Stochastic Adaptive Quasi-Newton Methods for Minimizing Expected Values

Chaoxu Zhou [*1] **Wenbo Gao** [*1] **Donald Goldfarb** [1]

Abstract

We propose a novel class of *stochastic, adaptive* methods for minimizing self-concordant functions which can be expressed as an expected value. These methods generate an estimate of the true objective function by taking the empirical mean over a sample drawn at each step, making the problem tractable. The use of adaptive step sizes eliminates the need for the user to supply a step size. Methods in this class include extensions of gradient descent (GD) and BFGS. We show that, given a suitable amount of sampling, the stochastic adaptive GD attains linear convergence in expectation, and with further sampling, the stochastic adaptive BFGS attains R-superlinear convergence. We present experiments showing that these methods compare favorably to SGD.

1. Introduction

We are concerned with minimizing functions of the form

$$\min_{x\in\mathbb{R}^n} F(x) = \mathbb{E}_\xi f(x,\xi) \tag{1}$$

Many common problems in statistics and machine learning can be put into this form. For instance, in the empirical risk minimization framework, a model is learned from a set $\{y_1,\dots,y_m\}$ of training data by minimizing an empirical loss function of the form

$$\min_x L(x) = \frac{1}{m}\sum_{i=1}^{m} f(x,y_i) \tag{2}$$

It is easy to see that this formulation is equivalent to taking ξ to be the uniform distribution on the points $\{y_1,\dots,y_m\}$.

An objective function of the form (1) is often impractical, as the distribution of ξ is generally unavailable, making it infeasible to analytically compute $\mathbb{E}f(x,\xi)$. This can be resolved by replacing the expectation $\mathbb{E}f(x,\xi)$ by the estimate (2). The strong law of large numbers implies that the sample mean $L(x)$ converges almost surely to $F(x)$ as the number of samples m increases, provided that the samples y_i are drawn independently from the distribution of ξ. However, even the concrete problem (2) is not a good target for classical optimization algorithms, as the amount of data m is frequently extremely large. A better strategy when optimizing (2) is to consider *subsamples* of the data to reduce the computational cost. This leads to *stochastic algorithms* where the objective function changes at each iteration by randomly selecting subsamples.

Many stochastic algorithms have been proposed which use this approach for solving (2), notably *stochastic gradient descent* (SGD) (Bottou, 2010), and variance-reduced extensions of SGD, such as SVRG (Johnson & Zhang, 2013), SAG (Schmidt et al., 2013), and SAGA (Defazio et al., 2014). These methods are *first-order* methods, extending gradient descent to the stochastic setting, and the latter three (variance-reduced) methods can be shown to converge linearly for strongly convex objectives. Linearly convergent stochastic Limited Memory BFGS algorithms (Byrd et al., 2016)(Moritz et al., 2016)(Gower et al., 2016) have also been proposed. It is then natural to consider stochastic extensions of quasi-Newton and second-order methods. One such method, the Newton Incremental Method (NIM) (Rodomanov & Kropotov, 2016), combines cyclic updating of a fixed collection of functions $f(x,y_1),\dots,f(x,y_m)$ with Newton's method, and attains local superlinear convergence.

One of the key obstacles to developing stochastic extensions of quasi-Newton methods is the necessity of selecting appropriate *step sizes*. The analysis of the global convergence of the BFGS method (Powell, 1976) and other members of *Broyden's convex class* (Byrd et al., 1987) assumes that Armijo-Wolfe inexact line search is used. This is rather undesirable for a stochastic algorithm, as line search is both computationally expensive and difficult to analyze in a probabilistic setting. However, there is a special class of functions, the *self-concordant functions*, whose properties allow us to compute an *adaptive step size* and thereby avoid performing line searches. In (Gao & Goldfarb, 2016), it is shown that the BFGS method

[*]Equal contribution [1]Dept. of Industrial Engineering and Operations Research, Columbia University. Correspondence to: Chaoxu Zhou <cz2364@columbia.edu>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

(Broyden, 1967) (Fletcher, 1970)(Goldfarb, 1970)(Shanno, 1970) with adaptive step sizes converges superlinearly when applied to self-concordant functions.

In this paper, our goal is to develop a stochastic quasi-Newton algorithm for self-concordant functions. We propose an iterative method of the following form. At the k-th iteration, we draw m_k i.i.d samples $\xi_1, \ldots, \xi_{m_k}$. Define the *empirical objective function* at the k-th iteration to be

$$F_k(x) = \frac{1}{m_k} \sum_{i=1}^{m_k} f(x, \xi_i) \tag{3}$$

Let H_k be a positive definite matrix. The next step direction is given by

$$d_k = -H_k \nabla F_k(x_k)$$

and the step size by

$$t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k}$$

where

$$\alpha_k = \frac{\nabla F_k(x_k)^T H_k \nabla F_k(x_k)}{\delta_k^2} \tag{4}$$
$$\delta_k = \sqrt{\nabla F_k(x_k)^T H_k \nabla^2 F_k(x_k) H_k \nabla F_k(x_k)}$$

The motivation for this step size is described in Section 4.

A key feature of these methods is that the step size t_k can be computed analytically, using only local information, and adapts itself to the local curvature. A fixed step size η is typically used in variance-reduced first-order methods, and this step size must be determined experimentally. The theoretical analysis that has been provided for these methods is of little help in choosing η, as often η is constrained to be impractically small, and moreover, is related to unknown constants such as the Lipschitz parameter of the gradient. Furthermore, a fixed η which was effective in one regime may become ineffective as the algorithm progresses, and enters regions of varying curvature.

