation in performance, as compared to the optimal complexity $\mathscr{O}(k^{-1})$. Such degradation can become even more pronounced when the underlying function is not strongly convex, as demonstrated through another example by Nemirovski et al. (2009).

Our illustration of the behavior of basic SGD on the function $f(x) = x^2/10$ is meant to convey the idea that step sizes cannot be chosen "too short" if the optimal complexity is to be guaranteed. However, "long steps" have been known to make SGD's trajectory "more noisy" (Nemirovski et al. 2009). Iterate-averaging is a technique that is intended as a balance between these extremes. Loosely speaking, iterate-averaging allows for using long steps within the basic SGD iteration, but then averages the resulting iterates offline, to account for the increased "noise" in the iterates. The general structure for such iterate-averaged SGD (for first-order oracles) is

$$\begin{aligned} x_{k+1} &= x_k - \alpha_k g(x_k, 1); \\ \tilde{x}_{k+1} &= \frac{1}{k+1}\sum_{i=1}^{k+1} x_i \; k = 0, 1, 2, \ldots. \end{aligned} \tag{12}$$

Iterate-averaging as an idea first seems to have appeared in Nemirovski and Yudin (1983) for a setting more general than Euclidean spaces, but was developed further by Polyak and Juditsky (1992) and by Nemirovski et al. (2009). A precise complexity rate result for the iterates resulting from (12) is given by Polyak and Juditsky (1992), and is stated here in a slightly modified form.

Theorem 7 (Polyak and Juditsky, 1992) Let the function f be L-smooth, twice differentiable, and c-strongly convex with a unique minimum attained at x^*. Let $B(x)$ denote the $p \times p$ matrix of second derivatives of f at $x \in \mathscr{D}$. Let the noise process $\varepsilon_k = G(x_k) - \nabla f(x_k)$ satisfy $\mathbb{E}[\varepsilon_k | \mathscr{F}_{k-1}] = 0$ and $\mathbb{E}[\|\varepsilon_k\|^2 | \mathscr{F}_{k-1}] + \|\nabla f(x_k)\|^2 \leq K_2(1 + \|x_{k-1}\|^2)$ almost surely for some $K_2 > 0$. Also, let $\mathbb{E}[\varepsilon_k \varepsilon_k^T | \mathscr{F}_{k-1}] \xrightarrow{p} S$, where S is a positive definite matrix. Let the step size sequence $\alpha_k = \alpha k^{-\beta}$, where $1/2 < \beta < 1$. Then, $\tilde{x}_{k+1} \to x^*$ almost surely, and $\sqrt{k}(x_k - x^*) \xrightarrow{d} N(0, V)$, where $V = B(x^*)^{-1} S (B(x^*)^{-1})^T$.

The crucial point to note about Polyak and Juditsky's iterate-averaging is that the convergence rate characterized in Theorem 7 is the best possible in an information-theoretic sense (see Section 3.5). And, this best rate, crucially dependent on the second derivative of the function f at the point x^*, is attained with no explicit estimation of the second-derivative (Hessian) matrix. Iterate-averaging and dynamic adaptive sampling methods are intimately connected and essentially do the same thing, but in different ways. The question of when to start averaging in iterate-averaging methods is a question of great practical importance about which little is currently known.

As noted earlier, a popular and general SGD technique that averages iterates akin to (12) is what has been called mirror descent (Nemirovski and Yudin 1983; Nemirovski et al. 2009). Apparently, mirror descent was introduced as a generalization of the gradient descent iteration for non-Euclidean spaces, where the x_k iterate and the gradient $\nabla f(x_k)$ may not be in the same space, thus rendering an iteration such as $x_k - \gamma \nabla f(x_k)$ meaningless. To understand mirror descent, let's restrict our brief discussion here to a compact convex set $X \subset \mathbb{R}^d$ equipped with an arbitrary norm $\|\cdot\|$, even though mirror descent is designed for non-Euclidean spaces. The idea of mirror descent is simple in principle. Since an iteration such as $x_k - \gamma \nabla f(x_k)$ is meaningless, mirror descent performs the SGD iteration on the dual space, inverts back

