A Linearly-Convergent Stochastic L-BFGS Algorithm

Philipp Moritz
University of California, Berkeley
pcmoritz@eecs.berkeley.edu

Robert Nishihara
University of California, Berkeley
rkn@eecs.berkeley.edu

Michael I. Jordan
University of California, Berkeley
jordan@eecs.berkeley.edu

Abstract

We propose a new stochastic L-BFGS algorithm and prove a linear convergence rate for strongly convex and smooth functions. Our algorithm draws heavily from a recent stochastic variant of L-BFGS proposed in Byrd et al. (2014) as well as a recent approach to variance reduction for stochastic gradient descent from Johnson and Zhang (2013). We demonstrate experimentally that our algorithm performs well on large-scale convex and non-convex optimization problems, exhibiting linear convergence and rapidly solving the optimization problems to high levels of precision. Furthermore, we show that our algorithm performs well for a wide-range of step sizes, often differing by several orders of magnitude.

1 Introduction

A trend in machine learning has been toward using more parameters to model larger datasets. As a consequence, it is important to design optimization algorithms for these large-scale problems. A typical optimization problem arising in this setting is empirical risk minimization. That is,

$$\min_w \frac{1}{N} \sum_{i=1}^{N} f_i(w), \tag{1}$$

where $w \in \mathbb{R}^d$ may specify the parameters of a machine learning model, and $f_i(w)$ quantifies how well the model w fits the ith data point. Two challenges arise when attempting to solve Equation 1. First, d may be extremely large. Second, N may be extremely large.

Appearing in Proceedings of the 19^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 41.

When d is small, Newton's method is often the algorithm of choice due to its rapid convergence (both in theory and in practice). However, Newton's method requires the computation and inversion of the Hessian matrix $\nabla^2 f(w)$, which may be computationally too expensive in high dimensions. As a consequence, practitioners are often limited to using first-order methods which only compute gradients of the objective, requiring $O(d)$ computation per iteration. The gradient method is the simplest example of a first-order method, but much work has been done to design quasi-Newton methods which incorporate information about the curvature of the objective without ever computing second derivatives. L-BFGS (Liu and Nocedal, 1989), the limited-memory version of the classic BFGS algorithm, is one of the most successful algorithms in this space. Inexact Newton methods are another approach to using second order information for large-scale optimization. They approximately invert the Hessian in $O(d)$ steps. This can be done by using a constant number of iterations of the conjugate gradient method (Dembo et al., 1982; Dembo and Steihaug, 1983; Nocedal and Wright, 2006).

When N is large, batch algorithms such as the gradient method, which compute the gradient of the full objective at every iteration, are slowed down by the fact that they have to process every data point before updating the model. Stochastic optimization algorithms get around this problem by updating the model w after processing only a small subset of the data, allowing them to make much progress in the time that it takes the gradient method to make a single step.

For many machine learning problems, where both d and N are large, stochastic gradient descent (SGD) and its variants are the most widely used algorithms (Robbins and Monro, 1951; Bottou, 2010; Bottou and LeCun, 2004), often because they are some of the few algorithms that can realistically be applied in this setting.

Given this context, much research in optimization has been directed toward designing better stochastic first-order algorithms. For a partial list, see (Kingma and

Ba, 2015; Sutskever et al., 2013; Duchi et al., 2011; Shalev-Shwartz and Zhang, 2013; Johnson and Zhang, 2013; Roux et al., 2012; Wang et al., 2013; Nesterov, 2009; Frostig et al., 2015; Agarwal et al., 2014). In particular, much progress has gone toward designing stochastic variants of L-BFGS (Mokhtari and Ribeiro, 2014a; Wang et al., 2014; Byrd et al., 2014; Bordes et al., 2009; Schraudolph et al., 2007; Sohl-Dickstein et al., 2014).

Unlike gradient descent, L-BFGS does not immediately lend itself to a stochastic version. The updates in the stochastic gradient method average together to produce a downhill direction in expectation. However, as pointed out in Byrd et al. (2014), the updates used in L-BFGS to construct the inverse Hessian approximation overwrite one another instead of averaging. Our algorithm addresses this problem in the same ways as Byrd et al. (2014), by computing Hessian vector products formed from larger minibatches.

Though stochastic methods often make rapid progress early on, the variance of the estimates of the gradient slow their convergence near the optimum. To illustrate this phenomenon, even if SGD is initialized at the optimum, it will immediately move to a point with a worse objective value. For this reason, convergence guarantees typically require diminishing step sizes. One promising line of work involves speeding up the convergence of stochastic first-order methods by reducing the variance of the gradient estimates (Johnson and Zhang, 2013; Roux et al., 2012; Defazio et al., 2014; Shalev-Shwartz and Zhang, 2013).

We introduce a stochastic variant of L-BFGS that incorporates the idea of variance reduction and has two desirable features. First, it obtains a guaranteed linear rate of convergence in the strongly-convex case. In particular, it does not require a diminishing step size in order to guarantee convergence (as partially evidenced by the fact that if our algorithm is initialized at the optimum it will stay there). Second, it performs very well on large-scale optimization problems, exhibiting a qualitatively linear rate of convergence in practice.

2 The Algorithm

We consider the problem of minimizing the function

$$f(w) = \frac{1}{N}\sum_{i=1}^{N} f_i(w) \tag{2}$$

over $w \in \mathbb{R}^d$. For a subset $\mathcal{S} \subseteq \{1, \dots, N\}$, we define the subsampled function $f_{\mathcal{S}}$ by

$$f_{\mathcal{S}}(w) = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f_i(w). \tag{3}$$

Our updates will use stochastic estimates of the gradient $\nabla f_{\mathcal{S}}$ as well as stochastic approximations to the inverse Hessian $\nabla^2 f_{\mathcal{T}}$. Following Byrd et al. (2014), we use distinct subsets $\mathcal{S}, \mathcal{T} \subseteq \{1, \dots, N\}$ in order to decouple the estimation of the gradient from the estimation of the Hessian. We let $b = |\mathcal{S}|$ and $b_H = |\mathcal{T}|$.

