S2CD: Semi-Stochastic Coordinate Descent

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

Zheng Qu
University of Edinburgh
United Kingdom, EH9 3FD
zheng.qu@ed.ac.uk

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

October 16, 2014

Abstract

We propose a novel reduced variance method—semi-stochastic coordinate descent (S2CD)—for the problem of minimizing a strongly convex function represented as the average of a large number of smooth convex functions: $f(x) = \frac{1}{n}\sum_i f_i(x)$. Our method first performs a deterministic step (computation of the gradient of f at the starting point), followed by a large number of stochastic steps. The process is repeated a few times, with the last stochastic iterate becoming the new starting point where the deterministic step is taken. The novelty of our method is in how the stochastic steps are performed. In each such step, we pick a random function f_i and a random coordinate j—both using nonuniform distributions—and update a single coordinate of the decision vector only, based on the computation of the j^{th} partial derivative of f_i at two different points. Each random step of the method constitutes an unbiased estimate of the gradient of f and moreover, the squared norm of the steps goes to zero in expectation, meaning that the method enjoys a reduced variance property. The complexity of the method is the sum of two terms: $O(n\log(1/\epsilon))$ evaluations of gradients ∇f_i and $O(\hat{\kappa}\log(1/\epsilon))$ evaluations of partial derivatives $\nabla_j f_i$, where $\hat{\kappa}$ is a novel condition number.

1 Introduction

In this paper we study the problem of unconstrained minimization of a strongly convex function represented as the average of a large number of smooth convex functions:

$$\min_{x\in\mathbb{R}^d} f(x) \equiv \frac{1}{n}\sum_{i=1}^{n} f_i(x). \tag{1}$$

Many computational problems in various disciplines are of this form. In machine learning, $f_i(x)$ represents the loss/risk of classifier $x \in \mathbb{R}^d$ on data sample i, f represents the empirical risk (=average loss), and the goal is to find classifier minimizing f. Often, an L2-regularizer of the form $\frac{\mu}{2}\|x\|^2$, for $\mu > 0$, is added to the loss, making it strongly convex and hence easier to minimize.

Assumptions. We assume that $f_i : \mathbb{R}^d \to \mathbb{R}$ is a differentiable convex function with Lipschitz continuous partial derivatives. Formally, we assume that for each $i \in [n] := \{1, 2, \dots, n\}$ and $j \in [d] := \{1, 2, \dots, d\}$ there exists $L_{ij} \geq 0$ such that for all $x \in \mathbb{R}^d$ and $h \in \mathbb{R}$,

$$f_i(x + he_j) \leq f_i(x) + \langle \nabla f_i(x), he_j\rangle + \frac{L_{ij}}{2}h^2, \tag{2}$$

where e_j is the j^{th} standard basis vector in $\mathbb{R}^d$, $\nabla f(x) \in \mathbb{R}^d$ the gradient of f at point x and $\langle\cdot,\cdot\rangle$ is the standard inner product. This assumption was recently used in the analysis the accelerated coordinate descent method APPROX [3]. We further assume that f is μ-strongly convex. That is, we assume that there exists $\mu > 0$ such that for all $x, y \in \mathbb{R}^d$,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|y - x\|^2. \tag{3}$$

Context. Batch methods such as gradient descent (GD) enjoy a fast (linear) convergence rate: to achieve ϵ-accuracy, GD needs $\mathcal{O}(\kappa \log(1/\epsilon))$ iterations, where κ is a condition number. The drawback of GD is that in each iteration one needs to compute the gradient of f, which requires a pass through the entire dataset. This is prohibitive to do many times if n is very large.

Stochastic gradient descent (SGD) in each iteration computes the gradient of a single randomly chosen function f_i only—this constitutes an unbiased (but noisy) estimate of the gradient of f—and makes a step in that direction [13, 7, 19]. The rate of convergence of SGD is slower, $\mathcal{O}(1/\epsilon)$, but the cost of each iteration is independent of n. Variants with nonuniform selection probabilities were considered in [20], a mini-batch variant (for SVMs with hinge loss) was analyzed in [17].

Recently, there has been progress in designing algorithms that achieve the fast $O(\log(1/\epsilon))$ rate without the need to scan the entire dataset in each iteration. The first class of methods to have achieved this are stochastic/randomized coordinate descent methods.

