Direct Acceleration of SAGA using Sampled Negative Momentum

Kaiwen Zhou[1] **Qinghua Ding**[1] **Fanhua Shang**[2]
James Cheng[1] **Danli Li**[1] **Zhi-Quan Luo**[3]

[1]Department of Computer Science and Engineering, The Chinese University of Hong Kong
[2]School of Artificial Intelligence, Xidian University
[3]Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong (Shenzhen)

Abstract

Variance reduction is a simple and effective technique that accelerates convex (or non-convex) stochastic optimization. Among existing variance reduction methods, SVRG and SAGA adopt unbiased gradient estimators and are the most popular variance reduction methods in recent years. Although various accelerated variants of SVRG (e.g., Katyusha and Acc-Prox-SVRG) have been proposed, the direct acceleration of SAGA still remains unknown. In this paper, we propose a directly accelerated variant of SAGA using a novel Sampled Negative Momentum (SSNM), which achieves the best known oracle complexity for strongly convex problems (with known strong convexity parameter). Consequently, our work fills the void of directly accelerated SAGA.

1 Introduction

In this paper, we consider optimizing the following composite finite-sum problem, which arises frequently in machine learning and statistics such as supervised learning and regularized empirical risk minimization (ERM):

$$\min_{x\in\mathbb{R}^d} \Big\{ F(x) \triangleq f(x) + h(x) \Big\}, \tag{1}$$

where $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$ is an average of n smooth and convex function $f_i(x)$, and $h(x)$ is a simple and convex (but possibly non-differentiable) function. Here, we also define $F_i(x) = f_i(x) + h(x)$ with $\nabla F_i(x) = \nabla f_i(x) + \partial h(x)$ and $\partial h(x)$ denotes a sub-gradient of $h(\cdot)$ at x, which will be used in the paper.

We focus on achieving a highly accurate solution for Problem (1), although for practical optimization tasks, such as supervised learning, low empirical risk may result in a high generalization error. In this paper, we treat Problem (1) as a pure optimization problem.

When $F(\cdot)$ in Problem (1) is strongly convex, traditional analysis shows that gradient descent (GD) yields a fast linear convergence rate but with a high per-iteration cost, and thus may not be suitable for problems with a very large n. As an alternative for large-scale problems, SGD [Robbins and Monro, 1951] uses only one or a mini-batch of gradients in each iteration, and thus enjoys a significantly lower per-iteration complexity than GD. However, due to the undiminished variance of the gradient estimator, vanilla SGD is shown to yield only a sub-linear convergence rate. Recently, stochastic variance reduced methods (e.g., SAG [Roux et al., 2012], SVRG [Johnson and Zhang, 2013], SAGA [Defazio et al., 2014], and their proximal variants, such as [Schmidt et al., 2017], [Xiao and Zhang, 2014] and [Konečný et al., 2016]) were proposed to solve Problem (1). All these methods are equipped with various variance reduction techniques, which help them achieve low per-iteration complexities comparable with SGD and at the same time maintain a faster linear convergence rate than GD (including accelerated GD). In terms of oracle complexity[1], these methods all achieve an $\mathcal{O}((n+\kappa)\log(1/\epsilon))$ complexity[2], as compared with $\mathcal{O}(n\sqrt{\kappa}\log(1/\epsilon))$ for accelerated deterministic methods (e.g., Nesterov's accelerated gradient descent [Nesterov, 2004]).

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

[1]Oracle complexity in this paper, denoted by $\mathcal{O}(\cdot)$, is the number of calls to Incremental First-order Oracle (IFO) + Proximal operator Oracle (PO).

[2]We denote $\kappa \triangleq \frac{L}{\mu}$ throughout the paper, which is known as the condition number of an L-smooth and μ-strongly convex function.

Table 1: Comparison of some accelerated variants of SVRG and SAGA. Here, we regard using reductions or proximal point variants as "Indirect" acceleration.

	Indirect	Direct
SVRG (or Prox-SVRG)	APPA & Catalyst	Katyusha & MiG
SAGA		**SSNM**
	Point-SAGA	

Inspired by the acceleration technique proposed in Nesterov's accelerated gradient descent [Nesterov, 2004], accelerated variants of stochastic variance reduced methods have been proposed in recent years, such as Acc-Prox-SVRG [Nitanda, 2014], APCG [Lin et al., 2014], APPA [Frostig et al., 2015], Catalyst [Lin et al., 2015], SPDC [Zhang and Xiao, 2015] and Katyusha [Allen-Zhu, 2017]. Among these algorithms, APPA and Catalyst achieve acceleration by using some carefully designed reduction techniques, which, however, result in additional log factors in their overall oracle complexities. Katyusha, as the first directly accelerated variant of SVRG, introduced the idea of negative momentum (or Katyusha momentum): regarding the gradient estimator of SVRG

$$\widetilde{\nabla} = \nabla f_i(x) - \nabla f_i(\tilde{x}) + \nabla f(\tilde{x}),$$

the negative momentum is a $(\tilde{x} - x)$ offset added (with decay) to each update in this epoch. One can interpreted it as the momentum provided by a previously randomly computed point. Then, by combining it with Nesterov's momentum, Katyusha yields the best known[3] oracle complexity $\mathcal{O}((n + \sqrt{\kappa n})\log(1/\epsilon))$ for strongly convex problems. More recent work [Zhou et al., 2018] shows that adding only negative momentum to SVRG is enough to achieve the best known oracle complexity for strongly convex problems, which results in a simple and scalable algorithm called MiG.

