mS2GD: Mini-Batch Semi-Stochastic Gradient Descent in the Proximal Setting

Jakub Konečný  
University of Edinburgh  
United Kingdom, EH9 3FD  
J.Konecny@sms.ed.ac.uk

Jie Liu  
Lehigh University  
Bethlehem, PA 18015  
jie.liu@lehigh.edu

Peter Richtárik  
University of Edinburgh  
United Kingdom, EH9 3FD  
peter.richtarik@ed.ac.uk

Martin Takáč  
Lehigh University  
Bethlehem, PA 18015  
takac.mt@gmail.com

Abstract

We propose a mini-batching scheme for improving the theoretical complexity and practical performance of semi-stochastic gradient descent applied to the problem of minimizing a strongly convex composite function represented as the sum of an average of a large number of smooth convex functions, and simple nonsmooth convex function. Our method first performs a deterministic step (computation of the gradient of the objective function at the starting point), followed by a large number of stochastic steps. The process is repeated a few times with the last iterate becoming the new starting point. The novelty of our method is in introduction of mini-batching into the computation of stochastic steps. In each step, instead of choosing a single function, we sample $b$ functions, compute their gradients, and compute the direction based on this. We analyze the complexity of the method and show that the method benefits from two speedup effects. First, we prove that as long as $b$ is below a certain threshold, we can reach predefined accuracy with less overall work than without mini-batching. Second, our mini-batching scheme admits a simple parallel implementation, and hence is suitable for further acceleration by parallelization.

1 Introduction

The problem we are interested in is to minimize a sum of two convex functions,

$$\min_{x \in \mathbb{R}^d} \{P(x) := f(x) + R(x)\}, \quad (1)$$

where $f$ is the average of a large number of smooth convex functions $f_i(x)$, i.e.,

$$f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x), \quad (2)$$

We further make the following assumptions:

Assumption 1. The regularizer $R : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is convex and closed. The functions $f_i : \mathbb{R}^d \to \mathbb{R}$ are differentiable and have Lipschitz continuous gradients with constant $L > 0$. That is,

$$||\nabla f_i(x) - \nabla f_i(y)|| \leq L ||x - y||,$$

for all $x, y \in \mathbb{R}^d$, where $|| \cdot ||$ is L2 norm.

Hence, the gradient of $f$ is also Lipschitz continuous with the same constant $L$. 

1
Assumption 2. $P$ is strongly convex with parameter $\mu > 0$, i.e., $\forall x, y \in \text{dom}(R)$,
\[
P(y) \geq P(x) + \xi^T(y - x) + \frac{\mu}{2} \|y - x\|^2, \quad \forall \xi \in \partial P(x),
\] where $\partial P(x)$ is the subdifferential of $P$ at $x$.

Let $\mu_f \geq 0$ and $\mu_R \geq 0$ be the strong convexity parameters of $f$ and $R$, respectively (we allow both of them to be equal to 0, so this is not an additional assumption). We assume that we have lower bounds available ($\nu_f \in [0, \mu_f]$ and $\nu_R \in [0, \mu_R]$).

Related work There has been intensive interest and activity in solving problems of \cite{1} in the past years. An algorithm that made its way into many applied areas is FISTA \cite{1}. However, this method is impractical in large-scale setting (big data). A popular approach can be equivalently written using the proximal operator $\text{prox}_h$.

Our Contributions. In this work, we combine the variance reduction ideas for stochastic gradient methods with mini-batching. In particular, develop and analyze mS2GD (Algorithm \cite{1}) – a mini-batch variant of S2GD \cite{8}; the proximal setting was motivated by SVRG \cite{7, 21}. This brief paper is closely related to the works on stochastic gradient methods with a technique of explicit variance reduction of stochastic approximation of the gradient. In particular, our method is a mini-batch variant of S2GD \cite{8}; the proximal setting was motivated by SVRG \cite{7, 21}.

A typical stochastic gradient descent (SGD) method will randomly sample $i^{th}$ function and then update the variable $x$ using $\nabla f_i(x)$ — an estimate of $\nabla f(x)$. Important limitation of SGD is that it is inherently sequential, and thus it is difficult to parallelize them. In order to enable parallelism, mini-batching—samples multiple examples per iteration—is often employed \cite{17, 3, 23, 20, 18}.

