Improved SVRG for Non-Strongly-Convex or Sum-of-Non-Convex Objectives

Zeyuan Allen-Zhu ZEYUAN@CSAIL.MIT.EDU
Princeton University

Yang Yuan YANGYUAN@CS.CORNELL.EDU
Cornell University

Abstract

Many classical algorithms are found until several years later to outlive the confines in which they were conceived, and continue to be relevant in unforeseen settings. In this paper, we show that SVRG is one such method: being originally designed for strongly convex objectives, it is also very robust in non-strongly convex or sum-of-non-convex settings.

More precisely, we provide new analysis to improve the state-of-the-art running times in both settings by either applying SVRG or its novel variant. Since non-strongly convex objectives include important examples such as Lasso or logistic regression, and sum-of-non-convex objectives include famous examples such as stochastic PCA and is even believed to be related to training deep neural nets, our results also imply better performances in these applications.[1]

1. Introduction

The fundamental algorithmic problem in optimization is to design efficient algorithms for solving certain *classes* of problems. By distinguishing between smooth and non-smooth functions, between weakly-convex and strongly-convex functions, between proximal and non-proximal functions, or even between convex and non-convex functions, the number of classes grows exponentially and it may be unrealistic to design a new algorithm for each specific class. Taking into account such "design complexity", it is beneficial to design a single method the works for multiple classes, or perhaps even more beneficial if this method is already widely used and happens to outlive the confines it was originally designed for. Easier done in practice, providing a support *theory* unifying the underlying classes for a specific method is particularly exciting, challenging, and sometimes even enlightening: the theoretical findings may further suggest experimentalists regarding how such a method should be best tuned in practice.

In this paper, we revisit the SVRG method by Johnson and Zhang (Johnson & Zhang, 2013) and explore its applications to either a non-strongly convex objective, or a *sum-of-non-convex* objective, or even both. We show faster convergence results for minimizing such objectives by either directly applying SVRG or modifying it in a novel manner.

Consider the following composite convex minimization:

$$\min_{x\in\mathbb{R}^d}\left\{F(x)\stackrel{\text{def}}{=}f(x)+\Psi(x)\stackrel{\text{def}}{=}\frac{1}{n}\sum_{i=1}^{n}f_i(x)+\Psi(x)\right\}\ . \tag{1.1}$$

Here, $f(x)=\frac{1}{n}\sum_{i=1}^{n}f_i(x)$ is a convex function that is written as a finite average of n smooth functions $f_i(x)$,[2] and $\Psi(x)$ is a relatively simple (but possibly non-differentiable) convex function, sometimes referred to as the *proximal* function. Suppose we are interested in finding an approximate minimizer $x\in\mathbb{R}^d$ satisfying $F(x)\le F(x^*)+\varepsilon$, where x^* is a minimizer of $F(x)$.

Examples. Problems of this form arise in many places in machine learning, statistics, and operations research. For instance, many *regularized empirical risk minimization (ERM)* problems fall into this category with convex $f_i(\cdot)$. In such problems, we are given n training examples $\{(a_1,\ell_1),\dots(a_n,\ell_n)\}$, where each $a_i\in\mathbb{R}^d$ is the feature vector of example i, and each $\ell_i\in\mathbb{R}$ is the label of example i. The following classification and regression problems are well-known examples of ERM:

- Ridge Regression: $f_i(x)=\frac{1}{2}(\langle a_i,x\rangle-\ell_i)^2+\frac{\sigma}{2}\|x\|_2^2$ and $\Psi(x)=0$.

[1] The full version of this paper can be found on `http://arxiv.org/abs/1506.01972`.

[2] In fact, even if each $f_i(x)$ is not smooth but only Lipschitz continuous, standard smoothing techniques such as Chapter 2.3 of (Hazan, 2015) can make each $f_i(x)$ smooth without sacrificing too much accuracy.

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

- Lasso: $f_i(x) = \frac{1}{2}(\langle a_i, x\rangle - \ell_i)^2$ and $\Psi(x) = \sigma\|x\|_1$.
- ℓ_1-Regularized Logistic Regression: $f_i(x) = \log(1 + \exp(-\ell_i\langle a_i, x\rangle))$ and $\Psi(x) = \sigma\|x\|_1$.

Another important problem that falls into this category is the *principle component analysis (PCA)* problem. Suppose we are given n data vectors $a_1, \dots, a_n \in \mathbb{R}^d$, denoting by $A = \frac{1}{n}\sum_{i=1}^n a_i a_i^T$ the normalized covariance matrix, Garber and Hazan (Garber & Hazan, 2015) showed that approximately finding the principle component of A is equivalent to minimizing $f(x) = \frac{1}{2}x^T(\mu I - A)x$ for some suitably chosen parameter $\mu > 0$. Therefore, defining $f_i(x) \stackrel{\text{def}}{=} \frac{1}{2}x^T(\mu I - a_i a_i^T)x$ and $\Psi(x) = 0$, this falls into Problem (1.1) with non-convex functions $f_i(\cdot)$.

Background of SVRG. Stochastic first-order methods perform the following updates to solve Problem (1.1):

$$x_{t+1} \leftarrow \arg\min_{y\in\mathbb{R}^d} \left\{ \frac{1}{2\eta}\|y - x_t\|_2^2 + \langle \xi_t, y\rangle + \Psi(y) \right\} ,$$

where η is the step length, and ξ_t is a random vector satisfying $\mathbb{E}[\xi_t] = \nabla f(x_t)$ which is referred to as the *stochastic gradient*. If the proximal function $\Psi(y)$ equals zero, the update simply reduces to $x_{t+1} \leftarrow x_t - \eta\xi_t$.

Given the "finite average" structure $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$, a classical choice is to set $\xi_t = \nabla f_i(x_t)$ for some random index $i \in [n]$ per iteration. Methods based on this choice are known as *stochastic gradient descent (SGD)*.

More recently, the convergence speed of SGD has been further improved with the *variance-reduction* technique (Johnson & Zhang, 2013; Shalev-Shwartz & Zhang, 2013; Schmidt et al., 2013; Shalev-Shwartz & Zhang, 2012; Xiao & Zhang, 2014; Defazio et al., 2014; Mairal, 2015). In all of these cited results, the authors have, in one way or another, shown that SGD can converge much faster if one makes a better choice of the stochastic gradient ξ_t, so that its variance $\mathbb{E}[\|\xi_t - \nabla f(x_t)\|_2^2]$ reduces as t increases.

One particular way to reduce the variance is the SVRG method described as follows (Johnson & Zhang, 2013). Keep a snapshot $\widetilde{x} = x_t$ after every m stochastic update steps (where m is some parameter), and compute the full gradient $\nabla f(\widetilde{x})$ only for such snapshots. Then, set $\xi_t = \nabla f_i(x_t) - \nabla f_i(\widetilde{x}) + \nabla f(\widetilde{x})$ as the stochastic gradient. One can verify that, under this choice of ξ_t, it satisfies $\mathbb{E}[\xi_t] = \nabla f(x_t)$ and $\lim_{t\to\infty}\mathbb{E}[\|\xi_t - \nabla f(x_t)\|_2^2] = 0$.