Although a considerable amount of work has been done for accelerating SVRG, another popular stochastic variance reduced method, SAGA, does not have a directly accelerated variant until recently. Accelerating frameworks such as APPA or Catalyst can be used to accelerate SAGA, but the reduction techniques proposed in these works are always difficult to implement and may also result in additional log factors in the overall oracle complexity. A notable variant of SAGA is Point-SAGA [Defazio, 2016]. Point-SAGA requires the proximal operator oracle of each $F_i(\cdot)$ and with the help of that, it can adopt a much larger learning rate than SAGA, which results in the accelerated complexity $\mathcal{O}((n+\sqrt{\kappa n})\log(1/\epsilon))$. Some accelerated variants of SVRG and SAGA are summarized in Table 1. However, the proximal operator of each $F_i(\cdot)$ may not be efficiently computed in practice. Even for logistic regression, we need to run an individual loop (Newton's method) for its proximal operator oracle. Therefore, a directly accelerated variant of SAGA is of real interests.

Following the idea of adding only negative momentum to SVRG [Zhou et al., 2018], we consider adding negative momentum to SAGA. However, unlike SVRG, which keeps a constant snapshot in each inner loop, the "snapshot" of SAGA is a table of points, each corresponding to the position that the component function gradient $\nabla f_i(\cdot)$ was lastly evaluated. Thus, it is non-trivial to directly accelerate SAGA. In this paper, we propose a novel *Sampled Negative Momentum* for SAGA. We further show that adding such a momentum has the same acceleration effect as adding negative momentum to SVRG.

Our contributions are summarized below:

- We propose a directly accelerated variant of SAGA. The acceleration technique is a combination of the negative momentum trick and a novel double sampling scheme, which we called *Sampled Negative Momentum*. We further prove that this accelerated variant achieves the best known oracle complexity for strongly convex problems, which is $\mathcal{O}((n+\sqrt{\kappa n})\log(1/\epsilon))$.
- We discuss some subtle differences on strongly convex assumptions when applying the acceleration technique. Such differences are always neglected in previous directly accelerated methods (e.g., Katyusha and MiG). Our discussion shows that the strongly convex assumption imposed in this paper can be adapted to other strongly convex assumption using a transforming trick.
- We provide a variant of the proposed algorithm for the non-smooth setting and prove that it achieves a lower $\mathcal{O}\left(\frac{\log(1/\epsilon)}{\sqrt{\epsilon}}\right)$ oracle complexity than the $\mathcal{O}(\frac{1}{\epsilon})$ derived in Point-SAGA [Defazio,

[3] According to [Arjevani, 2017], this rate can only be attained when μ is known. Without knowing μ, the best known rate is $\mathcal{O}((n+\kappa)\log(1/\epsilon))$ achieved by [Lei and Jordan, 2017] and [Xu et al., 2017]. We assume μ is known throughout the paper.

Algorithm 1 SAGA with Sampled Negative Momentum (SSNM)

Input: Iterations number K, initial point x_1, learning rate $\eta = \begin{cases} \sqrt{\frac{1}{3\mu n L}} & \text{if } \frac{n}{\kappa} \leq \frac{3}{4}, \\ \frac{1}{2\mu n} & \text{if } \frac{n}{\kappa} > \frac{3}{4}. \end{cases}$, parameter $\tau = \frac{n\eta\mu}{1+\eta\mu}$.

Initialize: "Points" table ϕ with $\phi_1^1 = \phi_2^1 = \ldots = \phi_n^1 = x_1$ and a running average for the gradients of "points" table.

1: **for** $k = 1, 2, \ldots, K$ **do**
2: 1. Sample i_k uniformly in $\{1, \ldots, n\}$ and compute the gradient estimator using the running average.
3: $y_{i_k}^k = \tau x_k + (1-\tau)\phi_{i_k}^k$;
4: $\widetilde{\nabla}_k = \nabla f_{i_k}(y_{i_k}^k) - \nabla f_{i_k}(\phi_{i_k}^k) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\phi_i^k)$;
5: 2. Perform a proximal step.
6: $x_{k+1} = \arg\min_x \left\{ h(x) + \langle \widetilde{\nabla}_k, x \rangle + \frac{1}{2\eta}\|x_k - x\|^2 \right\}$;
7: 3. Sample I_k uniformly in $\{1, \ldots, n\}$, take $\phi_{I_k}^{k+1} = \tau x_{k+1} + (1-\tau)\phi_{I_k}^k$. All other entries in the "points" table remain unchanged. Update the running average corresponding to the change in the "points" table.
8: **end for**

Output: x_{K+1}

2016].

- Since SSNM does not use the hybrid momentum in Katyusha, it has a simpler structure and potentially clearer intuition. We provide some insights by building connections between the negative momentum trick and the standard Nesterov's momentum in [Nesterov, 2004].

