Acceleration of SVRG and Katyusha X by Inexact Preconditioning

Yanli Liu [1] **Fei Feng** [1] **Wotao Yin** [1]

Abstract

Empirical risk minimization is an important class of optimization problems with many popular machine learning applications, and stochastic variance reduction methods are popular choices for solving them. Among these methods, SVRG and Katyusha X (a Nesterov accelerated SVRG) achieve fast convergence without substantial memory requirement. In this paper, we propose to accelerate these two algorithms by *inexact preconditioning*, the proposed methods employ *fixed* preconditioners, although the subproblem in each epoch becomes harder, it suffices to apply *fixed* number of simple subroutines to solve it inexactly, without losing the overall convergence. As a result, this inexact preconditioning strategy gives provably better iteration complexity and gradient complexity over SVRG and Katyusha X. We also allow each function in the finite sum to be nonconvex while the sum is strongly convex. In our numerical experiments, we observe an on average $8\times$ speedup on the number of iterations and $7\times$ speedup on runtime.

1. Introduction

Empirical risk minimization is an important class of optimization problems that has many applications in machine learning, especially in the large-scale setting. In this paper, we formulate it as the minimization of the following objective

$$F(x) = f(x) + \psi(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x), \qquad (1.1)$$

where the finite sum $f(x)$ is strongly convex, each $f_i(x)$ in the finite sum is smooth[1] and can be nonconvex, and the regularizer $\psi(x)$ is proper, closed, and convex, but may be nonsmooth. A nonzero $\psi(x)$ is desirable in many applications, for example, ℓ_1- regularization that induces sparsity in the solution. Allowing f_i to be nonconvex is also necessary in some applications, e.g., shift-and-invert approach to solve PCA (Saad, 1992).

1.1. Related Work

To obtain a high quality approximate solution $\hat{x}$ of (1.1), stochastic variance reduction algorithms are a class of preferable choices in the large scale setting where n is huge. If each f_i is $\sigma-$strongly convex and $L-$smooth, and $\psi = 0$, then SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014a), SAG (Roux et al., 2012), SARAH (Nguyen et al., 2017), SDCA (Shalev-Shwartz & Zhang, 2013), SDCA without duality (Shalev-Shwartz, 2016), and Finito/MISO (Defazio et al., 2014b; Mairal, 2013) can find such a $\hat{x}$ within $\mathcal{O}\big((n+\frac{L}{\sigma})\ln(\frac{1}{\varepsilon})\big)$ evaluations of component gradients ∇f_i, while vanilla gradient descent needs $\mathcal{O}(n\frac{L}{\sigma}\ln\frac{1}{\varepsilon})$ evaluations. Recently, SCSG improves this complexity to $\mathcal{O}\big((n\wedge\frac{L}{\sigma\varepsilon}+\frac{L}{\sigma})\ln\frac{1}{\varepsilon}\big)^{2}$. When $\psi \neq 0$, many of these algorithms can be extended accordingly and the same gradient complexity is preserved (Xiao & Zhang, 2014; Defazio et al., 2014a; Shalev-Shwartz & Zhang, 2016). Among these methods, SVRG has been a popular choice due to its low memory cost.

When the condition number $\frac{L}{\sigma}$ is large, the performances of these variance reduction methods may degenerate considerably. In view of this, there have been many schemes that incorporate second-order information into the variance reduction schemes. In (Gonen et al., 2016), the problem data is first transformed by linear sketching in order to decrease the condition number, then SVRG is applied. However, the strategy is only proposed for ridge regression and it is unclear whether it can be applied to other problems.

A larger family of algorithms, called Stochastic Quasi-Newton (SQN) methods, apply to more general settings. The idea is to first sample one or a few Hessian-vector products, then perform a L-BFGS type update on the approximate Hessian inverse H_k (Byrd et al., 2016; Moritz

[1] Department of Mathematics, University of California, Los Angeles, Los Angeles, CA, USA. Correspondence to: Yanli Liu <yanli@math.ucla.edu>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

[1] A function f is said to be smooth if its gradient ∇f is Lipschitz continuous.

[2] $a \wedge b := \min\{a, b\}$.

et al., 2016; Gower et al., 2016), then H_k is applied to the SVRG-type stochastic gradient as a preconditioner. That is,

$$w_{t+1} = w_t - \eta H_k \tilde{\nabla}_t,$$

where $\tilde{\nabla}_t$ is a variance-reduced stochastic gradient.

Linear convergence is established and competitive numerical performances are observed for SQN methods. However, the theoretical linear rate depends on the condition number of the approximate Hessian, which again depends poorly on the condition number of the objective, so it is not clear whether they are faster than SVRG in general. Furthermore, they do not support nondifferentiable regularizers nonconvexity of individual f_i. Recently, the first issue is partially resolved in (Lin et al., 2016), where the algorithm is at least as fast as SVRG. To deal with the second issue, (Wang et al., 2018) applied a H_k-preconditioned proximal mapping of ψ after H_k is applied to the variance reduced stochastic gradient, but in order to evaluate this mapping efficiently, H_k is required to be of the symmetric rank-one update form $\tau I_d + uu^T$, where $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix and $u \in \mathbb{R}^{\mathrm{d}}$. However, H_k is still ill-conditioned with a conditioner number of order $\mathcal{O}(\frac{1}{\varepsilon})$, therefore only a gradient complexity of order $\mathcal{O}\big((n+\kappa\frac{1}{\varepsilon})\ln(\frac{1}{\varepsilon})\big)$ can be guaranteed.

Another way of exploiting second-order information is to cyclically calculate one individual Hessian $\nabla^2 f_i$ (or an approximation of it) (Rodomanov & Kropotov, 2016; Mokhtari et al., 2018), linear and locally superlinear convergence are established. However, they require at least an $O(n)$ amount of memory to store the local variables, which will be substantial when n is large.

Aside from exploiting second-order information, it is also possible to apply Nesterov-type acceleration to SVRG. Recently, Katyusha (Allen-Zhu, 2017) and Katyusha X (Allen-Zhu, 2018) are developed in this spirit. Katyusha X also applies to the sum-of-nonconvex setting where each f_i can be nonconvex. There are also “Catalyst” accelerated methods (Lin et al., 2015), where a small amount of strong convexity $\frac{c}{2}\|x-y^k\|^2$ is added to the objective and is minimized inexactly at each step, then Nesterov acceleration is applied. However, Catalyst methods have an additional $\ln k$ factor in gradient complexity over Katyusha and Katyusha X.