Non-Strongly Convex Objectives. Although many variance-reduction based methods have been proposed, most of them, including SVRG, only has convergence guarantee of Problem (1.1) when the objective $F(x)$ is strongly convex. However, in many machine learning applications, $F(x)$ is simply *not* strongly convex. This is particularly true for Lasso (Tibshirani, 1996) and ℓ_1-Regularized Logistic Regression (Ng, 2004), two cornerstone problems extensively used for feature selections.

One way to get around this is to add a dummy regularizer $\frac{\lambda}{2}\|x\|_2^2$ to $F(x)$, and then apply any of the above methods. However, the weight of this regularizer, λ, needs to be chosen before the algorithm starts. This adds a lot of difficulty when applying such methods to real life: (1) one needs to tune λ by repeatedly executing the algorithm, and (2) the error of the algorithm does not converge to zero as time goes (in fact, it converges to $O(\lambda)$ so one needs to know the desired accuracy before the algorithm starts). Perhaps more importantly, adding the dummy regularizer hurts the performance of the algorithm both in theory and practice.

Another possible solution is to tackle the non-strongly convex case *directly* (Schmidt et al., 2013; Defazio et al., 2014; Mairal, 2015), without using any dummy regularizer. These methods are the so-called *anytime* algorithms: they can be interrupted at any time, and the training error tends to zero as the number of iterations increases.

While direct methods are much more convenient for practical uses, existing direct methods are much slower than indirect methods (i.e., methods via dummy regularization) at least in theory. More specifically, if the desired accuracy is ε and the smoothness of each $f_i(x)$ is L, then the *gradient complexities* [3] of the best known direct and indirect methods are respectively

$$O\Big(\frac{n+L}{\varepsilon}\Big) \qquad \text{and} \qquad O\Big((n+\frac{L}{\varepsilon})\log\frac{1}{\varepsilon}\Big) .$$

Therefore in theory, when n is usually dominating, indirect methods are faster but less convenient, while direct methods are slower but more convenient.

In this paper, we propose SVRG^{++}, a new method that solves the non-strongly convex case of Problem (1.1) *directly* with gradient complexity $O(n\log\frac{1}{\varepsilon} + \frac{L}{\varepsilon})$, outperforming both known direct and indirect methods. In particular, our complexity outperforms known direct methods (e.g., SAGA or SAG) by a factor $\widetilde{\Omega}(n/L)$ in the case when $L \le n$. Since L is usually on the order of $O(s)$ for large-scale machine learning problems where s is the sparsity of feature vectors and s can be much smaller than n, we claim that this outperformance may be significant in theory. On the practical side, SVRG^{++} is a direct, anytime method, which is convenient to use. We describe SVRG^{++} and the main techniques we use in Section 4.

Sum-of-Non-Convex Objectives. If $f(x)$ is σ-strongly convex while each $f_i(x)$ is non-convex but L-smooth, Shalev-Shwartz discovered that the SVRG method admits a gradient complexity of $O\big((n+\frac{L^2}{\sigma^2})\log\frac{1}{\varepsilon}\big)$ for minimizing $F(x)$ (Shalev-Shwartz, 2015) in the case of $\Psi(x) = 0$. A similar result has been independently re-discovered by Garber and Hazan (Garber & Hazan, 2015) and applied to

[3] Throughout this paper, we will use *gradient complexity* as an effective measure of an algorithm's running time. Usually, the total running time of an algorithm is $O(d)$ multiplied with its gradient complexity, because each $\nabla f_i(x)$ can be computed in $O(d)$ time.

the PCA problem. This setting is also believed to be happening (at least locally) on training deep neural nets (Johnson & Zhang, 2013; Shalev-Shwartz, 2015; Allen-Zhu & Hazan, 2016a).

Despite the missing proximal term $\Psi(x)$ in their analysis, the running time above is imperfect for two reasons.

- First, this complexity is not stable: even if we modify only one of $f_i(x)$ from convex to (a little bit) non-convex, the best known gradient complexity for SVRG immediately worsens to $O\big((n+\frac{L^2}{\sigma^2})\log\frac{1}{\varepsilon}\big)$ from $O\big((n+\frac{L}{\sigma})\log\frac{1}{\varepsilon}\big)$. In contrast, one should expect a more graceful decay of the performance as a function on the "magnitude" of the non-convexity, or perhaps even a *threshold* where the performance is totally unaffected if the magnitude is "below" this threshold.
- Second, the complexity does not take into account the asymmetry in smoothness. For instance, in PCA applications, each $f_i(x)$ can be very non-convex and its Hessian has eigenvalues between $-l < 0$ and $L > 0$ where l can be significantly larger than L. Can we take advantage of this asymmetry to get better running time?

In this paper, we prove that if each $f_i(x)$ is L-upper smooth and l-lower smooth (which means the Hessian of $f_i(x)$ has eigenvalues bounded between $[-l, L]$), the same SVRG method admits a gradient complexity of $O\big((n+\frac{L}{\sigma}+\frac{Ll}{\sigma^2})\log\frac{1}{\varepsilon}\big)$. This resolves both our aforementioned concerns. First, if $l = O(\sigma)$, our new result suggests that the convergence of SVRG is asymptotically the *same* as the convex case, meaning there is a threshold $O(\delta)$ that SVRG allows each $f_i(x)$ to be non-convex below this threshold for free. Second, in the $l > L$ case, our result implies a linear dependence on the non-convexity parameter l, rather than the quadratic one $O\big((n+\frac{l^2}{\sigma^2})\log\frac{1}{\varepsilon}\big)$ shown by prior work (Shalev-Shwartz, 2015; Garber & Hazan, 2015). To the best of our knowledge, this is the first time that upper and lower smoothness parameters are distinguished in order to prove convergence results for minimizing (1.1).

Our improvement on SVRG immediately leads to faster stochastic algorithms for PCA (Shamir, 2015; Garber & Hazan, 2015). Assume that $A = \frac{1}{n}\sum_{i=1}^n a_i a_i^T$ is a normalized covariance matrix where each $a_i \in \mathbb{R}^d$ has Euclidean norm at most 1. Let $\lambda \in [0,1]$ be the largest eigenvalue of A. Garber and Hazan showed that computing the leading eigenvector of A is, up to binary search preprocessing, equivalent to the sum-of-non-convex form of Problem (1.1), with upper smoothness $L = \lambda$ and lower smoothness $l = 1$.[4] Garber and Hazan further applied SVRG to minimize this objective and proved an overall running time $O\big((nd+\frac{d}{\delta^2})\log\frac{1}{\varepsilon}\big)$. Our result improves this running time to $O\big((nd+\frac{\lambda d}{\delta^2})\log\frac{1}{\varepsilon}\big)$. Since λ may be as small as $1/d$, this speed up is significant in theory.[5]

Since the original publication of this paper, our above PCA speed-up has also been translated to k-SVD, which is to compute the first k singular vectors of a given matrix (Allen-Zhu & Li, 2016).