When applied to (1), coordinate descent methods (CD) [8, 12] can, like SGD, be seen as an attempt to keep the benefits of GD (fast linear convergence) while reducing the complexity of each iteration. In CD we only compute a single partial derivative $\nabla_j f(x)$ at each iteration and update a single coordinate of vector x only. The partial derivative is also an unbiased estimate of the gradient – however, unlike the SGD estimate, its variance is low. Indeed, as one approaches the optimum, partial derivatives decrease to zero. While CD methods are able to obtain linear convergence, they typically need $O((d/\mu)\log(1/\epsilon))$ iterations when applied to (1) directly [1]. CD method typically significantly outperform GD, especially on sparse problems with a very large number variables/coordinates [8, 12].

An alternative to applying CD to (1) is to apply it to the dual problem; this is possible under certain additional structural assumptions on the functions f_i. This is the strategy employed by stochastic dual coordinate ascent (SDCA) [16], whose rate is $O((n+\kappa)\log(1/\epsilon))$. The condition number κ here is different (and larger) than the condition number appearing in the rate of GD. Despite this, this is vast improvement on the rates achieved by both GD and SGD. Accelerated [15] and mini-batch [17] variants of SDCA have also been proposed.

Recently, there has been progress in designing primal methods which match the fast rate of SDCA. Stochastic average gradient (SAG) [14], and more recently SAGA [1], move in a direction composed of old stochastic gradients. The semi-stochastic gradient descent (S2GD) [6] and stochastic variance reduced gradient (SVRG) [5, 18] methods employ a different strategy: one first computes the gradient of f, followed by $O(\kappa)$ steps where only stochastic gradients are computed. These are used to estimate the change of the gradient, and it is this direction which combines the old gradient and the new stochastic gradient information which used in the update.

Goal. In this work we develop a new method—semi-stochastic coordinate descent (S2CD)—for solving (1), enjoying a fast rate similar to SDCA/SAG/S2GD/SVRG/SAGA. S2CD can be seen as a hybrid between S2GD and CD. In particular, the complexity of our method is the sum of two terms: $n\log(1/\epsilon)$ evaluations of ∇f_i and $O(\hat{\kappa}\log(1/\epsilon))$ evaluations of $\nabla_j f_i$, where $\hat{\kappa}$ is a new condition number. While $\hat{\kappa} \geq \kappa$ appearing in the aforementioned methods, this is multiplied by the cost of the evaluation of a partial derivative $\nabla_j f_i$, which can be smaller than the evaluation cost of ∇f_i. Hence, our result can be both better or worse than previous results, depending on whether the increase of the condition number can or can not be compensated by the lower cost of the stochastic steps based on the evaluation of partial derivatives.

2 S2CD Algorithm

In this section we describe the Semi-Stochastic Coordinate Descent method (Algorithm 1). The method has an outer loop (an "epoch"), indexed by counter k, and an inner loop, indexed by t. At the beginning of epoch k, we compute and store the gradient of f at x_k. Subsequently, S2CD enters the inner loop in which a sequence of vectors $y_{k,t}$ for $t = 0, 1 \ldots, t_k$ is computed in a stochastic way, starting from $y_{k,0} = x_k$. The number t_k of stochastic steps in the inner loop is random, following a geometric law:

$$\mathbf{P}(t_k = T) = (1-\mu h)^{m-T}/\beta, \qquad T \in \{1, \ldots, m\}, \qquad \beta := \textstyle\sum_{t=1}^{m}(1-\mu h)^{m-t}.$$

[1]The complexity can be improved to $O(\frac{d\beta}{\tau\mu}\log(1/\epsilon))$ in the case when τ coordinates are updated in each iteration, where $\beta \in [1,\tau]$ is a problem-dependent constant [9]. This has been further studied for nonsmooth problems via smoothing [4], for arbitrary nonuniform distributions governing the selection of coordinates [11] and in the distributed setting [10, 2]. Also, efficient accelerated variants with $O(1/\sqrt{\epsilon})$ rate were developed [3, 2].