2 Preliminaries

In this paper, we consider Problem (1) in standard Euclidean space with the Euclidean norm denoted by $\|\cdot\|$. We use $\mathbb{E}$ to denote that the expectation is taken with respect to all randomness in one epoch. In order to further categorize the objective functions, we define that a convex function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be L-smooth if for all $x, y \in \mathbb{R}^d$, it holds that

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

and μ-strongly convex if for all $x, y \in \mathbb{R}^d$,

$$f(x) \geq f(y) + \langle \mathcal{G}, x - y \rangle + \frac{\mu}{2}\|x - y\|^2, \qquad (3)$$

where $\mathcal{G} \in \partial f(y)$, the set of sub-gradient of $f(\cdot)$ at y for non-differentiable $f(\cdot)$. If $f(\cdot)$ is differentiable, we can simply replace $\mathcal{G} \in \partial f(y)$ with $\mathcal{G} = \nabla f(y)$. Then we make the following assumption to identify the main objective condition (strongly convex) that is the focus of this paper:

Assumption 1 (Strongly Convex)**.** *In Problem* (1)*, each* $f_i(\cdot)$[4] *is* L*-smooth and convex,* $h(\cdot)$ *is* μ*-strongly convex.*

[4]In fact, if each $f_i(\cdot)$ is L-smooth, the averaged function $f(\cdot)$ is itself L-smooth — but probably with a smaller L. We keep using L as the smoothness constant for a consistent analysis.

3 Direct Acceleration of SAGA

Our proposed algorithm SSNM (SAGA with Sampled Negative Momentum) is formally given in Algorithm 1. As we can see, there are some unusual tricks used in Algorithm 1. Thus we elaborate some ideas behind Algorithm 1 by making the following remarks:

- *Coupled point* $y_{i_k}^k$ *correlates to the randomness of* i_k. Unlike the negative momentum used for Katyusha, which comes from a fixed snapshot $\tilde{x}$, the negative momentum of SAGA can only be found on a "points" table that changes over time. Thus, in SSNM, we choose to use the i_kth entry of the "points" table to provide the negative momentum, which makes the coupled point correlate to the randomness of sample i_k. In fact, all the possible coupled points y_i^k form a "coupled table". Although the table is never explicitly computed, we shall see that the concept of "coupled table" is critical in the proof of SSNM. The 3rd step in Algorithm 1 can thus be regarded as sampling a point in such a table.

- *"Biased" gradient estimator* $\tilde{\nabla}_k$. The expectation of the semi-stochastic gradient estimator $\tilde{\nabla}_k$ defined in Algorithm 1 is the average of the gradients computed in the "coupled table", $\mathbb{E}_{i_k}\left[\tilde{\nabla}_k\right] = \frac{1}{n}\sum_{i=1}^n \nabla f_i(y_i^k)$, which seems to be surprising as this expectation (except $\tilde{\nabla}_1$) does not correspond to any gradient of $f(\cdot)$, but can be used to show convergence to the optimal solution of $F(\cdot)$. In some sense, $\tilde{\nabla}_k$ is a "biased" gradient estimator.

- *Independent samples* I_k *and* i_k. The additional sample I_k is crucial for the convergence analysis of Algorithm 1, which chooses an index to store the

Table 2: Comparison of variants of SAGA (All complexities are for strongly convex objectives).

	Complexity	Requirements	Memory
SAGA	$\mathcal{O}((n+\kappa)\log(1/\epsilon))$	IFO of $f(\cdot)$, PO of $h(\cdot)$	$O(nd)$ or $O(n)$ for linear models.
Point-SAGA	$\mathcal{O}((n+\sqrt{\kappa n})\log(1/\epsilon))$	PO of each $F_i(\cdot)$	$O(nd)$ or $O(n)$ for linear models*.
SSNM	$\mathcal{O}((n+\sqrt{\kappa n})\log(1/\epsilon))$	IFO of $f(\cdot)$, PO of $h(\cdot)$	$O(nd)$

* A memory issue of Point-SAGA is discussed in the Supplementary Material E.

updated point in the "points" table. The insight of this choice is that it separates the randomness of x_{k+1} and the update index in the "points" table so as to make certain inequalities valid.

- *Two learning rates for two cases.* Using different parameter settings for different objective conditions (ill-condition and well-condition) is common for accelerated methods [Shalev-Shwartz and Zhang, 2014, Allen-Zhu, 2017, Zhou et al., 2018]. If some parameters such as L, μ are unknown, SSNM is still a practical algorithm with tuning only η and τ, as compared with Katyusha which has 4 parameters that need to be tuned. Note that we have tried to make the parameter settings in SSNM similar to Katyusha and MiG. We believe that it can help conduct some fair experimental comparisons with these methods.

- *Only one variable vector with a simple algorithm structure.* Same as MiG in [Zhou et al., 2018], SSNM only has one variable vector in the main loop. Coupled point $y_{i_k}^k$ can be computed whenever used and does not need to be explicitly stored. Moreover, SSNM has a one loop structure compared to those variants of SVRG. Such a structure is good for asynchronous implementation since algorithms with two loops in this setting always require a synchronization after each inner loop [Mania et al., 2017]. Moreover, the algorithm structure of SSNM is more elegant than Katyusha and MiG, both of which require a tricky weighted averaged scheme at the end of each inner loop[5].

Since the algorithms such as Point-SAGA and SAGA are closely related to SSNM, in the next subsection, we compare in details these different variants of SAGA.

3.1 Comparison with SAGA and Point-SAGA

As summarized in Table 2, in comparison, SSNM yields the same fast $\mathcal{O}((n+\sqrt{\kappa n})\log(1/\epsilon))$ convergence rate as Point-SAGA without requiring additional assumptions, demonstrating the advantage of direct acceleration. Weaker assumptions on the objective function make the algorithm more implementable. However, since SSNM requires storing the "points" table, the memory complexity of SSNM is always $O(nd)$. This is a disadvantage when the objective is a linear model such as linear logistic regression and ridge regression. It is well known that for these linear models, each gradient is just a weighting of the corresponding data vector. Thus, we can simply store a scalar to represent a gradient, which allows SAGA and Point-SAGA to have an $O(n)$ memory complexity for these problems. Note that for logistic regression, the proximal operator oracle required by Point-SAGA does not have a closed form solution. We may need to run several Newton steps for an inexact oracle as in [Defazio, 2016]. In comparison, the gradient oracle required by SSNM and SAGA is much easier to access.

