# Variance Reduction of Stochastic Gradients Without Full Gradient Evaluation

Florian Jarre, Heinrich Heine Univ., Düsseldorf, Germany
Felix Lieder, Heinrich Heine Univ., Düsseldorf, Germany

March 9, 2020

**Abstract**

A standard concept for reducing the variance of stochastic gradient approximations is based on full gradient evaluations every now and then. In this paper an approach is considered that – while approximating a local minimizer of a sum of functions – also generates approximations of the gradient and the function values without relying on full gradient evaluations.

## 1. Introduction, stochastic gradient descent

Estimation problems arising in internet applications and machine learning often have the form of minimizing a function $f : \mathbb{R}^n \to \mathbb{R}$ with

$$f(x) := \frac{1}{m} \sum_{i=1}^{m} f_i(x).$$

Here, the functions $f_i : \mathbb{R}^n \to \mathbb{R}$ are convex and differentiable for $1 \le i \le m$ and the number $m$ may be extremely large. To distinguish from $f$, the functions $f_i$ are also referred to as *elementary functions*. When $m$ is huge, it may be useful to approximate the function $f$ by selecting a non-empty random set $S \subset \{1, \ldots, m\}$ called "batch" or "mini-batch" in the sequel and defining

$$f_S(x) := \frac{1}{|S|} \sum_{i \in S} f_i(x). \tag{1}$$

Often, the function $f$ itself is the result of some random process, but this form of randomness is ignored in the following and only the random choice of $S$ is investigated. The simplest variant of stochastic gradient descent algorithm then generates an initial point $x^0 \in \mathbb{R}^n$ and defines

$$x^{k+1} := x^k - \alpha \nabla f_S(x^k) \qquad \text{for } k \ge 0. \tag{2}$$

Here, $S = S_k$ is obtained from an independent uniform distribution at each step $k$ while the choice of the step length $\alpha = \alpha_k > 0$ is not necessarily changed.

When $S$ is sampled uniformly from $\{1, \ldots, m\}$, the expected value of the gradient approximation defined by the batch satisfies

$$\mathbb{E}(\nabla f_S(x)) = \nabla f(x).$$

Hence, at an exact minimizer $x^*$ of $f$ the relation $\mathbb{E}(\nabla f_S(x^*)) = 0$ holds true. Nevertheless, for any particular $S \neq \{1, \ldots, m\}$ generally $\nabla f_S(x^*) \neq 0$, and hence, the iterates will deviate again from an exact minimizer $x^*$ should an iterate $x^k$ coincide with $x^*$ by coincidence. A convergence with a fixed step size $\alpha$ therefore is not possible. Several remedies have been proposed such as reducing the step lengths [5, 3], forming averages [4],

$$\bar{x}^0 := x^0, \qquad \bar{x}^k := \tfrac{k}{k+1}\, \bar{x}^{k-1} + \tfrac{1}{k+1}\, x^k \quad \text{for } k \geq 1,$$

or full gradient evaluations every now and then [2]. In this paper, the approach in [2] is modified as to avoid any full gradient evaluations.

The algorithm considered in this paper generates a sequence of iterates along with two estimates for the gradient of $f$ at the current iterate, the standard stochastic gradient estimate (1) based on a mini-batch and an averaged stochastic gradient approximation. The averaged gradient approximation at the previous iterate is corrected based on a stochastic approximation of the difference of the gradient at the previous and the current iterate and then combined with the stochastic gradient (1) at the current iterate. The weights of this combination are determined as to minimize the estimated variance of the averaged gradient, and batch size and step lengths are adaptively determined as to guarantee the best anticipated progress divided by the associated computational effort. Preliminary numerical examples illustrate the quality of the averaged stochastic gradient approximation.

## 2. An averaged stochastic gradient approach

We consider an iterative procedure that generates points $x^k \in \mathbb{R}^n$ for $k = 0, 1, 2, \ldots$ and assume that all functions $f_i$ for $1 \leq i \leq m$ are twice continuously differentiable. At each iteration $k$ a random set $S_k \subset \{1, \ldots, m\}$ of cardinality $c_k := |S_k| \geq \bar{c}$ for some parameter $\bar{c} \geq 2$ is taken from an independent uniform distribution and an approximate gradient

$$\nabla f_{S_k}(x^k) := \frac{1}{c_k} \sum_{\ell \in S_k} \nabla f_\ell(x^k) \tag{3}$$

is computed.