Our new methods are also capable of solving general problems of the form (1). This is in contrast to popular incremental-type methods such as SAG, SAGA, and NIM, which, because of their stored updating scheme, can only be applied to problems of the form (2) with a fixed data set $\{y_1, \ldots, y_m\}$. This opens up new avenues for the solutions of problems where new data can be sampled as the algorithm progresses, as opposed to having a fixed training set throughout. In this respect, our new method is akin to *Streaming SVRG* (Frostig et al., 2015), which is also able to solve (1) with similar conditions on the growth of the samples m_k.

By choosing the matrices H_k appropriately, we obtain stochastic extensions of classical methods. In particular, two choices of H_k will be of interest:

1. Taking $H_k = I$ yields the stochastic adaptive gradient descent method (SA-GD).
2. Fixing H_0, and then taking H_{k+1} to be the BFGS update of H_k, yields the stochastic adaptive BFGS method (SA-BFGS).

Our methods can be proven to converge under a suitable sampling regime. When the number of samples m_k is fixed, and large enough, both SA-GD and SA-BFGS converge R-linearly in expectation to an ϵ-optimal solution. To obtain R-superlinear convergence, the number of samples must be allowed to grow rapidly. This principle, of increasing the number of samples to match the desired rate of convergence, was previously applied to R-linear convergence in (Byrd et al., 2012). The SA-BFGS algorithm can be shown to converge *R-superlinearly almost surely* to the true optimal solution if the number of samples is increased so that m_k^{-1} converges R-superlinearly to 0.

This paper is organized as follows. In Section 2, we introduce our notation, and the technical assumptions needed for the analysis of our methods. In Section 3, we describe the relevant results from stochastic analysis which motivate our algorithms. In Section 4, we briefly describe the required theory of self-concordant functions. In Section 5, we formally define stochastic adaptive methods, and state the convergence results. In Section 6, we present preliminary numerical experiments, and conclude with a discussion in Section 7. Full proofs of the convergence theorems can be found in the supplementary materials.

2. Assumptions and Notation

The number of variables is n. We write $g(x) = \nabla F(x)$ and $G(x) = \nabla^2 F(x)$, and $g_k(x) = \nabla F_k(x)$ and $G_k(x) = \nabla^2 F_k(x)$ for the gradient and Hessian of the empirical objective function. In the context of a sequence of iterates $\{x_k\}_{k=0}^{\infty}$ generated by an algorithm, we also write g_k with no argument to denote $g_k(x_k)$, and G_k for $G_k(x_k)$. In the context of BFGS, H_k denotes the approximation of the inverse Hessian, and $B_k = H_k^{-1}$. The step direction (generally $-H_k g_k$) is denoted d_k, and the step size by t_k, so the actual step taken is $s_k = t_k d_k$.

The optimal solution of $\min_{x \in \mathbb{R}^n} F(x)$ is denoted x^*, and the optimal solution of the empirical problem $\min_{x \in \mathbb{R}^n} F_k(x)$ is denoted x_k^*. Note that x_k^* is a random variable.

Unless otherwise specified, the norm $\|\cdot\|$ is the 2-norm, or the operator 2-norm. The Frobenius norm is indicated as $\|\cdot\|_F$.

We make the following technical assumptions on $F(x)$ and $f(x, \xi)$. We will explain the motivation for these assumptions at the relevant points in the discussion.

Assumptions:

1. There exist constants $L \geq \ell > 0$ such that for every $x \in \mathbb{R}^n$ and every realization of ξ, the Hessian of f with respect to x satisfies

$$\ell I \preceq \nabla_x^2 f(x, \xi) \preceq LI$$

That is, $f(x, \xi)$ is strongly convex for all ξ, with the eigenvalues of $\nabla_x^2 f(x, \xi)$ bounded by ℓ and L. The *condition number*, denoted κ, is given by L/ℓ.

2. $F_k(x)$ is standard self-concordant for every possible sampling $\xi_1, \ldots, \xi_{m_k}$.

3. There exist compact sets $\mathcal{D}_0$ and $\mathcal{D}$ with $x^* \in \mathcal{D}$ and $\mathcal{D}_0 \subseteq \mathcal{D}$, such that if x_0 is chosen in $\mathcal{D}_0$, then for all possible realizations of the samples $\xi_1, \ldots, \xi_{m_k}$ for every k, the sequence of iterates $\{x_k\}_{k=0}^{\infty}$ produced by the algorithm is contained within $\mathcal{D}$. We use $D = \sup\{\|x - y\| : x, y \in \mathcal{D}\}$ for the diameter of $\mathcal{D}$.

Furthermore, we assume that the objective values and gradients are bounded:

$$u = \sup_{\xi} \sup_{x \in \mathcal{D}} f(x, \xi) < \infty$$
$$l = \inf_{\xi} \inf_{x \in \mathcal{D}} f(x, \xi) > -\infty$$
$$\gamma = \sup_{\xi} \sup_{x \in \mathcal{D}} \|\nabla f(x, \xi)\| < \infty$$

4. (For BFGS only) The Hessian $G(x)$ is Lipschitz continuous with constant L_H.

Note that Assumption 1 is standard when analyzing stochastic algorithms. The function $f(x, \xi)$ is commonly a loss function, and either $f(x, \xi)$ is itself strongly convex, or each $f(x, \xi)$ is weakly convex and the strong convexity is ensured by adding a quadratic regularization term to $F(x)$.

