Towards Memory-friendly Deterministic Incremental Gradient Method

Jiahao Xie **Hui Qian*** **Zebang Shen** **Chao Zhang**
College of Computer Science and Technology, Zhejiang University, China
{xiejh, qianhui, shenzebang, zczju}@zju.edu.cn

Abstract

Incremental Gradient (IG) methods are classical strategies in solving finite sum minimization problems. Deterministic IG methods are particularly favorable in handling massive scale problem due to its memory-friendly data access pattern. In this paper, we propose a new deterministic variant of the IG method SVRG that blends a periodically updated full gradient with a component function gradient selected in a cyclic order. Our method uses only $\mathcal{O}(1)$ extra gradient storage without compromising the linear convergence. Empirical results demonstrate that the proposed method is advantageous over existing incremental gradient algorithms, especially on problems that does not fit into physical memory.

1 Introduction

We consider the following finite sum minimization problem:

$$\min_{w\in\mathbb{R}^d} F(w) = \frac{1}{N}\sum_{i=1}^{N} f_i(w), \tag{1}$$

where each component function $f_i(w)$ is differentiable and potentially non-convex, and the objective function $F(w)$ is strongly convex. This formulation arises from many convex machine learning problems [Hastie et al., 2009], and also constitutes an important part of recent accelerated non-convex solvers [Carmon et al., 2016, Allen-Zhu, 2017] where f_i can be non-convex.

A classical method for solving Problem (1) is Gradient Descent (GD), which requires expensive full gradient computation per iteration. By exploiting the finite sum structure, Incremental Gradient (IG) methods select and process the component function one at a time to reduce the iteration cost, and hence are particularly favorable when N is huge. A key ingredient in designing IG methods is the component selection strategy. From such perspective, IG methods can be divided into two subclasses: the stochastic ones and the deterministic ones.

Stochastic Incremental Gradient (StocIG) methods date back to [Robbins and Monro, 1951] and have regained interest recently due to the surging demand for tackling large scale optimization problems. Typical StocIG methods, e.g. Stochastic Gradient Descent (SGD) and its Variance Reduced (VR) variants SVRG [Johnson and Zhang, 2013] and SAGA [Defazio et al., 2014], utilize the *with-replacement* sampling scheme in selecting component functions. While leading to unbiased gradient estimator and simple convergence analysis, such scheme requires random access to the data points. As an alternative, *without-replacement* sampling scheme has also been studied in the literature [Shamir, 2016, Gürbüzbalaban et al., 2015, Ying et al., 2017], which requires to reshuffle the whole dataset periodically. However, when the dataset is too large to fit into memory, both random access and random reshuffle entail heavy I/O cost, thereby rendering the StocIG methods impractical for the big data scenario. Typical deterministic IG algorithms like Deterministic Incremental Gradient (DIG) [Bertsekas, 2011] and Incremental Aggregate Gradient (IAG) [Blatt et al., 2007] process data in a cyclic scheme, which promotes the spatial locality property significantly and therefore reduces the page fault rate when handling huge dataset using limited memory. However, existing deterministic IG methods can be further improved. In DIG, to guarantee the convergence to the exact solution of Problem (1), diminishing step size is required and only sublinear convergence rate is obtained. In IAG, gradients of all N component functions need to be stored in order to ensure linear convergence which results in extra expensive writing operation per itera-

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84.
*Corresponding author.

Table 1: Summary of SGD, SVRG, SAGA, DIG, IAG, and SIG

Algorithm	SGD	SVRG	SAGA	DIG	IAG	SIG
Data Request of Update Rule (Read/Write)	R	R	R+W	R	R+W	R
Component Selection Scheme (Stochastic/Deterministic)	S	S	S	D	D	D
Extra Storage	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(N)$	$\mathcal{O}(1)$	$\mathcal{O}(N)$	$\mathcal{O}(1)$
Linear Convergence Rate	No	Yes	Yes	No	Yes	Yes

tion.

Inspired by the aforementioned works, we propose a new deterministic variant of SVRG with contributions listed as follows.

- By blending a periodically updated full gradient with a component function gradient selected in a cyclic order, we develop an algorithm named Snapshot-based Incremental Gradient (SIG) that uses only $\mathcal{O}(1)$ extra gradient storage. No random access is involved. Additionally, we incorporate the momentum technique into SIG to derive an efficient variant.
- The linear convergence of the proposed SIG algorithm is established under the assumptions that objective $F(w)$ is strongly convex and each component function is smooth. We make no assumption on the convexity of the individual component function. Moreover, our analysis also covers random shuffle as a special case.

We conduct numerical experiments and show that the SIG algorithm is comparable to SVRG in terms of convergence rate. More importantly, when the dataset is too large to fit into main memory, our algorithm shows significant advantage over existing incremental gradient algorithms, both stochastic and deterministic.