## 2.1 A simple error estimate

We begin by recalling a simple estimate about the angle between $\nabla f_{S_k}(x^k)$ and $\nabla f(x^k)$: The sample variance of the $j$-th component of $\nabla f_i(x^k)$ is given by

$$\frac{1}{c_k - 1} \sum_{\ell \in S_k} (\nabla f_\ell(x^k) - \nabla f_{S_k}(x^k))_j^2,$$

and the sample variance of the $j$-th component of $\nabla f_{S_k}(x^k)$ is obtained by multiplying this with $\frac{1}{c_k}$. Thus, the sample estimate for the trace $\mathbb{E}(\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\|^2)$ of the covariance matrix for the components $(\nabla f_{S_k}(x^k))_j$ of $\nabla f_{S_k}(x^k)$ is given by

$$V_g := \frac{1}{c_k(c_k - 1)} \sum_{\ell \in S_k} \|\nabla f_\ell(x^k) - \nabla f_{S_k}(x^k)\|_2^2. \tag{4}$$

(This estimate is meaningful only for $|S_k| \ll m$.) For some value $t > 0$ the probability that $\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\| \geq t$ holds true can then be bounded by a basic Chebyshev-type inequality:

$$t^2 \Pr(\{\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\|^2 \geq t^2\}) \leq \mathbb{E}(\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\|^2) \approx V_g,$$

and thus,

$$\Pr(\{\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\|^2 < t^2\}) \gtrapprox 1 - \tfrac{V_g}{t^2},$$

where the sign "$a \gtrapprox b$" is used as short hand notation for "$a \geq c$ for some $c \approx b$". For a given $\eta \in (0, 1)$ one can select

$$t := (\|\nabla f_{S_k}(x^k)\|^2 + \|\nabla f(x^k)\|^2 - 2\eta \|\nabla f_{S_k}(x^k)\| \|\nabla f(x^k)\|)^{1/2}.$$

Then, the probability that the cosine of the angle between $\nabla f_{S_k}(x^k)$ and $\nabla f(x^k)$ is at least $\eta$ is bounded below by

$$\begin{aligned}
& \Pr(\{\nabla f_{S_k}(x^k)^T \nabla f(x^k) \geq \eta \|\nabla f_{S_k}(x^k)\| \, \|\nabla f(x^k)\|\}) \\
= \; & \Pr(\{-\nabla f_{S_k}(x^k)^T \nabla f(x^k) \leq \tfrac{t^2 - \|\nabla f_{S_k}(x^k)\|^2 - \|\nabla f(x^k)\|^2}{2}\}) \\
= \; & \Pr(\{\|\nabla f_{S_k}(x^k) - \nabla f(x^k)\|^2 \leq t^2\}) \\
\gtrapprox \; & 1 - \tfrac{V_g}{t^2}.
\end{aligned}$$

This inequality essentially being the Chebyshev inequality, it is not very strong. Nevertheless it motivates our goal below to modify the standard stochastic gradient approach so that $V_g$ or some related quantity satisfies an inequality such as

$$V_g \leq \tfrac{1}{2} \|\nabla f_{S_k}(x^k)\|^2. \tag{5}$$

Assume that (5) holds. Then, for $\eta = 0$, guaranteeing that $\nabla f_{S_k}(x^k)$ is a descent direction, it follows that $t^2 \approx 2\|\nabla f_{S_k}(x^k)\|^2$, and thus the probability of $\nabla f_{S_k}(x^k)$ being a descent direction is at least $3/4$, and smaller values of $V_g$ lead better descent estimates.

The Chebyshev inequality also suggests the short notion that $V_g$ is referred to as *sample variance of the vector* $\nabla f_{S_k}(x^k)$ in the sequel.

## 2.2 The computation of an averaged stochastic gradient

We now discuss a modification of the standard stochastic gradient approach as to generate approximate gradients with reduced variance. Starting from $\nabla^A f(x^0) := \nabla f_{S_0}(x^0)$, in addition to $\nabla f_{S_{k+1}}(x^{k+1})$ a second approximation $\nabla^A f(x^{k+1})$ to $\nabla f(x^{k+1})$ is computed for $k \geq 0$ as follows: For brevity denote $x = x^k$, $x^+ = x^{k+1}$, and $\Delta x = x^{k+1} - x^k$. Then a set $\widehat{S}_k$ (possibly of smaller cardinality) is chosen independently and

$$\nabla^A f(x^+) := \rho \left( \nabla^A f(x) + \nabla f_{\widehat{S}_k}(x^+) - \nabla f_{\widehat{S}_k}(x) \right) + (1-\rho) \nabla f_{S_{k+1}}(x^+), \tag{6}$$

where $\rho \in (0,1)$ is a parameter discussed next. In any case, for any $\rho \in (0,1)$, using the linearity of the expected value, it follows by induction that $\mathbb{E}(\nabla^A f(x^+)) = \nabla f(x^+)$.