Following Johnson and Zhang (2013), we occasionally compute full gradients, which we use to reduce the variance of our stochastic gradient estimates.

The update rule for our algorithm will take the form

$$w_{k+1} = w_k - \eta_k H_k v_k.$$

In the gradient method, H_k is the identity matrix. In Newton's method, it is the inverse Hessian $(\nabla^2 f(w_k))^{-1}$. In our algorithm, as in L-BFGS, H_k will be an approximation to the inverse Hessian. Instead of the usual stochastic estimate of the gradient, v_k will be a stochastic estimate of the gradient with reduced variance.

Code for our algorithm is given in Algorithm 1. Our algorithm is specified by several parameters. It requires a step size η, a memory size M, and positive integers m and L. Every m iterations, the algorithm performs a full gradient computation, which it uses to reduce the variance of the stochastic gradient estimates. Every L iterations, the algorithm updates the inverse Hessian approximation. The vector s_r records the average direction in which the algorithm has made progress over the past $2L$ iterations. The vector y_r is obtained by multiplying s_r by a stochastic estimate of the Hessian. Note that this differs from the usual L-BFGS algorithm, which produces y_r by taking the difference between successive gradients. We find that this approach works better in the stochastic setting. The inverse Hessian approximation H_r is defined from the pairs (s_j, y_j) for $r - M + 1 \leq j \leq r$ using the standard L-BFGS update rule, which is described in Section 2.1. The user must also choose batch sizes b and b_H from which to construct the stochastic gradient and stochastic Hessian estimates.

In Algorithm 1 and below, we use I to refer to the identity matrix. We use $\mathcal{F}_{k,t}$ to denote the sigma algebra generated by the random variables introduced up to the time when the iteration counters k and t have the specified values. That is,

$$\mathcal{F}_{k,t} = \sigma\left(\begin{array}{c}\{\mathcal{S}_{k',t'} \,:\, k' < k \text{ or } k' = k \text{ and } t' < t\} \\ \cup\{\mathcal{T}_r : rL \leq mk + t\}\end{array}\right).$$

We will use $\mathbb{E}_{k,t}$ to denote the conditional expectation with respect to $\mathcal{F}_{k,t}$.

We define the inverse Hessian approximation H_r in Section 2.1. Note that we do not actually construct

Algorithm 1 Stochastic L-BFGS

Input: initial state w_0, parameters m, M, and L, batch sizes b and b_H, and step size η

Initialize $r = 0$
Initialize $H_0 = I$
for $k = 0, \ldots$ **do**
  Compute a full gradient $\mu_k = \nabla f(w_k)$
  Set $x_0 = w_k$
  for $t = 0, \ldots, m-1$ **do**
    Sample a minibatch $\mathcal{S}_{k,t} \subseteq \{1, \ldots, N\}$
    Compute a stochastic gradient $\nabla f_{\mathcal{S}_{k,t}}(x_t)$
    Compute a variance reduced gradient $v_t = \nabla f_{\mathcal{S}_{k,t}}(x_t) - \nabla f_{\mathcal{S}_{k,t}}(w_k) + \mu_k$
    Set $x_{t+1} = x_t - \eta H_r v_t$
    if $t \equiv 0 \bmod L$ **then**
      Increment $r \leftarrow r + 1$
      Set $u_r = \frac{1}{L} \sum_{j=t-L}^{t-1} x_j$
      Sample $\mathcal{T}_r \subseteq \{1, \ldots, N\}$ to define the stochastic approximation $\nabla^2 f_{\mathcal{T}_r}(u_r)$
      Compute $s_r = u_r - u_{r-1}$
      Compute $y_r = \nabla^2 f_{\mathcal{T}_r}(u_r) s_r$
      Define H_r as in Section 2.1
  Set $w_{k+1} = x_i$ for randomly chosen $i \in \{0, \ldots, m-1\}$

the matrix H_r because doing so would require $O(d^2)$ computation. In practice, we directly compute products of the form $H_r v$ using the two-loop recursion (Nocedal and Wright, 2006, Algorithm 7.4).

2.1 Construction of the Inverse Hessian Approximation H_r

To define the inverse Hessian approximation H_r from the pairs (s_j, y_j), we follow the usual L-BFGS method. Let $\rho_j = 1/s_j^\top y_j$ and recursively define

$$H_r^{(j)} = (I - \rho_j s_j y_j^\top)^\top H_r^{(j-1)} (I - \rho_j s_j y_j^\top) + \rho_j s_j s_j^\top, \quad (4)$$

for $r - M + 1 \le j \le r$. Initialize $H_r^{(r-M)} = (s_r^\top y_r / \|y_r\|^2) I$ and set $H_r = H_r^{(r)}$.

Note that the update in Equation 4 preserves positive definiteness (note that $\rho_j > 0$), which implies that H_r and each $H_r^{(j)}$ will be positive definite, as will their inverses.

3 Preliminaries

Our analysis makes use of the following assumptions.

Assumption 1. *The function $f_i \colon \mathbb{R}^n \to \mathbb{R}$ is convex and twice continuously differentiable for each $1 \le i \le N$.*

Assumption 2. *There exist positive constants λ and Λ such that*

$$\lambda I \preceq \nabla^2 f_{\mathcal{T}}(w) \preceq \Lambda I \quad (5)$$

for all $w \in \mathbb{R}^d$ and all nonempty subsets $\mathcal{T} \subseteq \{1, \ldots, N\}$. Note the lower bound trivially holds in the regularized case.

