On Data Preconditioning for Regularized Loss Minimization

Tianbao Yang†, Rong Jin∗‡, Shenghuo Zhu‡

†Department of Computer Science, the University of Iowa
∗Department of Computer Science and Engineering, Michigan State University
tianbao-yang@uiowa.edu, rongjin@cse.msu.edu, shenghuo@gmail.com

Abstract
In this work, we study data preconditioning, a well-known and long-existing technique, for boosting the convergence of first-order methods for regularized loss minimization. It is well understood that the condition number of the problem, i.e., the ratio of the Lipschitz constant to the strong convexity modulus, has a harsh effect on the convergence of the first-order optimization methods. Therefore, minimizing a small regularized loss for achieving good generalization performance, yielding an ill conditioned problem, becomes the bottleneck for big data problems. We provide a theory on data preconditioning for regularized loss minimization. In particular, our analysis exhibits an appropriate data preconditioner and characterizes the conditions on the loss function and on the data under which data preconditioning can reduce the condition number and therefore boost the convergence for minimizing the regularized loss. To make the data preconditioning practically useful, we endeavor to employ and analyze a random sampling approach to efficiently compute the preconditioned data. The preliminary experiments validate our theory.

1 Introduction
Many supervised machine learning tasks end up with solving the following regularized loss minimization problem:

\[
\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(x_i^\top w, y_i) + \frac{\lambda}{2} \|w\|^2_2, \tag{1}
\]

where \(x_i \in \mathcal{X} \subseteq \mathbb{R}^d\) denotes the feature representation, \(y_i \in \mathcal{Y}\) denotes the supervised information, \(w \in \mathbb{R}^d\) is the unknown decision vector and \(\ell(z, y)\) is a convex loss function with respect to \(z\). Examples can be found in classification (e.g., \(\ell(x^\top w, y) = \log(1 + \exp(-yx^\top w))\) for logistic regression) and regression (e.g., \(\ell(x^\top w, y) = (1/2)(x^\top w - y)^2\) for least square regression).

The first-order methods have become the dominant approaches for solving the optimization problem in (1), due to their light computation compared to the second-order methods (e.g., the Newton method). Because of the exponential growth of data, many stochastic optimization algorithms have emerged to further reduce the running time of full gradient methods [18]. One limitation of most first-order methods is that they suffer from a poor convergence if the condition number is small. For instance, stochastic gradient descent (e.g., Pegasos [23]) for solving (1) with a Lipschitz continuous loss function, yields a convergence rate of \(O\left(\frac{\bar{L}^2}{\lambda} \right)\), where \(\bar{L}\) is the Lipschitz constant of the loss function w.r.t \(w\). The convergence rate reveals that the smaller the condition number (i.e., \(\bar{L}^2 / \lambda\)), the worse the convergence. The same phenomenon occurs in optimizing a smooth loss function. Without loss of generality, the iteration complexity – the number of iterations required for achieving an \(\epsilon\)-optimal solution, of SDCA [25], SAG [22] and SVRG [12] for a \(L\)-smooth loss function (whose gradient is \(L\)-Lipschitz continuous) is \(O((n + \frac{L}{\lambda}) \log(\frac{1}{\epsilon}))\). Although the convergence is linear for smooth loss function, however, iteration complexity would be dominated by the condition number.
In Section 3 will exhibit how the data preconditioning affects the two ingredients. Moreover, some theoretical analysis indicates that the value of \( \lambda \) could be as small as \( 1/n \) in order to achieve a small generalization error [24][25]. Therefore, it arises as an interesting question “can we design first-order optimization algorithms that have less severe and even no dependence on the large condition number”?

While most previous works target on improving the convergence rate by achieving a better dependence on the number of iterations \( T \), few works have revolved around mitigating the dependence on the condition number. Bach and Moulines [3] provided a new analysis of the averaged stochastic gradient (ASG) algorithm for minimizing a smooth objective function with a constant step size. They established a convergence rate of \( O(1/T) \) without suffering from the small strong convexity modulus. Two recent works posted online [17, 28] proposed to use importance sampling instead of random sampling in stochastic gradient methods, leading to a dependence on the averaged Lipschitz constant of the individual loss functions instead of the worst Lipschitz constant. However, the convergence rate still badly depends on \( 1/\lambda \).