We denote $\nabla^A f(x^+)$ by "averaged stochastic gradient" (ASG) and note that $\nabla^A f(x^+)$ is a weighted average of the stochastic gradient $\nabla f_{S_{k+1}}(x^+)$ at $x^+$ and a correction of the ASG $\nabla^A f(x)$ at the "old" $x$.

We now turn to the determination of $\rho$. When evaluating $\nabla f_{\widehat{S}_k}(x^+)$, in analogy to $V_g$ also the sample variance

$$V_H := \frac{1}{c_k(c_k-1)} \sum_{\ell \in \widehat{S}_k} \| (\nabla f_\ell(x^+) - \nabla f_\ell(x)) - (\nabla f_{\widehat{S}_k}(x^+) - \nabla f_{\widehat{S}_k}(x)) \|_2^2$$

of $\nabla f_{\widehat{S}_k}(x^+) - \nabla f_{\widehat{S}_k}(x)$ is computed. Here, $\nabla f_{\widehat{S}_k}(x^+) - \nabla f_{\widehat{S}_k}(x) = H_k \Delta x$ where $H_k := \int_0^1 \nabla^2 f_{\widehat{S}_k}(x + t\Delta x) dt$, so that $V_H$ is an estimate for the variance of the vector $H_k \Delta x$ under the uniformly random choice of $\widehat{S}_k$. Note that $V_H = O(\|\Delta x\|_2^2)$ when $f$ is twice continuously differentiable, and thus, $V_H \to 0$ when the step length $\|\Delta x\|_2$ converges to zero, $\|\Delta x\|_2 \to 0$. This happens for example, if the generated sequence $\{x^k\}_{k \in \mathbb{N}}$ converges. In contrast $V_g$ will generally not converge to zero.

By construction, the three terms $\nabla^A f(x)$, $\nabla f_{\widehat{S}_k}(x^+) - \nabla f_{\widehat{S}_k}(x)$, and $\nabla f_{S_{k+1}}(x^+)$ are stochastically independent and thus, the variance $V_A(x^+)$ of the vector $\nabla^A f(x^+)$ is given by

$$V_A(x^+) = \rho^2 V_A(x) + \rho^2 V_H + (1-\rho)^2 V_g. \tag{7}$$

The minimizer of (7) is

$$\rho := \frac{V_g}{V_g + (V_A(x) + V_H)} \quad \text{with} \quad V_A(x^+) = \frac{V_g (V_A(x) + V_H)}{V_g + (V_A(x) + V_H)}. \tag{8}$$

If the quantities $V_g$ and $V_H$ were constant and the above process of replacing $V_A(x)$ with the above estimate $V_A(x^+)$ was iterated, the limit would be

$$-\frac{V_H}{2} + \frac{\sqrt{4V_g V_H + V_H^2}}{2} < \sqrt{V_g V_H} = O(\|\Delta x\|)$$

which is much smaller than $V_g$ if $V_H = O(\|\Delta x\|_2^2)$ is much smaller than $V_g$.

In our numerical implementation also the strategy is tested where $\widehat{S}_k$ is chosen $\widehat{S}_k = S_k$, so that only one additional evaluation, namely of $\nabla f_{S_k}(x^+)$, is needed. In this case, the two terms $\nabla^A f(x)$ and $\nabla f_{S_k}(x^+) - \nabla f_{S_k}(x)$ are not independent, and thus the above estimate (7) is merely an anticipation, but for large $k$ the correlation coefficients are very small since the influence of $\nabla f_{S_k}(x)$ on $\nabla^A f(x)$ is very small. In our numerical examples this slightly cheaper strategy worked well. Below, we summarize an algorithm using this simplified approach for computing $\nabla^A f(x^+)$ given $\nabla^A f(x)$ and an estimate $V_A(x)$:

**Algorithm 1** *(Updating ASG:)*
*Input: a randomly generated estimate $\nabla^A f(x)$ at some point $x = x^k$ satisfying $\mathbb{E}(\nabla^A f(x)) = \nabla f(x)$, an estimate $V_A(x)$ for the variance $\mathbb{E}(\|\nabla f(x) - \nabla^A f(x)\|^2)$, a mini-batch $S_k \subset \{1, \ldots, m\}$ with $c_k := |S_k|$ and associated gradients $\nabla f_\ell(x)$ for $\ell \in S_k$.*
*Output: an estimate $\nabla^A f(x^+)$ at some point $x^+ = x^{k+1}$ and an estimate $V_A(x^+)$ for the variance $\mathbb{E}(\|\nabla f(x^+) - \nabla^A f(x^+)\|^2)$.*

1. *Compute $\nabla_{S_k} f(x^+)$ ("old $S_k$") and set*
$$V_H := \frac{1}{c_k(c_k-1)} \sum_{\ell \in S_k} \|(\nabla f_\ell(x^+) - \nabla f_\ell(x)) - (\nabla f_{S_k}(x^+) - \nabla f_{S_k}(x))\|_2^2.$$
2. *Select $S_{k+1}$ independently of $S_k$, compute $\nabla_{S_{k+1}} f(x^+)$, and set*
$$V_g := \frac{1}{c_{k+1}(c_{k+1}-1)} \sum_{\ell \in S_{k+1}} \|\nabla f_\ell(x^+) - \nabla f_{S_{k+1}}(x^+)\|_2^2.$$
3. *Set $\rho := \frac{V_g}{V_g + V_A(x) + V_H}$ and*
$$\nabla^A f(x^+) := \rho \left(\nabla^A f(x) + \nabla f_{S_k}(x^+) - \nabla f_{S_k}(x)\right) + (1-\rho)\nabla f_{S_{k+1}}(x^+).$$
4. *Set $V_A(x^+) = \frac{V_g(V_A(x)+V_H)}{V_g + V_A(x) + V_H}$.*

**Note 1**

1. *Initially, at a first iterate $x^0$, one may define $\nabla^A f(x^0) := \nabla f_{S_k}(x^0)$ via (3) with $V_A(x^0) := V_g$ in (4).*
2. *The update is intended for points $x^+$ "not far from $x$" in the sense that the relative change of the gradients $\nabla f_l$ at $x$ and at $x^+$ is moderate. When the change is large, the value $\rho$ in Step 3. will be small and the update will return an estimate $\nabla^A f(x^+)$ close to the stochastic gradient $\nabla f_{S_{k+1}}(x^+)$*
3. *For a given set $S_k$ the gradient of $f_{S_k}$ is evaluated at two points $x$ and $x^+$ rather than just at one point $x$ as in standard SGD algorithms, but when the data associated with $f_{S_k}$ is kept in cache memory, the extra time for the second evaluation is less than the time needed for the first evaluation and the extra time needed to generate ASG is moderate.*

## 2.3 Limitations of the ASG

Unfortunately, there are two other points that limit the usefulness of the averaged stochastic gradient as proposed in Algorithm 1:

1. If the iterates $x^k$ approach a minimizer $x^*$ with $\nabla f(x^*) = 0$ and if the absolute error $\|\nabla^A f(x^k) - \nabla f(x^k)\|$ remains about constant, then the relative error $\|\nabla^A f(x^k) - \nabla f(x^k)\|/\|\nabla f(x^k)\|$ tends to infinity (because $\|\nabla f(x^k)\| \to 0$). Thus, it is not sufficient to start somehow with a fairly accurate approximation $\nabla^A f(x^0)$ to $\nabla f(x^0)$ and to make sure, the error does not increase, but the absolute error must decrease in order to maintain a certain relative accuracy.

2. Second, and in aggravation of the first point, if the iterates $x^k$ follow a certain random walk with fairly large step lengths but such that the average $\bar{x}^k$ of the iterates $x^k$ converges to a minimizer $x^*$, then the quantity $\|\Delta x\| = \|x^+ - x\|$ in *Step 1.* of Algorithm 1 may be large compared to $\|\bar{x}^k - x^*\|$, and thus,

$$\|\nabla^A f(x^+) - \nabla f(x^+)\| \approx O(\sqrt{V_A(x^+)}\,) \approx O(\|\Delta x\|^{1/2}) \gg \|\bar{x}^+ - x^*\| \approx \|\nabla f(\bar{x}^+)\|$$