Our results above are non-accelerated for the sum-of-non-convex setting. One can apply Catalyst (Lin et al., 2015; Frostig et al., 2015) to further improve its running time when σ is very small. Not surprisingly, our performance improvement carries to the accelerated setting as well.

Finally, we also prove that our proposed improvements on SVRG (for non-strongly convex objectives and for sum-of-non-convex objectives) can be put together, leading to a new algorithm $\texttt{SVRG}_{\texttt{nc}}^{++}$ that works for both non-strongly convex and sum-of-non-convex objectives. This gives faster algorithms and can be found in the full paper.

Roadmap. We discuss related work in Section 2 and provide notational background in Section 3. We state our result for non-strongly convex objectives in Section 4, for sum-of-non-convex objectives in Section 5 and 6. In Section 7 we perform experiments supporting our theory. Most of the technical proofs, as well as our $\texttt{SVRG}_{\texttt{nc}}^{++}$ method for solving both non-strongly convex and sum-of-non-convex objectives, are included in the full paper.

2. Other Related Work

The first published variance-reduction method is SAG (Schmidt et al., 2013). SAG obtains an $O(\log(1/\varepsilon))$ convergence (i.e., linear convergence) for strongly convex and smooth objectives, comparing to the $O(1/\varepsilon)$ rate of SGD (Hazan et al., 2007; Shalev-Shwartz & Singer, 2007). This $O(\log(1/\varepsilon))$ rate has also been obtained by several concurrent or subsequent works, such as SVRG, MISO and SAGA (Johnson & Zhang, 2013; Mairal, 2015; Defazio et al., 2014). SDCA (Shalev-Shwartz & Zhang, 2013) has also been discovered to be intrinsically performing some "variance reduction" procedure (Shalev-Shwartz, 2015).

Among the variance-reduction algorithms, only SAG, MISO, and SAGA can provide theoretical guarantees for directly solving non-strongly convex objectives (i.e., without adding a dummy regularizer). The best gradient complexity for direct methods before our work is $O(\frac{n+L}{\varepsilon})$ due

[4] Suppose that the eigengap between largest and second largest eigenvalues of A is $\delta = \lambda - \lambda_2$. Garber and Hazan showed that computing the principle component of A is, up to binary search preprocessing, equivalent to minimizing the objective $f(x) \stackrel{\text{def}}{=} \frac{1}{2}x^T(\mu I - A)x + b^T x$ where $\mu = \lambda + \delta$. If one defines $f_i(x) \stackrel{\text{def}}{=} \frac{1}{2}x^T(\mu I - a_i a_i^T)x + b^T x$, this minimization problem falls into the sum-of-non-convex setting of Problem (1.1), with upper smoothness $L = \mu \approx \lambda$ and lower smoothness $l = 1$.

[5] Garber and Hazan also applied acceleration schemes on top of SVRG, and obtained a running time $\widetilde{O}(\frac{n^{3/4}d}{\sqrt{\delta}})$. We can do the same thing here and improve their running time to $\widetilde{O}(\frac{n^{3/4}\lambda^{1/4}d}{\sqrt{\delta}})$ in the accelerated setting.

to SAG and SAGA. On the other hand, if one uses indirect methods, the best gradient complexity is $O\big((n+\frac{L}{\varepsilon})\log\frac{1}{\varepsilon}\big)$, where the asymptotic dependence on ε is weakened to $\frac{\log(1/\varepsilon)}{\varepsilon}$.

We work directly with smooth functions $f_i(x)$ rather than the more structured $f_i(x) \stackrel{\text{def}}{=} \phi_i(\langle x, a_i\rangle)$. In the structured case, AccSDCA (Shalev-Shwartz & Zhang, 2014), along with subsequent works (Lin et al., 2014; Zhang & Xiao, 2015), obtains a slightly better gradient complexity $O\big(\big(n+\min\big\{L/\varepsilon, \sqrt{nL/\varepsilon}\big\}\big)\log\frac{1}{\varepsilon}\big)$ for non-strongly convex objectives. This class of methods require one to work with the dual of the objective, require one to add dummy regularizer for non-strongly convex objectives (i.e., are indirect), and run only faster than the variance-reduction based methods when $n < \sqrt{L/\varepsilon}$.

Since the original submission of this paper, we learned several other related works from the anonymous reviewers. First, the SVRG method was independently discovered and published also by (Zhang et al., 2013). Second, the result of (Mahdavi et al., 2013) also uses doubling-epoch technique and can partially infer our results on $\texttt{SVRG}^{++}$ with a slightly more complicated proof and different algorithm.[6] Third, in a concurrent accepted paper to this ICML, Garber et al. (Garber et al., 2016) improved the original Garber-Hazan PCA result (Garber & Hazan, 2015) and thus solved a special case of our Theorem 6.1; their result has nothing to do with other theorems in this paper, especially Theorem 5.1 and D.1.[7]

In some concurrent works, the authors of (Allen-Zhu & Hazan, 2016b) obtained our same running time on $\texttt{SVRG}^{++}$ through reductions. However, their algorithm is not a direct one so cannot be practically as good as $\texttt{SVRG}^{++}$. Also after this paper is accepted, the author of (Allen-Zhu, 2016) provided a direct method for solving (1.1) but in an accelerated speed. As mentioned in (Allen-Zhu, 2016), his method can be combined with the technique in this paper to obtain a non-strongly convex accelerated running time.

3. Notations

Throughout this paper, we denote by $\|\cdot\|$ the Euclidean norm. We assume that each $f_i(\cdot)$ is differentiable and $\Psi(\cdot)$ is convex and lower semicontinuous.

We say that a differentiable function $f_i(\cdot)$ is *L-smooth* (or has L-Lipschitz continuous gradient) if:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \le L\|x-y\| \quad \forall x, y \in \mathbb{R}^d \ .$$

The above definition has several equivalent forms, and one of them says for all $x, y \in \mathbb{R}^d$:

$$-\frac{L}{2}\|y-x\|^2 \le f(y) - \big(f(x)+\langle\nabla f(x), y-x\rangle\big) \le \frac{L}{2}\|y-x\|^2 \ .$$

In this paper, we say $f_i(\cdot)$ is L-upper smooth if it satisfies

$$f(y) - \big(f(x)+\langle\nabla f(x), y-x\rangle\big) \le \frac{L}{2}\|y-x\|^2 \quad \forall x, y \in \mathbb{R}^d \ ,$$

and $f_i(\cdot)$ is l-lower smooth if it satisfies

$$f(y) - \big(f(x)+\langle\nabla f(x), y-x\rangle\big) \ge -\frac{l}{2}\|y-x\|^2 \quad \forall x, y \in \mathbb{R}^d \ .$$

Let us give a few examples: a convex differentiable function is 0-lower smooth; an L-smooth function is L-upper and L-lower smooth; a convex L-smooth function is L-upper and 0-lower smooth.