In this paper, we explore the data preconditioning for reducing the condition number of the problem (1). In contrast to many other works, the proposed data preconditioning technique can be applied together with any first-order methods to improve their convergences. Data preconditioning is a long-existing technique that is used to improve the condition number of a data matrix. In the general form, data preconditioning is to apply \( P^{-1} \) to the data, where \( P \) is a non-singular matrix. It has been employed widely in solving linear systems [1]. In the context of convex optimization, data preconditioning has been applied to conjugate gradient and Newton methods to improve their convergence for ill-conditioned problems [14]. However, it remains unclear how data preconditioning can be used to improve the convergence of first-order methods for minimizing a regularized empirical loss. In the context non-convex optimization, the data preconditioning by ZCA whitening has been widely adopted in learning deep neural networks from image data to speed-up the optimization [21][15], though the underlying theory is barely known. Interestingly, our analysis reveals that the proposed data preconditioner is closely related to ZCA whitening and therefore shed light on the practice widely deployed in deep learning. However, an inevitable critique on the usage of data preconditioning is the computational overhead pertaining to computing the preconditioned data. Thanks to modern cluster of computers, this computational overhead can be made as minimal as possible with parallel computations. We also endeavor to analyze and employ a random sampling approach to efficiently compute the preconditioned data.

In summary, our contributions include: (i) we present a theory on data preconditioning for the regularized loss optimization by introducing an appropriate data preconditioner (Section 3); (ii) we quantify the conditions under which the data preconditioning can reduce the condition number and therefore boost the convergence of the first-order optimization methods (Assumptions 1, 2); (iii) we present an efficient approach for computing the preconditioned data and validate the theory by experiments (Section 3.3)[4].

2 Preliminaries

To facilitate our analysis, we decouple the dependence on the data from the condition number. Henceforth, we denote by \( R \) the upper bound of the data norm, i.e., \( ||x||_2 \leq R \), and by \( L \) the Lipschitz constant of the scalar loss function \( \ell(z,y) \) or its gradient \( \ell'(z,y) \) with respect to \( z \) depending the smoothness of the loss function. Then the gradient w.r.t \( w \) of the loss function is bounded by \( ||\nabla_w \ell(w^\top x,y)||_2 = ||\ell'(w^\top x,y)x|| \leq LR \) if \( \ell(z,y) \) is a \( \lambda \)-Lipschitz continuous non-smooth function. Similarly, the second order gradient can be bounded by \( ||\nabla^2_w \ell(w^\top x,y)||_2 = ||\ell''(w^\top x,y)x^\top x|| \leq LR^2 \) assuming \( \ell(z,y) \) is a \( \lambda \)-smooth function. As a result, the condition number for a \( \lambda \)-Lipschitz continuous scalar loss function is \( LR^2/\lambda \) and is \( L^2R^2/\lambda \) for a \( \lambda \)-smooth loss function. In the sequel, we will refer to \( R \), i.e., the upper bound of the data norm as the data ingredient of the condition number, and refer to \( L/\lambda \) or \( L^2/\lambda \), i.e., the ratio of the Lipschitz constant to the strong convexity modulus as the functional ingredient of the condition number. The analysis in Section 3 will exhibit how the data preconditioning affects the two ingredients.

The condition number of the problem in (1) for the Lipschitz continuous loss function is referred to \( \bar{L}^2/\lambda \), and for the smooth loss function is referred to \( L/\lambda \), where \( L \) is the Lipschitz constant for the function and its gradient w.r.t \( w \), respectively.
Although the proposed data preconditioning can be applied to boost any first-order methods, we will restrict out attention to the stochastic gradient methods, which share the following updates for (1):

$$w_t = w_{t-1} - \eta_t \left( g(w_{t-1}; x) + \lambda w_{t-1} \right), \quad (2)$$

where $g(w_{t-1}; x)$ denotes a stochastic gradient of the loss that depends on the original data $x$. For example, the vanilla SGD for optimizing non-smooth loss uses $g(w_{t-1}; x) = \nabla \ell(w_{t-1} x_i; y_i) x_i$, where $i_t$ is randomly sampled. SAG and SVRG use a particularly designed stochastic gradient for minimizing a smooth loss.