leading to a poor approximation $\nabla^A f(x^+)$ in *Step 3.*

## 2.4 An adaptive selection of the batch size and the step length

Both points of the previous subsection can be addressed by controlling the step lengths $\alpha_k$ and the batch size $|S_{k+1}|$ in an ASG descent algorithm of the form

$$x^{k+1} := x^k - \alpha_k \nabla^A f(x^k).$$

Here, $\alpha_k$ and the cardinality of $S_{k+1}$ are to be chosen in some optimal fashion such that

$$V_A(x^{k+1}) \overset{(8)}{\leq} \tfrac{V_g(V_A(x^k)+V_H)}{V_g+V_A(x^k)+V_H} \quad \overset{!}{\leq} \quad \delta\|\nabla^A f(x^k)\|^2, \tag{9}$$

holds true for some parameter $\delta \in (0, \frac{1}{2}]$. If $\|\nabla^A f(x^k)\| \approx \|\nabla^A f(x^{k+1})\| \approx \|\nabla f(x^{k+1})\|$ then relation (9) implies in analogy to relation (5)

$$\Pr(\{\cos(\sphericalangle(\nabla f(x^{k+1}), \nabla^A f(x^{k+1}))) \geq 1 - \delta\}) \gtrapprox \tfrac{1}{2}.$$

Note that $V_g$ depends on the batch size of $S_{k+1}$ and $V_H$ depends on the step length $\alpha_k$: Assume for the moment that $\bar{V}_g$ is an estimate associated with the batch size $\bar{c} \geq 2$ and $\bar{V}_H$ is an estimate associated with a "targeted" step length $\bar{\alpha}_k$ that is discussed in the next subsection.

If $V_g := \bar{V}_g$ and $V_H := \bar{V}_H$ satisfy (9), then set $|S_{k+1}| = \bar{c}$ and $\alpha_k = \bar{\alpha}_k$.

If not, then consider the situation where instead the batch size is taken as $|S_{k+1}| = \gamma_k \bar{c}$ and the step length is set to $\alpha_k = \beta_k \bar{\alpha}_k$ with some parameters $\gamma_k \geq 1$ and $\beta_k \in (0, 1]$. Then, $V_g \approx \bar{V}_g/\gamma_k$ and $V_H \approx \beta_k^2 \bar{V}_H$.

Moreover, the progress of the algorithm at the $k$-th iteration is proportional to $\beta_k$ and the amount of work is roughly proportional to $2+\gamma_k$ (since there are 2 evaluations of $\nabla f_{S_k}$ in *Step 3.* of Algorithm 1, namely at $x^k$ and at $x^{k+1}$, and the evaluation of $\nabla f_{S_{k+1}}(x^{k+1})$ costs $\gamma_k$ times as much as the one of $\nabla f_{S_k}(x^{k+1})$). Thus maximizing the progress divided by the amount of the associated work leads to the problem

$$\max \ \tfrac{\beta}{2+\gamma} \ \mid \ \frac{\bar{V}_g}{\gamma}(V_A(x^k)+\beta^2\bar{V}_H) = \delta\|\nabla^A f(x^k)\|^2(\frac{\bar{V}_g}{\gamma}+V_A(x^k)+\beta^2\bar{V}_H),\ \beta\le 1,\ \gamma\ge 1.$$

The equality-constraint can be solved for $\gamma$ leading to

$$\gamma=\gamma(\beta):=\frac{\bar{V}_g(V_A(x^k)+\beta^2\bar{V}_H-\delta\|\nabla^A f(x)\|^2)}{\delta\|\nabla^A f(x)\|^2(V_A(x^k)+\beta^2\bar{V}_H)}. \tag{10}$$

A line search for $\beta\in[0,1]$ maximizing $\beta/(2+\gamma(\beta))$ either results in a value $\gamma<1$ in which case we fix $\gamma=1$ and solve the equality constraint for $\beta$ or in a value $\gamma\ge 1$ that can be rounded to the next integer or the next integer multiple of $1/\bar{c}$. (Note that maximizing $\beta/(2+\gamma)$ seemingly does not depend on how large $\bar{\alpha}_k$ is initially chosen, but the initial "targeted" values of $|S_k|$ and $\bar{\alpha}_k$ are hidden in the terms $\bar{V}_g$ and $\bar{V}_H$ in the right hand side of (10).)