We say a function $f(\cdot)$ is σ-strongly convex if

$$f(y) - \big(f(x)+\langle\nabla f(x), y-x\rangle\big) \ge \frac{\sigma}{2}\|y-x\|^2 \quad \forall x, y \in \mathbb{R}^d \ .$$

Note that for a twice differentiable function f, the above definitions are equivalent to the corresponding statements about the eigenvalues of $\nabla^2 f(x)$. Indeed, L-upper smoothness is equivalent to saying all eigenvalues are no more than L, l-lower smoothness is equivalent to saying all eigenvalues are no less than $-l$, and σ-strong convexity is saying all eigenvalues are at least σ.

4. $\texttt{SVRG}^{++}$ for Non-Strongly Convex Objectives

In this section we consider the case of Problem (1.1) when each $f_i(x)$ is a convex function and the objective is not necessarily strongly convex. Recall that this class of problems include Lasso and logistic regression as notable examples.

We propose our $\texttt{SVRG}^{++}$ algorithm for solving this case, see Algorithm 1. Given an initial vector x^ϕ, our algorithm is divided into S epochs. The s-th epoch consists of m_s stochastic gradient steps (see Line 8 of $\texttt{SVRG}^{++}$), where m_s doubles between every consecutive two epochs. This "doubling" feature distinguishes our method from all of the cited variance-reduction based methods.

Within each epoch, similar to SVRG, we compute the full gradient $\widetilde{\mu}_{s-1} = \nabla f(\widetilde{x}^{s-1})$ where $\widetilde{x}^{s-1}$ is the average point of the previous epoch. We then use $\widetilde{\mu}_{s-1}$ to define the variance-reduced stochastic gradient ξ, see Line 7 of $\texttt{SVRG}^{++}$. Unlike SVRG, our starting vector x_0^s of each epoch is set to be the ending vector $x_{m_{s-1}}^{s-1}$ of the previous epoch, rather than the average of the previous epoch.[8]

[6] Mahdavi et al. studied an oracle model where there are two gradient oracles, a stochastic one and a full-gradient one. Then, they prove comparable bounds to $\texttt{SVRG}^{++}$ but without supporting proximal terms and therefore do not directly apply to ERM problems such as Lasso or logistic regression.

[7] For the PCA problem, they produced the same $O\big((nd + \frac{\lambda d}{\delta^2})\log\frac{1}{\varepsilon}\big)$ running time as we do; however, their result is only about PCA so does not solve general sum-of-non-convex objectives; they also did not introduce upper or lower smoothness like we do.

[8] The theoretical convergence of SVRG relies on its Option II, that is to set the beginning vector of each epoch to be the *average* (or a random) vector of the previous epoch. However, the authors of SVRG conduct their experiment using the last vector rather

Algorithm 1 $\texttt{SVRG}^{++}(x^\phi, m_0, S, \eta)$

$\widetilde{x}^0 \leftarrow x^\phi$, $x_0^1 \leftarrow x^\phi$
for $s \leftarrow 1$ **to** S **do**
  $\widetilde{\mu}_{s-1} \leftarrow \nabla f(\widetilde{x}^{s-1})$
  $m_s \leftarrow 2^s \cdot m_0$
  for $t \leftarrow 0$ **to** $m_s - 1$ **do**
    Pick i uniformly at random in $\{1, \cdots, n\}$.
    $\xi \leftarrow \nabla f_i(x_t^s) - \nabla f_i(\widetilde{x}^{s-1}) + \widetilde{\mu}_{s-1}$
    $x_{t+1}^s = \arg\min_{y \in \mathbb{R}^d} \left\{ \frac{1}{2\eta} \|x_t^s - y\|^2 + \Psi(y) + \langle \xi, y \rangle \right\}$
  end for
  $\widetilde{x}^s \leftarrow \frac{1}{m_s} \sum_{t=1}^{m_s} x_t^s$
  $x_0^{s+1} \leftarrow x_{m_s}^s$
end for
return $\widetilde{x}^S$.

We state our main result for $\texttt{SVRG}^{++}$ as follows:

Theorem 4.1. *If each $f_i(x)$ is convex in Problem (1.1), then $\texttt{SVRG}^{++}(x^\phi, m_0, S, \eta)$ satisfies if m_0 and S are positive integers and $\eta = 1/(7L)$, then*

$$\mathbb{E}[F(\widetilde{x}^S) - F(x^*)] \le O\Big(\frac{F(x^\phi) - F(x^*)}{2^S} + \frac{L\|x^\phi - x^*\|^2}{2^S m_0}\Big). \tag{4.1}$$

In addition, $\texttt{SVRG}^{++}$ has a gradient complexity of $O(S \cdot n + 2^S \cdot m_0)$.

As a result, given an initial vector x^ϕ satisfying $\|x^\phi - x^*\|^2 \le \Theta$ and $F(x^\phi) - F(x^*) \le \Delta$ for parameters $\Theta, \Delta \in \mathbb{R}_+$, by setting $S = \log_2(\Delta/\varepsilon)$, $m_0 = L\Theta/\Delta$, and $\eta = 1/(7L)$, we obtain an $O(\varepsilon)$ approximate minimizer of $F(\cdot)$ with a total gradient complexity $O\big(n \log\big(\frac{\Delta}{\varepsilon}\big) + \frac{L\Theta}{\varepsilon}\big)$.

Our proof of Theorem 4.1 is included in the full paper.

4.1. Additional Improvements

Inspired by $\texttt{SVRG}^{++}$, we also introduce SVRG_Auto_Epoch, a variant of $\texttt{SVRG}^{++}$ where epoch length is automatically determined instead of doubled every epoch. Auto epoch is an attractive feature in practice because it enables the algorithm to perform well for different types of objectives.

The criterion we use to determine the termination of epoch s in SVRG_Auto_Epoch is based on the quality of the snapshot full gradient $\nabla f(\widetilde{x}^{s-1})$. Intuitively, if epoch length is too long, an algorithm may move too far from the snapshot point, meaning that the gradient estimator ξ may have a large variance. Following this intuition, for every iteration t, we record $\texttt{diff}_t = \|\nabla f_i(x_t^s) - \nabla f_i(\widetilde{x}^{s-1})\|_2^2$ because $\mathbb{E}_i[\texttt{diff}_t]$ is a very tight upper bound on the variance of the gradient estimator (see the proof of Lemma A.2). Under this notion, we decide the epoch termination of SVRG_Auto_Epoch as follows. Each epoch has a minimum length of $n/4$. From iteration $t = n/4$ onwards, we keep track of the average $\texttt{diff}_t$ in the last $n/4$ iterations, i.e., $\sum_{j=t-n/4+1}^{t} \texttt{diff}_j$. If this quantity is greater than half of the average $\texttt{diff}_j$ recorded from the previous epoch, we terminate the current epoch and start a new one.[9] SVRG_Auto_Epoch shows good performance in our experiments, and we leave it as an open question to prove a complexity result for this method.