A straightforward approach by exploring data preconditioning for the solving problem in (1) is by variable transformation. Let $\phi$ where

$$\phi(x) = \logistic loss obeys the above condition with $\beta$. Although the proposed data preconditioning can be applied to boost any first-order methods, we will restrict our attention to the stochastic gradient methods, which share the following updates for (1):

$$w_t = w_{t-1} - \eta_t \left( g(w_{t-1}; x) + \lambda w_{t-1} \right), \quad (2)$$

where $g(w_{t-1}; x)$ denotes a stochastic gradient of the loss that depends on the original data $x$. For example, the vanilla SGD for optimizing non-smooth loss uses $g(w_{t-1}; x) = \nabla \ell(w_{t-1} x_i; y_i) x_i$, where $i_t$ is randomly sampled. SAG and SVRG use a particularly designed stochastic gradient for minimizing a smooth loss.

A straightforward approach by exploring data preconditioning for the solving problem in (1) is by variable transformation. Let $\phi$ where

$$\phi(x) = \logistic loss obeys the above condition with $\beta$. Although the proposed data preconditioning can be applied to boost any first-order methods, we will restrict our attention to the stochastic gradient methods, which share the following updates for (1):

$$w_t = w_{t-1} - \eta_t \left( g(w_{t-1}; x) + \lambda w_{t-1} \right), \quad (2)$$

where $g(w_{t-1}; x)$ denotes a stochastic gradient of the loss that depends on the original data $x$. For example, the vanilla SGD for optimizing non-smooth loss uses $g(w_{t-1}; x) = \nabla \ell(w_{t-1} x_i; y_i) x_i$, where $i_t$ is randomly sampled. SAG and SVRG use a particularly designed stochastic gradient for minimizing a smooth loss.

A straightforward approach by exploring data preconditioning for the solving problem in (1) is by variable transformation. Let $\phi$ where

$$\phi(x) = \logistic loss obeys the above condition with $\beta$. Although the proposed data preconditioning can be applied to boost any first-order methods, we will restrict our attention to the stochastic gradient methods, which share the following updates for (1):

$$w_t = w_{t-1} - \eta_t \left( g(w_{t-1}; x) + \lambda w_{t-1} \right), \quad (2)$$

where $g(w_{t-1}; x)$ denotes a stochastic gradient of the loss that depends on the original data $x$. For example, the vanilla SGD for optimizing non-smooth loss uses $g(w_{t-1}; x) = \nabla \ell(w_{t-1} x_i; y_i) x_i$, where $i_t$ is randomly sampled. SAG and SVRG use a particularly designed stochastic gradient for minimizing a smooth loss.
Let $C = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^\top$ denote the sample covariance matrix. We define a smoothed covariance matrix $H$ as $H = \rho I + \frac{1}{n} \sum_{i=1}^{n} x_i x_i^\top = \rho I + C$, where $\rho = \lambda / \beta$. Thus, the transformed problem becomes

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \phi(w^\top x_i, y_i) + \frac{\beta}{2} w^\top H w.$$  

(4)

Using the variable transformation $v \leftarrow H^{1/2}w$, the above problem is equivalent to

$$\min_{v \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \phi(v^\top H^{-1/2}x_i, y_i) + \frac{\beta}{2} \|v\|^2.$$  

(5)

It can be shown that the optimal value of the above preconditioned problem is equal to that of the original problem **(1)**. As a matter of fact, so far we have constructed a data preconditioner as given by $P^{-1} = H^{-1/2}$ that transforms the original feature vector $x$ into a new vector $H^{-1/2}x$. It is worth noting that the data preconditioning $H^{-1/2}x$ is similar to the ZCA whitening transformation, which transforms the data using the covariance matrix, i.e., $C^{-1/2}x$ such that the data has identity covariance matrix. Whitening transformation has found many applications in image processing **[19]**, and it is also employed in independent component analysis **[11]** and optimizing deep neural networks **[21]** **[15]**.

### 3.2 Condition Number

Besides the data, there are two additional alterations: (i) the strong convexity modulus is changed from $\lambda$ to $\beta$ and (ii) the loss function becomes $\phi(z, y) = \ell(z, y) - 2z^2$. Before discussing the convergence rates of the first-order optimization methods for solving the preconditioned problem in (5), we elaborate on how the two ingredients of the condition number are affected: (i) the functional ingredient namely the ratio of the Lipschitz constant of the loss function to the strong convexity modulus and (ii) the data ingredient namely the upper bound of the data norm. We first analyze the change of the functional ingredient as summarized in the following lemma, whose proof and other proofs can be found in the supplementary material.