Algorithm 1 Semi-Stochastic Coordinate Descent (S2CD)

```
**parameters:** $m$ (max # of stochastic steps per epoch); $h > 0$ (stepsize parameter); $x_0 \in \mathbb{R}^d$ (starting point)
**for** $k = 0, 1, 2, \dots$ **do**
    Compute and store $\nabla f(x_k) = \frac{1}{n}\sum_i \nabla f_i(x_k)$
    Initialize the inner loop: $y_{k,0} \leftarrow x_k$
    Choose random length of the inner loop: let $t_k = T \in \{1, 2, \dots, m\}$ with probability $(1-\mu h)^{m-T}/\beta$
    **for** $t = 0$ to $t_k - 1$ **do**
        Pick coordinate $j \in \{1, 2, \dots, d\}$,with probability $p_j$
        Pick function index $i$ from the set $\{i \; : \; L_{ij} > 0\}$ with probability $q_{ij}$
        Update the $j^{th}$ coordinate: $y_{k,t+1} \leftarrow y_{k,t} - hp_j^{-1}\big(\nabla_j f(x_k) + \frac{1}{nq_{ij}}\left(\nabla_j f_i(y_{k,t}) - \nabla_j f_i(x_k)\right)\big)e_j$
    **end for**
    Reset the starting point: $x_{k+1} \leftarrow y_{k,t_k}$
**end for**
```

In each step of the inner loop, we seek to compute $y_{k,t+1}$, given $y_{k,t}$. In order to do so, we sample coordinate j with probability p_j and subsequently[2] sample i with probability q_{ij}, where the probabilities are given by

$$\omega_i := |\{j : L_{ij} \neq 0\}|, \qquad v_j := \sum_{i=1}^n \omega_i L_{ij}, \qquad p_j := v_j / \sum_{j=1}^d v_j, \qquad q_{ij} := \frac{\omega_i L_{ij}}{v_j}, \qquad p_{ij} := p_j q_{ij}. \tag{4}$$

Note that $L_{ij} = 0$ means that function f_i does not depend on the j^{th} coordinate of x. Hence, ω_i is the number of coordinates function f_i depends on – a measure of sparsity of the data[3]. It can be shown that f has a 1-Lipschitz gradient with respect to the weighted Euclidean norm with weights $\{v_j\}$ ([3, Theorem 1]). Hence, we sample coordinate j proportionally to this weight v_j. Note that p_{ij} is the joint probability of choosing the pair (i, j).

Having sampled coordinate j and function index i, we compute two partial derivatives: $\nabla_j f_i(x_k)$ and $\nabla_j f_i(y_{k,t})$ (we compressed the notation here by writing $\nabla_j f_i(x)$ instead of $\langle \nabla f_i(x), e_j \rangle$), and combine these with the pre-computed value $\nabla_j f(x_k)$ to form an update of the form

$$y_{k,t+1} \leftarrow y_{k,t} - hp_j^{-1} G e_j,$$

where G depends on i, j, x_k and $y_{k,t}$. Note that only a single coordinate of $y_{k,t}$ is updated. It can be easily observed that $\mathbf{E}_i[G \mid j] = \nabla_j f(y_{k,t})$. Hence, G is an unbiased estimate of the j^{th} partial derivative of f at $y_{k,t}$. Moreover,

$$\mathbf{E}_{ij}[hp_j^{-1}Ge_j] = \mathbf{E}_j[\mathbf{E}_i[hp_j^{-1}Ge_j \mid j]] = h\mathbf{E}_j[p_j^{-1}e_j \mathbf{E}_i[G \mid j]] = h\mathbf{E}_j[p_j^{-1}e_j \nabla_j f(y_{k,t})] = h\nabla f(y_{k,t}),$$

and hence the update step is an unbiased estimate of $\nabla f(y_{k,t})$ (scaled by h).

Special cases. In the case when $n = 1$, S2CD reduces to a stochastic CD algorithm with importance sampling[4] as studied in [8, 12], but written with many redundant computations. Indeed, the method in this case does not require the x_k iterates, and instead takes on the form: $y_{0,t+1} \leftarrow y_{0,t} - hp_j^{-1}\nabla_j f(y_{0,t})e_j$, where $p_j = L_{1j}/\sum_j L_{1j}$.

It is possible to extend the S2CD algorithm and results to the case when coordinates are replaced by (nonoverlapping) blocks of coordinates, as in [12]—we did not do it here for the sake simplicity. In such a setting, we would obtain semi-stochastic block coordinate descent. In the special case with all variables forming a single block, the algorithm reduces to the S2GD method described in [6], but with nonuniform probabilities for the choice of i—proportional to the Lipschitz constants of the gradient of the functions f_i (this is also studied in [18]). As in [18], the complexity result then depends on the average of the Lipschitz constants (see next section).

[2]In S2CD, as presented, coordinates j is selected first, and then function i is selected, according to a distribution conditioned on the choice of j. However, one could equivalently sample (i, j) with joint probability p_{ij}. We opted for the sequential sampling for clarity of presentation purposes.