2 Assumptions and Notations

Throughout this paper, we consider Problem (1) under following assumptions:

Assumption 1. *The component function f_i is L-Lipschitz smooth for every $i \in \{1, \dots, N\}$: for any $w_1, w_2 \in \mathbb{R}^d$,*

$$\|\nabla f_i(w_1) - \nabla f_i(w_2)\| \leq L\|w_1 - w_2\|. \tag{2}$$

Assumption 2. *$F(w)$ is μ-strongly convex: for any $w_1, w_2 \in \mathbb{R}^d$,*

$$(\nabla F(w_1) - \nabla F(w_2))^T (w_1 - w_2) \geq \mu\|w_1 - w_2\|^2. \tag{3}$$

We denote the Euclidean norm by $\|\cdot\|$. With Assumption 1, one can easily show that F is also L-smooth. The above assumptions imply that $\mu \leq L$ and we define the condition number of F as $\kappa = L/\mu$.

3 Preliminary

Existing IG methods can be written in the following compact form

$$w_{t+1} = w_t - \eta_t g_t,$$

where w_t is the iterate and g_t approximates the full gradient $\nabla F(w_t)$ using the selected component function. Different methods vary in their component function selection strategies and gradient approximation schemes. We summarize some representative algorithms in Table 1.

3.1 Gradient Approximation Scheme

We describe three principled gradient approximation schemes that construct g_t given f_{i_t} is selected.

Individual Approximation: SGD and DIG estimate $\nabla F(w_t)$ with the gradient of an individual component function $g_t = \nabla f_{i_t}(w_t)$. While reading only one data point in each iteration and requiring only $\mathcal{O}(1)$ gradient storage, such approximation introduces non-vanishing noise and compromises the linear convergence in GD. Both SGD and DIG use a diminishing step size and admit only a sublinear convergence rate.
Average Approximation: To reduce the noise, SAG and IAG approximate $\nabla F(w_t)$ with the average of historical component gradients. Specifically, $g_t = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(w_{\tau_i})$, where τ_i denotes the last iteration when the i^{th} component function is accessed. Using such gradient approximation rule, SAG and IAG obtain linear convergence with random and cyclical data access respectively. However, both methods maintain N component gradients, thus require a huge storage. Besides, such maintenance has to perform both READ and WRITE operations on the storage that scale with the problem dataset and can be even larger in some other interesting cases (refer to the Nonconvex solver).
Mix Approximation: An alternative to the average approximation scheme is to mix the current incremental gradient with the full gradient at some periodically

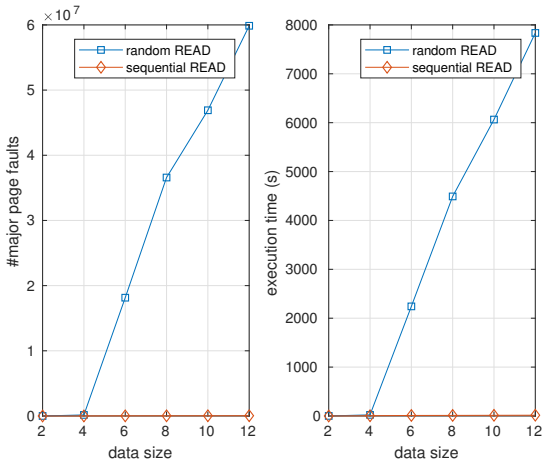

Figure 1: Comparison of major page faults and total execution time of two programs that read 10% data from datasets of diferernt sizes $(2, 4, \ldots, 12\text{GB})$ in random order and cyclic order respectively under 4GB available main memory.

updated snapshot $\tilde{w}$. Concretely,

$$g_t = \nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}) + \nabla F(\tilde{w}),$$

where $\nabla F(\tilde{w})$ is computed once and kept in memory. Such scheme reads one data point per iteration and only requires $\mathcal{O}(1)$ gradient storage, and if with-replacement sampling scheme is adopted, linear convergence can be obtained. However, there is no mix approximation counterpart in the deterministic IG literature.

3.2 Memory-Friendly Data Accessing Patterns

When the dataset is too large to fit into physical memory, certain data access patterns become more efficient than others due to the modern memory hierarchy [Silberschatz et al., 2014].

Compared to stochastic IG methods, deterministic IG methods leverage the superiority of sequential data access, which usually results in significantly fewer page faults and hence fewer swapping operations during component function selection from a huge population. Figure 1 compares the number of major page faults[1], and the execution time of two programs that perform the same amount of sequential and random READ operations respectively. It can be seen that sequential data access has much fewer major page faults than the random one as the data size becomes larger.

[1]A major page fault occurs when disk I/O is involved to satisfy the page request.

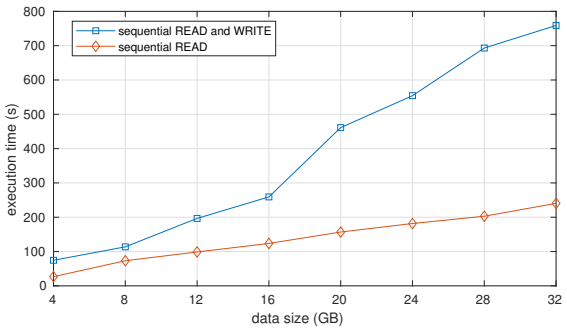

Figure 2: Execution time of two programs that access data in a cyclic order from datasets of different sizes $(4, 8, \ldots, 32\text{GB})$ under 4GB available main memory. One program performs READ operations which corresponds to the selection of component functions while the other performs READ and WRITE operations which corresponds to the maintenance of individual gradients.