1.2. Our Contributions

1. We propose to accelerate SVRG and Katyusha X by a *fixed* preconditioner, as opposed to time-varying preconditioners in SQN methods. And the subproblems are solved with *fixed* number of simple subroutines.

2. If the preconditioner captures the second order information of f, then there will be significant accelerations. With a good preconditioner M, when $\kappa_f \in (n^{\frac{1}{2}}, n^2 d^{-2})$, Algorithm 1 and Algorithm 2 are $\mathcal{O}(\frac{n^{\frac{1}{2}}}{\kappa_f})$ and $\mathcal{O}(\sqrt{\frac{n^{\frac{1}{2}}}{\kappa_f}})$ times faster than SVRG and Katyusha X in terms of gradient complexity, respectively. When $\kappa_f > n^2 d^{-2}$, these numbers become $\mathcal{O}(\frac{d}{\sqrt{n\kappa_f}})$ and $\mathcal{O}(\frac{d}{n^{\frac{3}{4}}})$. We also demonstrate these accelerations for Lasso and Logistic regression.

3. Our acceleration applies to the sum-of-nonconvex setting, where $f(x)$ in (1.1) is strongly convex, but each individual f_i can be nonconvex. We also allow a nondifferentiable regularizer $\psi(x)$.

2. Preliminaries and Assumptions

Throughout this paper, we use $\|\cdot\|$ for ℓ_2-norm and $\langle\cdot,\cdot\rangle$ for dot product, $\|\cdot\|_1$ denotes the ℓ_1-norm.

The preconditioner $M \succ 0$ is a symmetric, positive definite matrix. We write $\lambda_{\min}(M)$ and $\lambda_{\max}(M)$ as the smallest and the largest eigenvalues of M, respectively, and $\kappa(M) := \frac{\lambda_{\max}(M)}{\lambda_{\min}(M)}$ as the condition number of M. For $M \succ 0$, let $\|\cdot\|_M$ and $\langle\cdot,\cdot\rangle_M$ denote the norm and inner product induced by M, respectively, i.e., $\|x\|_M = \sqrt{x^T M x}$, $\langle x, y\rangle_M = x^T M y$.

We use $\lceil\cdot\rceil$ to denote the ceiling function. For $r \in (0, 1]$, $N \sim$ **Geom**(r) denotes a random variable N that obeys the geometric distribution, i.e., $N = k$ with probability $(1-r)^k r$ for $k \in \mathbb{N}$. We have $\mathbb{E}[N] = \frac{1-p}{p}$.

Definition 1. We say that $f : \mathbb{R}^{\mathrm{d}} \to \mathbb{R}$ is L_f-smooth, if it is differentiable and satisfies

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

We say that $f : \mathbb{R}^{\mathrm{d}} \to \mathbb{R}$ is L_f^M-smooth under $\|\cdot\|_M$, if it is differentiable and satisfies

$$f(y) \le f(x) + \langle\nabla f(x), y-x\rangle + \frac{L_f^M}{2}\|y-x\|_M^2, \forall x, y \in \mathbb{R}^{\mathrm{d}}\,.$$

Definition 2. We say that f is σ_f-strongly convex, if

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

We say that f is σ_f^M-strongly convex under $\|\cdot\|_M$, if

$$f(y) \ge f(x) + \langle\nabla f(x), y-x\rangle + \frac{\sigma_f^M}{2}\|y-x\|_M^2, \forall x, y \in \mathbb{R}^{\mathrm{d}}\,.$$

L_f^M-smoothness under $\|\cdot\|_M$ is equivalent to $\|\nabla f_i(x) - \nabla f_i(y)\|_{M^{-1}} \le L_f^M\|x-y\|_M$. Also, σ_f^M-strong convexity is equivalent to $\|\nabla f_i(x) - \nabla f_i(y)\|_{M^{-1}} \ge \sigma_f^M\|x-y\|_M$. Cf. Section 2 of (Shalev-Shwartz & Zhang, 2016).

Definition 3. We define the condition number of f under $\|\cdot\|_M$ as $\kappa_f^M := \frac{L_f^M}{\sigma_f^M}$.

When $M = I$, we have $\kappa_f^M = \kappa_f := \frac{L_f}{\sigma_f}$.

In this paper, we will choose M such that $\kappa_f^M \ll \kappa$. For example, if $f(x) = \frac{1}{2}x^T Q x$ where $Q \succ 0$ is ill-conditioned, by choosing $M = Q$ we have

$$\|\nabla f(x) - \nabla f(y)\|_{M^{-1}} \equiv \|x - y\|_Q,$$

which tells us that $L_f^M = \sigma_f^M = 1$ and $\kappa_f^M = 1$, while $\kappa_f = \kappa(Q) \gg 1$. That is, under $Q-$metric, $f(x)$ has a much smaller condition number and can be minimized easily.

Definition 4. For a proper closed convex function $\phi : \mathbb{R}^{\mathrm{d}} \to \mathbb{R} \cup \{+\infty\}$, its subdifferential at $x \in \mathrm{dom}(f)$ is written as

$$\partial\phi(x) = \{v \in \mathbb{R}^{\mathrm{d}} \mid \phi(z) \geq \phi(x) + \langle v, z - x\rangle \ \forall z \in \mathbb{R}^{\mathrm{d}}\}.$$

Definition 5. For a proper closed convex function $\phi : \mathbb{R}^{\mathrm{d}} \to \mathbb{R}$, its $M-$preconditioned proximal mapping with step size $\eta > 0$ is defined by

$$\mathbf{prox}_{\eta\psi}^M(x) = \arg\min_{y \in \mathbb{R}^{\mathrm{d}}}\{\psi(y) + \frac{1}{2\eta}\|x - y\|_M^2\}.$$

When $M = I$, this reduces to the classical proximal mapping.

Finally, let us list the assumptions that will be effective throughout this paper.

Assumption 1. In the objective function (1.1),

1. Each $f_i(x)$ is L_f-smooth and L_f^M-smooth under $\|\cdot\|_M$.
2. $f(x)$ is σ_f-strongly convex, and σ_f^M-strongly convex under $\|\cdot\|_M$, where $\sigma_f > 0$ and $\sigma_f^M > 0$.
3. The regularization term $\psi(x)$ is proper closed convex and $\mathbf{prox}_{\eta\psi}$ is easy to compute.