**Lemma 1.** If $\ell(z, y)$ is a $L$-Lipschitz continuous function, then $\phi(z, y)$ is $(L + \beta r)$-Lipschitz continuous for $|z| \leq r$. If $\ell(z, y)$ is a $L$-smooth function, then $\phi(z, y)$ is a $(L - \beta)$-smooth function.

**Lemma 1** indicates that after the data preconditioning the functional ingredient becomes $(L + \beta r)^2 / \beta$ for a $L$-Lipschitz continuous non-smooth loss function and $(L - \beta) / \beta$ for a $L$-smooth function.

Next, we analyze the upper bound of the preconditioned data $\tilde{x} = H^{-1/2}x$. Noting that $\|\tilde{x}\|_2^2 = x^\top H^{-1}x$, in what follows we will focus on bounding $\max_i x_i^\top H^{-1}x_i$. We first derive and discuss the bound of the expectation $E_i[x_i^\top H^{-1}x_i]$ that is useful in proving the expectational convergence bound. Many discussions also carry over to the upper bound for individual data. Let $\frac{1}{\sqrt{n}}X = \frac{1}{\sqrt{n}}(x_1, \ldots, x_n) = USV^\top$ be the singular value decomposition of $X$, where $U \in \mathbb{R}^{d \times d}$, $V \in \mathbb{R}^{n \times d}$ and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_d), \sigma_1 \geq \ldots \geq \sigma_d$, then $C = USV^\top X$ is the eigen-decomposition of $C$.

Thus, we have

$$E_i[x_i^\top H^{-1}x_i] = tr(H^{-1}C) = \sum_{i=1}^{d} \frac{\sigma_i^2}{\sigma_i^2 + \rho} \leq \gamma(C, \rho),$$

We refer to $\gamma(C, \rho)$ as the numerical rank of $C$ with respect to $\rho$. The first observation is that $\gamma(C, \rho)$ is a monotonically decreasing function in terms of $\rho$. It is straightforward to show that if $X$ is low rank, e.g., $\text{rank}(X) = k \ll d$, then $\gamma(C, \rho) < k$. If $C$ is full rank, the value of $\gamma(C, \rho)$ will be affected by the decay of its eigenvalues. Bach **[2]** has derived the order of $\gamma(C, \rho)$ in $\rho$ under two different decays of the eigenvalues of $C$. The conclusion is that if the eigenvalues of $C$ follow a polynomial decay $\sigma_i^2 = i^{-2\tau}, \tau \geq 1/2$, then $\gamma(C, \rho) \leq O(\rho^{-(1/2\tau)})$, and if the eigenvalues of $C$ satisfy an exponential decay $\sigma_i^2 = e^{-\tau i}$, then $\gamma(C, \rho) \leq O(\log \left( \frac{1}{\rho} \right))$. In statistics **[10]**, $\gamma(C, \rho)$ is also referred to as the effective degree of freedom. In order to prove high probability bounds, we have to derive the upper bound for individual $x_i^\top H^{-1}x_i$. To this end, we introduce the following measure to quantify the incoherence of $V$.

**Definition 1.** The generalized incoherence measure of an orthogonal matrix $V \in \mathbb{R}^{n \times d}$ w.r.t to $(\sigma_1^2, \ldots, \sigma_d^2)$ and $\rho > 0$ is

$$\mu(\rho) = \max_{1 \leq i \leq n} \frac{n}{\gamma(C, \rho)} \sum_{j=1}^{d} \frac{\sigma_j^2}{\sigma_j^2 + \rho} V_{ij}^2.$$
Similar to the incoherence measure introduced in the compressive sensing theory [7], the generalized incoherence also measures the degree to which the rows in $V$ are correlated with the canonical bases. We can also establish the relationship between the two incoherence measures. The incoherence of an orthogonal matrix $V \in \mathbb{R}^{n \times n}$ is defined as $\mu = \max_{ij} \sqrt{\sum_{k} V_{ik} V_{jk}}$ [7]. With simple algebra, we can show that $\mu(\rho) \leq \rho^2$. Since $\mu \in [1, \sqrt{n}]$, therefore $\mu(\rho) \in [1, n]$. It has been established in matrix completion [8] and matrix approximation [27] that highly coherent matrix $K = X^T X/n$ is difficult to recover by randomly sampling entries or columns. As manifested in the following analysis, the same logic applies to data preconditioning for reducing the condition number. Given the definition of $\mu(\rho)$, we have the following lemma on the upper bound of $x_i^T H^{-1} x_i$.