The WRITE operation is considerably more expensive than the READ operation, which is amplified when the swap-out occurs frequently. Figure 2 compares the execution time of a program with READ-only routines and a program with READ and WRITE routines. The latter one increases up to 3.4 times longer than the former one as the data size gets larger than the physical memory. Algorithms like SAG and IAG work in the same manner as the latter program, whose performance is undermined by the repeated write operations. Consequently, a memory-friendly method should circumvent such pitfall.

3.3 Convex Learning Tasks and Beyond

To further motivate our work, we give some important tasks that can be casted as Problem (1).

Empirical Risk Minimization Given a dataset $\{x_i, y_i\}_{i=1}^N$, the regularized Empirical Risk Minimization (ERM) with linear predictor is defined as

$$F(w) = \frac{1}{N}\sum_{i=1}^{N} \phi(w^\top x_i, y_i) + \frac{\mu}{2}\|w\|^2 \qquad (4)$$

where the empirical loss function $\phi(a, y)$ is usually convex with respect to the first parameter, e.g. $\phi(a, y) = \frac{1}{2}\|a - y\|^2$ in Ridge Regression and $\phi(a, y) = log(1 + \exp(-y \cdot a))$ in l_2-Logistic Regression. The regularization $\mu\|w\|^2/2$ prevents overfitting. By writing $f_i(w) = \phi(w^\top x_i, y_i) + \frac{\mu}{2}\|w\|^2$, we can phrase Problem (4) as Problem (1).

Subroutine in Non-convex Solver Without assuming the convexity on the individual component function, Problem (1) captures tasks beyond con-

Algorithm 1: SIG Algorithm

input : η, w_0^0, S
for $s = 0, 1, \ldots, S-1$ **do**
  compute $G = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(w_0^s)$
  for $t = 0, \ldots, N-1$ **do**
    $i_t = t + 1$
    $w_{t+1}^s = w_t^s - \eta\big(\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s) + G\big)$
  end
  $w_0^{s+1} = w_N^s$
end
return: w_0^S

vex ERM and is of independent interest [Shalev-Shwartz, 2016, Allen-Zhu and Yuan, 2016]. In particular, when minimizing nonconvex smooth finite sum $\Phi(w) = \frac{1}{N}\sum_{i=1}^{N} \phi_i(w)$, accelerated solvers like [Allen-Zhu, 2017, Carmon et al., 2016] iteratively update some $\tilde{w}$ so that a local minimum of $\Phi(w)$ exists in its vicinity. During the update, their algorithms call a subroutine to solve the following problem

$$\min_w \frac{1}{N}\sum_{i=1}^{N} \phi_i(w) + \frac{L}{2}\|w - \tilde{w}\|^2 \tag{5}$$

where L is the smoothness parameter of $\Phi(w)$. It can be checked that Problem (5) falls in the same category considered in this paper. Hence our work potentially accelerates solving important nonconvex problems in deep neural network.

4 Snapshot-based Incremental Gradient Algorithm

In this section, we propose a deterministic IG method that (1) accesses data in sequential order, (2) requires no maintenance of historical gradients, and (3) converges linearly to the global optimal, which is summarized in Algorithm 1. Our algorithm utilizes both the incremental gradient and the full gradient at some snapshot point, and hence is called Snapshot-based Incremental Gradient (SIG).

We use the following gradient approximation scheme in the s^{th} epoch:

$$g_t^s = \nabla f_{t+1}(w_t^s) - \nabla f_{t+1}(w_0^s) + \nabla F(w_0^s),$$

which is similar to SVRG. However, there are several fundamental distinctions. First, SVRG selects component functions randomly, which leads to inefficiency in handling massive problems; in contrast, SIG accesses data in a sequential and thereby memory-friendly pattern. Additionally, the inner loop length of SVRG depends on the condition number of the problem[2], but in SIG it is fixed to N, which avoids the complication in condition number estimation. Our settings of sequential data access and inner loop length are necessary to give the important observation

$$\sum_{t=0}^{N-1} g_t^s = \sum_{t=0}^{N-1} \nabla f_{t+1}(w_t^s), \tag{6}$$

and therefore

$$w_0^{s+1} = w_0^s - \eta\sum_{t=0}^{N-1} g_t^s = w_0^s - \eta\sum_{t=0}^{N-1} \nabla f_{t+1}(w_t^s), \tag{7}$$

which is vital to our analysis. In our analysis, we treat $\sum_{t=0}^{N-1} \nabla f_{t+1}(w_t^s)$ as a perturbed gradient evaluated at w_0^s which leads to a more succinct proof, compared to complicated Lyapunov construction in [Gurbuzbalaban et al., 2017]. Observation (7) also appears in the proof of DIG [Bertsekas, 2011], however, the perturbation to the gradient $\nabla F(w_0^s)$ does not vanish as the DIG algorithm progresses. Consequently only sublinear convergence is obtained for DIG as diminishing step size is necessary. As we shall see in Lemma 1, the design of g_t^s enables us to show that perturbation of gradient is bounded by the distance to the optimal and hence vanishes as the algorithm converges.