In addition to auto epoch, $\texttt{SVRG}^{++}$ can also be combined with other enhancements proposed for SVRG. For example, (Harikandeh et al., 2015) saves the time to compute full gradients at snapshot points by making them less accurate in the first a few epochs. (Konecný et al., 2014) uses mini-batch gradients per iteration to further decrease the variance. These ideas are orthogonal to our proposed techniques and therefore can be applied to further improve the performance of $\texttt{SVRG}^{++}$.

5. SVRG for Sum-of-Non-Convex Objectives I: Small Lower Smoothness

In this section we consider Problem (1.1) when each $f_i(x)$ is not necessarily convex, L-upper smooth, and l-lower smooth for some $0 \le l \le L$. We assume that $f(\cdot)$ is σ-strongly convex. For this class of objectives, the best known gradient complexity for stochastic gradient methods is $O\big((n + \frac{L^2}{\sigma^2}) \log \frac{1}{\varepsilon}\big)$ due to SVRG (Shalev-Shwartz, 2015).

This gradient complexity is essentially a factor L/σ greater than that for the convex case, that is $O\big((n + \frac{L}{\sigma}) \log \frac{1}{\varepsilon}\big)$. Following the intuition discussed in the introduction, we improve it to $O\big((n + \frac{L}{\sigma} + \frac{Ll}{\sigma^2}) \log \frac{1}{\varepsilon}\big)$, a quantity that is asymptotically the same as the convex setting when $l \le O(\sigma)$, and linearly degrades as l increases.

Recall that the original SVRG (Option II) works as follows (see Algorithm 2 for completeness). Given an initial vector x^ϕ, SVRG is divided into S epochs, each of length m for the same m across epochs. Within each epoch, SVRG computes the full gradient $\widetilde{\mu}_{s-1} = \nabla f(\widetilde{x}^{s-1})$ where $\widetilde{x}^{s-1}$ is the average point of the previous epoch. Then, SVRG uses $\widetilde{\mu}_{s-1}$ to define the variance-reduced version of the stochastic gradient ξ, see Line 6 of Algorithm 2. The starting vector x_0^s of each epoch is set to be the average vector of the previous epoch.[10]

We state our main result for SVRG in this section as follows:

than the average because it is more "natural". This present paper partially shows that this natural choice also has competitive performance, and therefore confirms the empirical finding of SVRG. (Similar result can also be obtained for the strongly convex case, which we exclude for simplicity.)

[9] We always set the first epoch to be of length $n/4$ and the second to be of length $n/2$.

[10] This choice of the starting vector is different from $\texttt{SVRG}^{++}$, but was the original choice made by SVRG. Similar result can also be obtained using the choice from $\texttt{SVRG}^{++}$.

Algorithm 2 $\texttt{SVRG}(x^\phi, m, S, \eta)$ (Johnson & Zhang, 2013)

$\widetilde{x}^0 \leftarrow x^\phi$, $x_0^1 \leftarrow x^\phi$
for $s \leftarrow 1$ **to** S **do**
  $\widetilde{\mu}_{s-1} \leftarrow \nabla f(\widetilde{x}^{s-1})$
  for $t \leftarrow 0$ **to** $m-1$ **do**
    Pick i uniformly at random in $\{1, \cdots, n\}$.
    $\xi \leftarrow \nabla f_i(x_t^s) - \nabla f_i(\widetilde{x}^{s-1}) + \widetilde{\mu}_{s-1}$
    $x_{t+1}^s = \arg\min_{y \in \mathbb{R}^d} \big\{ \frac{1}{2\eta}\|x_t^s - y\|^2 + \Psi(y) + \langle \xi, y\rangle \big\}$
  end for
  $\widetilde{x}^s \leftarrow \frac{1}{m}\sum_{t=1}^m x_t^s$
  $x_0^{s+1} \leftarrow \widetilde{x}^s$
end for
return $\widetilde{x}^S$.

Theorem 5.1. *If each $f_i(x)$ is L-upper and l-lower smooth in Problem (1.1) for $0 \le l \le L$, $f(x)$ is σ-strongly convex, $\eta = \min\{\frac{1}{21L}, \frac{\sigma}{63Ll}\}$ and $m \ge \frac{10}{\sigma\eta} = \Omega(\max\{\frac{L}{\sigma}, \frac{Ll}{\sigma^2}\})$, then $\texttt{SVRG}(x^\phi, m, S, \eta)$ satisfies*[a]

$$\mathbb{E}[F(\widetilde{x}^s) - F(x^*)] \le \frac{3}{4}\big(F(\widetilde{x}^{s-1}) - F(x^*)\big) \ . \quad (5.1)$$

Therefore, by setting $S = \log_{4/3}\big(\frac{F(x^\phi)-F(x^*)}{\varepsilon}\big)$, in a total gradient complexity of

$$O\Big(\Big(n + \frac{L}{\sigma}\max\big\{1, \frac{l}{\sigma}\big\}\Big)\log\frac{F(x^\phi)-F(x^*)}{\varepsilon}\Big) \ ,$$

we obtain an output $\widetilde{x}^S$ satisfying $\mathbb{E}[F(\widetilde{x}^s) - F(x^*)] \le \varepsilon$.

[a] Here we have assumed that the first $s-1$ epochs are fixed and the only randomness comes from epoch s.

The full proof of Theorem 5.1 is included in the full paper.

6. SVRG for Sum-of-Non-Convex Objectives II: Large Lower Smoothness

In this section we consider Problem (1.1) when each $f_i(x)$ is not necessarily convex, L-upper smooth, and l-lower smooth function for some $l \ge L$. We assume $f(\cdot)$ is σ-strongly convex. For this class of objectives, the best known gradient complexity for stochastic gradient methods is $O\big((n + \frac{l^2}{\sigma^2})\log\frac{1}{\varepsilon}\big)$ due to (Shalev-Shwartz, 2015).

This known gradient complexity is essentially a factor $l^2/L^2 \ge 1$ worse than that of the symmetric case (i.e., the case when $l = L$). In this section, we improve this factor to l/L which is quadratically faster than l^2/L^2. As we have explained in the introduction, this result improves the convergence for the best known stochastic algorithm for PCA.

We state our main result for SVRG in this section as follows, and its proof is included in the full paper.