We will typically force strong convexity to hold by adding a strongly-convex regularizer to our objective (which can be absorbed into the f_i's). These assumptions imply that f has a unique minimizer, which we denote by w_*.

Lemma 3. *Suppose that Assumption 1 and Assumption 2 hold. Let $B_r = H_r^{-1}$. Then*

$$\operatorname{tr}(B_r) \le (d + M)\Lambda$$
$$\det(B_r) \ge \frac{\lambda^{d+M}}{((d+M)\Lambda)^M}.$$

We prove Lemma 3 in Section 7.1.

Lemma 4. *Suppose that Assumption 1 and Assumption 2 hold. Then there exist constants $0 < \gamma \le \Gamma$ such that H_r satisfies*

$$\gamma I \preceq H_r \preceq \Gamma I \quad (6)$$

for all $r \ge 1$.

In Section 7.2, we prove Lemma 4 with the values

$$\gamma = \frac{1}{(d+M)\Lambda} \quad \text{and} \quad \Gamma = \frac{((d+M)\Lambda)^{d+M-1}}{\lambda^{d+M}}.$$

We will make use of Lemma 5, a simple result for strongly convex functions. We include a proof for completeness.

Lemma 5. *Suppose that f is continuously differentiable and strongly convex with parameter λ. Let w_* be the unique minimizer of f. Then for any $x \in \mathbb{R}^d$, we have*

$$\|\nabla f(x)\|^2 \geq 2\lambda(f(x) - f(w_*)).$$

Proof. By the strong convexity of f,

$$\begin{aligned} f(w_*) &\geq f(x) + \nabla f(x)^\top (w_* - x) + \frac{\lambda}{2}\|w_* - x\|^2 \\ &\geq f(x) + \min_v \left(\nabla f(x)^\top v + \frac{\lambda}{2}\|v\|^2 \right) \\ &= f(x) - \frac{1}{2\lambda}\|\nabla f(x)\|^2. \end{aligned}$$

The last equality holds by plugging in the minimizer $v = -\nabla f(x)/\lambda$. □

In Lemma 6, we bound the variance of our variance-reduced gradient estimates. The proof of Lemma 6, given in Section 7.3, closely follows that of Johnson and Zhang (2013, Theorem 1).

Lemma 6. *Let w_* be the unique minimizer of f. Let $\mu_k = \nabla f(w_k)$ and let $v_t = \nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_k) + \mu_k$ be the variance-reduced stochastic gradient. Conditioning on $\mathcal{F}_{k,t}$ and taking an expectation with respect to $\mathcal{S}$, we have*

$$\mathbb{E}_{k,t}[\|v_t\|^2] \leq 4\Lambda(f(x_t) - f(w_*) + f(w_k) - f(w_*)). \quad (7)$$

4 Convergence Analysis

Theorem 7 states our main result.

Theorem 7. *Suppose that Assumption 1 and Assumption 2 hold. Let w_* be the unique minimizer of f. Then for all $k \geq 0$, we have*

$$\mathbb{E}[f(w_k) - f(w_*)] \leq \alpha^k \mathbb{E}[f(w_0) - f(w_*)],$$

where the convergence rate α is given by

$$\alpha = \frac{1/(2m\eta) + \eta\Gamma^2\Lambda^2}{\gamma\lambda - \eta\Gamma^2\Lambda^2} < 1,$$

assuming that we choose $\eta < \gamma\lambda/(2\Gamma^2\Lambda^2)$ and that we choose m large enough to satisfy

$$\gamma\lambda > \frac{1}{2m\eta} + 2\eta\Gamma^2\Lambda^2. \quad (8)$$

Proof. Using the Lipschitz continuity of ∇f, which follows from Assumption 2, we have

$$\begin{aligned} & f(x_{t+1}) \quad (9) \\ &\leq f(x_t) + \nabla f(x_t)^\top (x_{t+1} - x_t) + \frac{\Lambda}{2}\|x_{t+1} - x_t\|^2 \\ &= f(x_t) - \eta \nabla f(x_t)^\top H_r v_t + \frac{\eta^2\Lambda}{2}\|H_k v_t\|^2. \end{aligned}$$

Conditioning on $\mathcal{F}_{k,t}$ and taking expectations in Equation 9, this becomes

$$\begin{aligned} & \mathbb{E}_{k,t}[f(x_{t+1})] \quad (10) \\ &\leq f(x_t) - \eta \nabla f(x_t)^\top H_r \nabla f(x_t) + \frac{\eta^2\Lambda}{2}\mathbb{E}_{k,t}\|H_k v_t\|^2, \end{aligned}$$

where we used the fact that $\mathbb{E}_{k,t}[v_t] = \nabla f(x_t)$. We then use Lemma 4 to bound the second and third terms on the bottom line of Equation 10 to get

$$\mathbb{E}_{k,t}[f(x_{t+1})] \leq f(x_t) - \eta\gamma\|\nabla f(x_t)\|^2 + \frac{\eta^2\Gamma^2\Lambda}{2}\mathbb{E}_{k,t}\|v_t\|^2.$$

Now, we bound $\mathbb{E}_{k,t}\|v_t\|^2$ using Lemma 6 and we bound $\|\nabla f(x_t)\|^2$ using Lemma 5. Doing so gives

$$\begin{aligned} & \mathbb{E}_{k,t}[f(x_{t+1})] \\ &\leq f(x_t) - 2\eta\gamma\lambda(f(x_t) - f(w_*)) \\ &\quad + 2\eta^2\Gamma^2\Lambda^2(f(x_t) - f(w_*) + f(w_k) - f(w_*)) \\ &= f(x_t) - 2\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)(f(x_t) - f(w_*)) \\ &\quad + 2\eta^2\Gamma^2\Lambda^2(f(w_k) - f(w_*)). \end{aligned}$$