5 Convergence Analysis

In our analysis, we consider a more general case in which the component functions can be processed in arbitrary order, i.e., $i_t = \sigma^s(t+1)$ where σ^s is an arbitrary permutation of $\{1, \ldots, N\}$. The sequential order ($i_t = t + 1$) is a special case that we are particularly interested in.

Since in SIG the component function we choose at each iteration is not independent of the previous ones, it is difficult to analyze the relations between two consecutive iterates. In Theorem 1, we bound the distance between the iterates at the start of two consecutive epochs instead.

Before diving into the convergence analysis of Algorithm 1, we present the following useful lemma.

Lemma 1. *Suppose that Assumption 1 holds and the step size η satisfies $\eta < \frac{1}{NL}$, then*

$$\sum_{t=0}^{N-1} \|\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s)\| \le \frac{\eta L^2 N^2}{2(1-\eta LN)}\|w_0^s - w^*\|$$

[2]There is a less known variant of SVRG proposed by [Hofmann et al., 2015] which does not use κ to determine the inner loop length, but instead updates the snapshot with probability 1/N at each iteration.

The proof of Lemma 1 is deferred to Appendix A.

Theorem 1. *Suppose that Assumption 1 and Assumption 2 hold. And assume that $\eta \le \mu/(2L^2N)$, the distance to optimum $\|w_0^s - w^*\|^2$ converges linearly:*

$$\|w_0^{s+1} - w^*\|^2 \le \alpha \|w_0^s - w^*\|^2,$$

where $\alpha = 1 - \frac{1}{2}\eta\mu N < 1$.

Remark 1. *Our analysis makes no assumption on the order in which we select component functions, and therefore our theorem covers both cyclical selection strategy and without-replacement sampling strategy. Our result extends [Shamir, 2016] which is only applicable to ridge regression. For practical consideration, we focus on the sequential order since such data accessing pattern is memory-friendly and is particularly efficient for extremely large dataset that cannot fit into physical memory.*

Remark 2. *Theorem 1 implies that Algorithm 1 takes $\mathcal{O}\big(\kappa^2 N \log(1/\epsilon)\big)$ component gradient evaluations to get an ϵ-accurate solution under Assumption 1 and 2 in the worst case. By contrast, SVRG has a complexity of only $\mathcal{O}\big(N + \kappa\sqrt{N}\log(1/\epsilon)\big)$ in expectation under the same assumptions [Allen-Zhu and Hazan, 2016]. With an additional assumption that each f_i is convex, SVRG, SAG, and SAGA have a smaller complexity of $\mathcal{O}\big((\kappa+N)\log(1/\epsilon)\big)$ in expectation. On the other hand, the worst-case complexity of IAG is $\mathcal{O}\big(\kappa N^2 \log(1/\epsilon)\big)$ under the same assumptions as ours. Therefore SIG is better than IAG when $N > \kappa$, which coincides with the problem of interest. Otherwise, IAG is better. Our numerical results in Section 7 show that SIG is comparable to stochastic IG methods with linear convergence rates such as SVRG and SAGA and is much better than IAG in terms of convergence rates.*

Proof. First, we show that our algorithm can be interpreted as a perturbed full gradient method. We rewrite the distance to the optimal $\|w_0^{s+1} - w^*\|$ using observation 7

$$\begin{aligned}\|w_0^{s+1} - w^*\| &= \|w_0^s - w^* - \eta \sum_{t=0}^{N-1} \nabla f_{i_t}(w_t^s)\| \\ &= \|w_0^s - w^* - \eta N \nabla F(w_0^s) \\ &\quad - \eta \sum_{t=0}^{N-1} \big(\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s)\big)\|\end{aligned} \tag{8}$$

Then we use triangle inequality twice to upper bound (8) as follows:

$$\begin{aligned}&\|w_0^{s+1} - w^*\| \\ &\le \|w_0^s - w^* - \eta N \nabla F(w_0^s)\| \\ &\quad + \eta \| \sum_{t=0}^{N-1} \big(\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s)\big)\| \\ &\le \|w_0^s - w^* - \eta N \nabla F(w_0^s)\| \\ &\quad + \eta \sum_{t=0}^{N-1} \|\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s)\|. \end{aligned} \tag{9}$$

The first term of (9) corresponds to the distance to optimun of the GD method. The second term can be seen as an additional noise.

Since F is μ-strongly convex and L-smooth, we have the following well-known result [Bubeck, 2014, Lemma 3.5]:

$$\begin{aligned}&\langle \nabla F(x) - \nabla F(y), x - y\rangle \\ &\ge \frac{\mu L}{\mu + L}\|x - y\|^2 + \frac{1}{\mu + L}\|\nabla f(x) - \nabla f(y)\|. \end{aligned} \tag{10}$$

Now we derive an upper bound of $\|w_0^s - w^* - \eta N \nabla F(w_0^s)\|$. Notice that

$$\begin{aligned}&\|w_0^s - w^* - \eta N \nabla F(w_0^s)\|^2 \\ &= \|w_0^s - w^*\|^2 + \eta^2 N^2 \|\nabla F(w_0^s)\|^2 \\ &\quad - 2\eta N (\nabla F(w_0^s) - \nabla F(w^*))^T (w_0^s - w^*) \\ &\le (1 - 2\eta N \frac{\mu L}{\mu + L})\|w_0^s - w^*\|^2 \\ &\quad + \eta N (\eta N - \frac{2}{\mu + L})\|\nabla F(w_0^s) - \nabla F(w^*)\|^2 \end{aligned} \tag{11}$$

where the first inequality holds because of (10).