to the primal space, and then projects to the original space X to ensure feasibility. To make this operation precise, mirror descent defines a mirror map $\Phi : \mathscr{D} \to \mathbb{R}$ that is strictly convex and differentiable, has the gradient taking all possible values in $\mathbb{R}^d$ and diverging on the boundary of $\mathscr{D}$, that is, $\lim_{x\to\partial\mathscr{D}} \|\nabla\Phi(x)\| = \infty$. The domain $\mathscr{D}$ of the mirror map is such that the original set X is a subset of its closure. Several example mirror maps have been presented; see for example Section 4.3 in (Bubeck 2015). Also, the Bregman divergence

$$D_\Phi(x,y) = \Phi(y) - \Phi(x) - \nabla\Phi(x)^T(y-x)$$

is used to accomplish the projection

$$\Pi_X^\Phi(y) := \arg\min D_\Phi(x,y)$$

in the primal space X. Given an iterate x_k, mirror descent first obtains the image $\Phi(x_k)$ of x_k through the mirror map $\Phi(\cdot)$, then performs the gradient step $\Phi(x_k) - \alpha_k \nabla g(x_k,m)$ to obtain the image $\nabla\Phi(y_{k+1})$ in the dual space, and inverts $\nabla\Phi(y_{k+1})$ to obtain y_{k+1}, which is finally projected back into the primal space. Formally, the iteration is written as

$$\begin{aligned}\nabla\Phi(y_{k+1}) &= \nabla\Phi(x_k) - \alpha_k g(x_k,m)\\ x_{k+1} &\in \Pi_X^\Phi(y_{k+1}).\end{aligned}$$

The "returned solution" $\tilde{x}_i^k$ after k steps, obtained by "non-uniform averaging" of the iterates $x_i, x_{i+1}, \ldots, x_k$, is given as

$$\begin{aligned}\tilde{x}_i^k &= \frac{\sum_{t=i}^k \gamma_t x_t}{\sum_{t=i}^k \gamma_t};\\ \gamma_t &= \frac{\theta D_{\Phi,X}\sqrt{\alpha}}{M_*\sqrt{t}}, \quad t = 1,2,\ldots,k\end{aligned}$$

where $\theta > 0$ is a chosen constant, the diameter

$$D_{\Phi,X} = \sqrt{2}\sup_{x\in X^o, z\in X}\left(\Phi(z) - \Phi(x) - \nabla\Phi(z)^T(x-Z)\right)^{1/2},$$

the constant α is the strong-convexity parameter of the function Φ, and $M_* \geq \sup_{x\in X}\mathbb{E}\left[\|g(x,m)\|_*^2\right]$.

As is noted by Nemirovski et al. (2009), mirror descent results in a convergence rate that is robust with respect to mis-estimation of the strong convexity parameter of the function f. The main complexity result for mirror descent given by them is

$$\mathbb{E}\left[f(\tilde{x}_i^k) - f(x^*)\right] \leq \frac{D_{\Phi,X}M_*}{\sqrt{\alpha k}}\left[\frac{2}{\theta}\frac{k}{k-i+1} + \frac{\theta}{2}\sqrt{\frac{k}{i}}\right]. \tag{13}$$

It can be observed that the convergence rate as implied by (13) is not optimal for smooth functions, but the rate implied by (13) is optimal for non-smooth functions.

3.5 Optimal SGD Methods

By *optimal methods* we mean SGD variants that achieve the fastest achievable convergence rate. More precisely, it was shown by Tsybakov and Polyak (1990) that when Problem P has a unique solution x^*, any linear recursive estimation procedure used on Problem P satisfies

$$\mathbb{E}\left[(X_k - x^*)(X_k - x^*)^T\right] \geq V k^{-1} + o(k^{-1}), \tag{14}$$

where $V = B(x^*)^{-1}S(B(x^*)^{-1})^T$ and $B(x^*)$ is the matrix of second derivatives (Hessian) at x^* and where for two $p \times p$ matrices A and B, $A \geq B$ means $A - B$ is positive definite. Hence, any procedure that attains the bound in (14) has been deemed *optimal*. For example, the iterate-averaging procedures (Polyak and Juditsky 1992) outlined in Section 3.3, and some of their precursors (Venter 1967; Fabian 1968), attain the bound in (14) and are, hence, thought to be optimal. Of course, the bound in (14) is a lower bound and not always attainable, as happens when the underlying function f is non-smooth.