**Lemma 2.** $x_i^T H^{-1} x_i \leq \mu(\rho) \gamma(C, \rho), \quad i = 1, \ldots, n$.

The theorem below states the condition number of the preconditioned problem (5).

**Theorem 2.** If $\ell(z, y)$ is a $L$-Lipschitz continuous function satisfying the condition in Assumption 2 then the condition number of the optimization problem in (5) is bounded by $\frac{(L + \beta)\mu(\rho) \gamma(C, \rho)}{\beta}$, where $\rho = \lambda/\beta$. If $\ell(z, y)$ is a $L$-smooth function satisfying the condition in Assumption 7 then the condition number of (5) is $\frac{(L + \beta)\mu(\rho) \gamma(C, \rho)}{\beta}$.

Following the above theorem and previous discussions on the condition number, we can establish the following conditions that ensure that the data preconditioning can reduce the condition number.

**Assumption 2.** Assume $\frac{(L + \beta)\mu(\rho) \gamma(C, \rho)}{\beta} \leq R^2$ if $\ell(z, y)$ is a $L$-Lipschitz continuous function; and assume $\frac{(L + \beta)\mu(\rho) \gamma(C, \rho)}{\beta} \leq \frac{R^2}{\lambda}$ if $\ell(z, y)$ is $L$-smooth.

**Remark 1:** In the above conditions, we make explicit the effect from the loss function and the data. In the right hand side, the quantity $R^2/\mu(\rho) \gamma(C, \rho)$ measures the ratio between the maximum norm of the original data and that of the preconditioned data. The left hand side depends on the property of the loss function and the value of $\lambda$. Due to the unknown value of $r$ for non-smooth optimization, we first discuss the indications of the condition for the smooth loss function and comment on the value of $r$ in Remark 2. Let us consider $\beta, L \approx \Theta(1)$ (e.g. in least square regression) and $\lambda = \Theta(1/n)$. Therefore $\rho = \lambda/\beta = \Theta(1/n)$. The condition in Assumption 2 for the smooth loss requires the ratio between the maximum norm of the original data and that of the preconditioned data is larger than $\Theta(1/n)$. If the eigenvalues of the covariance matrix follow an exponential decay, then $\gamma(C, \rho) = \Theta(1)$ and the condition indicates that $\mu(\rho) \leq \Theta(n R^2)$, which can be satisfied easily if $R > 1$ due to the fact $\mu(\rho) \leq n$. If the eigenvalues follow a polynomial decay $i^{-2\tau}, \tau \geq 1/2$, then $\gamma(C, \rho) \leq \Theta(n^{1/(2\tau)}) = O(n^{1/(2\tau)})$, then the condition indicates that $\mu(\rho) \leq O(n^{1-\tau} R^2)$, which means the faster the decay of the eigenvalues, the easier for the condition to be satisfied. If $\beta$ is a small value (e.g. in logistic regression), then the satisfaction of the condition depends on the balance between the factors $\lambda, L, \beta, \gamma(C, \rho), \mu(\rho), R^2$. In practice, if $\beta, L$ is known we can always check the condition by calculating the ratio between the maximum norm of the original data and that of the preconditioned data and comparing it with $\lambda/\beta - \lambda/L$. If $\beta$ is unknown, we can take a trial and error method by tuning $\beta$ to achieve the best performance.

**Remark 2:** Next, we comment on the value of $r$ for non-smooth optimization. It was shown in [23] the optimal solution $w_*$ to (1) can be bounded by $\|w_*\| \leq O(\frac{1}{\sqrt{\lambda}})$. Therefore we can ensure $|z| = w^T x \leq R/\sqrt{\lambda}$ and thus $r^2 \leq R^2/\lambda$. In the worse case $r^2 = R^2/\lambda$, the condition number of the preconditioned problem for non-smooth optimization is bounded by $O((R^2 \mu(\rho) \gamma(C, \rho))/\lambda)$. Then there may be no improvement for convergence. In practice, we usually observe $r < R/\sqrt{\lambda}$, especially when $\lambda$ is very small. On the other hand, when $\lambda$ is too small the step sizes $1/(\lambda \tau)$ of SGD on the original problem at the beginning of iterations are extremely large, making the optimization unstable.

