# SAGA: A fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives

El Mahdi CHAYTI

April 2019

# Contents

# 1 Introduction

In the article, a new (at the time) stochastic algorithm namely SAGA is introduced. This algorithm can be seen as an extension of the SAG (stochastic average gradient) and SVRG (stochastic variance reduced gradients). SAGA is shown to have better theoretical results than the before mentioned algorithms, it supports non-differentiable objectives through the use of the proximal operator. SAGA takes advantage of any inherent strong convexity and applies directly to non-strongly convex cases.

## Problemn setting

We consider the following general form optimization problem :

$$\min_{x \in \mathbf{R}^d} F(x) := f(x) + h(x) \tag{1}$$

where $f(x) := \dfrac{\sum_{i=1}^{n} f_i(x)}{n}$
$f(x)$ is the differentiable part of the objective function and $h$ is the part of the objective function that is not differentiable. We suppose for the sake of applicability of proximal methods that it is simple to compute the proximal operator related to $h$.

We note that we are in a convex setting so both f and h are convex functions, furthermore the functions $f_i$ are supposed to be L-smooth (their gradients are L Lipschitz continuous). The special case where the $f_i$ are $\mu$-strongly convex will be also discussed.

# 2 Variance reduction :

SVRG, SAG and SAGA are a family of algorithms that were build with the goal of reducing the variance of the classical stochastic gradient algorithm. The idea behind these algorithms is to estimate the gradient at a given step in a non-trivial way (we consider trivial to be gradients of one of the functions $f_i$ as it is done in SGD) so as to reduce the variance of the estimate.
Before going any further, one might ask why what is the problem with SGD? the problem is that due to variance SGD can't attain a linear convergence rate with constant step size even in the very special case of strong convexity(GD does have a linear rate in this case), this is what we can deduce from the following inequality about SGD (we will not prove it, it is classical) :

$$E[\|x_k - x_*\|^2] \leq (1 - s\mu)^t \|x_0 - x_*\|^2 + \frac{s}{\mu} B^2$$

$B$ is an expected uniform bound of the stochastic gradients, and $s$ is the constant step size.
We deduce from this equation that for convergence to be, $\alpha$ must be close to 0 in order for the variance term to vanish, and close to $\dfrac{1}{\mu}$ to cancel the linear term (linear in logarithmic scale). In the hope of surpassing this problem, variance reduction was considered.

In general, when we are given a value $V = E[X]$ ($X$ is a random variable) that we wish to estimate it by some random variable that has a variance lower than the variance of $X$, we can do the following :
look for a random variable $Z$ such that $cov(X, Z) \geq 0$ then estimate $V = E[X]$ by : $X_Z = \alpha(X - Z) + E[Z]$ with $\alpha$ being a parameter $\in [0, 1]$ that controls the bias, $\alpha = 1$ means $X_Z$ is unbiased.
In fact

$$var(X_Z) = \alpha^2(var(X) + var(Z) - 2cov(X, Z))$$

which means that it suffices that $var(Z) \leq 2cov(X, Z)$ for $X_Z$ to have less variance than $X$. In other words, the highly correlated $Z$ with $X$ the lower the variance.

This method is the one applied in SVRG and SAGA, however SAG deviates a little bit from this paradigm as its gradient estimates are **biased** in contrast to two algorithms mentioned before.

For SVRG : $\alpha = 1$ and at step t, the random variable $X = \nabla f_i(x^t)$ and $Z = \nabla f_i(\tilde{x})$ with i being a uniform index taking from the set $1, \cdots, n$ and $\tilde{x}$ is a reference point.
In practice the reference point is fixed for m steps and then changed to either the last value of $x^t$ or the average of the last m steps.
As for the SAGA algorithm, $\alpha$ and $X$ are the same while $Z$ now is equal to $Z = \nabla f_i(x_i^t)$ with $\{x_i^t, i = 1 \cdots n\}$ are n points that makes it possible to keep track of the n last values of $X$ (the gradients of $f_i$), well, not exactly the last n values of X but the last n gradients, which is the intuition behind these algorithms as taking more past information when going from one step to the following will mean having less variance.

SAG is exactly the same as SAGA except for $\alpha$ that takes the value $\frac{1}{n}$ instead (i.e SAG uses biased estimates of the gradient at each step while SAGA uses unbiased estimates)

Here is the SAGA algorithm in detail :
**Algorithm 1.** *SAGA*
*Set* $x_0 = 0$ *and* $x_0^i = x_0$ *for all* $i \in \{1, \cdots, n\}$, *choose a step size* $s > 0$
*Set up a table to keep the gradients :* $[\nabla f_1(x_0^1) \cdots \nabla f_n(x_0^n)]$
*For* $t = 0 \cdots T$ *do :*
*1 - sample* $i \in \{1, \cdots, n\}$
*2-* $g_t = \nabla f_i(x_t) - \nabla f_i(x_t^i) + \frac{1}{n} \sum_{j=1}^n \nabla \nabla f_j(x_t^j)$
*3-* $x_{t+1} = prox_\gamma^h(x_t - s g_t)$
*4- Update only the i-th grad::* $\nabla f_i(x_{t+1}^i) = \nabla f_i(x_t)$
*Output* $x_T$