For optimizing non-smooth (expectation) functions over the compact convex set $X \subset \mathbb{R}^p$, the bound analogous to (14), given by Nemirovski and Yudin (1983), is stated as follows. Suppose f is a convex and Lipschitz-continuous function satisfying $|f(x) - f(y)| \leq M\|x - y\|$, $\forall x, y \in X$, and denote $f^* := \inf_{x \in X} f(x)$. Then, for $p \geq \mathscr{O}(1)k$ where $\mathscr{O}(1)$ is a universal constant,

$$\mathbb{E}[f(X_k) - f^*] \geq \mathscr{O}(1)Mk^{-1/2}. \tag{15}$$

The mirror descent iterate-averaging method (Nemirovski et al. 2009), outlined in Section 3.3, attains the bound in (15).

Another interpretation of optimality comes from statistics. The famous Cramér-Rao bound implies that a lower bound for any unbiased estimator $T(Y_1, Y_2, \ldots, Y_n)$ of $\theta^* \in \mathbb{R}^p$ constructed using random copies $Y_1, Y_2, \ldots, Y_n$ (not necessarily independent) of a random vector Y having density $g(\cdot; \theta^*)$ satisfies, under certain simple regularity conditions (Casella and Berger 2002, pp. 335) on g, that

$$\mathrm{Var}(T(Y_1, Y_2, \ldots, Y_n)) \geq I(\theta^*)^{-1}; \; I(\theta^*) = -\left(\mathbb{E}\left[\frac{\partial}{\partial \theta^2}\sum_{i=1}^{n} \log g(Y_i; \theta)\right]\right), \tag{16}$$

where the matrix $I(\theta)$ is called the Fisher information matrix at θ.

To interpret (16) in the context of the current tutorial, suppose the objective function f is strongly convex and twice-differentiable. Then, any unbiased estimator of x^* that is constructed from k iid observations of the zero gradient at x^* necessarily has variance that exceeds the right-hand side of (16), which in turn can be shown to coincide with Vk^{-1} appearing in (14). It is in this sense that the iterate-averaging procedures (Polyak and Juditsky 1992) outlined in Section 3.3, and some earlier methods (Venter 1967; Fabian 1968) placing additional stringency are said to be efficient. Of course, the question of whether there exist biased estimators having mean squared errors lower than the right-hand side of (16) can be answered to be negative except in pathological conditions on the dependence structure of the sample-paths. We are aware of no analogues to (16) in the non-smooth and non-strongly convex contexts.

In Table 1, we list optimal algorithms for different contexts. The algorithm by Polyak and Juditsky and the mirror-descent algorithm are iterate-averaging algorithms discussed in Section 3.3. The RSAG algorithm of Ghadimi and Lan (2013) is an accelerated gradient descent algorithm that is not discussed in this tutorial; neither are methods for non-convex stochastic optimization. The optimal bound for the smooth non-convex context is not known.

Table 1: Optimal algorithms for various contexts.

	Non-Convex	Convex	Strongly Convex
Smooth	?	RSAG	Polyak and Juditsky
Non-Smooth	✗	Mirror-Descent	Mirror-Descent

4 CONCLUDING REMARKS

Solutions to the stochastic optimization problem as posed in Problem P lie at the heart of many machine learning and big data settings. Modern variants of SGD form the principal solutions to Problem P, and over the last two decades, impressive progress has been made in gaining a deeper understanding of the behavior of

these algorithms, especially in terms of their complexity rates and their practical performance. Perhaps more importantly, there has been a proliferation of (sometimes dramatic) examples where difficult classification and image processing problems modeled using deep learning have been solved quite satisfactorily using one of the variants outlined in this tutorial. As of this writing, a search is on for an effective way to use second-order information within these variants.

REFERENCES

Asmussen, S., and P. W. Glynn. 2007. *Stochastic Simulation: Algorithms and Analysis*. New York: Springer.

Bandeira, A. S., K. Scheinberg, and L. N. Vicente. 2014. "Convergence of Trust-Region Methods Based on Probabilistic Models". *SIAM Journal on Optimization* 24(3):1238–1264.

Bayraksan, G., and D. P. Morton. 2009. "A Sequential Sampling Procedure for Stochastic Programming". *Operations Research* 59(4):898–913.

Bertsekas, D., and J. Tsitsiklis. 1996. *Neuro-dynamic Programming*. Cambridge, Massachusetts: Athena Scientific.