Since $\eta \le \mu/(2NL^2) \le 1/(2NL) \le 1/(N\mu + NL)$, we have $\eta N - 2/(\mu + L) < 0$. On the other hand, it follows from (3) and Jensen's inequality that $\|\nabla F(w) - \nabla F(w*)\| \ge \mu \|w - w*\|$. Therefore,

$$\begin{aligned}&\|w_0^s - w^* - \eta N \nabla F(w_0^s)\|^2 \\ &\le \big[1 - 2\eta N \frac{\mu L}{\mu + L} + \eta N \mu^2 (\eta N - \frac{2}{\mu + L})\big]\|w_0^s - w^*\|^2 \\ &= \big(1 - 2\eta N \mu + \eta^2 N^2 \mu^2\big)\|w_0^s - w^*\|^2 \\ &= (1 - \eta N \mu)^2 \|w_0^s - w^*\|^2. \end{aligned} \tag{12}$$

By computing the square root of both sides, we obtain an upper bound of the first term of (9):

$$\|w_0^s - w^* - \eta N \nabla F(w_0^s)\| \le (1 - \eta N \mu)\|w_0^s - w^*\| \tag{13}$$

Next, we proceed to upper bound the noise term $\sum_{t=0}^{N-1} \|\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s)\|$ using Lemma 1. Notice that Lemma 1 holds under Theorem 1's conditions

since $\eta \leq \mu/(2L^2N)$ implies that $\eta < \frac{1}{NL}$. Combining (9), (13), and Lemma 1, we have

$$\begin{aligned}&\|w_0^{s+1} - w^*\| \\ &\leq \Big(1 - \eta\mu N + \frac{\eta^2 N^2 L^2}{2(1-\eta NL)}\Big)\|w_0^s - w^*\|. \end{aligned} \tag{14}$$

Since $1 - \eta NL \geq 1 - \frac{\mu}{2L} \geq \frac{1}{2}$, we have

$$\frac{\eta^2 N^2 L^2}{2(1-\eta NL)} \leq \eta^2 N^2 L^2 \leq \frac{1}{2}\eta N\mu. \tag{15}$$

Combining (14), and (15), we have

$$\begin{aligned}\|w_0^{s+1} - w^*\|^2 &\leq \big(1 - \frac{1}{2}\eta\mu N\big)\|w_0^s - w^*\|^2 \\ &= \alpha\|w_0^s - w^*\|^2, \end{aligned} \tag{16}$$

where $\alpha = 1 - \frac{1}{2}\eta\mu N < 1$. This concludes the proof of Theorem 1.

□

6 Accelerated Variant with Momentum

A common approach to accelerate a first order method is to use the momentum technique. [Nesterov, 2013] shows the effectiveness of such technique in accelerating full gradient descent methods. Katyusha [Allen-Zhu, 2016] attempts to extend such technique to the stochastic incremental gradient literature. In the deterministic IG literature, [Gurbuzbalaban et al., 2017] proposes IAG-M, an IAG variant with momentum, by adding an extra momentum term to the IAG update.

We attempt to accelerate SIG with the multiple momentum technique from [Allen-Zhu, 2016] and propose a variant named SIG-M which is summarized in Algorithm 2. Like SIG, the number of iterations in one epoch of SIG-M is the same as the data size. We update w and maintain two auxiliary variables, y and z at each iteration. The iterate w_{t+1}^s is the weighted sum of w_0^s, z_t^s and y_t^s. The two momentum parameters $\tau_1, \tau_2 \in [0, 1]$ are input arguments of the algorithm and $\tau_1 + \tau_2 \leq 1$. In the next section, we compare the performance of SIG-M with SIG by conducting numerical experiments on several standard real datasets.

7 Experiment

In this section, we conduct numerical experiments to demonstrate the performance of SIG and SIG-M. We focus on the ℓ_2-regularized Logistic Regression (LR)

$$\min_{w\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \log\big(1 + \exp(-y_i X_i^T w)\big) + \frac{\lambda}{2}\|w\|_2^2, \tag{17}$$

Algorithm 2: SIG-M Algorithm

input : α, w_0^0, S, τ_1, τ_2
$y_0^0 = z_0^0 \leftarrow w_0^0$;
for $s = 0, 1, \ldots, S-1$ **do**
  compute $G = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(w_0^s)$
  for $t = 0, \ldots, N-1$ **do**
    $i_t = t + 1$
    $w_{t+1}^s = \tau_1 z_t^s + \tau_2 w_0^s + (1 - \tau_1 - \tau_2) y_t^s$
    $z_{t+1}^s = z_t^s - \alpha\big(\nabla f_{i_t}(w_t^s) - \nabla f_{i_t}(w_0^s) + G\big)$
    $y_{t+1}^s = w_{t+1}^s + \tau_1(z_{t+1}^s - z_t^s)$
  end
  $w_0^{s+1} = w_N^s$, $y_0^{s+1} = y_N^s$, $z_0^{s+1} = z_N^s$
end
return: w_0^S

and the ℓ_2-regularized squared hinge loss (SVM)

$$\min_{w\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \big([1 - y_i X_i^T w]_+\big)^2 + \frac{\lambda}{2}\|w\|_2^2, \tag{18}$$

where $X_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ denote the feature vector and label of sample i, respectively.