As the right hand side of (9) changes with each iteration it may happen that the quantity $\gamma$ in (10) is rather large in some iteration where the right hand side was reduced. To limit the reductions of the right hand side of (9), a monotonicity can be enforced setting $\tau_0:=\infty$, $\tau_k:=\min\{\tau_{k-1},\ \|\nabla^A f(x^k)\|^2\}$ for $k\ge 1$ and testing

$$\tfrac{V_g(V_A(x^k)+V_H)}{V_g+V_A(x^k)+V_H} \ \overset{!}{\le} \ \delta\tau_k, \tag{11}$$

instead of (9). To further limit the computational effort in such iteration the following "computational safeguard" can be used: Fix an upper bound $\bar{\gamma}>1$ (e.g. $\bar{\gamma}=10$). If $\gamma>\bar{\gamma}$ then redefine

$$\beta:=\beta\cdot\tfrac{\bar{\gamma}}{\gamma} \quad \text{followed by} \quad \gamma:=\bar{\gamma}. \tag{12}$$

This safeguard limits the computational effort but may effect that (11) is violated in a given iteration. If, nevertheless, $V_A(x^{k+1})<V_A(x^k)$ then there is reason for the anticipation that (11) can be satisfed again in a future iteration.

If this anticipation does not substantiate, the goal of enforcing (11) is "too expensive" to maintain. In this case, preliminary numerical experiments indicate that a variance reduction of $\nabla^A f(x^k)$ can still be reached by an Algorithm as outlined in Section 2.7 below.

## 2.5 The initial targeted parameter values

The initial targeted batch size $\bar{c}\ge 2$ may depend on the computer hardware such as the maximum number of elementary functions whose data can be loaded into cache memory or

the number of parallel processors that can be used to simultaneously evaluate gradients of the elementary functions. In our experiments we arbitrarily fix $\bar{c} = 30$ as initial targeted batch size.

To address the choice of an initial targeted step length we recall the above observation that long steps decrease the accuracy of the ASG. This motivates the aim of staying "reasonably close" to the steepest descent path. Moreover, when following a small neighborhood of the steepest descent path with adaptively chosen short steps, possible ill-conditioning of $f$ is also addressed adaptively, in spite of the fact of only having approximate first order information available about the function $f$. The aim of remaining somewhat close to the steepest descent path leads to a targeted step length $\alpha_k$ based on the previous iteration: Consider the estimate

$$\chi_k := \frac{2\|\nabla f_{S_{k-1}}(x^{k-1}) - \nabla f_{S_{k-1}}(x^k)\|}{\|\nabla f_{S_{k-1}}(x^{k-1})\| + \|\nabla f_{S_{k-1}}(x^k)\|} \tag{13}$$

of the relative change of the gradient of $f$ between $x^{k-1}$ and $x^k$. If $\chi_k$ is very small a longer step might still have remained within a small neighborhood of the steepest descent path and might have led to more progress in reducing the objective function. If it is too large, then it might happen that $f(x^k) > f(x^{k-1})$ (but we are not willing to invest the computational effort to verify this). Thus, the targeted step length for the next iteration is aimed at leading to $\chi_{k+1} \stackrel{!}{=} \bar{\chi}$ for some moderately small value $\bar{\chi} \in (0, 1/2)$. More precisely, let $\Delta x^{k-1} := x^k - x^{k-1}$ and assume that $\chi_k \approx const \cdot \|\Delta x^{k-1}\|$. If $\chi_{k+1}$ is determined by the same constant "*const*" we obtain

$$\|\Delta x^k\| \stackrel{!}{=} \tfrac{\bar{\chi}}{\chi_k} \|\Delta x^{k-1}\|$$

and a targeted initial step $\bar{\alpha}_k := \frac{\bar{\chi}\|\Delta x^{k-1}\|}{\chi_k \|\nabla^A f(x^k)\|}$.

## 2.6 An adaptive estimate of the function values

When the step lengths are chosen rather short and the batch size not too small, one can also use an approach similar to the estimation of the averaged gradient to render an estimate of the function value as well. We first note that if $f$ was a quadratic function, then the equation

$$f(x^+) = f(x) + \frac{1}{2}(\nabla f(x) + \nabla f(x^+))^T \Delta x$$

would hold as an exact identity, where again, $x^+ := x + \Delta x$. More generally, for three times differentiable $f$, the error of the above Taylor expansion is in the order of $\|\Delta x\|_2^3$ and is ignored in the analysis below.