Taking expectations over all random variables, summing over $t = 0, \ldots, m-1$, and using a telescoping sum gives

$$\begin{aligned} & \mathbb{E}[f(x_m)] \\ &\leq \mathbb{E}[f(x_0)] + 2m\eta^2\Gamma^2\Lambda^2\mathbb{E}[f(w_k) - f(w_*)] \\ &\quad - 2\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)\left(\sum_{t=0}^{m-1}\mathbb{E}[f(x_t)] - mf(w_*)\right) \\ &= \mathbb{E}[f(w_k)] + 2m\eta^2\Gamma^2\Lambda^2\mathbb{E}[f(w_k) - f(w_*)] \\ &\quad - 2m\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)\mathbb{E}[f(w_{k+1}) - f(w_*)]. \end{aligned}$$

Rearranging the above gives

$$\begin{aligned} 0 &\leq \mathbb{E}[f(w_k) - f(x_m)] + 2m\eta^2\Gamma^2\Lambda^2\mathbb{E}[f(w_k) - f(w_*)] \\ &\quad - 2m\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)\mathbb{E}[f(w_{k+1}) - f(w_*)] \\ &\leq \mathbb{E}[f(w_k) - f(w_*)] + 2m\eta^2\Gamma^2\Lambda^2\mathbb{E}[f(w_k) - f(w_*)] \\ &\quad - 2m\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)\mathbb{E}[f(w_{k+1}) - f(w_*)] \\ &= (1 + 2m\eta^2\Gamma^2\Lambda^2)\mathbb{E}[f(w_k) - f(w_*)] \\ &\quad - 2m\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)\mathbb{E}[f(w_{k+1}) - f(w_*)]. \end{aligned}$$

The second inequality follows from the fact that $f(w_*) \leq f(x_m)$. Using the fact that $\eta < \gamma\lambda/(2\Gamma^2\Lambda^2)$, it follows that

$$\begin{aligned} & \mathbb{E}[f(w_{k+1}) - f(w_*)] \\ &\leq \frac{1 + 2m\eta^2\Gamma^2\Lambda^2}{2m\eta(\gamma\lambda - \eta\Gamma^2\Lambda^2)}\mathbb{E}[f(w_k) - f(w_*)]. \end{aligned}$$

Since we chose m and η to satisfy Equation 8, it follows that the rate α is less than one. This completes the proof. □

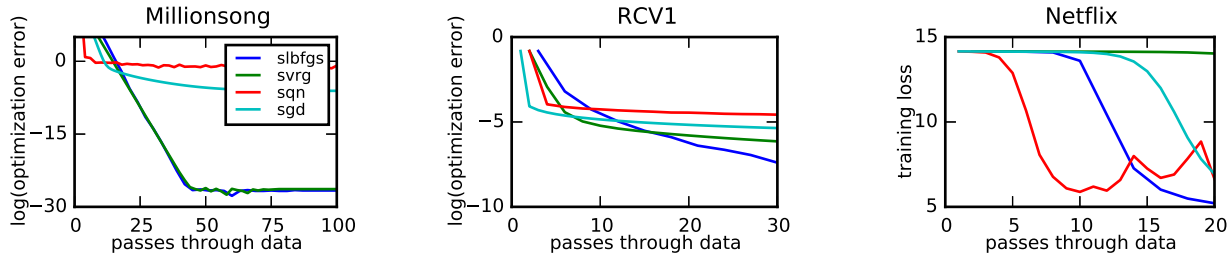

Figure 1: The left figure plots the log of the optimization error as a function of the number of passes through the data for SLBFGS, SVRG, SQN, and SGD for a ridge regression problem (Millionsong). The middle figure does the same for a support vector machine (RCV1). The right plot shows the training loss as a function of the number of passes through the data for the same algorithms for a matrix completion problem (Netflix).