**Remark 3:** We can also analyze the straightforward approach by solving the preconditioned problem in (3) using $P^{-1} = H^{-1/2}$. The bound of the data ingredient follows the same analysis. The functional ingredient is $O(L(\sigma_0^2 + \rho)/\lambda)$ due to that $\lambda u^T H^{-1} u \geq \lambda(\sigma_0^2 + \rho)\|u\|^2_2$. If $\lambda \ll \sigma_0^2$, then the condition number of the preconditioned problem still heavily depends on $1/\lambda$. Therefore, solving the naive preconditioned problem (3) with $P^{-1} = H^{-1/2}$ may not boost the convergence, which is also verified in Section 4 by experiments.
Finally, we use the example of SAG for solving least square regression to demonstrate the benefit of data preconditioning. Similar analysis carries on to other variance reduced stochastic optimization algorithms \cite{Bach,Moulines}. When $\lambda = 1/n$ the iteration complexity of SAG would be dominated by $O(R^2n \log(1/\epsilon))$ \cite{Bach} – tens of epochs depending on the value of $R^2$. However, after data preconditioning the iteration complexity becomes $O(n \log(1/\epsilon))$ if $n \geq R^2$, where $R$ is the upper bound of the preconditioned data, which would be just few epochs. In comparison, Bach and Moulines’ algorithm \cite{Bach} suffers from an $O(d \log R^2)$ iteration complexity that could be much larger than $O(n \log(1/\epsilon))$, especially when required $\epsilon$ is small and $R$ is large. Our empirical studies in Section 4 indeed verify these results.

### 3.3 Efficient Data Preconditioning

Now we proceed to address the third question, i.e., how to efficiently compute the preconditioned data. The data preconditioning using $H^{-1/2}$ needs to compute the square root inverse of $H$ times $x$, which usually costs a time complexity of $O(d^3)$. On the other hand, the computation of the preconditioned data for least square regression is as expensive as computing the closed form solution, which makes data preconditioning not attractive. In this section, we analyze an efficient data preconditioning by random sampling. As a compromise, we might lose some gain in convergence. The key idea is to construct the preconditioner by sampling a subset of $m$ training data, denoted by $\mathcal{D} = \{\hat{x}_1, \ldots, \hat{x}_m\}$. Then we construct new loss functions for individual data as,

$$
\psi(w^T x_i, y_i) = \begin{cases} 
\ell(w^T x_i, y_i) - \frac{\hat{\beta}}{2} (w^T x_i)^2 & \text{if } x_i \in \mathcal{D} \\
\ell(w^T x_i, y_i) & \text{otherwise}
\end{cases}
$$

Then we can show that the original problem is equivalent to

$$
\min_{v \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \psi(v^T \hat{H}^{-1/2} x_i, y_i) + \frac{\hat{\beta}}{2} \|v\|_2^2.
$$

(6)

where $\hat{H} = \hat{\rho}I + \frac{1}{m} \sum_{i=1}^{m} \hat{x}_i \hat{x}_i^T$, $\hat{\rho} = \frac{m}{n} \hat{\rho}$, $\hat{\beta} = \frac{m}{n} \beta$. Thus, $\hat{H}^{-1/2} x_i$ defines the new preconditioned data. It can be shown that the time complexity for computing $\hat{H}^{-1} x$ is just $O(m^2 d)$. Due to the limit of space, we include the discussions about the time complexity for computing the preconditioned data using $\hat{H}$ in supplemental material. It is worth noting that the random sampling approach has been used previously to construct the stochastic Hessian \cite{Bach}. Here, we analyze its impact on the condition number. The same analysis about the Lipschitz constant of the loss function carries over to $\psi(z, y)$, except that $\psi(z, y)$ is at most $L$-smooth if $\ell(z, y)$ is $L$-smooth. The following theorem allows us to bound the norm of the preconditioned data using $\hat{H}$.