3. Stochastic Framework

Our analysis is based on a uniform convergence law, a standard technique in learning theory and empirical processes. The central idea is that $F_k(x)$ is an empirical mean estimating $F(x)$, and we can closely control the error $|F_k(x) - F(x)|$ over all $x \in \mathcal{D}$ by varying the sample size m_k. The relevant stochastic analysis can be found in (W. van der Vaart & Wellner, 1996) and (Goldfarb et al., 2017). These powerful techniques are required to analyze second-order stochastic methods, since inverting the product of a sampled Hessian and sampled gradient generates both dependence and non-linearity. In comparison, first-order stochastic methods can be analyzed by evaluating the expected value and variance of the sampled gradient. It is also useful to compare this approach with that used in (Byrd et al., 2012) and (Friedlander & Goh, 2013), where the variance of the gradient is controlled by pointwise estimates or pointwise tail bounds.

Theorem 3.1 (Corollary 1, (Goldfarb et al., 2017))**.** *Let $\delta > 0$ and $0 < \epsilon < \min\{D, \frac{\delta}{2L}\}$. Then*

$$\mathbb{P}(\sup_{x \in \mathcal{D}} |F_k(x) - F(x)| > \delta) \leq c(\epsilon) \exp\left(-\frac{m_k(\delta - 2L\epsilon)^2}{2(u-l)^2}\right)$$

where $c(\epsilon) = 2n^{n/2} D^n \epsilon^{-n}$. If $m_k \geq 3$, then

$$\mathbb{E} \sup_{x \in \mathcal{D}} |F_k(x) - F(x)| \leq C\sqrt{\frac{\log m_k}{m_k}}$$
$$\mathbb{E}|F_k(x_k^*) - F(x^*)| \leq C\sqrt{\frac{\log m_k}{m_k}}$$

where C is a constant (depending only on F) given by

$$4(|u| + |l|) n^{n/2} D^n \exp\left[-n\left(\log \frac{u-l}{2\sqrt{2}L}\right)\right] + (u-l)\sqrt{n+1}$$

Assumption 3 is required for the uniform law of Theorem 3.1 to hold. The set $\mathcal{D}_0$ is assumed to be a region where, if x_0 is chosen within $\mathcal{D}_0$, the path of the stochastic algorithm will remain within the larger set $\mathcal{D}$. For practical purposes, we may take $\mathcal{D}$ to be an arbitrarily large bounded set in order to ensure convergence, though this worsens the complexity bounds provided by Theorem 3.1. We note that the constant C is very much a 'worst-case' bound, and for almost every problem arising in practice, the expected difference will be much smaller than Theorem 3.1 suggests. Rather, the crucial implication is that the expected difference diminishes at the rate $\sqrt{\frac{\log m_k}{m_k}}$, allowing a sharp level of control by adjusting m_k.

4. Self-Concordant Functions and Adaptive Methods

A convex function $f : \mathbb{R}^n \to \mathbb{R}$ is *self-concordant* if there exists a constant κ such that for every $x \in \mathbb{R}^n$ and every $h \in \mathbb{R}^n$, we have

$$|\nabla^3 f(x)[h, h, h]| \leq \kappa(\nabla^2 f(x)[h, h])^{3/2}$$

If $\kappa = 2$, f is *standard self-concordant*. Self-concordant functions were introduced by Nesterov and Nemirovski in the context of interior point methods (Nesterov & Nemirovski, 1994).

Many common problems have self-concordant formulations. At least one method, the DiSCO algorithm of Zhang and Xiao (2015), is tailored for distributed self-concordant optimization and has been applied to many regression problems. Convex quadratic objective functions have third

derivatives equal to zero, and are therefore trivially standard self-concordant. In particular, least squares regression is self-concordant. In (Zhang & Xiao, 2015), it is also shown that regularized regression, with either logistic loss or hinge loss, is self-concordant.

For self-concordant functions, the notion of a *local norm* is especially useful. Given a convex function f, the local norm with respect to f at the point x is given by

$$\|h\|_x = \sqrt{h^T \nabla^2 f(x) h}$$

Consider an iterative method for minimizing self-concordant functions with steps given as follows. On the k-th step, the step direction d_k is given by $d_k = -H_k \nabla f(x_k)$ for some positive definite matrix H_k, and the step size is given by $t_k = \frac{\alpha_k}{1+\alpha_k \delta_k}$, where $\delta_k = \|d_k\|_{x_k}$ and $\alpha_k = \frac{g_k^T H_k g_k}{\delta_k^2}$.

Methods of this type have been analyzed in (Tran-Dinh et al., 2015) and (Gao & Goldfarb, 2016). In (Gao & Goldfarb, 2016), the above choice of α_k is shown to guarantees a decrease in the function value.

Theorem 4.1 (Lemma 4.1, (Gao & Goldfarb, 2016))**.** *Let $\rho_k = \nabla f(x_k)^T H_k \nabla f(x_k)$. If α_k is chosen to be $\alpha_k = \frac{\rho_k}{\delta_k^2}$, then*

$$f(x_k + t_k d_k) \leq f(x_k) - \omega(\eta_k),$$

where $\eta_k = \frac{\rho_k}{\delta_k}$ and the function $\omega(z) = z - \log(1+z)$.

We make Assumption 2 in order to apply Theorem 4.1 to the empirical objective functions $F_k(x)$. A natural question is whether Assumption 2 can be relaxed to the assumption that $f(x, \xi)$ is self-concordant for all ξ, and $F(x)$ is standard self-concordant. In the case where ξ is finitely supported, it is possible to scale $F(x)$ so that we may use this weaker assumption.