Assume that the estimate of $f(x)$ is denoted by $f^A(x)$ with an estimated variance $V_{f,A}(x)$ that is initially set to $V_{f,A}(x^0) := V_f(x^0)$. Then we set

$$f^A(x^+) := \tilde{\rho}\left(f^A(x) + \frac{1}{2}(\nabla^A f(x) + \nabla^A f(x^+))^T \Delta x\right) + (1 - \tilde{\rho}) f_{S_{k+1}}(x^+) \tag{14}$$

where $\tilde{\rho}$ is chosen as to minimize

$$\rho^2(V_{f,A}(x) + V_A(x)\|\Delta x\|^2) + (1-\rho)^2 V_f$$

with

$$V_f := V_f(x^+) := \frac{1}{c_{k+1}(c_{k+1}-1)} \sum_{\ell \in S_{k+1}} |f_\ell(x^+) - f_{S_{k+1}}(x^+)|^2. \tag{15}$$

leading to

$$\tilde{\rho} = \frac{V_f}{V_f + V_{f,A}(x) + V_A(x)\|\Delta x\|_2^2} \tag{16}$$

and to the estimated Variance $V_{f,A}(x^+)$ for $f^A(x^+)$,

$$V_{f,A}(x^+) = \frac{V_f(V_{f,A}(x) + V_A(x)\|\Delta x\|_2^2)}{V_f + V_{f,A}(x) + V_A(x)\|\Delta x\|_2^2}. \tag{17}$$

## 2.7 An overall algorithm

The algorithmic details of the previous subsections lead to the following overall algorithm that generates an approximate minimizer of $f$ along with estimates for the gradient and function value at the approximate minimizer. (The gradient estimate gives some approximate lower bound for the distance to an exact local minimizer.)

**Algorithm 2** *(ASG descent algorithm for minimizing a function $f$:)*
*Select*
– *a stopping tolerance $\epsilon \in (0,1)$ (e.g. $\epsilon = 10^{-4}$) and a maximum iteration number "maxit"*
– *a desired accuracy $\delta \in (0, 1/2)$ for the approximate gradients (e.g. $\delta = 0.2$),*
– *a parameter $\bar{\chi} \in (0, 1/2)$ for the closeness to the steepest descent path (e.g. $\bar{\chi} = 0.2$),*
– *an initial batch size $\bar{c} \in \{2, \ldots, m\}$ (e.g. $\bar{c} = \min\{m, 30\}$)*
– *a maximum factor $1 \leq \bar{\gamma}$ to increase the initial batch size (e.g. $\bar{\gamma} = \min\{10, \frac{m}{\bar{c}}\}$).*
*Further choose*
– *a first step length $\bar{\alpha}_0 = \alpha_0 > 0$ (e.g. $\alpha_0 = 1$),*
– *subsets $S_0,\ S_1 \subset \{1, \ldots, m\}$ with $|S_0| = |S_1| = \bar{c}$, and set $\gamma_0 := 1$.*
*Set $\nabla^A f(x^0) := \nabla f_{S_0}(x^0)$, $tol := \epsilon\|\nabla^A f(x^0)\|_2$, $\Delta x^0 := -\bar{\alpha}_0 \nabla^A f(x^0)$, $x^1 := x^0 + \Delta x^0$.*
*Compute $\nabla^A f(x^1)$ and $V_A := V_A(x^1)$ with Algorithm 1. Record $V_g$, $V_H$. Set $k := 1$.*
*Compute $f^A(x^1)$ as in (14), (15), (16), and (17).*

*While $\|\nabla^A f(x^1)\| > tol$ and $k < maxit$ do*

1. *Set $\bar{\alpha}_k := \frac{\bar{\chi}\|\Delta x^{k-1}\|}{\chi_k \|\nabla^A f(x^k)\|}$ with $\chi_k$ as in (13).*
2. *Correct $V_g := V_g/\gamma_{k-1}$ and $V_H := V_H \cdot \bar{\alpha}_k^2/\alpha_{k-1}^2$ (corresponding to $\beta_k = \gamma_k = 1$).*
3. *Temporarily select $\beta_k = \gamma_k = 1$ and test whether (11) is satisfied.*

4. *If not then*

   (a) *Define* $\gamma(\beta)$ *as in (10).*

   (b) *Maximize* $\frac{\beta}{2+\gamma(\beta)}$ *for* $\beta \in [0,1]$ *with optimal values* $\beta^*$ *and* $\gamma(\beta^*)$.