**Theorem 3.** For any $\delta \leq 1/2$, If $m \geq \frac{2}{\delta^2} (\mu(\hat{\rho}) \gamma(C, \hat{\rho}) + 1)(t + \log d)$, with a probability $1 - e^{-t}$, 

$$
x_i^T \hat{H}^{-1} x_i \leq (1 + 2\delta) \mu(\hat{\rho}) \gamma(C, \hat{\rho}), \forall i = 1, \ldots, n
$$

The theorem indicates that the upper bound of the preconditioned data is only scaled up by a small constant factor with an overwhelming probability compared to that using all data points to construct the preconditioner under moderate conditions. Thus, similar conditions can be established for the data preconditioning using $\hat{H}^{-1/2}$ to improve the convergence rate. Moreover, varying $m$ may exhibit a tradeoff between the two ingredients understood as follows. Since $\gamma(C, \hat{\rho})$ is a monotonically decreasing function w.r.t $\rho$, therefore $\gamma(C, \hat{\rho}) \leq \gamma(C, \rho)$ and the data ingredient $x_i^T \hat{H}^{-1} x_i$ increases as $m$ increases. On the other hand, the functional ingredient $L/\beta$ would decrease as $m$ increases. Finally, we are aware of that when original data is sparse the preconditioned data may become dense, which may increase the per-iteration cost. It would pose stronger conditions for the data preconditioning to take effect. In our experiments, we focus on dense data sets.

### 4 Experiments

We first present some simulation results to verify our theory. To control the inherent data properties (i.e., numerical rank and incoherence), we generate synthetic data. We first generate a standard Gaussian matrix $M \in \mathbb{R}^{d \times n}$ and then compute its SVD $M = USV^T$. We use $U$ and $V$ as the left and right singular vectors to construct the data matrix $X \in \mathbb{R}^{d \times n}$. In this way, the incoherence
Figure 1: Synthetic data: (a) compares the condition number of the preconditioned problem (solid lines) with that of the original problem (dashed lines of the same color) by varying the value of \( \beta \) (a property of the data) and varying the decay of the eigenvalues. (b) compares the condition number by varying the value of \( \lambda \) (measuring the difficulty of the problem) and varying the decay of the eigenvalues. (c) compares the condition number vs the values of \( \lambda \) by varying the value of \( \beta \) (a property of the loss function) and varying the decay of the eigenvalues. (d) compares convergence of SVRG using full data and sub-sampled data for constructing the preconditioner on the synthetic regression data with proposed data preconditioning with the straightforward approach by solving (4) (simple-precond).

Next, we present some experimental results on convergence. In our experiments we focus on two tasks namely least square regression and logistic regression, and we study two variance reduced SGDs namely stochastic average gradient (SAG) \(^{22}\) and stochastic variance reduced SGD (SVRG) \(^{12}\). For SVRG, we set the step size as \( 0.1/L \), where \( L \) is the smoothness parameter of the individual loss function plus the regularization term in terms of \( w \). The number of iterations for the inner loop in SVRG is set to \( 2n \) as suggested by the authors. For SAG, the theorem indicates the step size is less than \( 1/(16L) \) while the authors have reported that using large step sizes like \( 1/L \) could yield better performances. Therefore we use \( 1/L \) as the step size unless otherwise specified. Note that we are not aiming to optimize the performances by using pre-trained initializations \(^{12}\) or by tuning the step sizes. Instead, the initial solution for all algorithms are set to zeros and the step sizes used in our experiments are either suggested in previous papers or have been observed to perform well in practice. In all experiments, we compare the convergence vs the number of epochs.
We generate synthetic data as described above. For least square regression, the response variable is generated by \( y = w^\top x + \epsilon \), where \( w_k \sim \mathcal{N}(0, 100) \) and \( \epsilon \sim \mathcal{N}(0, 0.01) \). For logistic regression, the label is generated by \( y = \text{sign}(w^\top x + \epsilon) \). Figure 2 shows the objective curves for minimizing the two problems by SVRG, SAG w/ and w/o data preconditioning. The results clearly demonstrate data preconditioning can significantly boost the convergence. To further justify the proposed theory of data preconditioning, we also compare with the straightforward approach that solves the preconditioned problem in (3) with the same data preconditioner. Due to the limit of space, we only show one result in Figure 1(c). More results and discussions can be found in supplement. These results verify that using the straightforward data preconditioning may not boost the convergence. We also validate the performance of the efficient data preconditioning in Section 3.3. We generate a synthetic data as before with \( d = 5000 \) features and with eigenvalues following the poly-0.5 decay, and plot the convergence of SVRG for solving least square regression with different preconditioners, including \( H^{-1/2} \) and \( \tilde{H}^{-1/2} \) with different values of \( m \). The results are shown in Figure 1(d), which demonstrate that using \( m = 100 \) training samples for constructing the data preconditioner is sufficient for data preconditioning to boost the convergence.