[3]The quantity $\omega := \max_i \omega_i$ (degree of partial separability of f) was used in the analysis of a large class of randomized parallel coordinate descent methods in [9]. The more informative quantities $\{\omega_i\}$ appear in the analysis of parallel/distributed/mini-batch coordinate descent methods [10, 3, 2].

[4]A parallel CD method in which every subset of coordinates can be assigned a different probability of being chosen/updated was analyzed in [11].

Note that the algorithm, as presented, assumes the knowledge of μ. We have done this for simplicity of exposition: the method works also if μ is replaced by some lower bound in the method, which can be set to 0 (see [6]). The change to the complexity results will be only minor and all our conclusions hold. Likewise, it is possible to give a $O(1/\epsilon)$ complexity result in the non-strongly convex case f.

3 Complexity Result

In this section, we state and describe our complexity result. An important step in our analysis is to show that the (unbiased) estimator $p_j^{-1}Ge_j$ of $\nabla f(y_{k,t})$ has low variance. To this end, we show that

$$\mathbf{E}_{ij}[\|p_j^{-1}Ge_j\|^2] \leq 4\hat{L}\left(f(y_{k,t}) - f(x_*)\right) + 4\hat{L}\left(f(x_k) - f(x_*)\right),$$

where

$$\hat{L} := \tfrac{1}{n}\textstyle\sum_{j=1}^d v_j \overset{(4)}{=} \tfrac{1}{n}\sum_{j=1}^d \sum_{i=1}^n \omega_i L_{ij}. \tag{5}$$

Note that as $y_{k,t} \to x_*$ and $x_k \to x_*$, the bound decreases to zero. This is the main feature of modern fast stochastic gradient methods: the gradient estimate has diminishing variance. Note that the standard SGD method does not have this property: indeed, there is no reason for $\mathbf{E}_i\|\nabla f_i(x)\|^2$ to be small even if $x = x_*$. Once the above inequality is established, the analysis follows similar steps as in [6, 5] and we obtain the following result (cf [6, Thm 4]):

Theorem 1 (Complexity of S2CD)**.** *If $0 < h < 1/(2\hat{L})$, then for all $k \geq 0$ we have:*

$$\mathbf{E}[f(x_{k+1}) - f(x_*)] \leq \left(\frac{(1-\mu h)^m}{(1-(1-\mu h)^m)(1-2\hat{L}h)} + \frac{2\hat{L}h}{1-2\hat{L}h}\right)\mathbf{E}[f(x_k) - f(x_*)].$$

By analyzing the above result (one can follow the steps in [6, Theorem 6]), we get the following useful corollary:

Corollary 2. *Fix the number of epochs $k \geq 1$, error tolerance $\epsilon \in (0,1)$ and let $\Delta := \epsilon^{1/k}$ and $\hat{\kappa} := \hat{L}/\mu$. If we run Algorithm 1 with stepsize h and m set as*

$$h = \frac{\Delta}{(4+2\Delta)\hat{L}}, \qquad m \geq \left(\frac{4}{\Delta}+2\right)\log\left(\frac{2}{\Delta}+2\right)\hat{\kappa}, \tag{6}$$

then $\mathbf{E}[f(x_k) - f(x_*)] \leq \epsilon(f(x_0) - f(x_*))$. In particular, for $k = \lceil\log(1/\epsilon)\rceil$ we have $\frac{1}{\Delta} \leq \exp(1)$, and we can pick

$$k = \lceil\log(1/\epsilon)\rceil, \qquad h = \frac{\Delta}{(4+2\Delta)\hat{L}} \approx \frac{1}{(4\exp(1)+2)\hat{L}} \approx \frac{1}{12.87\hat{L}}, \qquad m \geq 26\hat{\kappa}. \tag{7}$$

If we run S2CD with the parameters set as in (7), then in each epoch the gradient of f is evaluated once (this is equivalent to n evaluations of ∇f_i), and the partial derivative of some function f_i is evaluated $2m \approx 52\hat{\kappa} = O(\hat{\kappa})$ times. If we let C_{grad} be the average cost of evaluating the gradient ∇f_i and C_{pd} be the average cost of evaluating the partial derivative $\nabla_j f_i$, then the total work of S2CD can be written as

$$(n\mathcal{C}_{grad} + m\mathcal{C}_{pd})k \overset{(7)}{=} \mathcal{O}\left((n\mathcal{C}_{grad} + \hat{\kappa}\mathcal{C}_{pd})\log(1/\epsilon)\right), \tag{8}$$