For a general objective, all the three methods have the same memory complexity. In such a case, SSNM is apparently superior to the other two algorithms. Note that the exact proximal operator oracle for a general objective is always hard to be efficiently evaluated.

4 Theory

In this section, we theoretically analyze the performance of SSNM. First, we give a variance bound of the stochastic gradient estimator of SSNM shown in Lemma 1. Since the stochastic gradient estimator of SSNM is computed at a coupled point that contains randomness, the variance bound for SSNM, unlike most of the variance bounds in previous work, is built with respect to the expectation of the "biased" gradient estimator[6]. The proofs of Lemma 1 and Theorem 1 are given in the Supplementary Material.

Lemma 1 (Variance Bound)**.** *Using the same notations as in Algorithm 1, we can bound the variance of*

[5]These two algorithms can adopt an uniformly average scheme, but in this case, both algorithms require certain restarting tricks, which make them less implementable.

[6]Other methods using biased gradient estimators include SARAH [Nguyen et al., 2017], JacSketch [Gower et al., 2018]

the stochastic gradient estimator $\widetilde{\nabla}_k$ *as*

$$\mathbb{E}_{i_k}\left[\left\|\widetilde{\nabla}_k - \frac{1}{n}\sum_{i=1}^n \nabla f_i(y_i^k)\right\|^2\right] \le 2L\left(\frac{1}{n}\sum_{i=1}^n \big(f_i(\phi_i^k) - f(y_i^k) - \langle \nabla f_i(y_i^k), \phi_i^k - y_i^k\rangle\big)\right).$$

Now we can formally present the main theorem of SSNM below. As stated in [Allen-Zhu, 2017], the major task of the negative momentum is to cancel the additional inner product term shown in the variance bound so as to keep a close connection in each iteration. As we shall see shortly, our proposed sampled negative momentum effectively cancels the inner product term, which is where the acceleration comes from.

Theorem 1. *Let* $x^\star$ *be the solution of Problem* (1)*, define the following Lyapunov function* T*, which is the same as the one in SAGA [Defazio et al., 2014]:*

$$T^k \triangleq T(x_k, \phi^k) \triangleq \frac{1}{n\eta\mu}\left(\frac{1}{n}\sum_{i=1}^n F_i(\phi_i^k) - F(x^\star) - \frac{1}{n}\sum_{i=1}^n \langle \nabla F_i(x^\star), \phi_i^k - x^\star\rangle\right) + \frac{1}{2\eta n}\|x_k - x^\star\|^2.$$

If Assumption 1 holds, then by choosing $\tau = \frac{n\eta\mu}{1+\eta\mu}$*, the steps of Algorithm 1 satisfy the following contraction for the Lyapunov function in expectation (conditional on* T^k*):*

$$\mathbb{E}_{i_k, I_k}\big[T^{k+1}\big] \le (1+\eta\mu)^{-1} T^k.$$

Thus, by carefully choosing η*, we have the following inequalities in two cases:*

(I) (For ill-conditioned problems). *If* $\frac{n}{\kappa} \le \frac{3}{4}$*, with* $\eta = \sqrt{\frac{1}{3\mu n L}}$ *it holds that*

$$\mathbb{E}\big[\|x_{K+1} - x^\star\|^2\big] \le \left(1 + \sqrt{\frac{1}{3n\kappa}}\right)^{-K}\left(\frac{2}{\mu}\left(F(x_1) - F(x^\star)\right) + \|x_1 - x^\star\|^2\right).$$

The above inequality implies that in order to reduce the squared norm distance to ϵ*, we have an* $\mathcal{O}(\sqrt{\kappa n}\log(1/\epsilon))$ *oracle complexity as* $\epsilon \to 0$ *in expectation.*

(II) (For well-conditioned problems). *If* $\frac{n}{\kappa} > \frac{3}{4}$*, by choosing* $\eta = \frac{1}{2\mu n}$*, we have*

$$\mathbb{E}\big[\|x_{K+1} - x^\star\|^2\big] \le \left(1 + \frac{1}{2n}\right)^{-K}\left(\frac{2}{\mu}\big(F(x_1) - F(x^\star)\big) + \|x_1 - x^\star\|^2\right).$$

This inequality implies that in this case we have an $\mathcal{O}(n\log(1/\epsilon))$ *oracle complexity as* $\epsilon \to 0$ *in expectation.*

Thus, for strongly convex objectives, SSNM yields a fast $\mathcal{O}((n + \sqrt{\kappa n})\log(1/\epsilon))$ rate, which matches the best known oracle complexity achieved by accelerated SVRG [Frostig et al., 2015, Allen-Zhu, 2017].

4.1 Some subtle differences on strongly convex assumption

Recall that the strongly convex assumption for SAGA is imposed on each $f_i(\cdot)$ (or the average $f(\cdot)$ as an extension) [Defazio et al., 2014]. In comparison, SSNM requires the strong convexity of $h(\cdot)$ (in Assumption 1), which seems to be critical in the proof. Below we show that the strong convexity assumption of each $f_i(\cdot)$ can be efficiently transformed into Assumption 1.