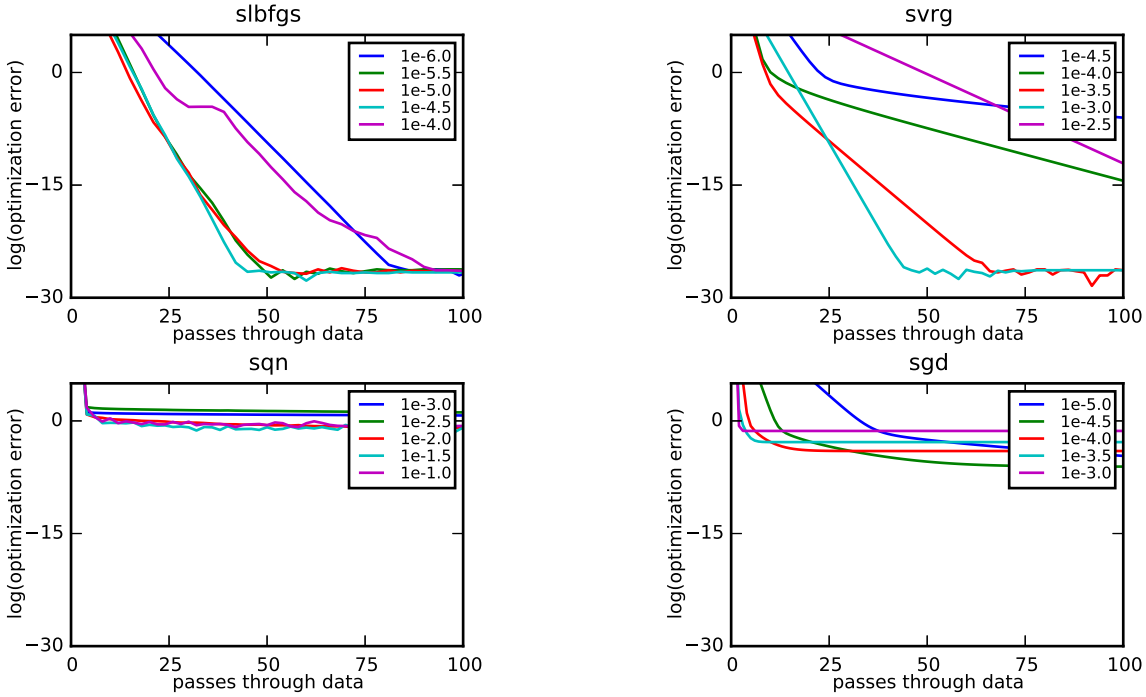

Figure 2: These figures show the log of the optimization error for SLBFGS, SVRG, SQN, and SGD on a ridge regression problem (millionsong) for a wide range of step sizes.

5 Related Work

There is a large body of work that attempts to improve on stochastic gradient descent by reducing variance. Shalev-Shwartz and Zhang (2013) propose stochastic dual coordinate ascent (SDCA). Roux et al. (2012) propose the stochastic average gradient method (SAG). Johnson and Zhang (2013) propose the stochastic variance reduced gradient (SVRG). Wang et al. (2013) develop an approach based on the construction of control variates. More recently, Frostig et al. (2015) devise an online version of SVRG that uses streaming estimates of the gradient to perform variance reduction.

Similarly, a number of stochastic quasi-Newton methods have been proposed. Bordes et al. (2009) propose a variant of stochastic gradient descent that makes use of second order information. Mokhtari and Ribeiro (2014a) analyze the straightforward application of L-BFGS in the stochastic setting and prove a $O(1/k)$ convergence rate in the strongly-convex setting. Byrd et al. (2014) propose a modified version of L-BFGS in the stochastic setting and prove a $O(1/k)$ convergence rate in the strongly-convex setting. Sohl-Dickstein et al. (2014) propose a stochastic quasi-Newton method for minimizing sums of functions by maintaining a separate approximation of the inverse Hessian for each function in the sum. Schraudolph et al. (2007) develop a stochastic version of L-BFGS for the online convex optimization setting. Wang

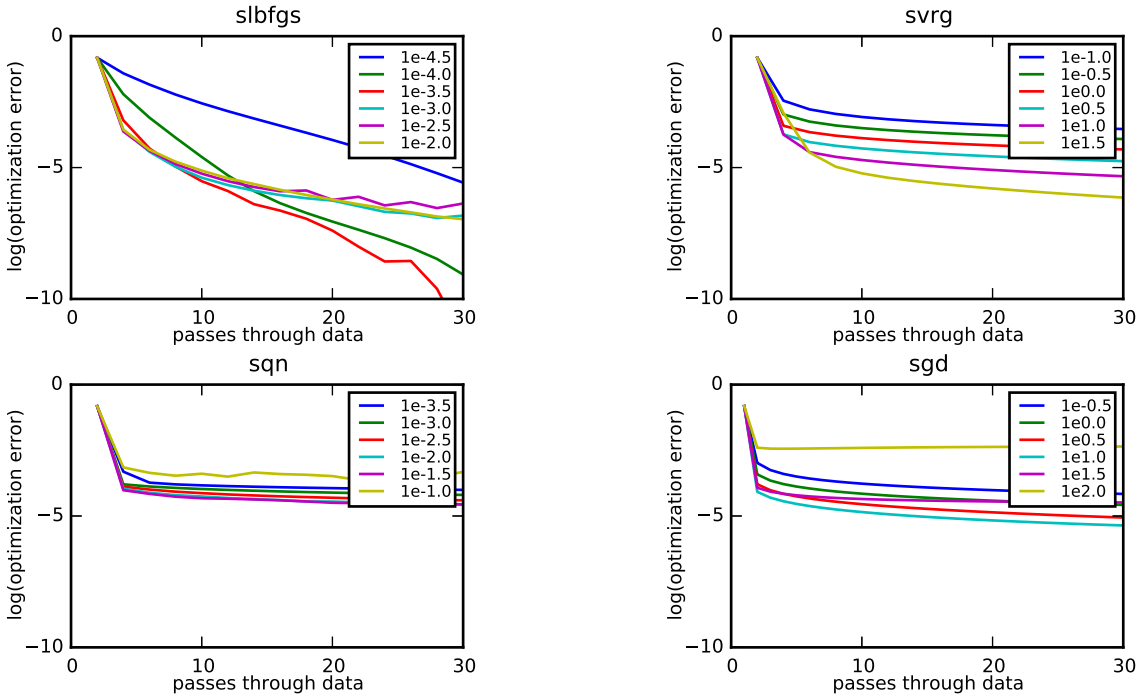

Figure 3: These figures show the log of the optimization error for SLBFGS, SVRG, SQN, and SGD on a support vector machine (RCV1) for a wide range of step sizes.

et al. (2014) prove the convergence of various stochastic quasi-Newton methods in the nonconvex setting. Our work differs from the preceding in that we guarantee a linear rate of convergence.

Lucchi et al. (2015) independently propose a variance-reduction procedure to speed up stochastic quasi-Newton methods and to achieve a linear rate of convergence. Their approach to updating the inverse-Hessian approximation is similar to that of L-BFGS, whereas our method leverages Hessian-vector products to stabilize the approximation.