Lemma 4.2. *Suppose that ξ is finitely supported on $\{a_1, \ldots, a_m\}$, with $p_i = \mathbb{P}(\xi = a_i)$. Suppose that $f(x, a_i)$ is self-concordant with constant κ_i. Let $\theta = \frac{1}{4} \frac{\max \kappa_i^2}{\min p_i}$. Then the scaled function $\overline{F}(x) = \mathbb{E}[\theta f(x, \xi)]$ is standard self-concordant, and every empirical objective function $\overline{F}_k(x)$ is standard self-concordant.*

Proof. Observe that $\theta f(x, a_i) p_i$ is self-concordant with constant $\theta^{-1/2} p_i^{-1/2} \kappa_i \leq 2$. We deduce that $\mathbb{E}\overline{F}(x) = \sum_{i=1}^m \theta f(x, a_i) p_i$ is standard self-concordant. Furthermore, since $\sum_{i=1}^m p_i = 1$, we have $\min p_i \leq \frac{1}{m}$. Thus, $\frac{1}{m} \theta f(x, a_i)$ is self-concordant with constant $\theta^{-1/2} m^{-1/2} \kappa_i \leq 2$, which implies that $\overline{F}_k(x)$ is standard self-concordant for every possible $\overline{F}_k(x)$. □

However, if we do not assume that each $f(x, \xi)$ is self-concordant, or if ξ is not compactly supported, it is unclear whether every possible $F_k(x)$ will be standard self-concordant, even when $F(x)$ is standard self-concordant. Thus, we impose Assumption 2 in our analysis, while observing that it is unnecessary in the practical case where ξ is derived from a finite data set.

Algorithm 1 SA-GD

Input: x_0, $\{m_0, m_1, \ldots\}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample $\xi_1, \ldots, \xi_{m_k}$ i.i.d from ξ
  Compute:

$$g_k = \nabla F_k(x_k), \quad d_k = -g_k,$$
$$\delta_k = \sqrt{g_k^T G_k(x_k) g_k},$$
$$\alpha_k = \frac{g_k^T g_k}{\delta_k^2}, \quad t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k},$$

  where $F_k(x) = \frac{1}{m_k} \sum_{i=1}^{m_k} f(x, \xi_i)$.
  Set $x_{k+1} = x_k + t_k g_k$.
end for

5. Stochastic Adaptive Methods

Our basic approach is to sample an empirical objective function $F_k(x)$ at each step, and then compute the step direction and adaptive step size (4) using F_k. For particular choices of the matrices H_k, we recover analogues of classical methods such as gradient descent, L-BFGS, and BFGS.

5.1. Stochastic Adaptive GD

When $H_k = I$ for every k, the resulting method is *stochastic adaptive gradient descent* (SA-GD), which is given as Algorithm 1. The number of samples drawn at each iteration is left as an input to the algorithm, and (weak) bounds on the required number m_k can be inferred from the convergence analysis.

Theorem 5.1. *Let $\epsilon > 0$. Suppose that at each iteration, we draw m i.i.d samples. We may choose $k = \widetilde{O}(\log \epsilon^{-1})$ and $m = \widetilde{O}(\epsilon^{-2} \log \epsilon^{-1})$ so that the SA-GD method converges in expectation to an ϵ-optimal solution after k steps. Here, $\widetilde{O}(\cdot)$ absorbs constants depending only on F, namely C and κ.*

A more precise quantitative form of Theorem 5.1, and its proof, appear as Theorem B.1 in the supplementary materials. The argument can be sketched as follows. Theorem 4.1 implies that the adaptive step size t_k produces a linear decrease towards the optimal value of F_k; that is, $F_k(x_{k+1}) - F_k(x_k^*) \leq r(F_k(x_k) - F_k(x_k^*))$. Here r is a deterministic constant depending only on ℓ, L, and γ. By iterating, we can bound the *true* gap $F(x_{k+1}) - F(x^*)$ in terms

Algorithm 2 SA-BFGS

Input: x_0, H_0, $\{m_0, m_1, \ldots\}$, $\beta < 1$
for $k = 0, 1, 2, \ldots$ **do**
  Sample $\xi_1, \ldots, \xi_{m_k}$ i.i.d from ξ
  Compute

$$g_k = \nabla F_k(x_k), \quad d_k = -H_k g_k$$
$$\delta_k = \sqrt{d_k^T G_k(x_k) d_k},$$
$$\alpha_k = \frac{g_k^T H_k g_k}{\delta_k^2}, \quad t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k},$$

  where $F_k(x) = \frac{1}{m_k} \sum_{i=1}^{m_k} f(x, \xi_i)$.
  Set $g_{k+1} = \nabla F_k(x_k + t_k d_k)$
  Set $y_k = g_{k+1} - g_k$
  if $g_{k+1}^T d_k < \beta g_k^T d_k$ **then**
    Set $d_k = g_k$
    Recompute δ_k, α_k, t_k
    Set $H_{k+1} = H_k$
  else
    Set $H_{k+1} = (I - \frac{s_k y_k^T}{s_k^T y_k}) H_k (I - \frac{y_k s_k^T}{s_k^T y_k}) + \frac{s_k s_k^T}{s_k^T y_k}$
  end if
  Set $x_{k+1} = x_k + t_k d_k$.
end for

of $r^k(F_0(x_1) - F_0(x_0^*))$ and $r^j \sup_{x \in \mathcal{D}} |F_{k+1-j}(x) - F(x)|$ for $1 \leq j \leq k$. The first term can be made smaller than ϵ by taking k to be $O(\log \epsilon^{-1})$, and the second can be bounded with Theorem 3.1.

5.2. Stochastic Adaptive BFGS

By updating H_k using the BFGS formula, we obtain the *stochastic adaptive BFGS* (SA-BFGS) method, which is given in Algorithm 2.

In Algorithm 2, we use the standard BFGS update with $y_k = g_k(x_{k+1}) - g_k(x_k)$. Another option is to replace y_k with the action of the Hessian on s_k, so $y_k = G_k(x_k)s_k$. In general, we must compute $G_k(x_k)d_k$ when finding the adaptive step size, so we can re-use the result of that computation instead of computing an extra gradient $g_k(x_{k+1})$.