Transforming the strong convexity assumption from holding for all $f_i(\cdot)$ to Assumption 1: Suppose we have an objective in the form (1) with each $f_i(\cdot)$ L-smooth and μ-strongly convex, $h(\cdot)$ convex and proper (the main assumption of SAGA). By defining $f_i'(\cdot) = f_i(\cdot) - \frac{\mu}{2}\|\cdot\|^2$ for each $f_i(\cdot)$ and $h'(\cdot) = h(\cdot) + \frac{\mu}{2}\|\cdot\|^2$, the optimal solution of minimizing $F'(\cdot) = \frac{1}{n}\sum_{i=1}^n f_i'(\cdot) + h'(\cdot)$ is equivalent to that of (1) and it can be verified that each $f_i'(\cdot)$ is $(L-\mu)$-smooth and convex, $h'(\cdot)$ is μ-strongly convex. Moreover, the proximal operator $\mathrm{prox}_{h'}^{\eta}(v) \triangleq \arg\min_x\{h'(x) + \frac{1}{2\eta}\|x - v\|^2\}, \forall v \in \mathbb{R}^d$ can be efficiently computed as

$$\mathrm{prox}_{h'}^{\eta}(v) = \mathrm{prox}_{h}^{\eta/(1+\eta\mu)}\left(\frac{v}{1+\eta\mu}\right).$$

Conversely, Assumption 1 may not be reducible to the strong convexity assumption of each $f_i(\cdot)$ using the above trick, since the modified regularizer $h(\cdot) - \frac{\mu}{2}\|\cdot\|^2$ may not be as "proper" as $h(\cdot)$.

Directly accelerated variants of SVRG (e.g., Katyusha and MiG) also require a strongly convex regularizer to achieve acceleration. This requirement can be weakened by adopting a restarting scheme for MiG (Algorithm 3 with Option II in [Zhou et al., 2018]) [7], which only requires $F(\cdot)$ to be strongly convex and thus keeps the same assumption as in Prox-SVRG [Xiao and Zhang, 2014]. Unfortunately, we found that the similar trick does not work for SSNM. The best we can achieve is to slightly weaken the strong convexity assumption to be imposed on each $F_i(\cdot)$, but it requires an additional upper bound $F(x) - F(x^\star) \le \frac{L_F}{2}\|x - x^\star\|^2$ for all $x \in \mathbb{R}^d$, where L_F is potentially much larger than

[7] Similar restarting trick can be used for Katyusha to weaken the strongly convex assumption.

L ($L_F = L$ when $h(\cdot) \equiv 0$). Moreover, the algorithm structure will be more complicated than Algorithm 1. Thus, we decided not to include the variant here.

The main limitation of the Lyapunov function used to prove the convergence of SSNM (and many SAGA-like algorithms) is that it does not contain an additive error term for $F(\cdot)$, unlike the convergence of SVRG (and its variants). We include a discussion on this difference in the Supplementary Material C.

4.2 Non-smooth extension

Problem (1) with non-smooth but L_1-Lipschitz continuous $f_i(\cdot)$, strongly convex $h(\cdot)$ is also prevalent in machine learning, e.g., L2-SVM. To solve this type of problems, the most direct solution is using subgradient methods (e.g., Pegasos [Shalev-Shwartz et al., 2011] with an $\mathcal{O}(\frac{1}{\epsilon})$ rate). As an accelerated variant of SAGA, Point-SAGA also obtains an $\mathcal{O}(\frac{1}{\epsilon})$ rate for a similar type of objectives [Defazio, 2016]. In comparison, Point-SAGA requires the exact proximal operator of each $f_i(\cdot)$ but does not show improvement on the bound. In this subsection, we consider extending SSNM into this setting by utilizing the proximal information of each $f_i(\cdot)$, which results in a convergence rate faster than $\mathcal{O}(\frac{1}{\epsilon})$.

Following [Orabona et al., 2012], we apply *Moreau-Yosida regularization* for each $f_i(\cdot)$, which results in a smooth approximation $f_i^\beta(\cdot)$ (with $\beta > 0$) defined as

$$\forall v \in \mathbb{R}^d, f_i^\beta(v) = \inf_{x \in \mathbb{R}^d} \left\{ f_i(x) + \frac{1}{2\beta}\|x - v\|^2 \right\}.$$

Then, it is clear that $\text{prox}_{f_i}^\beta(v)$ returns the point that attains the infimum in $f_i^\beta(v)$. As proven in Proposition 12.29 [Bauschke et al., 2011], $f_i^\beta(\cdot)$ is $\frac{1}{\beta}$-smooth and its gradient can be computed as $\nabla f_i^\beta(x) = \frac{1}{\beta}(x - \text{prox}_{f_i}^\beta(x)), \forall x \in \mathbb{R}^d$. Moreover, we have the following properties to further bound the error in this smooth approximation:

Lemma 2 (Lemma 2.2, [Orabona et al., 2012])**.** *Let $f_i(\cdot)$ be an L_1-Lipschitz continuous and convex function, then for any $x \in \mathbb{R}^d$, $\beta > 0$*

$$f_i^\beta(x) \le f_i(x) \le f_i^\beta(x) + \frac{\beta L_1^2}{2}.$$

Thus, by defining a "smoothed" objective $F^\beta(\cdot) = \frac{1}{n}\sum_{i=1}^n f_i^\beta(\cdot) + h(\cdot)$, we can use SSNM to minimize $F^\beta(\cdot)$, which leads to the following corollary:

Corollary 1. *Using Algorithm 1 to minimize $F^\beta(\cdot)$ defined above, and by choosing $\beta = \frac{\mu\epsilon}{4L_1^2}$, where $\epsilon > 0$ (small enough) is the required accuracy, in order to achieve $\|x_{K+1} - x^\star\|^2 \le \epsilon$ at the output point x_{K+1}, where $x^\star$ is the solution of minimizing the original $F(\cdot)$, we need an $\mathcal{O}\left(\left(n + \frac{\sqrt{n}L_1}{\sqrt{\epsilon}\mu}\right)\log(1/\epsilon)\right)$ oracle complexity in expectation.*

Proof. Denote the optimal solution of minimizing $F^\beta(\cdot)$ as $x_\beta^\star$. With the strong convexity of $F(\cdot)$, we can bound the difference between $x_\beta^\star$ and $x^\star$ as

$$\|x_\beta^\star - x^\star\|^2 \le \frac{2}{\mu}\big(F(x_\beta^\star) - F(x^\star)\big).$$

Based on Lemma 2, we have the following inequalities:

$$F(x_\beta^\star) \le F^\beta(x_\beta^\star) + \frac{\mu\epsilon}{8} \overset{(\star)}{\le} F^\beta(x^\star) + \frac{\mu\epsilon}{8} \le F(x^\star) + \frac{\mu\epsilon}{8},$$

where $(\star)$ holds due to the optimality of $x_\beta^\star$.

Thus, we conclude that $\|x_\beta^\star - x^\star\|^2 \le \frac{\epsilon}{4}$, which is based on the choice of β.

Following Theorem 1, in order to reduce the squared norm distance $\|x_{K+1} - x_\beta^\star\|^2$ at the output point x_{K+1} to $\frac{\epsilon}{4}$, we need $\mathcal{O}\left(\left(n + \sqrt{\frac{n}{\beta\mu}}\right)\log(1/\epsilon)\right)$ oracle calls. Note that the above results imply that x_{K+1} satisfies

$$\|x_{K+1} - x^\star\|^2 \le 2\|x_{K+1} - x_\beta^\star\|^2 + 2\|x_\beta^\star - x^\star\|^2 \le \epsilon.$$

□

The above results imply an $\mathcal{O}\left(\frac{\log(1/\epsilon)}{\sqrt{\epsilon}}\right)$ bound to solve the non-smooth objectives, which is superior to the $\mathcal{O}(\frac{1}{\epsilon})$ obtained by Point-SAGA. In order to avoid the log factor in the bound, we can use the AdaptSmooth in [Allen-Zhu and Hazan, 2016]. However, as mentioned in Section 4.1, in order to satisfy the HOOD property in [Allen-Zhu and Hazan, 2016], we need an additional upper bound $F(x) - F(x^\star) \le \frac{L_F}{2}\|x - x^\star\|^2$ for all $x \in \mathbb{R}^d$, which rules out certain choices of $h(\cdot)$, such as the indicator function of a closed convex set. Moreover, a $\log(L_F/\mu)$ factor will appear in the oracle complexity bound after using the AdaptSmooth. Thus, we omit further discussions about eliminating the log factor here.

5 Some insights about the negative momentum trick

In [Allen-Zhu, 2017], the negative momentum (or Katyusha momentum) is described as a "magnet" that reduces the error of the semi-stochastic gradient estimator for variance reduced algorithms. Thus, the author combined this idea with Nesterov's momentum (or "positive" momentum) to achieve acceleration. However, as shown in [Zhou et al., 2018] as well as

this work, it seems that merely using the negative momentum trick is enough to obtain the same accelerated convergence rate, which makes this acceleration somewhat "counter-intuitive". In theory, it is clear that with the help of negative momentum, we can adopt a much tighter variance bound. However, this theoretical effect does not explain the source of acceleration. In this section, we try to build a connection between the negative momentum and the standard Nesterov's momentum in [Nesterov, 2004].

For simplicity, we mainly focus on the objective (1) with $h(\cdot) \equiv 0$ in this section. First, consider the deterministic case with $n = 1$, Algorithm 1 degenerates into an algorithm with the following key steps (with $z \in \mathbb{R}^d$ denoting the one item "points" table ϕ):

$$\begin{aligned} y_k &= \tau x_k + (1-\tau) z_k; \\ x_{k+1} &= x_k - \eta \nabla f(y_k); \\ z_{k+1} &= \tau x_{k+1} + (1-\tau) z_k. \end{aligned}$$

Note that we can completely eliminate the sequence $\{x_k\}$, which results in a simple scheme below.

$$\begin{aligned} z_{k+1} &= y_k - \eta\tau \nabla f(y_k); \\ y_{k+1} &= z_{k+1} + (1-\tau)(z_{k+1} - z_k). \end{aligned}$$

By carefully choosing parameters η and τ, we recover the original Nesterov's accelerated gradient method with constant stepsize [Nesterov, 2004]. This observation motivates us to formulate the key steps in SSNM (Algorithm 1) and MiG[8] into the following schemes (outer loops are omitted for simplicity):