Theorem 6.1. *If each $f_i(x)$ is L-upper and l-lower smooth in Problem (1.1) for $l \ge L \ge 0$, $f(x)$ is σ-strongly convex, $\eta = \frac{\sigma}{25Ll}$ and $m \ge \frac{4}{\sigma\eta} = \Omega(\frac{Ll}{\sigma^2})$, then $\texttt{SVRG}(x^\phi, m, S, \eta)$ satisfies*

$$\mathbb{E}[F(\widetilde{x}^s) - F(x^*)] \le \frac{3}{4}\big(F(\widetilde{x}^{s-1}) - F(x^*)\big) \ . \quad (6.1)$$

Therefore, by setting $S = \log_{4/3}\big(\frac{F(x^\phi)-F(x^*)}{\varepsilon}\big)$, in a total gradient complexity of

$$O\Big(\Big(n + \frac{Ll}{\sigma^2}\Big)\log\frac{F(x^\phi)-F(x^*)}{\varepsilon}\Big) \ ,$$

we obtain an output $\widetilde{x}^S$ satisfying $\mathbb{E}[F(\widetilde{x}^s) - F(x^*)] \le \varepsilon$.

7. Experiments on ERM

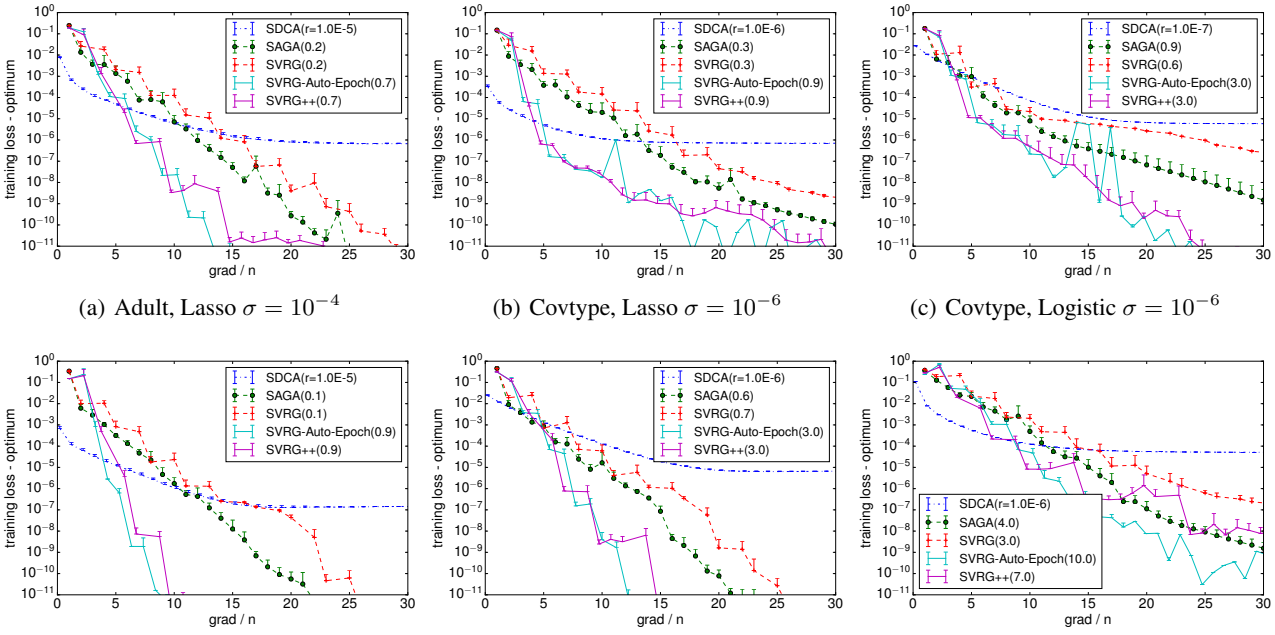
We confirm our theoretical findings using four real-life datasets: (1) the `Adult` dataset ($32,561$ examples and 123 features), (2) the `Covtype` dataset ($581,012$ examples and 54 features), (3) the `Ijcnn1` dataset (49990 examples and 22 features), and (4) the 2nd class of the `MNIST` dataset ($60,000$ examples and 780 features) (Fan & Lin). In order to make easy comparisons between different datasets, we scale each data vector down by the *average* Euclidean norm of the whole data set. This step is for comparison only and not necessary in practice.

We perform 3 classification tasks: *Lasso, ridge regression*, and *ℓ_1-regularized logistic regression*. As described in the introduction, Lasso and logistic regression do *not* admit strongly convex objectives, while the ridge objective is strongly convex. We consider four different values $\sigma \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, where σ is either the weight in regularizer $\frac{\sigma}{2}\|x\|_2^2$ for ridge, or that in regularizer $\sigma\|x\|_1^2$ for Lasso and logistic regression.

We have implemented the following algorithms:

- $\texttt{SVRG}^{++}$ with initial epoch length $m_0 = n/4$.
- `SVRG_Auto_Epoch` as we described in Section 4.1.
- `SVRG` (Johnson & Zhang, 2013; Xiao & Zhang, 2014) with (their suggested) epoch length $m = 2n$. (Recall that, in theory, `SVRG` is not designed for non-strongly convex objectives and $F(\cdot)$ needs to be added by a dummy regularizer for Lasso and logistic regression. However, in our experiments, we observed that this dummy regularizer is not necessary, so have neglected the regularized version of `SVRG` for a clean comparison.)
- `SAGA` (Defazio et al., 2014).
- `SDCA` (Shalev-Shwartz & Zhang, 2013; 2012) with Option I (steepest descent). Since `SDCA` works only with strongly convex objectives, a dummy regularizer has to be introduced for Lasso and Logistic regression.

For each algorithm above except `SDCA`, we tune the step length carefully from the set $\{a \times 10^{-k} : a \in \{1, 2, \ldots, 9\}, k \in \mathbb{Z}\}$ for each plot. For `SDCA` on Lasso and



(a) Adult, Lasso $\sigma = 10^{-4}$ (b) Covtype, Lasso $\sigma = 10^{-6}$ (c) Covtype, Logistic $\sigma = 10^{-6}$

(d) Ijcnn1, Lasso $\sigma = 10^{-5}$ (e) Ijcnn1, Logistic $\sigma = 10^{-5}$ (f) Mnist, Logistic $\sigma = 10^{-4}$

Figure 1. Selected performance comparisons for lasso and logistic regression using Tuning Type I. Our comprehensive comparisons for (1) other regularizer weights, (2) both lasso and ridge regression, and (3) both parameter tuning types can be found in the full paper.

logistic regression, we also tune the weight of its dummy regularizer from the set $\{10^{-k}, 2\times 10^{-k}, 5\times 10^{-k} : k \in \mathbb{Z}\}$. To make our comparison stronger, we adopt an anonymous reviewer's suggestion and consider two types of parameter tuning. In Tuning Type I, we select the best curve based on the training objective performance in the entire 30 passes to the dataset. In Tuning Type II, we select the best parameter only based on method's performance in the first 4 passes to the dataset. Tuning Type II might be more realistic for experimentalists who need to quickly pick the best parameters of the algorithms.