For technical reasons, our SA-BFGS procedure tests whether the Wolfe condition is satisfied for the adaptive step size t_k. If not, we revert to taking a SA-GD step. This is an artifact of our analysis, and under suitable conditions on the growth of the samples m_k, there will be some point after which the Wolfe condition is necessarily satisfied on every step. In practice, this test can be omitted.

There are also two possible ways to implement SA-BFGS. For problems with n at most medium-sized, it is possible to explicitly store the matrix H_k, and compute d_k by a matrix product $-H_k g_k$. For n very large, it is infeasible to store H_k, and we can instead store the pairs (s_k, y_k) and compute $-H_k g_k$ using a two-loop recursion (Nocedal, 1980). This corresponds to stochastic adaptive L-BFGS (SA-LBFGS) if we limit the number of past pairs (s_k, y_k) to only the h most recent, and to SA-BFGS if we store everything. The amount of storage used by SA-LBFGS surpasses that of SA-BFGS as h approaches n.

Theorem 5.1 holds for SA-LBFGS, since Theorem B.1 holds for any method where $\{H_k\}$ has uniformly bounded eigenvalues. Thus, SA-LBFGS also converges in expectation to an ϵ-optimal solution after $k = \widetilde{O}(\log \epsilon^{-1})$ steps given samples of size $m = \widetilde{O}(\epsilon^{-2} \log \epsilon^{-1})$, though now the constants within the big-O are also dependent on h.

As with SA-GD, one can obtain an ϵ-optimal solution in expectation from SA-BFGS with a fixed amount of sampling:

Theorem 5.2. *Let $\epsilon > 0$. Suppose that at each iteration, we draw m i.i.d samples. We may choose $k = \widetilde{O}(\log \epsilon^{-1})$ and $m = \widetilde{O}(\epsilon^{-2}(\log \epsilon^{-1})^3)$ so that the SA-BFGS method converges in expectation to an ϵ-optimal solution after k steps. Here, $\widetilde{O}(\cdot)$ absorbs constants depending only on F, namely C and κ.*

This theorem follows from Lemma C.9 in the supplementary materials. Note that the complexity bound is in fact weaker than that of SA-GD. This occurs because the initial matrices $H_0, \ldots, H_k$ may be poor approximations of $(\nabla^2 F(x))^{-1}$, in which case the BFGS directions may perform poorly. We have little theoretical control over H_k except in the limit, and this is reflected in the weaker iteration bounds for ϵ-optimality.

For convergence to the true optimal solution x^*, it is necessary for the number of samples m_k to increase over time. In fact, by increasing the number of samples appropriately, the SA-BFGS algorithm can be shown to converge to x^* *R-superlinearly* with probability 1.

Theorem 5.3. *Suppose that we draw m_k samples on the k-th step, where m_k^{-1} converges R-superlinearly to 0. Then the SA-BFGS algorithm converges R-superlinearly to the optimal solution x^* almost surely.*

The complete proof is left to Lemma C.1 in the supplementary materials. We briefly sketch it here.

Taking m_k to be an increasing sequence, we can guarantee that the sequence $\{\eta_k = \frac{\rho_k}{\delta_k}\}$ converges to 0. From basic results on the adaptive step size, this implies that t_k eventually satisfies the Armijo-Wolfe conditions, and therefore the algorithm eventually uses the BFGS direction, and performs a BFGS update on H_k, at every step. Our test for the Wolfe condition ensures that in every iteration, we either can control the progress by analyzing the BFGS up-

date H_{k+1}, or by using our earlier results on SA-GD steps. Together, these facts imply that SA-BFGS converges R-linearly almost surely, and consequently, the sum of distances $\sum_{k=0}^{\infty} \|x_k - x^*\|$ is finite almost surely.

Following the classical argument, we analyze the evolution of H_{k+1} to show that the steps taken by SA-BFGS converge to the Newton step. This is precisely the well-known *Dennis-Moré condition* (Dennis Jr. & Moré, 1974) for Q-superlinear convergence; however, in the stochastic setting, we have an additional error term resulting from the variance in the sampling of the gradients and Hessians. We apply Theorem 3.1 to show that $g_k(x)$ and $G(x)$ rapidly converge to the true gradient and Hessian, and thus SA-BFGS attains R-superlinear convergence.

Increasing m_k at this rate is clearly infeasible in reality, and it is non-trivial to determine a level of sampling which produces (iteration-wise) superlinear convergence in a reasonable amount of real time. We note, for comparison, that the Streaming SVRG method (Frostig et al., 2015) requires sample sizes for the gradient that grow at a geometric rate, and that the dynamic batch method (Byrd et al., 2012) obtains R-linear convergence in expectation by increasing the sample sizes at a geometric rate. A superlinear increase in the number of samples may be unavoidable as a condition for superlinear convergence, given the statistical error of the sample.

In practice, we are often uninterested in finding the true minimizer, and instead aim to quickly find an approximate solution. The theoretical R-superlinear rate of SA-BFGS suggests that SA-BFGS or SA-LBFGS may offer practical speedups over first-order stochastic methods, even without using substantial sampling. One intuitive heuristic would be to draw fewer samples in the early phase of the algorithm, when computing the gradient with high accuracy is less valuable, and then increasing the number of samples as the solutions approach optimality to reduce fluctuation.