Next, we present some experimental results on real data. We choose two data sets, the million songs data (MSD) [4] for regression and the CIFAR-10 data [13] for classification. The task on MSD is to predict the year of a song based on the audio features. Following the previous work, we map the target variable of year from 1922 \( \sim 2011 \) into \([0, 1]\). The task on CIFAR-10 is to predict the object in \( 32 \times 32 \) RGB images. Following [13], we use the mean centered pixel values as the input. To perform logistic regression on CIFAR-10, we train a binary classifier to classify dogs from cats with a total of 10000 images. The experiment results and the setup are shown in Figure 3 in which we also report the convergence of Bach and Moulines’ ASG algorithm [3] on the original problem with a step size \( c/\tilde{L}^2 \), where \( c \) is tuned in a range from 1 to 10. The step size for both SAG and SVRG is set to \( 1/\tilde{L} \). The results again demonstrate that the data preconditioning could yield significant speed-up in convergence, and the efficient data preconditioning could be useful. Finally, we briefly comment on the running time. We observe that the per-iteration time on the preconditioned problem is similar to that on the original problem. Since the convergence on the preconditioned problem is faster than that on the original problem, therefore the optimization time on the preconditioned problem, is shorter than that on the original problem. The computational overhead of the efficient data preconditioning on MSD is 1.68 seconds and on CIFAR-10 is 0.56 second, which are marginal as compared with the running time for one epoch, roughly 9 seconds on MSD and 0.65 second on CIFAR-10.

5 Conclusions

We have presented a theory of data preconditioning for boosting the convergence of first-order optimization methods for the regularized loss minimization. We characterized the conditions on the loss function and the data under which the condition number of the regularized loss minimization problem can be reduced and thus the convergence can be improved. We also presented an efficient method for computing the preconditioned data and analyzed the condition number using the efficient preconditioning. Our experimental results validate our theory and demonstrate the potential advantage of the data preconditioning for solving ill-conditioned big data optimization problems.
References

[1] O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, New York, NY, USA, 1994.

[2] F. Bach. Sharp analysis of low-rank kernel matrix approximations. In *COLT*, pages 185–209, 2013.

[3] F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $o(1/n)$. In *NIPS*, pages 773–781, 2013.

[4] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, pages 591–596, 2011.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[6] R. H. Byrd, G. M. Chin, W. Neveitt, and J. Nocedal. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, pages 977–995, 2011.

[7] E. J. Candès and J. Romberg. Sparsity and incoherence in compressive sampling. *Inverse Problems*, 23:969–985, 2007.

[8] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inf. Theor.*, 56:2053–2080, 2010.

[9] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., 2001.

[11] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Netw.*, pages 411–430, 2000.

[12] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, pages 315–323, 2013.

[13] A. Krizhevsky. Learning multiple layers of features from tiny images. Master’s thesis, 2009.

[14] S. Langer. *Preconditioned Newton Methods for Ill-posed Problems*. 2007.

[15] Y. LeCun, L. Bottou, G. Orr, and K. Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1998.

[16] J. Martens. Deep learning via hessian-free optimization. In *ICML*, pages 735–742, 2010.

[17] D. Needell, N. Srebro, and R. Ward. Stochastic gradient descent and the randomized kaczmarz algorithm. *CoRR*, 2013.

[18] Y. Nesterov. *Introductory lectures on convex optimization : a basic course*. Applied optimization. Kluwer Academic Publ., Boston, Dordrecht, London, 2004.

[19] M. Petrou and P. Bosdogianni. *Image processing - the fundamentals*. Wiley, 1999.

[20] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *ICCV*, pages 1762–1769, 2011.

[21] M. Ranzato, A. Krizhevsky, and G. E. Hinton. Factored 3-way restricted boltzmann machines for modeling natural images. In *AISTATS*, pages 621–628, 2010.

[22] M. W. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *CoRR*, abs/1309.2388, 2013.

[23] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: primal estimated sub-gradient solver for svm. *Math. Program.*, 127(1):3–30, 2011.

[24] S. Shalev-Shwartz and N. Srebro. Svm optimization: inverse dependence on training set size. In *ICML*, pages 928–935, 2008.

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

[26] K. Sridharan, S. Shalev-Shwartz, and N. Srebro. Fast rates for regularized objectives. In *NIPS*, pages 1545–1552, 2008.
[27] A. Talwalkar and A. Rostamizadeh. Matrix coherence and the nystrom method. In *UAI*, pages 572–579, 2010.

[28] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. 2014.