6 Experimental Results

To probe our theoretical results, we compare Algorithm 1 (SLBFGS) to the stochastic variance-reduced gradient method (SVRG) (Johnson and Zhang, 2013), the stochastic quasi-Newton method (SQN) (Byrd et al., 2014), and stochastic gradient descent (SGD). We evaluate these algorithms on several popular machine learning models, including ridge regression, support vector machines, and matrix completion. Our experiments show the effeciveness of the algorithm on real-world problems that are not neccessarily (strongly) convex.

Because SLBFGS and SVRG require computations of the full gradient, each epoch requires an additional pass through the data. Additionally, SLBFGS and SQN require Hessian-vector-product calculations, each of which is about as expensive as a gradient calculation Pearlmutter (1994). The number of Hessian-vector-product computations per epoch introduced by this is $(b_H N)/(bL)$, which in our experiments is either N or $2N$. To incorporate these additional costs, our plots show error with respect to the number of passes through the data (that is, the number of gradient or Hessian-vector-product computations divided by N). For this reason, the first iterations of SLBFGS, SVRG, SQN, and SGD all begin at different times, with SGD appearing first and SLBFGS appearing last.

For all experiments, we set the batch size b to either 20 or 100, we set the Hessian batch size b_H to $10b$ or $20b$, we set the Hessian update interval L to 10, we set the memory size M to 10, and we set the number of stochastic updates m to N/b. We optimize the learning rate via grid search. SLBFGS and SVRG use a constant step size. For SQN and SGD, we try three different step-size schemes: constant, $1/\sqrt{t}$, and $1/t$, and we report the best one. All experiments are initialized with a vector of zeros, except for the matrix completion problem, where in order to break symmetry, we initialize the experiments with a vector of standard normal random variables scaled by 10^{-5}.

First, we performed ridge regression on the million-song dataset (Bertin-Mahieux et al., 2011) consisting of approximately 4.6×10^5 data points. We set the reg-

ularization parameter $\lambda = 10^{-3}$. In this experiment, both SLBFGS and SVRG rapidly solve the problem to high levels of precision. Second, we trained a support vector machine on RCV1 (Lewis et al., 2004), with approximately 7.8×10^6 data points. We set the regularization parameter to $\lambda = 0$. In this experiment, SGD and SQN make more progress initially as expected, but SLBFGS finds a better optimum. Third, we solve a nonconvex matrix completion problem on the Netflix Prize dataset, as formulated in Recht and Ré (2013), with approximately 10^8 data points. We set the regularization parameter to $\lambda = 10^{-4}$. The poor performance of SVRG and SGD on this problem may be accounted for by the fact that the algorithms are initialized near the vector of all zeros, which is a stationary point (though not the optimum). Presumably the use of curvature information helps SLBFGS and SQN escape the neighborhood of the all zeros vector faster than SVRG and SGD.

Figure 1 plots a comparison of these methods on the three problems. For the convex problems, we plot the logarithm of the optimization error with respect to a precomputed reference solution. For the nonconvex problem, we simply plot the objective value as the global optimum is not necessarily known.

6.1 Robustness to Choice of Step Size

In this section, we illustrate that SLBFGS performs well on convex problems for a large range of step sizes. The windows in which SVRG, SQN, and SGD perform well are much narrower. In Figure 2, we plot the performance of SLBFGS, SVRG, SQN, and SGD for ridge regression on the millionsong dataset for step sizes varying over a couple orders of magnitude. In Figure 3, we show a similar plot for a support vector machine on the RCV1 dataset. In both cases, SLBFGS performs well, solving the problem to a high degree of precision over a large range of step sizes, whereas the performance of SVRG, SQN, and SGD degrade much more rapidly with poor step-size choices.

7 Proofs of Preliminaries

7.1 Proof of Lemma 3

The analysis below closely follows many other analyses of the inverse Hessian approximation used in L-BFGS (Nocedal and Wright, 2006; Byrd et al., 2014; Mokhtari and Ribeiro, 2014a,b), and we include it for completeness.

Note that $s_j^\top y_j = s_j \nabla^2 f_{\mathcal{T}_j}(u_j) s_j$, it follows from Assumption 2 that

$$\lambda \|s_j\|^2 \leq s_j^\top y_j \leq \Lambda \|s_j\|^2. \tag{11}$$

Similarly, letting $z_j = (\nabla^2 f_{\mathcal{T}_j}(u_j))^{1/2} s_j$ and noting that

$$\frac{\|y_j\|^2}{s_j^\top y_j} = \frac{z_j^\top \nabla^2 f_{\mathcal{T}_j}(u_j) z_j}{z_j^\top z_j},$$

Assumption 2 again implies that

$$\lambda \leq \frac{\|y_j\|^2}{s_j^\top y_j} \leq \Lambda. \tag{12}$$

Note that using the Sherman-Morrison-Woodbury formula, we can equivalently write Equation 4 in terms of the Hessian approximation $B_r = H_r^{-1}$ as

$$B_r^{(j)} = B_r^{(j-1)} - \frac{B_r^{(j-1)} s_j s_j^\top B_r^{(j-1)}}{s_j^\top B_r^{(j-1)} s_j} + \frac{y_j y_j^\top}{y_j^\top s_j}. \tag{13}$$

We will begin by bounding the eigenvalues of B_r. We will do this indirectly by bounding the trace and determinant of B_r. We have

$$\begin{aligned}
\operatorname{tr}(B_r^{(j)}) &= \operatorname{tr}(B_r^{(j-1)}) - \frac{\|B_r^{(j-1)} s_j\|^2}{s_j^\top B_r^{(j-1)} s_j} + \frac{\|y_j\|^2}{y_j^\top s_j} \\
&\leq \operatorname{tr}(B_r^{(j-1)}) + \frac{\|y_j\|^2}{y_j^\top s_j} \\
&\leq \operatorname{tr}(B_r^{(j-1)}) + \Lambda.
\end{aligned}$$