6. Numerical Experiments

We compared several implementations of SA-GD and SA-BFGS against the original SGD method and the *robust SGD* method (Nemirovski et al., 2009) on a penalized least squares problem with random design. That is, the objective function has the form

$$\min_{w \in \mathbb{R}^p} L(w) = \mathbb{E}(Y - X^T w)^2 + \frac{1}{2}\|w\|_2^2$$

where $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}$ are random variables with finite second moments. We base our model on a standard linear regression problem with Gaussian errors, so $Y = X^T\beta + \epsilon$ for a deterministic vector $\beta \in \mathbb{R}^p$ and $\epsilon \sim N(0,1)$ is a noise component. X was drawn according to a multivariate $N(0, \Sigma(\rho))$, where $\Sigma(\rho) = (1-\rho^2)I_p + \rho^2 J$ (here J is the all-ones matrix). By varying ρ, we control the condition number of the expected Hessian. We tested problems of size $p = 100$ and $p = 500$.

The following eight algorithms were tested:

SGD: SGD with fixed $m_k = p$ and diminishing step sizes $t_k = \frac{1}{k+1000}$ for problems with $p = 100$, and $t_k = \frac{1}{k+5000}$ for $p = 500$.

SA-GD: SA-GD with fixed $m_k = p$.

SA-GD-I: SA-GD with increasing samples $m_k = \frac{1}{2}p + 1.01^k$.

SA-BFGS: SA-BFGS with fixed $m_k = p$. We do not test for the Wolfe condition at each step, and instead always take the adaptive BFGS step and perform a BFGS update.

SA-BFGS-I: SA-BFGS with increasing samples $m_k = \frac{1}{2}p + 1.01^k$. We do not test for the Wolfe condition at each step, and instead always take the adaptive BFGS step and perform a BFGS update.

SA-BFGS-GD: SA-BFGS with increasing samples $m_k = \frac{1}{2}p + 1.01^k$. We test for the Wolfe condition and switch to taking a SA-GD step when the adaptive step t_k for SA-BFGS fails the test.

R-S-GD-C: Robust SGD with constant step size $t_k = \frac{1}{\sqrt{N}}$, where N is the pre-determined number of steps. At the k-th iteration, output the average of the previous $k/2$ solutions.

R-S-GD-V: Robust SGD with diminishing step size $t_k = \frac{1}{\sqrt{k}}$. At the k-th iteration, output the average of the previous $k/2$ solutions.

We also tested the Streaming SVRG algorithm with the hyperparameters specified in Corollary 4 (Frostig et al., 2015). However, this algorithm was difficult to tune and performed poorly, so we have omitted it from the figures.

The algorithms prefixed by "SA-" used the adaptive step size. For SGD, we followed the standard practice of using a diminishing and non-summable step size $t_k = \frac{a}{k+b}$. The values $a = 1, b = \{1000, 5000\}$ in our test were chosen experimentally. The SA-BFGS algorithms were implemented using a two-loop recursion to compute $H_k g_k$ when $p = 500$. This significantly improved their performance compared to storing the matrix H_k explicitly.

Figure 1 shows the performance of each algorithm on a series of problems with varying problem size p and parameter ρ. The y-axis measures the gap $\log(f(x(t)) - f(x^*))$ of the solution $x(t)$ obtained by the algorithm after using t

seconds of CPU time. The algorithms were implemented in Matlab 2015a, and the system was an Intel i5-5200U running Ubuntu.

The increasing sample sizes used in SA-GD-I do not appear to reduce its variance, compared to SA-GD with m_k fixed, which goes against our initial expectations. In plots (a), (b), (e), and (f), SA-GD-I appears to exhibit the same fluctuations as SA-GD when the points approach optimality. In plot (c), SA-GD-I briefly surpasses SA-GD before the objective value jumps again. It is only in plot (d) that SA-GD-I appears to descend more consistently than SA-GD. This suggests that, for these problem instances, the rate of sampling $m_k = O(1.01^k)$ could even be increased further, and that the tradeoff between taking more steps, versus taking fewer steps with more accurate estimates, favors more accurate steps.

Likewise, SA-BFGS-I failed to consistently perform better than SA-BFGS with m_k fixed. SA-BFGS-I seemed to perform relatively better when the dimension was larger, whereas SA-BFGS was faster when $p = 100$. One notable example was plot (f), where SA-BFGS initially descends rapidly before abruptly stalling, and is quickly surpassed by SA-BFGS-I. The data going beyond 60 seconds of CPU time verified that SA-BFGS continued to stall, and never managed to obtain a log-error of less than 1. Our interpretation is that for this problem, the fixed sample size is too small to obtain an accurate solution. It also appears that SA-BFGS requires larger samples to obtain a stable solution when the problem is particularly ill-conditioned. In the plots (e), and particularly (f), where $\rho = 0.9$, SA-BFGS rapidly reaches the point where variance in the gradient estimates introduces too much noise to make further progress.

What is also interesting is that SA-GD often outperforms SA-BFGS if both algorithms use the same fixed sample size. While somewhat disappointing, there is a natural reason for this. BFGS optimizes a local *quadratic* model of the objective, and is performant when its approximation H_k resembles the true Hessian. The Hessian exhibits greater variance than the gradient, simply by virtue of having n^2 components compared to n, and we generally expect that more sampling is needed to accurately estimate the Hessian. With the same amount of sampling, SA-BFGS is therefore 'noisier' than SA-GD relative to the true function.

We also see this reflected in the performance of SA-BFGS-GD, which was able to outperform SA-BFGS and SA-BFGS-I when $p = 100$ despite the additional cost of checking the Wolfe condition. For $p = 500$, it is less clear whether a better strategy is simply to perform SA-BFGS-I. One strategy that might be effective is a fixed pattern of 'switching' between GD and BFGS steps, with a greater number of GD steps in the early phase.

Predictably, robust SGD was the most stable method (given the lack of variance-reduction in the other methods). Both R-S-GD-C and R-S-GD-V exhibited almost no fluctuations, while improving at a reasonable rate. The R-S-GD-C method with a constant step size outperformed R-S-GD-V, which we find counterintuitive given that R-S-GD-C used a smaller step size than R-S-GD-V. This method of variance-reduction by averaging the solutions seems to be effective and can be applied to any iterative method.