In each plot, we run 10 times the experiments and plot both the mean and the variance. Since our plots are in log scale, we only keep the upper error bar to make the plots easier to read. In other words, the *lower* end of each error bar represents the *mean* of each data point.

Performance Comparison. We have picked a representative regularizer weight σ for each of the eight analysis tasks (lasso or logistic regression on one of the four datasets), and presented the performance plots using Tuning Type I in Figure 1. For the results on other values of σ as well as those for ridge regression, see Figure 3, 4, 5, and 6 in the appendix. We have also included plots using Tuning Type II in Figure 7, 8, 9, and 10 in the appendix.

In all of our plots, the y-axis represents the training objective value minus the minimum, and the x-axis represents the number of passes to the dataset. Here, following the tradition, one iteration of each algorithm counts as $1/n$ pass of the dataset, and the snapshot full-gradient computation of `SVRG`, `SVRG`$^{++}$, and `SVRG_Auto_Epoch` counts as one additional pass.

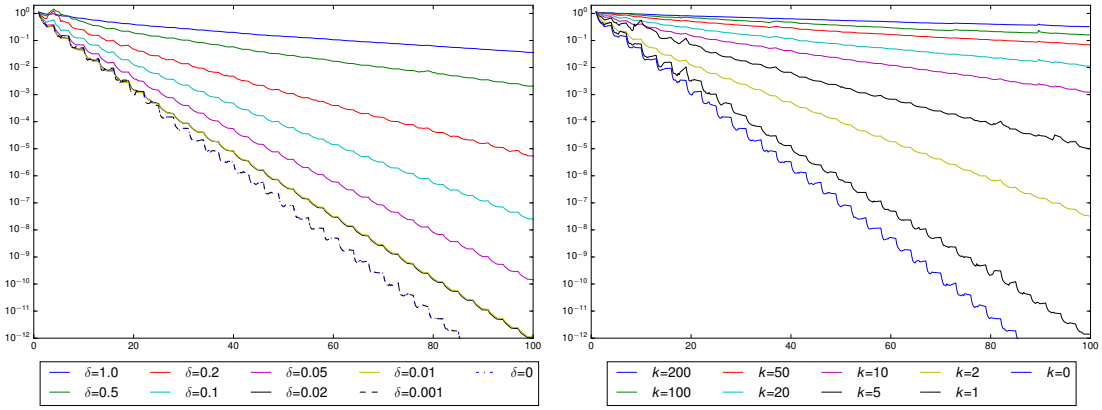
In the legend of each plot, we use `SDCA`$(r = r_0)$ to denote that r_0 is the weight of the best-tuned dummy regularizer. For every other algorithm, we use `Alg`(η) to denote that η is the best-tuned step length for algorithm `Alg`.

We make the following observations from this experiment:

- `SVRG`$^{++}$ and `SVRG_Auto_Epoch` consistently outperform `SVRG` in all the plots, indicating that they do improve over `SVRG` in non-strongly convex settings.
- `SVRG`$^{++}$ and `SVRG_Auto_Epoch` outperform `SAGA` in most cases, and are at least comparable to `SAGA` in the rest cases. This is not surprising because `SAGA` is also a direct algorithm for non-strongly convex objectives.
- `SVRG`$^{++}$ and `SVRG_Auto_Epoch` significantly outperform indirect methods via dummy regularization (i.e., `SDCA`) in the non-strongly convex settings. For ridge regression which is strongly convex, `SDCA` is comparable to other methods (see the figures in the appendix).

8. Experiments for Sum-of-Non-Convex Objectives

To verify our theoretical findings in Section 5 and 6, we run `SVRG` on a sum-of-non-convex objective built from synthet-



(a) parameterized by δ (b) parameterized by k

Figure 2. Performance analysis on sum-of-non-convex objectives. Note that the curves for $\delta = 0.001, 0.01, 0.02$ have overlapped in (a).

ically generated data. We generate $n = 500$ random vectors $a_1, \dots, a_{500} \in \mathbb{R}^d$ from the $d = 200$ dimensional unit cube and then normalize them to have Euclidean norm 1. Define the covariance matrix $A \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n a_i a_i^T$, and we consider the minimization problem[11]

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) \stackrel{\text{def}}{=} \frac{x^T A x}{2} + bx \right\}$$

for some randomly generated vector b.

The matrix A we generated has minimum eigenvalue equal to 7.02×10^{-4}, and thus $f(x)$ is strongly convex with parameter 7.02×10^{-4}. Next, we decompose $f(x)$ into an average of $f_i(x)$, each being non-convex with upper and lower smoothness parameters that we can control.

More specifically, given n diagonal matrices $D_1, \cdots, D_n$ satisfying $D_1 + \cdots + D_n = 0$, by setting $f_i(x) \stackrel{\text{def}}{=} \frac{x^T(a_i a_i^T + D_i)x}{2} + bx$, we have $f(x) = \frac{1}{n}\sum_i f_i(x)$. Under this construction, each f_i is non-convex if D_i has negative entries in the diagonals. We now consider two different ways to build $D_1, \dots, D_n$.

Remark 8.1. We do not perform real-life PCA experiments for the following reason. Recall Garber and Hazan reduced PCA to minimizing $f(x) = \frac{1}{2}x^T(\mu I - A)x + b^T x$. For all interesting choices of μ, our result in Theorem 6.1 is faster than theirs by the same constant factor $\lambda \in [1/d, 1]$, which is the largest eigenvalue of A. Therefore, by varying μ and comparing the plots, it is impossible to observe anything interesting: in particular, one cannot conclude our theoretical bound is tighter in practice. In contrast, our carefully designed synthetic experiment allows us to control the upper and lower smoothness parameters, and therefore to observe the improvements of our theorems directly.

Our first experiment is parameterized by a given value $\delta \in [0, 1]$. For each $j \in [d]$, we randomly select half of the indices $i \in [n]$ and assign its j-th diagonal $(D_i)_{jj}$ to be δ; for the other half of the indices i we assign $(D_i)_{jj}$ to be $-\delta$. In this way, we satisfy $D_1 + \cdots + D_n = 0$ and for each $i \in [n]$, we have $-\delta I \le \nabla^2 f_i(x) \le (1+\delta)I$. In other words, each function $f_i(x)$ is $L \approx 1$ upper smooth and exactly $l = \delta$ lower smooth. This corresponds to the $l \le L$ regime studied by Section 5.