The first equality follows from the linearity of the trace operator. The second equality follows from the fact that $\operatorname{tr}(AB) = \operatorname{tr}(BA)$. The fourth relation follows from Equation 12. Since

$$\operatorname{tr}(B_r^{(0)}) = d \frac{\|y_r\|^2}{s_r^\top y_r} \leq d\Lambda,$$

it follows inductively that

$$\operatorname{tr}(B_k) \leq (d + M)\Lambda.$$

Now to bound the determinant, we write

$$\begin{aligned}
\det(B_r^{(j)}) &= \det(B_r^{(j-1)}) \\
&\quad \det\left(I - \frac{s_j s_j^\top B_r^{(j-1)}}{s_j^\top B_r^{(j-1)} s_j} + \frac{(B_r^{(j-1)})^{-1} y_j y_j^\top}{y_j^\top s_j}\right) \\
&= \det(B_r^{(j-1)}) \frac{y_j^\top s_j}{s_j^\top B_r^{(j-1)} s_j} \\
&= \det(B_r^{(j-1)}) \frac{y_j^\top s_j}{\|s_j\|^2} \frac{\|s_j\|^2}{s_j^\top B_r^{(j-1)} s_j} \\
&\geq \det(B_r^{(j-1)}) \frac{\lambda}{\lambda_{\max}(B_r^{(j-1)})} \\
&\geq \det(B_r^{(j-1)}) \frac{\lambda}{\operatorname{tr}(B_r^{(j-1)})} \\
&\geq \det(B_r^{(j-1)}) \frac{\lambda}{(d+M)\Lambda}.
\end{aligned}$$

The first equality uses $\det(AB) = \det(A)\det(B)$. The second equality follows from the identity

$$\begin{aligned} &\det(I + u_1 v_1^\top + u_2 v_2^\top) \\ &= (1 + u_1^\top v_1)(1 + u_2^\top v_2) - (u_1^\top v_2)(v_1^\top u_2) \end{aligned} \tag{14}$$

by setting $u_1 = -s_j$, $v_1 = (B_r^{(j-1)} s_j)/(s_j^\top B_r^{(j-1)} s_j)$, $u_2 = (B_r^{(j-1)})^{-1} y_j$, and $v_2 = y_j/(y_j^\top s_j)$. See Dennis and Moré (1977, Lemma 7.6) for a proof, or simply note that Equation 14 follows from two applications of the identity $\det(A + uv^\top) = (1 + v^\top A^{-1} u)\det(A)$ when $I + u_1 v_1^\top$ is invertible and by continuity when it isn't. The third equality follows by multiplying the numerator and denominator by $\|s_j\|^2$. The fourth relation follows from Equation 11 and from the fact that $s_j^\top B_r^{(j-1)} s_j \leq \lambda_{\max}(B_r^{(j-1)})\|s_j\|^2$. The fifth relation uses the fact that the largest eigenvalue of a positive definite matrix is bounded by its trace. The sixth relation uses the previous bound on $\mathrm{tr}(B_r^{(j-1)})$. Since

$$\det(B_r^{(0)}) = \left(\frac{\|y_r\|^2}{s_r^\top y_r}\right)^d \geq \lambda^d,$$

it follows inductively that

$$\det(B_r) \geq \frac{\lambda^{d+M}}{((d+M)\Lambda)^M}.$$

7.2 Proof of Lemma 4

Using Lemma 3 as well as the fact that H_r is positive definite, we have

$$\lambda_{\max}(B_r) \leq \mathrm{tr}(B_r) \leq (d+M)\Lambda.$$

and

$$\lambda_{\min}(B_r) \geq \frac{\det(B_r)}{\lambda_{\max}(B_r)^{d-1}} \geq \frac{\lambda^{d+M}}{((d+M)\Lambda)^{d+M-1}}.$$

Since we defined $B_r = H_r^{-1}$, it follows that

$$\frac{1}{(d+M)\Lambda} I \preceq H_r \preceq \frac{((d+M)\Lambda)^{d+M-1}}{\lambda^{d+M}} I.$$

7.3 Proof of Lemma 6

Define the function $g_{\mathcal{S}}(w) = f_{\mathcal{S}}(w) - f_{\mathcal{S}}(w_*) - \nabla f_{\mathcal{S}}(w_*)^\top (w - w_*)$ to get the linearization of $f_{\mathcal{S}}$ around the optimum w_*, and note that $g_{\mathcal{S}}$ is minimized at w_*. It follows that for any w, we have

$$\begin{aligned} 0 = g_{\mathcal{S}}(w_*) &\leq g_{\mathcal{S}}\left(w - \frac{1}{\Lambda}\nabla g_{\mathcal{S}}(w)\right) \\ &\leq g_{\mathcal{S}}(w) - \frac{1}{2\Lambda}\|\nabla g_{\mathcal{S}}\|^2. \end{aligned}$$

Rearranging, we have

$$\begin{aligned} &\|\nabla f_{\mathcal{S}}(w) - \nabla f_{\mathcal{S}}(w_*)\|^2 \\ &\leq 2\Lambda(f_{\mathcal{S}}(w) - f_{\mathcal{S}}(w_*) - \nabla f_{\mathcal{S}}(w_*)^\top (w - w_*)). \end{aligned}$$

Averaging over all possible minibatches $\mathcal{S} \subseteq \{1, \ldots, N\}$ of cardinality b and using the fact that $\nabla f(w_*) = 0$, we see that

$$\begin{aligned} &\binom{N}{b}^{-1} \sum_{|\mathcal{S}|=b} \|\nabla f_{\mathcal{S}}(w) - \nabla f_{\mathcal{S}}(w_*)\|^2 \\ &\leq 2\Lambda(f(w) - f(w_*)). \end{aligned} \tag{15}$$