We used six datasets in our experiments: MNIST[3], rcv1.binary[4], a9a[4], avazu[4], criteo[4], and HIGGS[4]. Detailed information of these datasets and models are listed in Table 2. Note that in our experiments, all data points in each dataset has been normalized, as suggested by [Xiao and Zhang, 2014]. The optimal point w^* is obtained by running the SIG method for a sufficiently long time. And we use the best-tuned step sizes for all algorithms in our experiments, typically from $0.1/L$ to $1/L$. For SIG-M, we set $\tau_2 = 1/2$ and $\tau_1 = a/(s + b)$ during the s^{th} epoch where a and b are hyperparameters, as suggested by [Allen-Zhu, 2016].

7.1 Small Size Dataset

The numerical results of SIG and SIG-M over three small datsets (MNIST, rcv1 and a9a) are shown in Figure 3. We compare both SIG and SIG-M with the baselines including full gradient descent method (GD), some stochastic IG methods including SVRG under with-replacement sampling, Katyusha, SVRG under random reshuffling (SVRG-RR), and SAGA under random reshuffling (SAGA-RR), and deterministic IG methods including DIG and IAG. It can be seen that the performance of SIG is better than GD, DIG, and IAG methods. It's worth noting that while SIG converges with a step size which is close to those

[3] http://yann.lecun.com/exdb/mnist/

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

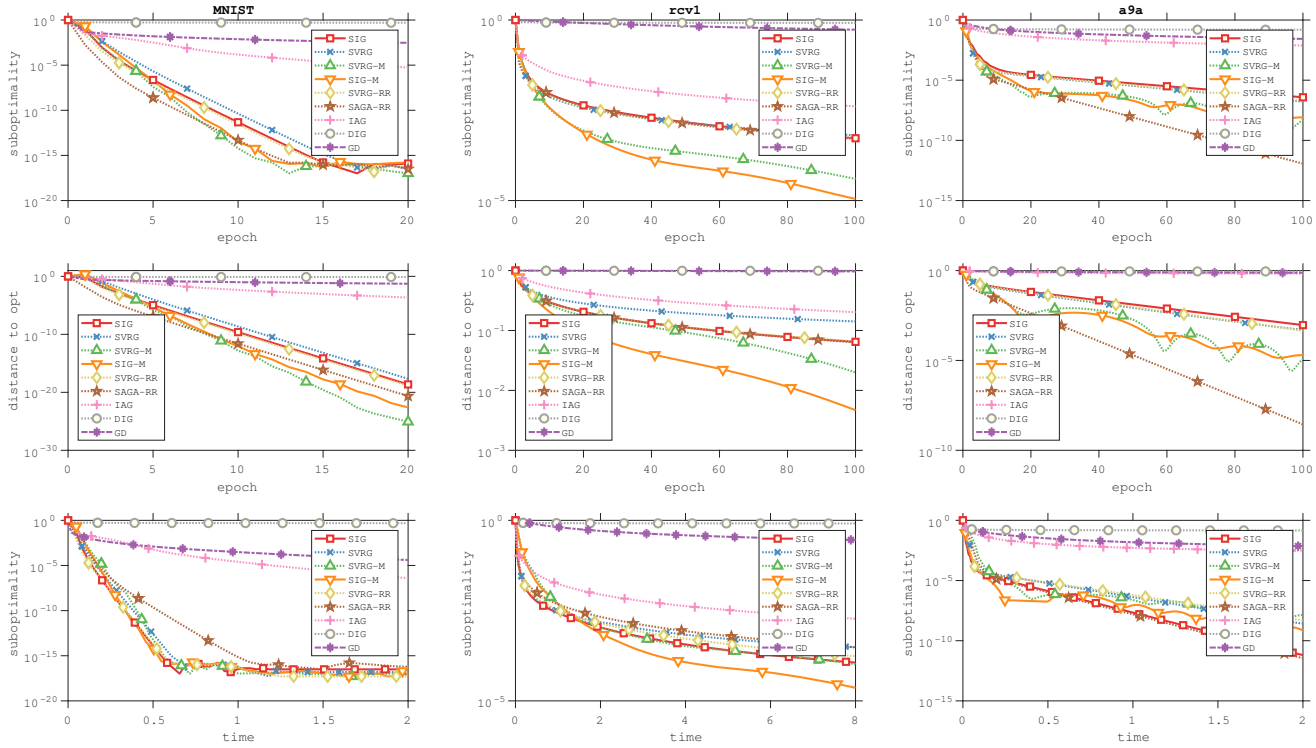

Figure 3: Comparison of SIG, SIG-M and other algorithms on different problems over three datasets: MNIST, rcv1.binary, and a9a. Each column corresponds to results on one dataset respectively. The plots on the first row measure the suboptimality $(F(w_0^s) - F(w^*))/(F(w_0^0) - F(w^*))$ versus number of epochs. The second row measures the relative distance to the optimal $\|w_0^s - w^*\|^2/\|w_0^0 - w^*\|^2$ versus number of epochs. The third row measures the suboptimality versus execution time.