SSNM

$$\widetilde{\nabla}_k^{(1)} = \nabla f_{i_k}(y_{i_k}^k) - \nabla f_{i_k}(\phi_{i_k}^k) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\phi_i^k);$$

$$\phi_{I_k}^{k+1} = y_{I_k}^k - \eta\tau \widetilde{\nabla}_k^{(1)};$$

$$y_{i_{k+1}}^{k+1} = \phi_{I_k}^{k+1} + \underline{(1-\tau)(\phi_{i_{k+1}}^{k+1} - \phi_{I_k}^k)}.$$

MiG

for $k = 1 \dots m$:

$$\widetilde{\nabla}_k^{(2)} = \nabla f_{i_k}(y_k^s) - \nabla f_{i_k}(\tilde{x}_s) + \nabla f(\tilde{x}_s);$$

$$y_{k+1}^s = y_k^s - \eta\tau \widetilde{\nabla}_k^{(2)};$$

$$\tilde{x}_{s+1} = \frac{1}{m}\sum_{k=1}^m y_{k+1}^s;$$

$$y_1^{s+1} = y_{m+1}^s + \underline{(1-\tau)(\tilde{x}_{s+1} - \tilde{x}_s)}.$$

[8]We adopt the uniform averaged scheme of MiG (Algorithm 3 with Option II in [Zhou et al., 2018]) for simplicity.

The underlined parts of both algorithms can be regarded as the source of acceleration, since setting $\tau = 1$ makes both algorithms degenerate into SAGA or Prox-SVRG[9]. A more careful analysis shows that: For MiG, the momentum $\tilde{x}_{s+1} - \tilde{x}_s$ is provided every m stochastic steps, where $m = \Theta(n)$ as suggested by the analysis in [Zhou et al., 2018]; for SSNM, although a little bit messy in randomness, we can observe that in expectation, every n steps, the momentum is provided by the newly computed iterate. In comparison, the momentum in Acc-Prox-SVRG [Nitanda, 2014] is added in every stochastic step. However, as analyzed in [Nitanda, 2014], in pure stochastic setting (mini-batch size is 1)[10], no acceleration can be guaranteed for Acc-Prox-SVRG in theory. The intuition here is that we may not trust the momentum provided in every stochastic step; instead, we trust the momentum provided by the average information of n stochastic steps.

Based on the above observation, we may understand the negative momentum in SSNM and MiG as the Nesterov's momentum based on average information, in addition to attaining tighter variance bounds.

6 Experiments

In this section, we conducted experiments on training an ℓ_2-logistic regression model to examine the practical performance of SSNM as well as to justify our theoretical results. Detailed experimental setup and parameter settings are given in the Supplementary Material D.

The experiments were designed as some ill-conditioned problems (with very small λ), since this is the regime in which all the accelerated first-order methods take effect. We tested SAGA, SSNM, Katyusha and MiG with their theoretical parameter settings (Point-SAGA is excluded since it requires a different oracle).

We report the results in Figure 1. From the results, we can make the following observations to justify the accelerated convergence rate:

- *Similar convergence results comparing with other accelerated algorithms.* In fact, we are surprised by the excellent performance of SSNM on the covtype dataset. For this dataset, SSNM is even significantly faster than Katyusha and MiG in

[9]In fact, setting $\tau = 1$ does not make SSNM and MiG exactly the same as SAGA and Prox-SVRG. For SSNM, the update index for the "points" table is different; for MiG, the initial point y_1^{s+1} for the new epoch is different.

[10]Pure stochastic setting is important since it is proven that in order to achieve the optimal convergence rate per data access, we should always choose a mini-batch size of 1 for a family of variance reduction methods [Liu and Hsieh, 2018].

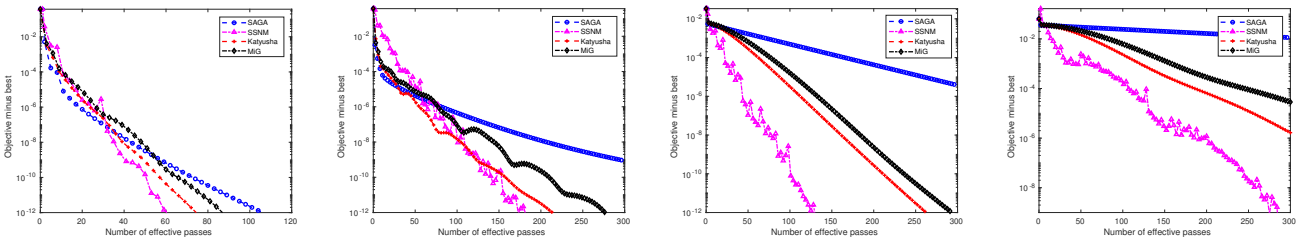

Figure 1: Evaluations of SAGA, SSNM, Katyusha and MiG on the a9a dataset with $\lambda = 10^{-6}$ and 10^{-7} (the first two figures) and the covtype dataset with $\lambda = 10^{-8}$ and 10^{-9} (the last two figures).

terms of the number of epochs (though in theory, Katyusha and MiG yield the same convergence rate as SSNM). The fast convergence of SSNM in practice may imply that the algorithm could potentially benefit many applications.

- *Around 3 times slow-down when κ is 10 times larger.* It can be observed that using the same dataset, when we divide λ by 10 (the same as multiply κ by 10), approximately $\sqrt{10}$ times slow-down ($\sqrt{10}$ times more oracle calls required to achieve the same accuracy) is recorded for all the accelerated methods. In comparison, SAGA shows significant slow-down when κ is increased in both experiments. This observation justifies the $\sqrt{\kappa}$ dependency for accelerated methods.