2 Proximal Algorithms

A popular proximal gradient approach to solving \cite{1} is to form a sequence $\{y_k\}$ via
\[
y_{k+1} = \arg \min_{x \in \mathbb{R}^d} \left[U_k(x) \right] = f(y_k) + \nabla f(y_k)^T(x - y_k) + \frac{1}{2h}\|x - y_k\|^2 + R(x).
\]
Note that $U_k$ in an upper bound on $P$ if $h > 0$ is a stepsize parameter satisfying $1/h \geq L$. This procedure can be equivalently written using the proximal operator as follows:
\[
y_{k+1} = \text{prox}_{hR}(y_k - h\nabla f(y_k)), \quad \text{prox}_{R}(z) = \arg \min_{x \in \mathbb{R}^d} \left\{ \frac{1}{2}\|x - z\|^2 + R(x) \right\}.
\]
In large-scale setting it is more efficient to instead consider the stochastic proximal gradient approach, in which the proximal operator is applied to a stochastic gradient step:
\[
y_{k+1} = \text{prox}_{hR}(y_k - hv_k),
\]
where $v_k$ is a stochastic estimate of $\nabla f(y_k)$. Of particular relevance to our work are the the SVRG \cite{7}, S2GD \cite{8} and Prox-SVRG \cite{21} methods where the stochastic estimate of $\nabla f(y_k)$ is of the form
\[
v_k = \nabla f(x) + \frac{1}{nq_i}(\nabla f_i(y_k) - \nabla f_i(x)),
\]
where $x$ is an “old” reference point for which the gradient $\nabla f(x)$ was already computed in the past, and $i_k \in \{1, 2, \ldots, n\}$ is a random index equal to $i$ with probability $q_i > 0$. Notice that $v_k$ is an unbiased estimate of the gradient:
\[
\mathbb{E}v_k \nabla f(x) + \sum_{i=1}^{n} q_i \frac{1}{nq_i}(\nabla f_i(y_k) - \nabla f_i(x)) \nabla f(y_k).
\]
SVRG, S2GD and Prox-SVRG, as well as our new mS2GD method, update the points \( y_k \) in an inner loop, and the reference point \( x \) in an outer loop. This ensures that \( v_k \) has low variance, which ultimately leads to extremely fast convergence.

## 3 Mini-batch S2GD

We now describe the mS2GD method (Algorithm 1).

**Algorithm 1 mS2GD**

1. **Input:** \( m \) (max # of stochastic steps per epoch); \( h > 0 \) (stepsize); \( x_0 \in \mathbb{R}^d \) (starting point); minibatch size \( b \in [n] \)
2. **for** \( k = 0, 1, 2, \ldots \) **do**
3. Compute and store \( g_k \leftarrow \nabla f(x_k) = \frac{1}{n} \sum_i \nabla f_i(x_k) \)
4. Initialize the inner loop: \( y_{k,0} \leftarrow x_k \)
5. Let \( t_k \leftarrow t \in \{1, 2, \ldots, m\} \) with probability \( q_t \) given by (6)
6. **for** \( t = 0 \) to \( t_k - 1 \) **do**
7. Choose mini-batch \( A_{kt} \subset [n] \) of size \( b \), uniformly at random
8. Compute a stochastic estimate of \( \nabla f(y_{kt}) \): \( y_{kt} \leftarrow y_k - \frac{1}{b} \sum_{i \in A_{kt}} (\nabla f_i(y_{kt}) - \nabla f_i(x_k)) \)
9. **end for**
10. Set \( x_{k+1} \leftarrow y_{k,t_k} \)
11. **end for**

The main step of our method (Step 8) is given by the update (4), however with the stochastic estimate of the gradient instead formed using a mini-batch of examples \( A_{kt} \subset [n] \) of size \( |A_{kt}| = b \). We run the inner loop for \( t_k \) iterations, where \( t_k = t \in \{1, 2, \ldots, m\} \) with probability \( q_t \) given by

\[
q_t = \frac{1}{\gamma} \left( \frac{1 - h \mu_f}{1 + h \nu_R} \right)^{m-t}, \quad \text{with} \quad \gamma = \sum_{t=1}^{m} \left( \frac{1 - h \mu_f}{1 + h \nu_R} \right)^{m-t}.
\]  

(6)

## 4 Complexity Result

In this section, we state our main complexity result and comment on how to optimally choose the parameters of the method.

**Theorem 1.** Let Assumptions 1 and 2 be satisfied and let \( x_* \) be the minimizer of the objective function. In addition, assume that the stepsize satisfies \( 0 < h < \min \left\{ \frac{1}{1 + h \nu_R}, \frac{1}{2L\alpha(b)} \right\} \) and that \( m \) is sufficiently large so that

\[
\rho \leq \frac{1}{\gamma h} \left( \frac{1 - h \mu_f}{1 + h \nu_R} \right)^m \mu + \frac{4h^2 L\alpha(b)}{1 + h \nu_R} \left( \frac{1 - h \mu_f}{1 + h \nu_R} \right)^m \gamma \leq 1,
\]  

(7)

where \( \alpha(b) = \frac{n - b}{b(n-1)} \). Then mS2GD has linear convergence in expectation:

\[
\mathbb{E}(P(x_k) - P(x_*)) \leq \rho^k (P(x_0) - P(x_*)).
\]