of SVRG and SAGA, IAG can only converge with a much smaller step size, and is thus worse than SIG in practice even when $N < \kappa$. Moreover, the performance of SIG is comparable to and sometimes better than the stochastic VR methods SVRG, SVRG-RR, and SAGA-RR. It is reasonable to conjecture that our bound of convergence rate in theoretical analysis is possibly conservative. And one can see that SIG-M outperforms SIG.

7.2 Large Size Dataset

Figure 4 shows the suboptimality versus time of SIG and SIG-M compared with some IG methods on three large datasets (avazu, criteo, and HIGGS). We restricted the maximal amount of main memory usage to 2GB for avazu and HIGGS and 8GB for criteo to simulate the scenario in which the dataset is too large to fit into main memory instead of using extremely large datasets due to time limit. One can see that SIG and SIG-M outperform DIG, IAG, SVRG, and Katyusha in terms of execution time. Moreover, SIG-M outperforms SIG. We note that for sparse datasets, all incremental gradient methods in our experiments are implemented with the lagged update technique proposed by [Schmidt et al., 2017]. The basic idea of the lagged update technique is to defer the update of some coordinate of the iterate w until the next iteration it is accessed. More details can be found in [Schmidt et al., 2017, section 4].

Figure 5 compares the computation time with data transfer time for SIG and SVRG algorithms with limited main memory usage of different sizes. It is shown that for the SIG method which selects the component functions in a cyclic scheme, the fraction of time spent on data transfer remains low. However, for the SVRG method which selects the component functions in a stochastic scheme, the fraction of time spent on data transfer becomes higher (up to 50%) as the amount of available main memory gets smaller. The main reason is that when the dataset cannot fit into main memory, more page faults occur in the random data access routine than in the sequential data access routine. There-

Table 2: Datasets and regularization parameters.

dataset	data size	dimension	λ	model
MNIST	60,000 (0.09GB)	784	10^{-4}	LR
rcv1.binary	20,242 (0.77GB)	47,236	10^{-8}	LR
a9a	32,561 (0.01GB)	123	10^{-7}	SVM
avazu	14,596,137 (3.37GB)	999,990	10^{-8}	LR
criteo	45,840,617 (26.98GB)	1,000,000	10^{-9}	LR
HIGGS	11,000,000 (4.31GB)	28	10^{-8}	SVM

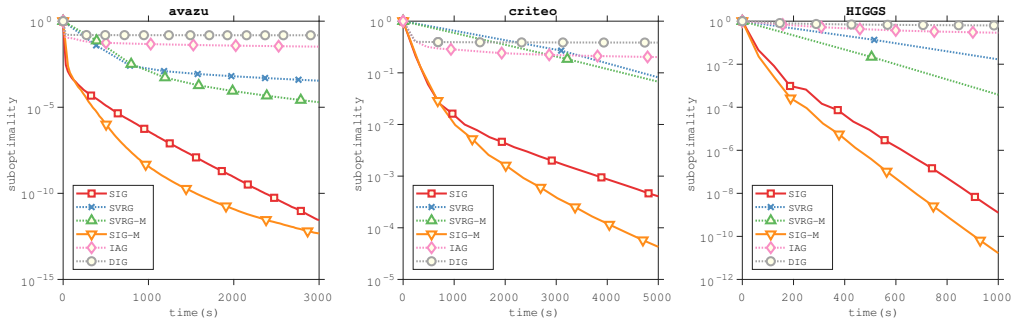

Figure 4: Comparison of SIG, SIG-M and other IG methods on avazu, criteo, and HIGGS dataset. The x-axis stands for execution time. The y-axis stands for the suboptimality.

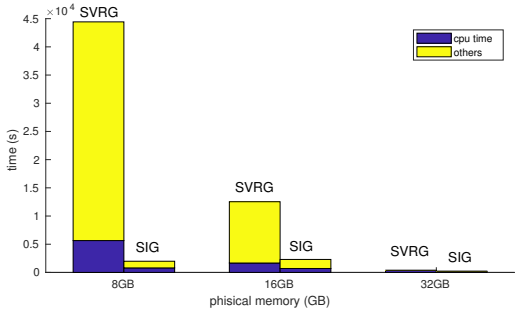

Figure 5: Comparison of the CPU time and data transfer time in inner loops of SIG and SVRG on criteo dataset with limited physical memory usage of different sizes. We ran both algorithms for 10 epochs.