Bollapragada, R., R. Byrd, and J. Nocedal. 2018. *Adaptive Sampling Strategies for Stochastic Optimization*. https://arxiv.org/pdf/1710.11258.pdf, accessed July 10, 2018.

Borkar, V. S. 1997. "Stochastic Approximation with Two Time Scales". *Systems and Control Letters* 29:291–294.

Bottou, L., F. Curtis, and J. Nocedal. 2016. *Optimization Methods for Large-Scale Machine Learning*. https://arxiv.org/abs/1606.04838, accessed July 10, 2018.

Bubeck, S. 2011. "Introduction to Online Optimization". https://www.microsoft.com/en-us/research/wp-content/uploads/2017/01/BubeckLectureNotes.pdf, accessed July 10, 2018.

Bubeck, S. 2015. "Convex Optimization: Algorithms and Complexity". *Foundations and Trends in Machine Learning* 8(3–4):231–358.

Byrd, R. H., G. M. Chin, J. Nocedal, and Y. Wu. 2012. "Sample Size Selection for Optimization Methods for Machine Learning". *Mathematical Programming, Series B* 134:127–155.

Casella, G., and R. L. Berger. 2002. *Statistical Inference*. 2nd ed. Pacific Grove, CA: Duxbury.

Chang, K., J. Hong, and H. Wan. 2013. "Stochastic Trust-Region Response-Surface Method (STRONG) – A New Response-Surface Framework for Simulation Optimization". *INFORMS Journal on Computing* 25(2):230–243.

Chen, H., and B. W. Schmeiser. 2001. "Stochastic Root Rinding via Retrospective Approximation". *IIE Transactions* 33(3):259–275.

Chen, R., M. Menickelly, and K. Scheinberg. 2016. "Stochastic Optimization Using a Trust-Region Method and Random Models". *Mathematical Programming* 169(2):447–487.

Defazio, A., F. Bach, and S. Lacoste-Julien. 2014. *SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives*. https://arxiv.org/pdf/1407.0202v3.pdf, accessed July 10, 2018.

Deng, G., and M. C. Ferris. 2009. "Variable-number Sample-path Optimization". *Mathematical Programming* 117(1–2):81–109.

Duchi, J., E. Hazan, and Y. Singer. 2011. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". *Journal of Machine Learning Research* 12:2121–2159.

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

Friedlander, M., and M. Schmidt. 2012. "Hybrid Deterministic-Stochastic Methods for Data Fitting". *SIAM Journal on Scientific Computing* 34(3):A1380–A1405.

Ghadimi, S., and G. Lan. 2013. "Stochastic First- and Zeroth-Order Methods for Nonconvex Stochastic Programming". *SIAM Journal on Optimization* 23(4):2341–2368.

Glasserman, P. 2004. *Monte Carlo Methods in Financial Engineering*. New York: Springer.

Hashemi, F., S. Ghosh, and R. Pasupathy. 2014. "On Adaptive Sampling Rules for Stochastic Recursions". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk et al., 3959–3970. Piscataway, New Jersey: IEEE.

Hashemi, F., R. Pasupathy, and M. R. Taaffe. 2018. "The Adaptive Sampling Gradient Method: Optimizing Smooth Functions with an Inexact Oracle.". In review.

Hastie, T., R. Tibshirani, and M. Wainwright. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Boca Raton, FL: Chapman and Hall/CRC.

Homem-de-Mello, T. 2003. "Variable-sample Methods for Stochastic Optimization". *ACM Transactions on Modeling and Computer Simulation* 13(2):108–133.

Johnson, R., and T. Zhang. 2013. "Accelerating Stochastic Gradient Descent Using Predictive Variance Reduction". In *Proceedings of the 2013 NIPS Conference*, edited by C. J. C. Burges et al. December 5^{th}–10^{th}, Lake Tahoe, California. https://papers.nips.cc/paper/4937-accelerating-stochastic-gradient-descent-using-predictive-variance-reduction.pdf.

Kim, S., R. Pasupathy, and S. G. Henderson. 2015. "A Guide to Sample Average Approximation". In *Handbook of Simulation Optimization*, edited by M. Fu, Chapter 8, 207–243. New York: Springer.