Numerical results for ERM (empirical risk minimization) problems can be found in section D in the supplementary material.

7. Discussion

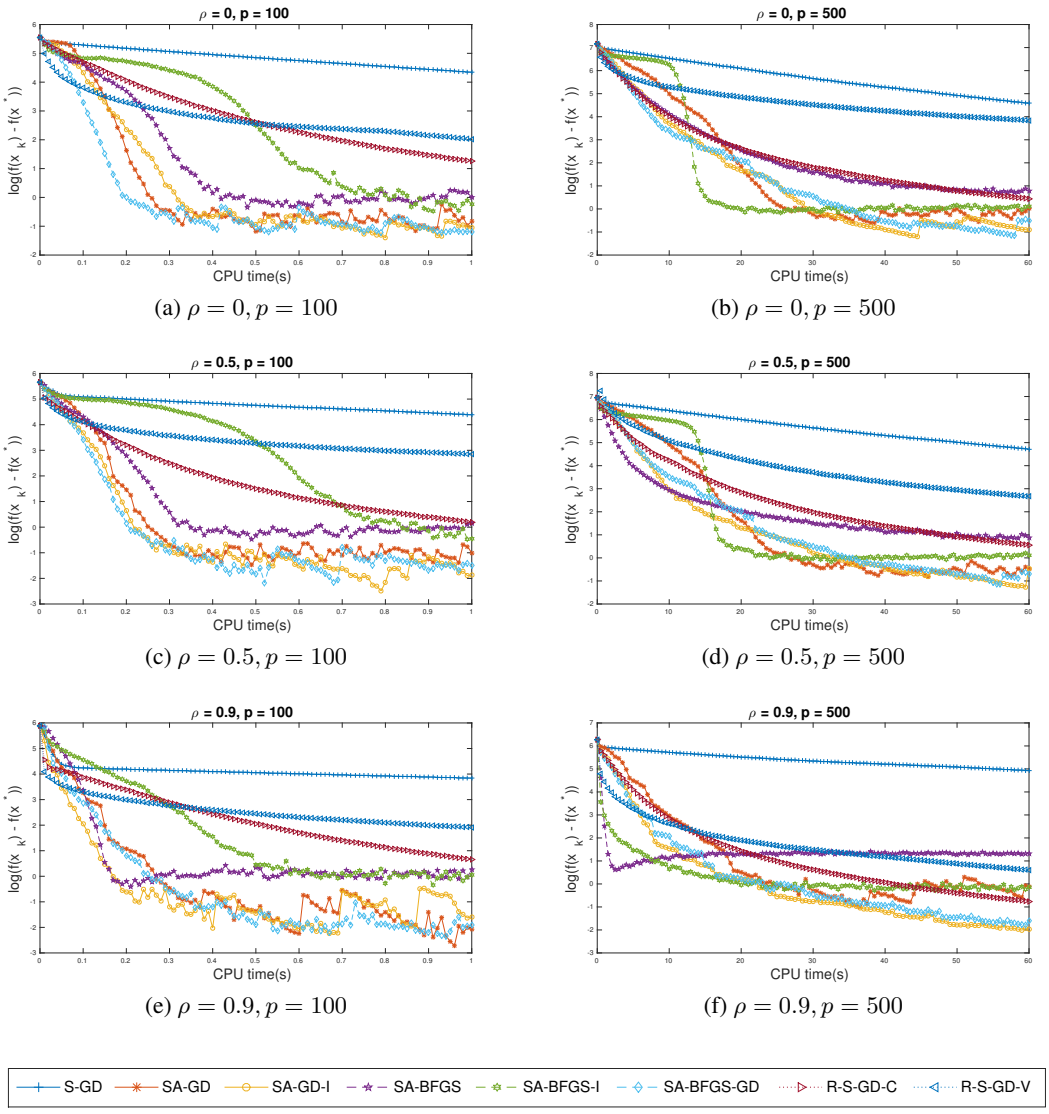
For self-concordant objective functions, adaptive methods eliminate the need to choose a step size, which is a big advantage. This is independent of a new difficulty which arises for *stochastic* methods, which is the choice of the sample size m_k. From our experiments, we see that choosing m_k appropriately is crucial. Unlike SGD and SVRG, where it is often most effective to use mini-batches of size 1, SA-BFGS and SA-GD work best with comparatively larger samples.

Note that SA-GD and SA-BFGS are not *purely* first-order methods, as computing the adaptive step size requires calculating the Hessian-vector product $G_k(x_k)d_k$. However, it is often possible to calculate Hessian-vector products efficiently, and at far less cost than computing the full Hessian $\nabla^2 F_k(x)$. Consider, for example, a a logistic regression problem where the empirical loss function is

$$L(x) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i a_i^T x})$$

for sampled data $\{(a_i, y_i) : a_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\}$. Let A denote the $n \times m$ matrix with columns a_i. The gradient is given by $A^T\beta$, where β is the vector $\beta_i = -y_i e^{-y_i a_i^T x}/(1 + e^{-y_i a_i^T x})$, and the Hessian is given by ABA^T, where B is the diagonal matrix with entries $B_{ii} = e^{-y_i a_i^T x}/(1 + e^{-y_i a_i^T x})^2$. To compute the product $d^T\nabla^2 L(x)d$, it suffices to compute $\sum_{i=1}^m B_{ii}(a_i^T d)^2$, and this requires approximately the same number of arithmetic operations as computing $\nabla L(x)$. Thus, computing δ_k for the adaptive step size requires roughly the same amount of work as one additional gradient calculation. Also, as mentioned in Section 5, we may replace the BFGS update in SA-BFGS with a modified update using the Hessian action $y_k = G_k(x_k)d_k$ to save effort.