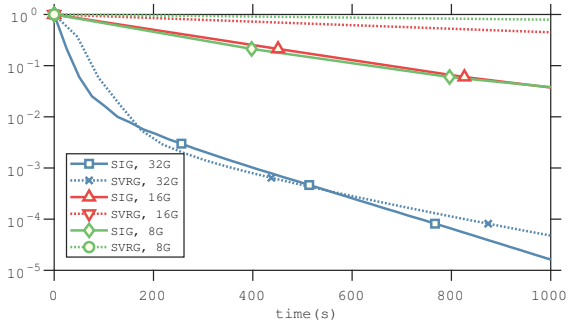

Figure 6: Comparison of SVRG and SIG algorithms on criteo dataset with limited physical memory usage of different sizes.

fore, for extremely large datasets, the SIG method is more efficient than the stochastic IG methods in terms of per iteration cost. Figure 6 shows the suboptimality versus time of SVRG and SIG with limited main memory usage of different sizes. We find that SIG converges faster than SVRG in terms of execution time as the amount of available main memory gets smaller.

8 Discussion

In this paper, we proposed a deterministic incremental gradient method, SIG, for solving large-scale optimization problems of minimizing strongly convex finite sums where the component functions are smooth but potentially non-convex. We proved that the SIG method enjoys a linear convergence rate. We further proposed a method named SIG-M, which aims to use momentums to accelerate the SIG method. Our numerical results show that SIG outperforms existing deterministic IG methods including DIG and IAG and is comparable to stochastic VR methods such as SVRG and SAGA-RR. And the use of momentums accelerates the convergence of SIG. Moreover, the proposed methods are time-efficient on large-scale optimization problems.

We note that SIG can be easily extended to the proximal case for solving a more general class of problems, $\min_{w\in\mathbb{R}^d} \sum_{i=1}^N f_i(w) + r(w)$, where $r(w)$ is convex and possibly non-differentiable. The convergence analysis of the proximal case is left for future work.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No: 61472347, 61672376, 61751209), and Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ18F020002.

References

[Allen-Zhu, 2016] Allen-Zhu, Z. (2016). Katyusha: The first direct acceleration of stochastic gradient methods. *arXiv preprint arXiv:1603.05953*.

[Allen-Zhu, 2017] Allen-Zhu, Z. (2017). Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 89–97, International Convention Centre, Sydney, Australia. PMLR.

[Allen-Zhu and Hazan, 2016] Allen-Zhu, Z. and Hazan, E. (2016). Variance reduction for faster non-convex optimization. In *International Conference on Machine Learning*, pages 699–707.

[Allen-Zhu and Yuan, 2016] Allen-Zhu, Z. and Yuan, Y. (2016). Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pages 1080–1089.

[Bertsekas, 2011] Bertsekas, D. P. (2011). Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, 2010(1-38):3.

[Blatt et al., 2007] Blatt, D., Hero, A. O., and Gauchman, H. (2007). A convergent incremental gradient method with a constant step size. *SIAM Journal on Optimization*, 18(1):29–51.

[Bubeck, 2014] Bubeck, S. (2014). Theory of convex optimization for machine learning. *arXiv preprint arXiv:1405.4980*, 15.

[Carmon et al., 2016] Carmon, Y., Duchi, J. C., Hinder, O., and Sidford, A. (2016). Accelerated methods for non-convex optimization. *arXiv preprint arXiv:1611.00756*.

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

[Gürbüzbalaban et al., 2015] Gürbüzbalaban, M., Ozdaglar, A., and Parrilo, P. (2015). Why random reshuffling beats stochastic gradient descent. *arXiv preprint arXiv:1510.08560*.

[Gurbuzbalaban et al., 2017] Gurbuzbalaban, M., Ozdaglar, A., and Parrilo, P. (2017). On the convergence rate of incremental aggregated gradient algorithms. *SIAM Journal on Optimization*, 27(2):1035–1048.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). Overview of supervised learning. In *The elements of statistical learning*, pages 9–41. Springer.

[Hofmann et al., 2015] Hofmann, T., Lucchi, A., Lacoste-Julien, S., and McWilliams, B. (2015). Variance reduced stochastic gradient descent with neighbors. In *Advances in Neural Information Processing Systems*, pages 2305–2313.

[Johnson and Zhang, 2013] Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323.

[Nesterov, 2013] Nesterov, Y. (2013). *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

[Robbins and Monro, 1951] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

[Schmidt et al., 2017] Schmidt, M., Le Roux, N., and Bach, F. (2017). Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112.

[Shalev-Shwartz, 2016] Shalev-Shwartz, S. (2016). Sdca without duality, regularization, and individual convexity. In *International Conference on Machine Learning*, pages 747–754.

[Shamir, 2016] Shamir, O. (2016). Without-replacement sampling for stochastic gradient methods. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 46–54. Curran Associates, Inc.

[Silberschatz et al., 2014] Silberschatz, A., Galvin, P. B., and Gagne, G. (2014). *Operating system concepts essentials*. John Wiley & Sons, Inc.

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

[Ying et al., 2017] Ying, B., Yuan, K., and Sayed, A. H. (2017). Convergence of variance-reduced stochastic learning under random reshuffling. *arXiv preprint arXiv:1708.01383*.