Remark 1.

1. In Assumption 1, we only require $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$ to be strongly convex, while each $f_i(x)$ can be nonconvex.
2. Several common choices of regularizers have simple proximal mappings. For example, when $\psi(x) = \lambda\|\cdot\|_1$ with $\lambda > 0$, $\mathbf{prox}_{\eta\psi}$ can be computed component wise as

$$\mathbf{prox}_{\eta\psi}(x) = \mathrm{sign}(x)\max\{|x| - \eta\lambda, 0\}.$$

3. Proposed Algorithms

As discussed in Sec. 1, SVRG and Katyusha X suffer from ill-conditioning like other first order methods. In this section, we propose to accelerate them by applying inexact preconditioning. Let us illustrate the idea as follows,

1. We would like to apply a preconditioner $M \succ 0$ to the gradient descent step in SVRG. i.e.,

$$\begin{aligned} w_{t+1} &= \mathbf{prox}_{\eta\psi}^M(w_t - \eta M^{-1}\tilde{\nabla}_t) \\ &= \arg\min_{y \in \mathbb{R}^d}\{\psi(y) + \frac{1}{2\eta}\|y - w_t\|_M^2 + \langle\tilde{\nabla}_t, y\rangle\}. \end{aligned} \tag{3.1}$$

where $\tilde{\nabla}_t$ is a variance-reduced stochastic gradient. When $\psi = 0$ and this minimization is solved exactly, we have $w_{t+1} = w_t - \eta M^{-1}\tilde{\nabla}_t$, which is a preconditioned gradient update.

2. However, solving (3.1) exactly may be expensive and impractical. In fact it suffices to solve it *highly inexactly* by *fixed* number of simple subroutines.

We summarize the resulted algorithm in Algorithm 1 and call it Inexact Preconditioned(IP-) SVRG. Compared to SVRG, the only difference lies in line 7.

Algorithm 1 Inexact Preconditioned SVRG(iPreSVRG)

Input: $F(\cdot) = \psi(\cdot) + \frac{1}{n}\sum_{i=1}^n f_i(\cdot)$, initial vector x^0, step size $\eta > 0$, preconditioner $M \succ 0$, number of epochs K.
Output: vector x^K

```
 1: for $k \leftarrow 0, ..., K-1$ do
 2:     $D^k \sim$ Geom($\frac{1}{m}$);
 3:     $w_0 \leftarrow x^k$, $g \leftarrow \nabla f(x^k)$;
 4:     for $t \leftarrow 0, ..., D^k$ do
 5:         pick $i_t \in \{1, 2, ..., n\}$ uniformly at random;
 6:         $\tilde{\nabla}_t = g + \big(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(w_0)\big)$;
 7:         $w_{t+1} \approx \arg\min_{y \in \mathbb{R}^d}\{\psi(y) + \frac{1}{2\eta}\|y - w_t\|_M^2 + \langle\tilde{\nabla}_t, y\rangle\}$;
 8:     end for
 9:     $x^{k+1} \leftarrow w_{D+1}$;
10: end for
```

Remark 2.

1. In line 2, the epoch length D^k obeys a geometric distribution and $\mathbb{E}[m^k] = m - 1$, this is for the purpose of simplifying analysis (motivated by (Lei & Jordan, 2017; Allen-Zhu, 2018)), in practice one can just set $D^k = m - 1$. In our experiments, this still brings significant accelerations.
2. The choice of m affects the performance. Intuitively, a larger m means more gradient evaluations per epoch, but also more progress per epoch. Theoretically, we show that $m = \lceil\frac{n}{1+pd}\rceil$ gives faster convergence than SVRG, where p is the number of subroutines used in Line 7.
3. In line 6, one can also sample a batch of gradients instead of one. It is straightforward to generalize our convergence results in Sec. 4 to this setting.

4. If $M = I$, line 7 reduces to
$$w_{t+1} = \mathbf{prox}_{\eta\psi}(w_t - \eta\tilde{\nabla}_t),$$
and Algorithm 1 reduces to SVRG.

For $M \not\propto I$, line 7 contains an optimization problem that may not have a closed form solution:

$$\arg\min_{y\in\mathbb{R}^d}\{\psi(y) + \frac{1}{2\eta}\|y - w_t\|_M^2 + \langle\tilde{\nabla}_t, y\rangle\}. \tag{3.2}$$

To solve it inexactly, we propose to apply *fixed* number of iterations of some simple subroutines, which are initialized at w_t. This procedure is summarized in Procedure 1.

Procedure 1 Procedure for solving (3.2) inexactly

Input: Iterator S, iterator step size $\gamma > 0$, number of iterations $p \geq 1$, problem data $\eta > 0, w_t, M \succ 0, \tilde{\nabla}_t, \psi(\cdot)$.
Output: vector w_{t+1}

```
1: $w_{t+1}^0 \leftarrow w_t$;
2: for $i \leftarrow 0, ..., p-1$ do
3:     $w_{t+1}^{i+1} = S(w_{t+1}^i, \eta, M, \tilde{\nabla}_t, \psi)$;
4: end for
5: $w_{t+1} \leftarrow w_{t+1}^p$;
```

Remark 3. In Procedure 1, there are many choices for the iterator S, for example, one can use proximal gradient, FISTA (Beck & Teboulle, 2009) (or equivalently, Nesterov acceleration (Nesterov, 2013)), and FISTA with restart (Odonoghue & Candes, 2015). Under these choices, line 3 is easy to compute. For example, when S is the proximal gradient step, line 3 of Procedure 1 becomes

$$w_{t+1}^{i+1} = \mathbf{prox}_{\gamma\psi}(w_{t+1}^i - \frac{\gamma}{\eta}M(w_{t+1}^i - w_t) - \gamma\tilde{\nabla}_t).$$

Now, let us also apply the inexact preconditioning idea to Katyusha X (Algorithm 2 of (Allen-Zhu, 2018)). Similar to Katyusha X, we first apply a momentum step, then one epoch of iPreSVRG (i.e., line $2 \sim 9$ of Algorithm 1).