Our second experiment is parameterized by a given value $k \in [1, n]$. For each $j \in [d]$, consider the j-th diagonal entry of all the matrices, $(D_1)_{jj}, (D_2)_{jj}, \dots (D_n)_{jj}$. We randomly select one of these entries and set it to be $-k$, and the rest $n-1$ of them to be $\frac{k}{n-1}$. Under this definition, we have $D_1 + \cdots + D_n = 0$ and for each $i \in [n]$, we have $-kI \le \nabla^2 f_i(x) \le (1 + k/(n-1))I$. In other words, each function $f_i(x)$ is approximately $L \approx 1$ upper smooth and $l = k$ lower smooth. This corresponds to the $l \ge L$ regime studied by Section 6.

We run SVRG (with the best tuned step length) for both experiments, and plot the performance in Figure 2. We make the following observations from the plots:

- In Figure 2(a), we observe that the performance SVRG is approximately linearly proportional to $lL = O(\delta)$ for large δ, as compared to $L^2 = O(1)$ from prior work. More importantly, SVRG is robust against small non-convexity parameter l. Indeed, for $l = \delta \le 0.02$, the convergence of SVRG is as fast as the convex case (i.e., $\delta = 0$ case). This confirms our theoretical finding in Section 5 and particularly confirms the existence of a threshold $O(\sigma)$ where the performance of SVRG only starts to degrade when l is above this threshold.
- In Figure 2(b), we see that the performance of SVRG is approximately linearly proportional to $lL = O(k)$, as compared to $l^2 = O(k^2)$ from prior work. This confirms our finding in Section 6.

[11] Since $x^* = A^{-1}b$ this is a linear system problem.

References

Allen-Zhu, Zeyuan. Katyusha: The First Truly Accelerated Stochastic Gradient Method. *ArXiv e-prints*, abs/1603.05953, March 2016.

Allen-Zhu, Zeyuan and Hazan, Elad. Variance Reduction for Faster Non-Convex Optimization. In *ICML*, 2016a.

Allen-Zhu, Zeyuan and Hazan, Elad. Optimal Black-Box Reductions Between Optimization Objectives. *ArXiv e-prints*, abs/1603.05642, March 2016b.

Allen-Zhu, Zeyuan and Li, Yuanzhi. Even Faster SVD Decomposition Yet Without Agonizing Pain. *ArXiv e-prints*, abs/xxxx.xxxxx, 2016.

Defazio, Aaron, Bach, Francis, and Lacoste-Julien, Simon. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. In *NIPS*, 2014.

Fan, Rong-En and Lin, Chih-Jen. LIBSVM Data: Classification, Regression and Multi-label. Accessed: 2015-06.

Frostig, Roy, Ge, Rong, Kakade, Sham M., and Sidford, Aaron. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *ICML*, volume 37, pp. 1–28, 2015.

Garber, Dan and Hazan, Elad. Fast and simple PCA via convex optimization. *ArXiv e-prints*, September 2015.

Garber, Daniel, Hazan, Elad, Jin, Chi, Kakade, Sham M., Musco, Cameron, Netrapalli, Praneeth, and Sidford, Aaron. Robust shift-and-invert preconditioning: Faster and more sample efficient algorithms for eigenvector computation. In *ICML*, 2016.

Harikandeh, Reza, Ahmed, Mohamed Osama, Virani, Alim, Schmidt, Mark, Konečný, Jakub, and Sallinen, Scott. Stopwasting my gradients: Practical svrg. In Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2242–2250. Curran Associates, Inc., 2015.

Hazan, Elad. DRAFT: Introduction to online convex optimimization. *Foundations and Trends in Machine Learning*, XX(XX):1–168, 2015.

Hazan, Elad, Agarwal, Amit, and Kale, Satyen. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, August 2007. ISSN 0885-6125.

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

Konecný, Jakub, Liu, Jie, Richtárik, Peter, and Takác, Martin. ms2gd: Mini-batch semi-stochastic gradient descent in the proximal setting. *CoRR*, abs/1410.4744, 2014.

Lin, Hongzhou, Mairal, Julien, and Harchaoui, Zaid. A Universal Catalyst for First-Order Optimization. In *NIPS*, 2015.

Lin, Qihang, Lu, Zhaosong, and Xiao, Lin. An Accelerated Proximal Coordinate Gradient Method and its Application to Regularized Empirical Risk Minimization. In *NIPS*, pp. 3059–3067, 2014.

Mahdavi, Mehrdad, Zhang, Lijun, and Jin, Rong. Mixed optimization for smooth functions. In *Advances in Neural Information Processing Systems*, pp. 674–682, 2013.

Mairal, Julien. Incremental Majorization-Minimization Optimization with Application to Large-Scale Machine Learning. *SIAM Journal on Optimization*, 25(2):829–855, April 2015. ISSN 1052-6234. Preliminary version appeared in ICML 2013.

Nesterov, Yurii. *Introductory Lectures on Convex Programming Volume: A Basic course*, volume I. Kluwer Academic Publishers, 2004. ISBN 1402075537.

Ng, Andrew Y. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the 21st International Conference on Machine Learning*, ICML 2004, pp. 78. ACM, 2004.

Schmidt, Mark, Le Roux, Nicolas, and Bach, Francis. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388*, pp. 1–45, 2013. Preliminary version appeared in NIPS 2012.

Shalev-Shwartz, Shai. SDCA without Duality. *arXiv preprint arXiv:1502.06177*, pp. 1–7, 2015.

Shalev-Shwartz, Shai and Singer, Yoram. Logarithmic regret algorithms for strongly convex repeated games. Technical report, The Hebrew University, 2007.

Shalev-Shwartz, Shai and Zhang, Tong. Proximal Stochastic Dual Coordinate Ascent. *arXiv preprint arXiv:1211.2717*, pp. 1–18, 2012.

Shalev-Shwartz, Shai and Zhang, Tong. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14:567–599, 2013.

Shalev-Shwartz, Shai and Zhang, Tong. Accelerated Proximal Stochastic Dual Coordinate Ascent for Regularized Loss Minimization. In *ICML*, pp. 64–72, 2014.

Shamir, Ohad. A Stochastic PCA and SVD Algorithm with an Exponential Convergence Rate. In *Proceedings of The 32nd International Conference on Machine Learning*, ICML 2015, pp. 144—-153, 2015.

Tibshirani, Robert. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Xiao, Lin and Zhang, Tong. A Proximal Stochastic Gradient Method with Progressive Variance Reduction. *SIAM Journal on Optimization*, 24(4):2057—-2075, 2014. ISSN 1052-6234.

Zhang, Lijun, Mahdavi, Mehrdad, and Jin, Rong. Linear convergence with condition number independent access of full gradients. In *Advances in Neural Information Processing Systems*, pp. 980–988, 2013.

Zhang, Yuchen and Xiao, Lin. Stochastic Primal-Dual Coordinate Method for Regularized Empirical Risk Minimization. In *ICML*, 2015.