This work represents only a preliminary step in the development of stochastic quasi-Newton methods. There are several key questions that remain open:



(a) $\rho = 0, p = 100$ (b) $\rho = 0, p = 500$

(c) $\rho = 0.5, p = 100$ (d) $\rho = 0.5, p = 500$

(e) $\rho = 0.9, p = 100$ (f) $\rho = 0.9, p = 500$

Figure 1. Experimental results for $p = 100, 500$ and varying ρ. The x-axis is the elapsed CPU time and the y-axis is $\log(f(x) - f(x^*))$.

1. Theorem 5.3 partially resolves a question posed by Moritz et al. (Moritz et al., 2016), by proving superlinear convergence of a stochastic algorithm under rather restrictive conditions. It would be strengthened greatly if we could prove superlinear convergence under weaker conditions on m_k.

2. The theory developed for adaptive step sizes only applies to self-concordant functions, but the adaptive step size itself can be interpreted for non-self-concordant functions, as an adjustment based on the local curvature. It would be of great interest to extend adaptive methods to general convex functions, thereby replacing both inexact line search and fixed step sizes on a large class of problems.

3. How can variance-reduction be applied to SA-GD and SA-BFGS? The control variates used in SVRG are effective, though costly, and perhaps difficult to compute for functions of the form $\mathbb{E}f(x, \xi)$. Another possible technique is to output an average of the solutions, as is done by robust SGD, though this introduces an additional hyperparameter (the number of past solutions to average). Better variance-reduction strategies might compensate for noisy estimates, allowing us to reduce the number of samples needed in practice.

4. What heuristics can be developed for the sample sizes m_k to improve the stability and speed up performance of SA-GD and SA-BFGS?

References

Bottou, Léon. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Physica-Verlag HD, 2010.

Broyden, Charles G. Quasi-Newton methods and their application to function minimisation. *Mathematics of Computation*, 21(99):368–381, 1967.

Byrd, Richard H., Nocedal, Jorge, and Yuan, Ya-Xiang. Global convergence of a class of quasi-Newton methods on convex problems. *Siam. J. Numer. Anal.*, (5):1171–1190, 1987.

Byrd, Richard H, Chin, Gillian M, Nocedal, Jorge, and Wu, Yuchen. Sample size selection in optimization methods for machine learning. *Mathematical Programming*, 134 (1):127–155, 2012.

Byrd, Richard H., Hansen, S. L., Nocedal, Jorge, and Singer, Yoram. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

Defazio, Aaron, Bach, Francis, and Lacoste-Julien, Simon. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.

Dennis Jr., John E. and Moré, Jorge J. Characterization of superlinear convergence and its application to quasi-Newton methods. *Math. Comp.*, 28(106):549–560, 1974.

Fletcher, Roger. A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317–322, 1970.

Friedlander, Michael P and Goh, Gabriel. Tail bounds for stochastic approximation. *arXiv preprint arXiv:1304.5586*, 2013.

Frostig, Roy, Ge, Rong, Kakade, Sham M, and Sidford, Aaron. Competing with the empirical risk minimizer in a single pass. In *COLT*, pp. 728–763, 2015.

Gao, Wenbo and Goldfarb, Donald. Quasi-Newton methods: Superlinear convergence without line search for self-concordant functions. *in review. arXiv:1612.06965*, 2016.

Goldfarb, Donald. A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24(109):23–26, 1970.

Goldfarb, Donald, Iyengar, Garud, and Zhou, Chaoxu. Linear convergence of stochastic Frank-Wolfe variants. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1066–1074, 2017.

Gower, Robert, Goldfarb, Donald, and Richtárik, Peter. Stochastic block BFGS : Squeezing more curvature out of data. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1869–1878, 2016.

Johnson, Rie and Zhang, Tong. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pp. 315–323, 2013.

Moritz, Philipp, Nishihara, Robert, and Jordan, Michael I. A linearly-convergent stochastic L-BFGS algorithm. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–258, 2016.

Nemirovski, Arkadi, Juditsky, Anatoli, Lan, Guanghui, and Shapiro, Alexander. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Nesterov, Yurii and Nemirovski, Arkadi. *Interior-point polynomial algorithms in convex programming*. Society for Industrial and Applied Mathematics, Philadelphia, 1994.

Nocedal, Jorge. Updating quasi-Newton matrices with limited storage. *Math. Comp.*, 35(151):773–782, 1980.

Powell, Michael J. D. Some global convergence properties of a variable metric algorithm for minimization without exact line searches. In Cottle, Richard and Lemke, C.E. (eds.), *Nonlinear Programming*, volume IX. SIAM-AMS Proceedings, 1976.

Rodomanov, Anton and Kropotov, Dmitry. A superlinearly-convergent proximal newton-type method for the optimization of finite sums. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 2597–2605, 2016.

Schmidt, Mark, Le Roux, Nicolas, and Bach, Francis. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, pp. 1–30, 2013.

Shanno, David F. Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

Tran-Dinh, Quoc, Kyrillidis, Anastasios, and Cevher, Volkan. Composite self-concordant minimization. *Journal of Machine Learning Research*, 16(Mar):371–416, 2015.

W. van der Vaart, Aad. and Wellner, Jon A. *Weak Convergence and Empirical Processes*. Springer New York, 1996.

Zhang, Yuchen and Xiao, Lin. Disco: Distributed optimization for self-concordant empirical loss. In *JMLR: Workshop and Conference Proceedings*, volume 32, pp. 362–370, 2015.