Algorithm 2 Inexact Preconditioned Katyusha X(iPreKatX)

Input: $F(x) = \psi(x) + \frac{1}{n}\sum_{i=1}^n f_i(x)$, initial vector x^0, step size $\eta > 0$, preconditioner $M \succ 0$, momentum weight $\tau \in (0, 1]$, number of epochs K.
Output: vector y^K

```
$y_{-1} = y_0 \leftarrow x_0$;
for $k \leftarrow 0, ..., K-1$ do
    $x_{k+1} \leftarrow \frac{\frac{3}{2}y_k + \frac{1}{2}x_k - (1-\tau)y_{k-1}}{1+\tau}$;
    $y_{k+1} \leftarrow$ Algorithm $1^{\text{1ep}}(F, M, x_{k+1}, \eta)$;
end for
```

Remark 4.

1. When $\tau = \frac{1}{2}$, one can show that $x_{k+1} \equiv y_k$, and Algorithm 2 reduces to Algorithm 1.
2. When $M = I$ and the proximal mapping is solved exactly, Algorithm 2 reduces to Katyusha X.
3. The convergence of Algorithm 2 is established when $\tau = \frac{1}{2}\sqrt{\frac{1}{2}m\eta\sigma_f^M}$. In practice, we found that many other choices of τ also work.

4. Main Theory

In this section, we proceed to establish the convergence of Algorithm 1 and Algorithm 2. The key idea is that when the preconditioned proximal gradient update in (3.2) is solved inexactly as in Procedure 1, the error can be bounded by $\|w_{t+1} - w_t\|_M$, under which we can still establish the overall convergence of Algorithm 1 and Algorithm 2. Combine this with the fixed number of simple subroutines in Procedure 1, we obtain a much lower gradient complexity when $\kappa_f > n^{\frac{1}{2}}$.

All the proofs in this section are deferred to the supplementary material.

First, Let us analyze the error in the optimality condition of (3.2) when it is solved inexactly by FISTA with restart as in Procedure 1. Specifically,

Let $h_1(y) = \psi(y)$ and $h_2(y) = \frac{1}{2\eta}\|y - w_t\|_M^2 + \langle\tilde{\nabla}, y\rangle$, then the subproblem (3.2) can be written as

$$\min_y \Psi(y) = h_1(y) + h_2(y).$$

Therefore, FISTA with restart applied to (3.2) can be summarized in the following algorithm.

Algorithm 3 FISTA with restart for solving (3.2)

Input: Iterator S, iterator step size $\gamma > 0$, number of iterations $p \geq 1$, problem data $\eta > 0, w_t, h_1(y) = \psi(y)$ and $h_2(y) = \frac{1}{2\eta}\|y - w_t\|_M^2 + \langle\tilde{\nabla}, y\rangle$.

```
$w_{t+1}^{(0,0)} = u_{t+1}^{(0,1)} \leftarrow w_t, \theta_0 = 1$
for $i \leftarrow 0, ..., r-1$ do
    for $j \leftarrow 0, ..., p_0 - 1$ do
        $\theta_0 = 1$;
        $w_{t+1}^{(i,j+1)} = \mathbf{prox}_{\gamma h_1}\big(u_{t+1}^{(i,j+1)} - \gamma\nabla h_2(u_{t+1}^{(i,j+1)})\big)$;
        $\theta_{j+1} = \frac{1+\sqrt{1+4\theta_j^2}}{2}$;
        $u_{t+1}^{(i,j+2)} = w_{t+1}^{(i,j+1)} + \frac{\theta_j - 1}{\theta_{j+1}}(w_{t+1}^{(i,j+1)} - w_{t+1}^{(i,j)})$;
    end for
    $w_{t+1}^{(i+1,0)} = u_{t+1}^{(i+1,1)} \leftarrow w_{t+1}^{(i,p_0)}$
end for
$w_{t+1} \leftarrow w_{t+1}^{(r-1,p_0)}$;
```

Lemma 1. *Take Assumption 1. Suppose in Procedure 1, we choose S as the iterator of FISTA with restart[1] every $p_0 = \lceil 2e\sqrt{\kappa(M)}\rceil$ steps, with step size $\gamma = \frac{\eta}{\lambda_{\max}(M)}$ and* *restart it $(r-1)$ times (that is, $p = rp_0$ iterations in total). Then, $w_{t+1} = w_{t+1}^{(r-1,p_0)}$ is an approximate solution to (3.2) that satisfies*

$$\mathbf{0} \in \partial\psi(w_{t+1}) + \frac{1}{\eta}M(w_{t+1} - w_t) + \tilde{\nabla}_t + M\varepsilon_{t+1}^p, \tag{4.1}$$

$$\|\varepsilon_{t+1}^p\|_M \leq \frac{c(p)}{\eta}\|w_{t+1} - w_t\|_M, \tag{4.2}$$

where

$$c(p) = 14\kappa(M)\frac{\tau^p}{1-\tau^p},$$

and

$$\tau = (\frac{4\kappa(M)}{p_0^2})^{\frac{1}{2p_0}} \leq \exp(-\frac{1}{2e\sqrt{\kappa(M)}+1}) < 1.$$

With Lemma 1, the overall convergences of Algorithm 1 and 2 can be established. The analysis is similar to that of (Allen-Zhu, 2018).

Theorem 1. *Under Assumption 1, let $x^* = \arg\min_x F(x)$, $64\kappa_f^M c^2(p) \leq 1$, $\eta \leq \frac{1}{2\sqrt{m}L_f^M}$, and $m \geq 4$. Then the iPreSVRG in Algorithm 1 satisfies*

$$\mathbb{E}[F(x^k) - F(x^*)] \leq \mathcal{O}\big((\frac{1}{1+\frac{1}{4}m\eta\sigma^M})^k\big). \tag{4.3}$$

Theorem 2. *Under Assumption 1, let $x^* = \arg\min_x F(x)$, $64\kappa_f^M c^2(p) \leq 1$, $\tau = \frac{1}{2}\sqrt{\frac{1}{2}m\eta\sigma_f^M}$, $\eta \leq \frac{1}{2\sqrt{m}L_f^M}$, and $m \geq 4$. Then the iPreKatX in Algorithm 2 satisfies*

$$\mathbb{E}[F(x^k) - F(x^*)] \leq \mathcal{O}\big((\frac{1}{1+\frac{1}{2}\sqrt{\frac{1}{2}m\eta\sigma^M}})^k\big). \tag{4.4}$$

Remark 5. When $M = I$, we have $c(p) = 0$, and Theorems 1 and 2 recovers the Theorems D.1 and 4.3 of (Allen-Zhu, 2018).