If we consider the special case \( \nu_f = 0, \nu_R = 0 \) (i.e., if the algorithm does not have any nontrivial good lower bounds on \( \mu_f \) and \( \mu_R \)), we obtain

\[
\rho = \frac{1}{\mu h (1 - 4L\alpha(b)m)} + 4L\alpha(b)(m + 1) \frac{\mu h (1 - 4L\alpha(b)m)}{1 - 4L\alpha(b)m}.
\]  

(8)

In the special case when \( b = 1 \) we get \( \alpha(b) = 1 \) in the above theorem, the rate given by (8) exactly recovers the rate achieved by Prox-SVRG [21]. If moreover \( R = 0 \), this rate is very similar (albeit very slightly weaker) to the one achieved by S2GD (see Eq (12) in [8]) in the case when the method does not have access to any nontrivial lower bound on \( \mu \).
4.1 Mini-batch speedup

In order to be able to see the speed-up we can gain from the mini-batch strategy, and due to many parameters in the complexity result (Theorem 1) we need to fix some of the parameters. For simplicity, we will use $\nu_f$ and $\nu_R$ equal to 0, so we can analyse (8) instead of (7). Let us consider the case when we also fix $k$ (number of outer iterations). Once the parameter $k$ is fixed and in order to get some $\epsilon$ accuracy, we get the value of $\rho$ which will guarantee the result.

Let us now fix target decrease in single epoch $\rho = \rho_*$. For any $1 \leq b \leq n$, define $(h^b_\ast, m^b_\ast)$ to be the optimal pair stepsize-size of the inner loop, such that $\rho < \rho_*$. This pair is optimal in the sense that $m^b_\ast$ is the smallest possible — because we are interested in minimizing the computational effort, thus minimizing $m$. If we set $b = 1$, we recover the optimal choice of parameters without mini-batching. If $m^b_\ast \leq m^1_\ast/b$, then we can reach the same accuracy with less evaluations of gradient of a function $f_i$. The following theorem states the formula for $h^b_\ast$ and $m^b_\ast$. Equation (9) shows that as long as the condition $\tilde{h}^b \leq \frac{1}{L}$ is satisfied, $m^b_\ast$ is decreasing at a rate faster than $1/b$. Hence, we can attain the same accuracy with less work, compared to the case when $b = 1$.

**Theorem 2.** Fix target $\rho = \rho_*$, where $\rho$ is given by (5) and $\rho_* \in (0, 1)$. If we consider the mini-batch size $b$ fixed, then the choice of stepsize $h^b_\ast$ and size of inner loop $m^b_\ast$ that minimizes the work done — the number of gradients evaluated — while having $\rho \leq \rho_*$, is given by:

$$\tilde{h}^b := \sqrt{(\frac{1+\rho}{\rho^2})^2 + \frac{1}{4\mu\alpha(b) L}} - \frac{1+\rho}{\mu^2}.$$  

If $\tilde{h}^b \leq \frac{1}{L}$ then $h^b_\ast = \tilde{h}^b$ and

$$m^b_\ast = \frac{4}{\sqrt{\frac{\mu^2}{\alpha^2(b) L} + 4(1+\rho)^2 - 2(1+\rho)}} = 8\alpha(b)\sqrt{1 + \rho + \frac{1}{4\alpha^2(b) L} \mu^2 \rho^2 + (1+\rho)^2}.$$  

Otherwise $h^b_\ast = \frac{1}{L}$ and $m^b_\ast = \frac{L/\mu + 4\alpha(b)}{\mu - 4\alpha(b)(1+\rho)}$.

5 Experiments

In this section we present a preliminary experiment, and an insight into the possible speedup by parallelism. Figure 1 shows experiments for L2-regularized logistic regression on the RCV1 dataset. We compare S2GD (blue, squares) and mS2GD (green circles) with mini-batch size $b = 8$, without any parallelism. The figure demonstrates that one can achieve the same accuracy with less work. The green dashed line is the ideal (most likely practically unachievable) result with parallelism (we divide passes through data by $b$). For comparison, we also include SGD with constant stepsize (purple, stars), chosen in hindsight to optimize performance. Figure 2 shows the possible speedup in terms of work done, formalized in Theorem 2. Notice that up to a certain threshold, we do not need any more work to achieve the same accuracy (red straight line is ideal speedup; blue curvy line is what mS2GD achieves).

Figure 1: mS2GD needs fewer data passes than other methods (RCV1 data). 

Figure 2: Speedup from parallelism for $\rho = 0.01$ (left) and $\rho = 0.1$ (right). Parameters: $L = 1, n = 1000, \mu = 1/n$.

\[^1\text{Available at \url{http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/}}\]
References