The complexity results of methods such as S2GD/SVRG [6, 5, 18] and SAG/SAGA [14, 1]—in a similar but not identical setup to ours (these papers assume f_i to be L_i-smooth)—can be written in a similar form:

$$\mathcal{O}\left((n\mathcal{C}_{grad} + \kappa\mathcal{C}_{grad})\log(1/\epsilon)\right), \tag{9}$$

where $\kappa = L/\mu$ and either $L = L_{max} := \max_i L_i$ ([14, 5, 6, 1]), or $L = L_{avg} := \frac{1}{n}\sum_i L_i$ ([18]). The difference between our result (8) and existing results (9) is in the term $\hat{\kappa}\mathcal{C}_{pd}$ – previous results have $\kappa\mathcal{C}_{grad}$ in that place. This difference constitutes a trade-off: while $\hat{\kappa} \geq \kappa$ (we comment on this below), we clearly have $\mathcal{C}_{pd} \leq \mathcal{C}_{grad}$. The comparison of the quantities $\kappa\mathcal{C}_{grad}$ and $\hat{\kappa}\mathcal{C}_{pd}$ is not straightforward and is problem dependent.

Let us now comment how do the condition numbers $\hat{\kappa}$ and $\kappa_{avg} = L_{avg}/\mu$ compare. It can be show that (see [12]) $L_i \leq \sum_{j=1}^d L_{ij}$ and, moreover, this inequality can be tight. Since $\omega_i \geq 1$ for all i, we have

$$\hat{\kappa} = \frac{\hat{L}}{\mu} \overset{(5)}{=} \frac{1}{\mu n}\sum_{j=1}^d\sum_{i=1}^n \omega_i L_{ij} \geq \frac{1}{\mu n}\sum_{i=1}^n\sum_{j=1}^d L_{ij} \geq \frac{1}{\mu n}\sum_{i=1}^n L_i = \frac{L_{avg}}{\mu} = \kappa_{avg}.$$

Finally, $\hat{\kappa}$ can be smaller or larger than $\kappa_{max} := L_{max}/\mu$.

References

[1] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. arXiv preprint arXiv:1407.0202, 2014.

[2] Olivier Fercoq, Zheng Qu, Peter Richtárik, and Martin Takáč. Fast distributed coordinate descent for non-strongly convex losses. In IEEE Workshop on Machine Learning for Signal Processing, 2014.

[3] Olivier Fercoq and Peter Richtárik. Accelerated, parallel and proximal coordinate descent. arXiv preprint arXiv:1312.5799, 2013.

[4] Olivier Fercoq and Peter Richtárik. Smooth minimization of nonsmooth functions with parallel coordinate descent methods. arXiv preprint arXiv:1309.5885, 2013.

[5] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in Neural Information Processing Systems, pages 315–323, 2013.

[6] Jakub Konečný and Peter Richtárik. Semi-stochastic gradient descent methods. arXiv preprint arXiv:1312.1666, 2013.

[7] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. SIAM Journal on Optimization, 19(4):1574–1609, 2009.

[8] Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM Journal on Optimization, 22(2):341–362, 2012.

[9] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. arXiv preprint arXiv:1212.0873, 2012.

[10] Peter Richtárik and Martin Takáč. Distributed coordinate descent for learning with big data. arXiv preprint arXiv:1310.2059, 2013.

[11] Peter Richtárik and Martin Takáč. On optimal probabilities in stochastic coordinate descent methods. arXiv preprint arXiv:1310.3438, 2013.

[12] Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. Mathematical Programming, 144(1-2):1–38, 2014.

[13] Herbert Robbins and Sutton Monro. A stochastic approximation method. The Annals of Mathematical Statistics, pages 400–407, 1951.

[14] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. arXiv preprint arXiv:1309.2388, 2013.

[15] Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. arXiv preprint arXiv:1309.2375, 2013.

[16] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. The Journal of Machine Learning Research, 14(1):567–599, 2013.

[17] Martin Takáč, Avleen Bijral, Peter Richtárik, and Nathan Srebro. Minibatch primal and dual methods for support vector machines. In Proceedings of the 30th International Conference on Machine Learning, 2013.

[18] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. arXiv preprint arXiv:1403.4699, 2014.

[19] Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In Proceedings of the 21st International Conference on Machine Learning, page 116. ACM, 2004.

[20] Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling. arXiv preprint arXiv:1401.2753v1, 2014.