In Theorems 1 and 2, we need the number of simple subroutines p to be large enough such that $64\kappa_f^M c^2(p) \leq 1$, the following Lemma provides a sufficient condition for this.

Lemma 2. *If the subproblem iterator S in Procedure 1 is FISTA with restart every $p_0 = \lceil 2e\sqrt{\kappa(M)}\rceil$ steps, and with step size $\gamma = \frac{\eta}{\lambda_{\max}(M)}$, then, in order for $64\kappa_f^M c^2(p) \leq 1$ to hold, it suffices to choose*

$$p = (2e\sqrt{\kappa(M)}+1)\ln\frac{\sqrt{\kappa_f^M}\kappa(M) + \sqrt{c_1}}{c_1} \tag{4.5}$$
$$= \mathcal{O}\bigg(\sqrt{\kappa(M)}\ln\big(\sqrt{\kappa_f^M}\kappa(M)\big)\bigg)$$

where $c_1 = \frac{1}{64*14^2}$.

With (4.3), (4.4), and (4.5), we can now calculate the gradient complexities of Algorithm 1 and Algorithm 2, but let us first do that for SVRG and Katyusha X.

In Assumption 1, we have assumed that $\mathbf{prox}_{\eta\psi}(\cdot)$ is cheap to evaluate, therefore, each epoch of SVRG needs $n+m$ gradient evaluations, which is also true for Katyusha X. As a result, the gradient complexity for SVRG and Katyusha X to reach $\varepsilon-$suboptimality are:

$$C_1(m,\varepsilon) = \mathcal{O}(\frac{n+m}{\ln(1+\frac{1}{4}m\eta\sigma)}\ln\frac{1}{\varepsilon}), \tag{4.6}$$

$$C_2(m,\varepsilon) = \mathcal{O}(\frac{n+m}{\ln(1+\frac{1}{2}\sqrt{\frac{1}{2}m\eta\sigma})}\ln\frac{1}{\varepsilon}). \tag{4.7}$$

For Algorithm 1 and Algorithm 2, each iteration in Procedure 1 is at most as expensive as d gradient computations[1] and is operated p times, therefore, one epoch of iPreSVRG/iPreKatX needs at most $n + (1+pd)m$ gradient computations.

Consequently, we can write the the gradient complexity for Algorithm 1 and Algorithm 2 to reach $\varepsilon-$suboptimality as:

$$C_1'(m,\varepsilon) = \mathcal{O}(\frac{n+(1+pd)m}{\ln(1+\frac{1}{4}m\eta\sigma^M)}\ln\frac{1}{\varepsilon}), \tag{4.8}$$

$$C_2'(m,\varepsilon) = \mathcal{O}(\frac{n+(1+pd)m}{\ln(1+\frac{1}{2}\sqrt{\frac{1}{2}m\eta\sigma^M})}\ln\frac{1}{\varepsilon}). \tag{4.9}$$

Remark 6.

1. According to Lemma 2, when S is FISTA with restart, it suffices to choose p by (4.5).
2. When the preconditioner M is chosen appropriately, the step size η in (4.8) and (4.9) can be much larger than that of (4.6) and (4.7).

Finally, we can compare $C_1(m,\varepsilon)$, $C_2(m,\varepsilon)$ with $C_1'(m,\varepsilon)$, $C_2'(m,\varepsilon)$, respectively. It turns out that there is a significant speedup when $\kappa > n^{\frac{1}{2}}$.

Theorem 3. *Take Assumption 1. Let the iterator S in Procedure 1 be FISTA with restart, and an appropriate preconditioner M is chosen such that κ_f and $\kappa(M)$ are of the same order, and κ_f^M is small compared to them, then*

1. *if $\kappa_f > n^{\frac{1}{2}}$ and $\kappa_f < n^2 d^{-2}$, then*

$$\frac{\min_{m\geq 1} C_1'(m,\varepsilon)}{\min_{m\geq 1} C_1(m,\varepsilon)} \leq \mathcal{O}(\frac{n^{\frac{1}{2}}}{\kappa_f}). \tag{4.10}$$

[1] FISTA with restart can be replaced with any iterator with Q-linear convergence on the iterates. In our experiments, FISTA also works, and a simple choice of $p = 20$ is enough.

[1] For each iteration of Procedure 1, the most expensive step is multiplying M to some vector, which is often cheaper than d gradient computations.

2. *if $\kappa_f > n^{\frac{1}{2}}$ and $\kappa_f > n^2 d^{-2}$, then*

$$\frac{\min_{m\geq 1} C_1'(m,\varepsilon)}{\min_{m\geq 1} C_1(m,\varepsilon)} \leq \mathcal{O}(\frac{d}{\sqrt{n\kappa_f}}). \tag{4.11}$$

Theorem 4. *Take Assumption 1. Let the iterator S in Procedure 1 be FISTA with restart, and an appropriate preconditioner M is chosen such that κ_f and $\kappa(M)$ are of the same order, and κ_f^M is small compared to them, then*

1. *if $\kappa_f > n^{\frac{1}{2}}$ and $\kappa_f < n^2 d^{-2}$, then*

$$\frac{\min_{m\geq 1} C_2'(m,\varepsilon)}{\min_{m\geq 1} C_2(m,\varepsilon)} \leq \mathcal{O}(\sqrt{\frac{n^{\frac{1}{2}}}{\kappa_f}}). \tag{4.12}$$

2. *If $\kappa_f > n^{\frac{1}{2}}$ and $\kappa_f > n^2 d^{-2}$, then*

$$\frac{\min_{m\geq 1} C_2'(m,\varepsilon)}{\min_{m\geq 1} C_2(m,\varepsilon)} \leq \mathcal{O}(\frac{d}{n^{\frac{3}{4}}}). \tag{4.13}$$

In Section 5, we provide practical choices of M for Lasso and Logistic regression.

5. Experiments

To investigate the practical performance of Algorithms 1 and 2, we test on three problems: Lasso, logistic regression, and a synthetic sum-of-nonconvex problem. For the first two, each function in the finite sum is convex. To guarantee that the objective is strongly convex, a small ℓ_2-regularization is added to Lasso and logistic regression.