   (c) *If* $\gamma(\beta^*) < 1$ *set* $\gamma(\beta^*) = 1$ *and* $\beta^* = \sqrt{\frac{\delta\|\nabla^A f(x^k)\|^2(V_g+V_A)-V_gV_A}{V_H(V_g-\delta\|\nabla f^A(x^k)\|^2)}} \in (0,1]$.

   (d) *If* $\gamma(\beta^*) > \bar{\gamma}$ *apply the computational safeguard (12).*

   (e) *Set* $\beta_k := \beta^*$ *and* $\gamma_k := \gamma(\beta^*)$.

5. *Select* $S_{k+1}$ *of cardinality* $\lfloor \gamma_k \bar{c} \rfloor$ *and set* $\alpha_k := \beta_k \bar{\alpha}_k$.

6. *Set* $\Delta x^k := -\alpha_k \nabla^A f(x^k)$ *and* $x^{k+1} := x^k + \Delta x^k$.

7. *Determine* $\nabla^A f(x^{k+1})$ *and* $V_A := V_A(x^{k+1})$ *with Algorithm 1. Record* $V_g$, $V_H$.

8. *Compute* $f^A(x^{k+1})$ *as in (14), (15), (16), and (17).*

9. *Set* $k := k+1$.

**Note 2**

1. *Upon termination the point* $x^k$ *is an approximate minimizer with approximate gradient* $\nabla^A f(x^k)$. *If* $k <$ *maxit then the norm of the approximate gradient is reduced by a factor at least* $\epsilon$ *compared to* $\|\nabla^A f(x^0)\|_2$.

2. *If* $\gamma(\beta^*) > \bar{\gamma}$ *in Step 4. (d) above, one may give up the aim of maintaining (11) and switch to an algorithmic variant that limits the computational effort per iteration.*

# 3. Preliminary numerical experiments

The concepts introduced in this article are not intended to be exclusive, but to be possibly used in conjunction with other stochastic gradient approaches, for example in a final phase. In the implementation for the results below, a certain minimum step length along the negative averaged gradient was required – at the expense of accuracy of the averaged gradient estimate. The numerical experiments aim at testing in how far the estimates for the norm of the gradient and for function values may differ from the anticipated bounds under such condition.

Some very preliminary examples were tested with the MNIST Test-set for the classification of hand-written digits. The test sets consists of 60000 Training data points in $\mathbb{R}^{784}$. Logistic regression for the individual digits was applied and the digit with highest agreement was chosen.

In the experiments 9237 out of 10000 test data were classified correctly. The rather high classification error appears to be due to the logistic model; increasing the final accuracy in the test runs results in almost no improvement of the classification error.

Each digit took about 750 iterations with a batch size of at least 30 (adaptively increased as described in Algorithm 2). The overall time for all digits on a standard desktop computer was 207 sec.

The initial norm of the gradients was about 1 and the final norm of the gradients was about $10^{-2}$ in our test runs. We note that the existence of a minimizer might not be given in this model.

Typical values of the algorithmic parameters at the end are (with large variations):

$$V_g \approx 10^{-2}, \qquad V_A \approx 10^{-4}, \qquad V_H \approx 10^{-8}$$

rendering a first experimental confirmation of the approximation strategy of $\nabla f(x)$ and $f(x)$.

# References


[1] Xiao-Bo Jin, Xu-Yao Zhang, Kaizhu Huang and Guang-Gang Geng (2018): Stochastic Conjugate Gradient Algorithm with Variance Reduction, IEEE Transactions on Neural Networks and Learning Systems, 1-12.

[2] Johnson, R., and Zhang, T. (2013): Accelerating Stochastic Gradient Descent Using Predictive Variance Reduction, in: Proceedings of the 26th International Conference on Neural Information Processing Systems - Vol. 1, NIPS'13, Curran Associates Inc., USA, 315–323.

[3] Nemirovski, A.S., Juditsky, A., Lan, G., and Shapiro, A. (2009): Robust stochastic approximation approach to stochastic programming. SIAM J. Optim., **19**(4), 15741609.

[4] Polyak, B.T., Juditsky, A.B. (1992): Acceleration of stochastic approximation by averaging. SIAM J. on Control and Opt., **30**(4), 838–855

[5] Rakhlin, A., Shamir, O., Sridharan, K. (2012): Making Gradient Descent Optimal for Strongly Convex Stochastic Optimization. ICML **12**, 1571-1578.