Now, let $\mu_k = \nabla f(w_k)$ and $v_t = \nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_k) + \mu_k$. Conditioning on $\mathcal{F}_{k,t}$ and taking an expectation with respect to $\mathcal{S}$, we find

$$\begin{aligned} \mathbb{E}_{k,t}[\|v_t\|^2] &\leq 2\mathbb{E}_{k,t}[\|\nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_*)\|^2] \\ &\quad + 2\mathbb{E}_{k,t}[\|\nabla f_{\mathcal{S}}(w_k) - \nabla f_{\mathcal{S}}(w_*) - \mu_k\|^2] \\ &\leq 2\mathbb{E}_{k,t}[\|\nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_*)\|^2] \\ &\quad + 2\mathbb{E}_{k,t}[\|\nabla f_{\mathcal{S}}(w_k) - \nabla f_{\mathcal{S}}(w_*)\|^2] \\ &\leq 4\Lambda(f(x_t) - f(w_*) + f(w_k) - f(w_*)). \end{aligned} \tag{16}$$

The first inequality uses the fact that $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$. The second inequality follows by noting that $\mu_k = \mathbb{E}_{k,t}[\nabla f_{\mathcal{S}}(w_k) - \nabla f_{\mathcal{S}}(w_*)]$ and that $\mathbb{E}[\|\xi - \mathbb{E}[\xi]\|^2] \leq \mathbb{E}[\|\xi\|^2]$ for any random variable ξ. The third inequality follows from Equation 15.

8 Discussion

This paper introduces a stochastic version of L-BFGS and proves a linear rate of convergence in the strongly convex case. Theorem 7 captures the qualitatively linear rate of convergence of SLBFGS, which is reflected in our experimental results. We expect SLBFGS to outperform other stochastic first-order methods in poorly conditioned settings where curvature information is valuable as well in settings where we wish to solve the optimization problem to high precision.

There are a number of interesting points to address in future work. The proof of Theorem 7 and many similar proofs used to analyze quasi-Newton methods result in constants that scale poorly with the problem size. At a deeper level, the point of studying quasi-Newton methods is to devise algorithms that lie somewhere along the spectrum from gradient descent to Newton's method, reaping the computational benefits of gradient descent and the rapid convergence of Newton's method. Many of the proofs in the literature, including the proof of Theorem 7, bound the extent to which the quasi-Newton method deviates from gradient descent by bounding the extent to which the

inverse Hessian approximation deviates from the identity matrix. Those bounds are then used to show that the quasi-Newton method does not perform too much worse than gradient descent. A future avenue of research is to study if stochastic quasi-Newton methods can be designed that provably exhibit superlinear convergence as has been done in the non-stochastic case.

References

A. Agarwal, O. Chapelle, M. Dudík, and J. Langford. A reliable effective terascale linear learning system. *The Journal of Machine Learning Research*, 15(1): 1111–1133, 2014.

T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *International Conference on Music Information Retrieval*, 2011.

A. Bordes, L. Bottou, and P. Gallinari. SGD-QN: Careful quasi-Newton stochastic gradient descent. *The Journal of Machine Learning Research*, 10: 1737–1754, 2009.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics*, pages 177–186, 2010.

L. Bottou and Y. LeCun. Large scale online learning. *Advances in neural information processing systems*, 16:217, 2004.

R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-Newton method for large-scale optimization. *arXiv preprint arXiv:1401.7020*, 2014.

A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems 27*, pages 1646–1654, 2014.

R. S. Dembo and T. Steihaug. Truncated-Newton algorithms for large-scale unconstrained optimization. *Mathematical Programming*, 26(2):190–212, 1983.

R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982.

J. E. Dennis, Jr and J. J. Moré. Quasi-Newton methods, motivation and theory. *SIAM review*, 19(1): 46–89, 1977.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

R. Frostig, R. Ge, S. M. Kakade, and A. Sidford. Competing with the empirical risk minimizer in a single pass. In *Conference on Learning Theory*, 2015.

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

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.

D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

A. Lucchi, B. McWilliams, and T. Hofmann. A variance reduced stochastic Newton method. *arXiv preprint arXiv:1503.08316*, 2015.

A. Mokhtari and A. Ribeiro. Global convergence of online limited memory BFGS. *arXiv preprint arXiv:1409.2045*, 2014a.

A. Mokhtari and A. Ribeiro. RES: Regularized stochastic BFGS algorithm. *IEEE Transactions on Signal Processing*, 62(23):6089–6104, 2014b.

Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120 (1):221–259, 2009.

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

B. A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1):147–160, 1994.

B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

N. L. Roux, M. Schmidt, and F. R. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems*, pages 2663–2671, 2012.

N. N. Schraudolph, J. Yu, and S. Günter. A stochastic quasi-Newton method for online convex optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 436–443, 2007.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss. *The Journal of Machine Learning Research*, 14(1):567–599, 2013.

J. Sohl-Dickstein, B. Poole, and S. Ganguli. Fast large-scale optimization by unifying stochastic gradient and quasi-Newton methods. In *International Conference on Machine Learning*, 2014.

I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147, 2013.

C. Wang, X. Chen, A. J. Smola, and E. P. Xing. Variance reduction for stochastic gradient optimization. In *Advances in Neural Information Processing Systems*, pages 181–189, 2013.

X. Wang, S. Ma, and W. Liu. Stochastic quasi-Newton methods for nonconvex stochastic optimization. *arXiv preprint arXiv:1412.1196*, 2014.