In the following, we compare SVRG, iPreSVRG, Katyusha X, and iPreKatX on four datasets from LIBSVM[1]: `w1a.t` (47272 samples, 300 features), `protein` (17766 samples, 357 features), `cod-rna.t` (271617 samples, 8 features), `australian` (690 samples, 14 features), and one synthetic dataset. The implementation settings are listed below,

1. We choose the epoch length $m = 100$ in all experiments, since we found that the choices $m \in \{\frac{n}{4}, \frac{n}{2}, n\}$ need more gradient evaluations.

2. For iPrePDHG and iPreKatX, we use FISTA as the subproblem iterator S. If the preconditioner M is diagonal, then the number of subroutines for solving the subproblem is $p = 1$, if not, then we set $p = 20$.

3. In all the experiments, we tune the step size η and momentum weight τ to their optimal.

4. All algorithms are initialized at $x^0 = \mathbf{0}$.

5. All algorithms are implemented in Matlab R2015b. To be fair, except for the subproblem routines for inexact preconditioning, the other parts of the code are identical in all algorithms. The experiments are conducted on a Windows system with Intel Core i7 2.6 GHz CPU. The code is available at:

   `https://github.com/uclaopt/IPSVRG`.

5.1. Lasso

We formulate Lasso as

$$\underset{x\in\mathbb{R}^d}{\text{minimize}}\ \frac{1}{2n}\sum_{i=1}^{n}(a_i^T x - b_i)^2 + \lambda_1\|x\|_1 + \lambda_2\|x\|_2^2, \tag{5.1}$$

where $a_i \in \mathbb{R}^d$ are feature vectors and $b_i \in \mathbb{R}$ are labels. Note that the first term is equivalent to $\frac{1}{2n}\|Ax - b\|^2$, where $A = (a_1, a_2, ..., a_n)^T \in \mathbb{R}^{n\times d}$ and $b = (b_1, b_2, \ldots, b_n) \in \mathbb{R}^n$.

For Lasso as in (5.1), we provide two choices of preconditioner M,

1. When d is small, we choose
$$M_1 = \frac{1}{n}A^T A,$$
this is the exact Hessian of the smooth part of the objective.

2. When d is large and $A^T A$ is diagonally dominant, we choose
$$M_2 = \frac{1}{n}\text{diag}(A^T A) + \alpha I,$$
where $\alpha > 0$. In this case, the subproblem (3.2) can be solved exactly with $p = 1$ iteration.

Our numerical results are presented in the following figures. We didn't observe significant accelerations of Katyusha X over SVRG and iPreKatX over iPrePDHG, and we suspect the reason is that $m = 100$ and the optimal choices of step size η make $m\eta\sigma_f > 1$ or $m\eta\sigma_f^M > 1$, thus the complexity in (4.7) and (4.9) are not better than (4.6) and (4.8), respectively.

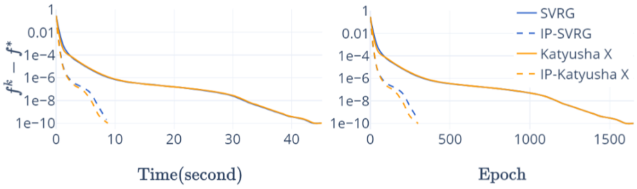

Figure 5.1. Lasso on `w1a.t`, $(n, d) = (47272, 300)$, $\lambda_1 = 10^{-3}, \lambda_2 = 10^{-8}$. For iPreSVRG and iPreKatX: $\eta_1 = 0.005$; For SVRG and Katyusha X: $\eta_2 = 0.08$; For Katyusha X and iPreKatX: $\tau = 0.45$, $M = M_2$ with $\alpha = 0.01$.

[1] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

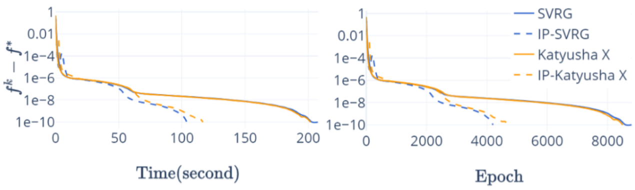

Figure 5.2. Lasso on `protein`, $(n, d) = (17766, 357)$, $\lambda_1 = 10^{-4}, \lambda_2 = 10^{-6}$, $\eta_1 = 0.008$, $\eta_2 = 0.2$, $\tau = 0.2$, $M = M_2$ with $\alpha = 0.008$.

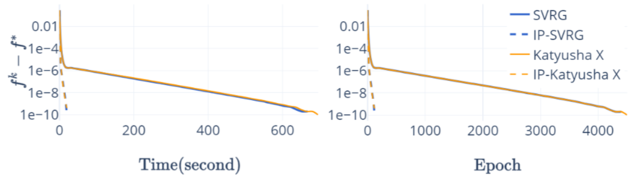

Figure 5.3. Lasso on `cod-rna.t`, $(n, d) = (271617, 8)$, $\lambda_1 = 10^{-2}, \lambda_2 = 1$, $\eta_1 = 1$, $\eta_2 = 5 \times 10^{-6}$, $\tau = 0.45$, $M = M_1$, subproblem iterator step size $\gamma = 3 \times 10^{-6}$.

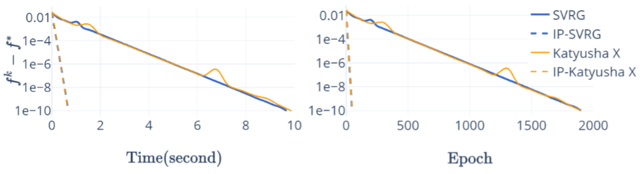

Figure 5.4. Lasso on `australian`, $(n, d) = (690, 14)$, $\lambda_1 = 2, \lambda_2 = 10^{-8}$, $\eta_1 = 0.01$, $\eta_2 = 8 \times 10^{-10}$, $\tau = 0.49$, $M = M_1$,$\gamma = 5 \times 10^{-10}$.

5.2. Logistic Regression

We formulate Logistic regression as

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \ln\left(1 + \exp(-b_i \cdot a_i^T x)\right) + \lambda_1 \|x\|_1 + \lambda_2 \|x\|_2^2, \tag{5.2}$$

where again $a_i \in \mathbb{R}^d$ are feature vectors and $b_i \in \mathbb{R}$ are labels.