Kushner, H. J., and G. G. Yin. 2003. *Stochastic Approximation and Recursive Algorithms and Applications*. New York: Springer.

Lai, T. L. 2003. "Stochastic Approximation". *The Annals of Statistics* 31(2):391–406.

Markowitz, H. M., G. P. Todd, and W. F. Sharpe. 2000. *Mean-variance Analysis in Portfolio Choice and Capital Markets*. New York: Wiley.

Nelson, B. L. 2013. *Foundations and Methods of Stochastic Simulation: A First Course*. New York: Springer.

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.

Nemirovski, A. S., and D. B. Yudin. 1983. *Problem Complexity and Method Efficiency in Optimization*. New York: Wiley.

Nesterov, Y. 2004. *Introductory Lectures on Convex Optimization: A Basic Course*. New York: Springer Science + Business Media, LLC.

Nocedal, J., and S. J. Wright. 2006. *Numerical Optimization*. Berlin: Springer.

Pasupathy, R. 2010. "On Choosing Parameters in Retrospective-Approximation Algorithms for Stochastic Root Finding and Simulation Optimization". *Operations Research* 58(4/1):889–901.

Pasupathy, R., and B. W. Schmeiser. 2009. "Retrospective-Approximation Algorithms for Multidimensional Stochastic Root-finding Problems". *ACM Transactions on Modeling and Computer Simulation* 19(2), article 5.

Polyak, B. 1987. *Introduction to Optimization*. New York: Optimization Software, Inc.

Polyak, B. T., and A. B. Juditsky. 1992. "Acceleration of Stochastic Approximation by Averaging". *SIAM Journal on Control and Optimization* 30(4):838–855.

Robbins, H., and S. Monro. 1951. "A Stochastic Approximation Method". *Annals of Mathematical Statistics* 22(3):400–407.

Schmidt, M., N. L. Roux, and F. Bach. 2013. "Minimizing Finite Sums with the Stochastic Average Gradient". Technical report, INRIA, hal-0086005.

Shalev-Shwartz, S., and T. Zhang. 2013. "Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization". *Journal of Machine Learning Research* 14:567–599.

Shapiro, A., D. Dentcheva, and A. Ruszczynski. 2009. *Lectures on Stochastic Programming: Modeling and Theory*. Philadelphia, PA: SIAM.

Shashaani, S., F. S. Hashemi, and R. Pasupathy. 2018. "ASTRO-DF: A Class of Adaptive Sampling Trust-Region Algorithms for Derivative-Free Simulation Optimization". *SIAM Journal on Optimization*. Under minor revision.

Shashaani, S., S. R. Hunter, and R. Pasupathy. 2016. "ASTRO-DF: Adaptive Sampling Trust-Region Optimization Algorithms, Heuristics, and Numerical Experience". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. K. Roeder et al., 554–565. Piscataway, New Jersey: IEEE.

Tsybakov, A., and B. T. Polyak. 1990. "Optimal Order of Accuracy of Search Algorithms in Stochastic Optimization". *Problemy Peredachi Informatsii* 26(2):45–53.

Venter, H. J. 1967. "An Extension of the Robbins-Monro Procedure". *Annals of Mathematical Statistics* 38:181–190.

Yousefian, F., A. Nedić, and U. Shanbhag. 2012. "On Stochastic Gradient and Subgradient Methods with Adaptive Steplength Sequences". *Automatica* 48(1):56–67.

AUTHOR BIOGRAPHIES

DAVID NEWTON is a Ph.D. student in the Department of Statistics at Purdue University. His dissertation addresses questions in the area of stochastic optimization. His e-mail address is newton34@purdue.edu.

RAGHU PASUPATHY is an associate professor in the Department of Statistics at Purdue University. His research interests lie broadly in Monte Carlo methods with a specific focus on simulation optimization. He is a member of INFORMS, IIE, and ASA, and serves as an associate editor for Operations Research and INFORMS Journal on Computing. His email address is pasupath@purdue.edu.

FARZAD YOUSEFIAN is an assistant professor in the school of Industrial Engineering and Management at Oklahoma State University. His research interests lie in the development of computational methods for solving stochastic optimization and equilibrium problems arising from machine learning and multi-agent systems. He is the recipient of the best theoretical paper award in the 2013 Winter Simulation Conference. His email address is farzad.yousefian@okstate.edu.