And here is SAG :
**Algorithm 2.** *SAG*
*Set* $x_0 = 0$ *and* $x_0^i = x_0$ *for all* $i \in \{1, \cdots, n\}$, *choose a step size* $s > 0$
*Set up a table to keep the gradients :* $[\nabla f_1(x_0^1) \cdots \nabla f_n(x_0^n)]$
*For* $t = 0 \cdots T$ *do :*
*1 - sample* $i \in \{1, \cdots, n\}$
*4- Update only the i-th grad:* $\nabla f_i(x_{t+1}^i) = \nabla f_i(x_t)$
*2-* $g_t = \frac{1}{n} \sum_{j=1}^n \nabla \nabla f_j(x_t^j)$
*3-* $x_{t+1} = prox_\gamma^h(x_t - s g_t)$
*Output* $x_T$

# 3 Theory

In the article non-strongly and strongly convex cases were discussed, convergence rates for both cases were given.
In the strongly case and with a step size $s = \frac{1}{2(\mu n + L)}$ we have the following inequality :

$$E[\|x_k - x_*\|^2] \leq (1 - \frac{\mu}{2(\mu n + L)})^k [\|x_0 - x_*\|^2 + \frac{\mu}{\mu n + L}(f(x_0) - \langle f^{'}(x_*), x_0 - x_* \rangle - f(x_*))]$$

It is indeed a linear convergence rate with a constant step size.
In the non-strongly convex case, it has been established that with a step size $s = \frac{1}{3L}$ we have the following convergence guarantee :

$$E[|F(\bar{x}_k := \frac{1}{k} \sum_{t=1}^k x_t) - F(x_*)|] \leq \frac{4n}{k} [\frac{2L}{n} \|x_0 - x_*\|^2 + f(x_0) - \langle f^{'}(x_*), x_0 - x_* \rangle - f(x_*)]$$

It has been also shown that with this same step size the algorithms adapts automatically to strong convexity and gives pretty much similar results. If functions $f_i$ happen to be $\mu$-strongly convex, then :

$$E[\|x_k - x_*\|^2] \leq (1 - \min(\frac{1}{4n}, \frac{\mu}{3L}))^k [\|x_0 - x_*\|^2 + \frac{2n}{3L}(f(x_0) - \langle f^{'}(x_*), x_0 - x_* \rangle - f(x_*))]$$

We have almost the same linear convergence rate without having to know the strong convexity constant.

# 4 Ideas for the proofs

The main idea is to consider the following lyapounov function :

$$T^k(x_k, \{x_k^i\}_{i=1}^n) = \frac{1}{n}\sum_{j=1}^{n}(f_j(x_k^j) - f(x_*) - \langle f_j^{'}(x_k^j), x_k^j - x_*\rangle) + c\|x^k - x_*\|^2$$

With $c = \frac{1}{2\gamma(1-\gamma\mu)n}$, $\gamma = \frac{1}{2(\mu n + L)}$.
Then we prove that for $\kappa = \frac{1}{\gamma\mu}$, $T$ is such :

$$E[T^{k+1}] \leq (1 - \frac{1}{\kappa})T^k$$

This gives : $E[T^k] \leq (1 - \frac{1}{\kappa})^k T^0$
Using the convexity of the functions $f_j$ it is easy to see that $c\|x^k - x_*\|^2 \leq T^k$ which yields the result for strong convexity.
The non-strongly convex case is treated almost the same. The same Lyaponouv function as before is considered, and a lemma that uses the definition of L-smoothness of the functions $f_i$ is used.

# 5 Applications

SGD, SAG and SAGA were applied to the data given for homeworks in the case of ridge regression (square loss) with a regularization parameter equal to $1e^{-5}$ :

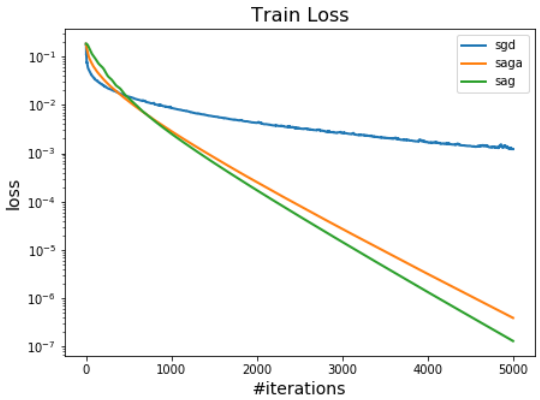


Figure 1: training loss

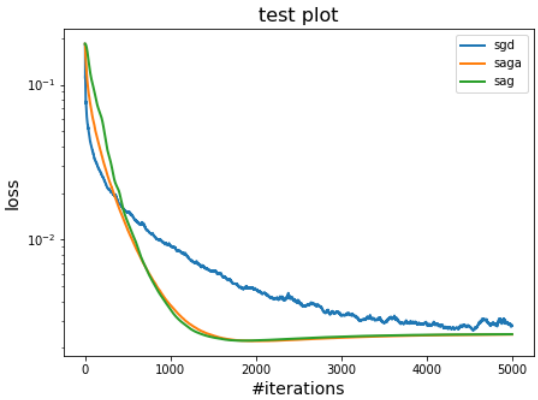


Figure 2: test loss

The linear convergence rate can be seen clearly from the training loss plot.
For details about the implementation please see the notebook or the HTML file accompanying this report.