For Logistic regression as in (5.2), the Hessian of the smooth part can be expressed as

$$H = \frac{1}{n} \sum_{i=1}^{n} \frac{\exp(-b_i a_i^T x)}{\left(1 + \exp(-b_i a_i^T x)\right)^2} b_i^2 a_i a_i^T \preccurlyeq \frac{1}{4n} B^T B,$$

where $B = \text{diag}(b)A = \text{diag}(b)(a_1, a_2, ..., a_n)^T$. Inspired by this[1], we provide two choices of preconditioner M,

1. When d is small, we choose

$$M_1 = \frac{1}{4n} B^T B.$$

2. When d is large and $B^T B$ is diagonally dominant, we choose

$$M_2 = \frac{1}{4n} \text{diag}(B^T B) + \alpha I,$$

where $\alpha > 0$. In this case, the subproblem (3.2) can be solved exactly with $p = 1$ iteration.

Our results are presented in the following figures, again, we didn't observe a significant acceleration of Katyusha X over SVRG and iPreKatX over iPrePDHG, due to the same reason mentioned in the last subsection.

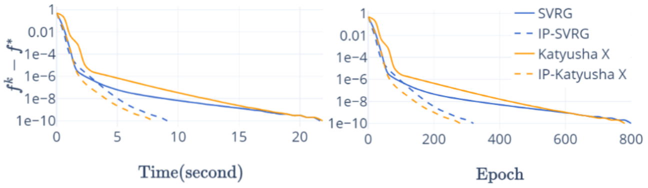

Figure 5.5. Logistic regression on `w1a.t`, $(n, d) = (47272, 300)$, $\lambda_1 = 5 \times 10^{-4}, \lambda_2 = 10^{-8}$, $\eta_1 = 0.06$, $\eta_2 = 4$, $\tau = 0.4$, $M = M_2$ with $\alpha = 0.005$.

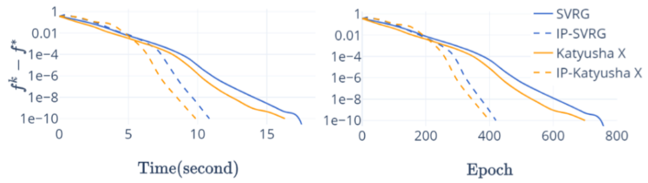

Figure 5.6. Logistic regression on `protein`, $(n, d) = (17766, 357)$, $\lambda_1 = 10^{-4}, \lambda_2 = 10^{-8}$, $\eta_1 = 1.5$, $\eta_2 = 10$, $\tau = 0.3$, $M = M_2$ with $\alpha = 0.05$.

[1] Here is a heuristic justification: By Definition 1 we know that $L_f^M = 1$; Since $\frac{\exp(-b_i a_i^T x)}{\left(1+\exp(-b_i a_i^T x)\right)^2} \to 0$ only when x is unbounded, we know that if the iterates x^k of our algorithms are bounded, then $H(x^k) \succcurlyeq \frac{c}{n} B^T B$ for some $c > 0$, which gives $\sigma_f^M = 4c$ according to Definition 2. When c is not too small, one can expect $\kappa_f^M = \frac{1}{4c} \ll \kappa_f$.

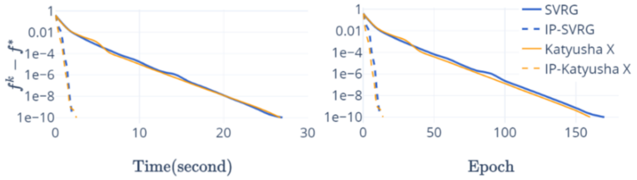

Figure 5.7. Logistic regression on `cod-rna.t`, $(n, d) = (271617, 8)$, $\lambda_1 = 0.1, \lambda_2 = 10^{-8}$, $\eta_1 = 1$, $\eta_2 = 3 \times 10^{-5}$, $\tau = 0.4$, $M = M_1$, $\gamma = 2 \times 10^{-5}$.

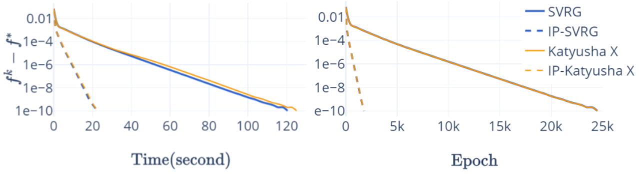

Figure 5.8. Logistic regression on `australian`, $(n, d) = (690, 14)$, $\lambda_1 = 0.5$, $\lambda_2 = 10^{-8}$, $\eta_1 = 1$, $\eta_2 = 10^{-6}$, $\tau = 0.2$, $M = M_1$, $\gamma = 2 \times 10^{-7}$.

5.3. Sum-of-nonconvex Example

Similar to (Allen-Zhu & Yuan, 2016), we generate a sum-of-nonconvex example by the following procedure:

We take n normalized random vector $a_i \in \mathbb{R}^{\mathrm{d}}$, and also d vectors of the form $g_i = (0, ...0, 5i, 0, ...0)$, where the nonzero element is at ith coordinate.

And the sum-of-nonconvex problem is given by

$$\underset{x \in \mathbb{R}^{\mathrm{d}}}{\text{minimize}} \ \frac{1}{2n} \sum_{i=1}^{n} x^T (c_i c_i^T + D_i) x + b^T x + \lambda_1 \|x\|_1, \quad (5.3)$$

where $n = 2000, d = 100$, and $\lambda_1 = 10^{-3}$.

$$c_i = \begin{cases} a_i + g_i & i = 1, 2, ..., d, \\ a_i & \text{otherwise.} \end{cases}$$

$$D_i = \begin{cases} -100I & i = 1, 2, ..., \frac{n}{2}, \\ 100I & \text{otherwise.} \end{cases}$$

Since the sum of D_i's is 0, they do not affect the condition number of the whole problem. However, it makes most of the first half of f_i to be highly nonconvex. Overall, the condition number of this problem is equal to that of $\sum_{i=1}^{n} c_i c_i^T$, which is approximately 10000 in our tested data.