Another observation is that accelerated methods seem to perform worse in the experiments on the a9a dataset at first several passes. We conjecture that this is because the objective is locally well-conditioned around the initial point. For well-conditioned problem, accelerated methods do not yield a faster rate in theory. In practice, we always found that a smaller amount of momentum yields a better performance. Non-accelerated methods (SVRG, SAGA) always perform better in this case, since they are the accelerated methods without momentum. In the parameter schemes of SSNM, MiG, and Katyusha, the amounts of negative momentum are all set to be $\geq 1/2$ for simplicity in the proofs. To achieve more consistent performance, we can use parameter schemes with a smaller amount of momentum.

However, as also reported in Figure 1, the convergence of SSNM, though very fast, is somewhat unstable compared with the other three methods. This can be explained by the double sampling trick used in SSNM, which greatly increases the uncertainty inside each iteration.

An empirical comparison with Point-SAGA for ridge regression is also given in the Supplementary Material E for reference.

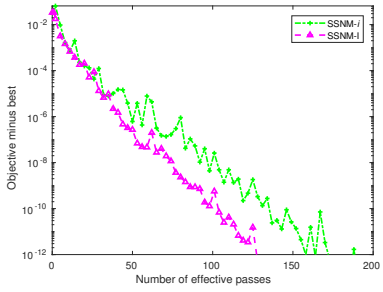

Figure 2: Comparison of using sample i_k (SSNM-i) or I_k (SSNM-I) in 7th step of SSNM on the covtype dataset with $\lambda = 10^{-8}$.

6.1 Effectiveness of sample I_k

A natural question is that: can we use sample i_k (the sample of stochastic gradient) instead of an independent sample I_k in the 7th step of Algorithm 1? We empirically evaluated the effect of sample I_k as shown in Figure 2. As we can see, using sample i_k makes the algorithm even more unstable and slower in convergence comparing with using an independent sample I_k. This effect can probably be explained by some kind of variance cumulation when using the sample i_k.

7 Conclusions

In this paper, we proposed SSNM, an accelerated variant of SAGA, which uses the *Sampled Negative Momentum* trick. Our theoretical results show that SSNM achieves the best known oracle complexity for strongly convex problems and our experiments justified such improvements for the ill-conditioned problems. Although memory consumption of SSNM is higher than SAGA and some other variants, considering its good performance and general objective assumption, SSNM is still potentially beneficial in practice.

Acknowledgements

We thank the reviewers for their valuable comments. This work was supported in part by ITF 6904945 from the Innovation and Technology Commission HKSAR, and Grants (CUHK 14208318 & 14222816) from the Hong Kong RGC. Fanhua Shang was supported by the National Natural Science Foundation of China (Nos. 61876220 and 61876221), and the Science Foundation of Xidian University (No. 10251180018).

References

Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *STOC*, pages 1200–1205, 2017.

Z. Allen-Zhu and E. Hazan. Optimal black-box reductions between optimization objectives. In *NIPS*, pages 1606–1614, 2016.

Y. Arjevani. Limitations on variance-reduction and acceleration schemes for finite sums optimization. In *NIPS*, pages 3540–3549, 2017.

H. H. Bauschke, P. L. Combettes, et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

A. Defazio. A simple practical accelerated method for finite sums. In *NIPS*, pages 676–684, 2016.

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

R. Frostig, R. Ge, S. Kakade, and A. Sidford. Unregularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *ICML*, pages 2540–2548, 2015.

R. M. Gower, P. Richtárik, and F. Bach. Stochastic quasi-gradient methods: Variance reduction via jacobian sketching. *arXiv:1805.02632*, 2018.

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

J. Konečný, J. Liu, P. Richtárik, , and M. Takáč. Mini-batch semi-stochastic gradient descent in the proximal setting. *IEEE J. Sel. Top. Sign. Proces.*, 10(2): 242–255, 2016.

L. Lei and M. Jordan. Less than a single pass: Stochastically controlled stochastic gradient. In *AISTATS*, pages 148–156, 2017.

H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In *NIPS*, pages 3366–3374, 2015.

Q. Lin, Z. Lu, and L. Xiao. An accelerated proximal coordinate gradient method. In *NIPS*, pages 3059–3067, 2014.

X. Liu and C.-J. Hsieh. Fast variance reduction method with stochastic batch size. In *ICML*, pages 3179–3188, 2018.

H. Mania, X. Pan, D. Papailiopoulos, B. Recht, K. Ramchandran, and M. I. Jordan. Perturbed iterate analysis for asynchronous stochastic optimization. *SIAM J. Optim.*, 27(4):2202–2229, 2017.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publ., Boston, 2004.

L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *ICML*, pages 2613–2621, 2017.

A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *NIPS*, pages 1574–1582, 2014.

F. Orabona, A. Argyriou, and N. Srebro. Prisma: Proximal iterative smoothing algorithm. *arXiv preprint arXiv:1206.2372*, 2012.

H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 1951.

N. L. Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *NIPS*, pages 2672–2680, 2012.

M. Schmidt, N. L. Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162:83–112, 2017.

S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *ICML*, pages 64–72, 2014.

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.

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

Y. Xu, Q. Lin, and T. Yang. Adaptive svrg methods under error bound conditions with unknown growth parameter. In *NIPS*, pages 3277–3287, 2017.

Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In *ICML*, pages 353–361, 2015.

K. Zhou, F. Shang, and J. Cheng. A simple stochastic variance reduced algorithm with fast convergence rates. In *ICML*, pages 5980–5989, 2018.