Since $\sum_{i=1}^{n} c_i c_i^T$ is diagonally dominant, we select $M = \text{diag}(\frac{1}{n} \sum_{i=1}^{n} c_i c_i^T) + \alpha I$ as the preconditioner. Our algorithms also have significant acceleration in this sum-of-nonconvex setting.

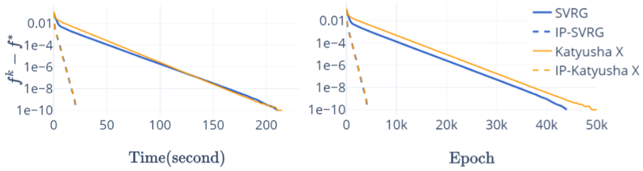

Figure 5.9. Sum-of-nonconvex on synthetic data. $\lambda_1 = 10^{-3}$, $\alpha = 15$. $\eta_1 = 0.015$, $\eta_2 = 10^{-4}$, $\tau = 0.45$.

6. Conclusions and Future Work

In this paper, we propose to accelerate SVRG and Katyusha X by inexact preconditioning, with an appropriate preconditioner, both can be provably accelerated in terms of iteration complexity and gradient complexity. Our algorithms admits a nondifferentiable regularizer, as well as nonconvexity of individual functions. We confirm our theoretical results on Lasso, Logistic regression, and a sum-of-nonconvex example, where simple choices of preconditioners lead to significant accelerations.

There are still open questions left for us to address in the future: (a) Do we have theoretical guarantee when the subproblem iterator S is chosen as faster schemes such as APCG (Lin et al., 2014), NU_ACDM (Allen-Zhu et al., 2016), and A2BCD (Hannah et al., 2018a)? (b) In general, how to choose a simple preconditioner that can greatly reduce the condition number of the problem? (c) Is it possible to apply this inexact preconditioning technique to other stochastic algorithms?

Acknowledgements

We would like to thank Yunbei Xu for helpful discussions on the idea of inexact preconditioning. We also thank the reviewers for their valuable comments.

This work is supported in part by the National Key R&D Program of China 2017YFB02029, AFOSR MURI FA9550-18-1-0502, NSF DMS-1720237, and ONR N0001417121.

References

Allen-Zhu, Z. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1200–1205. ACM, 2017.

Allen-Zhu, Z. Katyusha X: Practical Momentum Method for

Stochastic Sum-of-Nonconvex Optimization. In *ICML*, 2018.

Allen-Zhu, Z. and Yuan, Y. Improved SVRG for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pp. 1080–1089, 2016.

Allen-Zhu, Z., Qu, Z., Richtárik, P., and Yuan, Y. Even faster accelerated coordinate descent using non-uniform sampling. In *International Conference on Machine Learning*, pp. 1110–1119, 2016.

Bauschke, H. H., Combettes, P. L., et al. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, volume 2011. Springer, 2017.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

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

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

Defazio, A., Domke, J., and Caetano. Finito: A faster, permutable incremental gradient method for big data problems. In *International Conference on Machine Learning*, pp. 1125–1133, January 2014b.

Gonen, A., Orabona, F., and Shalev-Shwartz, S. Solving ridge regression using sketched preconditioned SVRG. In *International Conference on Machine Learning*, pp. 1397–1405, 2016.

Gower, R., Goldfarb, D., and Richtárik, P. Stochastic block BFGS: squeezing more curvature out of data. In *International Conference on Machine Learning*, pp. 1869–1878, 2016.

Hannah, R., Feng, F., and Yin, W. A2BCD: An asynchronous accelerated block coordinate descent algorithm with optimal complexity. *arXiv preprint arXiv:1803.05578*, 2018a.

Hannah, R., Liu, Y., O'Connor, D., and Yin, W. Breaking the span assumption yields fast finite-sum minimization. In *Advances in Neural Information Processing Systems*, pp. 2314–2323, 2018b.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pp. 315–323, 2013.

Lei, L. and Jordan, M. Less than a single pass: stochastically controlled stochastic gradient. In *Artificial Intelligence and Statistics*, pp. 148–156, 2017.

Lin, H., Mairal, J., and Harchaoui, Z. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pp. 3384–3392, 2015.

Lin, H., Mairal, J., and Harchaoui, Z. An inexact variable metric proximal point algorithm for generic quasi-newton acceleration. *arXiv preprint arXiv:1610.00960*, 2016.

Lin, Q., Lu, Z., and Xiao, L. An accelerated proximal coordinate gradient method. In *Advances in Neural Information Processing Systems*, pp. 3059–3067, 2014.

Mairal, J. Optimization with first-order surrogate functions. In *International Conference on Machine Learning*, pp. 783–791, February 2013.

Mokhtari, A., Eisen, M., and Ribeiro, A. IQN: An incremental quasi-Newton method with local superlinear convergence rate. *SIAM Journal on Optimization*, 28(2): 1670–1698, 2018.

Moritz, P., Nishihara, R., and Jordan, M. A linearly-convergent stochastic L-BFGS algorithm. In *Artificial Intelligence and Statistics*, pp. 249–258, 2016.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87. Springer Science & Business Media, 2013.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning*, pp. 2613–2621, 2017.

Odonoghue, B. and Candes, E. Adaptive restart for accelerated gradient schemes. *Foundations of computational mathematics*, 15(3):715–732, 2015.

Rodomanov, A. and Kropotov, D. A superlinearly-convergent proximal Newton-type method for the optimization of finite sums. In *International Conference on Machine Learning*, pp. 2597–2605, 2016.

Roux, N. L., Schmidt, M., and Bach, F. R. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25*, pp. 2663–2671. Curran Associates, Inc., 2012.

Saad, Y. *Numerical Methods for Large Eigenvalue Problems*. Manchester University Press, 1992.

Shalev-Shwartz, S. SDCA without duality, regularization, and individual convexity. In *International Conference on Machine Learning*, pp. 747–754, 2016.

Shalev-Shwartz, S. and Zhang, T. Stochastic dual coordinate ascent methods for regularized loss. *J. Mach. Learn. Res.*, 14(1):567–599, February 2013.

Shalev-Shwartz, S. and Zhang, T. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 155(1-2):105–145, January 2016.

Wang, X., Wang, X., and Yuan, Y.-x. Stochastic proximal quasi-Newton methods for non-convex composite optimization. *Optimization Methods and Software*, pp. 1–27, 2018.

Xiao, L. and Zhang, T. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.