Primal Method for ERM with Flexible Mini-batching Schemes and Non-convex Losses*

Dominik Csiba † Peter Richtárik ‡

June 7, 2015

Abstract

In this work we develop a new algorithm for regularized empirical risk minimization. Our method extends recent techniques of Shalev-Shwartz [02/2015], which enable a dual-free analysis of SDCA, to arbitrary mini-batching schemes. Moreover, our method is able to better utilize the information in the data defining the ERM problem. For convex loss functions, our complexity results match those of QUARTZ, which is a primal-dual method also allowing for arbitrary mini-batching schemes. The advantage of a dual-free analysis comes from the fact that it guarantees convergence even for non-convex loss functions, as long as the average loss is convex. We illustrate through experiments the utility of being able to design arbitrary mini-batching schemes.

1 Introduction

Empirical risk minimization (ERM) is a very successful and immensely popular paradigm in machine learning, used to train a variety of prediction and classification models. Given examples $A_1, \dots, A_n \in \mathbb{R}^{d\times m}$, loss functions $\phi_1, \dots, \phi_n : \mathbb{R}^m \to \mathbb{R}$ and a regularization parameter $\lambda > 0$, the L2-regularized ERM problem is an optimization problem of the form

$$\min_{w\in\mathbb{R}^d} \left[P(w) := \frac{1}{n}\sum_{i=1}^{n} \phi_i(A_i^\top w) + \frac{\lambda}{2}\|w\|^2 \right] \tag{1}$$

Throughout the paper we shall assume that for each i, the loss function ϕ_i is l_i-smooth with $l_i > 0$. That is, for all $x, y \in \mathbb{R}^m$ and all $i \in [n] := \{1, 2, \dots, n\}$, we have

$$\|\nabla\phi_i(x) - \nabla\phi_i(y)\| \le l_i\|x - y\|. \tag{2}$$

Further, let $L_1, \dots, L_n > 0$ be constants for which the inequality

$$\|\nabla\phi_i(A_i^\top w) - \nabla\phi_i(A_i^\top z)\| \le L_i\|w - z\| \tag{3}$$

holds for all $w, z \in \mathbb{R}^d$ and all i and let $L := \max_i L_i$. Note that we can always bound $L_i \le l_i\|A_i\|$. However, L_i can be better (smaller) than $l_i\|A_i\|$.

*The authors acknowledge support from the EPSRC Grant EP/K02325X/1, Accelerated Coordinate Descent Methods for Big Data Optimization.

†School of Mathematics, The University of Edinburgh, United Kingdom (e-mail: cdominik@gmail.com)

‡School of Mathematics, The University of Edinburgh, United Kingdom (e-mail: peter.richtarik@ed.ac.uk)

1.1 Background

In the last few years, a lot of research effort was put into designing new efficient algorithms for solving this problem (and some of its modifications). The frenzy of activity was motivated by the realization that SGD [1], not so long ago considered the state-of-the-art method for ERM, was far from being optimal, and that new ideas can lead to algorithms which are far superior to SGD in both theory and practice. The methods that belong to this category include SAG [2], SDCA [3], SVRG [4], S2GD [5], mS2GD [6], SAGA [7], S2CD [8], QUARTZ [9], ASDCA [10], prox-SDCA [11], IPROX-SDCA [12], A-PROX-SDCA [13], AdaSDCA [14], SDNA [15]. Methods analyzed for arbitrary mini-batching schemes include NSync [16], ALPHA [17] and QUARTZ [9].

In order to find an ϵ-solution in expectation, state of the art (non-accelerated) methods for solving (1) only need

$$O((n+\kappa)\log(1/\epsilon))$$

steps, where each step involves the computation of the gradient $\nabla\phi_i(A_i^\top w)$ for some randomly selected example i. The quantity κ is the condition number. Typically one has $\kappa = \frac{\max_i l_i\|A_i\|^2}{\lambda}$ for methods picking i uniformly at random, and $\kappa = \frac{\sum_i l_i\|A_i\|^2}{n\lambda}$ for methods picking i using a carefully designed data-dependent importance sampling. Computation of such a gradient typically involves work which is equivalent to reading the example A_i, that is, $O(nnz(A_i)) \leq dm$ arithmetic operations.

1.2 Contributions

In this work we develop a new algorithm for the L2-regularized ERM problem (1). Our method extends a technique recently introduced by Shalev-Shwartz [18], which enables a dual-free analysis of SDCA, to *arbitrary mini-batching schemes.* That is, our method works at each iteration with a random subset of examples, chosen in an i.i.d. fashion from an arbitrary distribution. Such flexible schemes are useful for various reasons, including i) the development of distributed or robust variants of the method, ii) design of importance sampling for improving the complexity rate, iii) design of a sampling which is aimed at obtaining efficiencies elsewhere, such us utilizing NUMA (non-uniform memory access) architectures, and iv) streamlining and speeding up the processing of each mini-batch by means of assigning to each processor approximately even workload so as to reduce idle time (we do experiments with the latter setup).

In comparison with [18], our method is able to better *utilize the information in the data examples* $A_1,\ldots,A_n$, leading to a better *data-dependent* bound. For convex loss functions, our complexity results match those of QUARTZ [9] in terms of the rate (the logarithmic factors differ). QUARTZ is a primal-dual method also allowing for arbitrary mini-batching schemes. However, while [9] only characterize the decay of expected risk, we also give bounds for the *sequence of iterates*. In particular, we show that for convex loss functions, our method enjoys the rate (Theorem 2)

$$\max_i\left(\frac{1}{p_i}+\frac{l_i v_i}{\lambda p_i n}\right)\log\left(\frac{(L+\lambda)E^{(0)}}{\lambda\epsilon}\right),$$

where p_i is the probability that coordinate i is updated in an iteration, $v_1,\ldots,v_n>0$ are certain "stepsize" parameters of the method associated with the sampling and data (see (6)), and $E^{(0)}$ is a constant depending on the starting point. For instance, in the special case picking a single example at a time uniformly at random, we have $p_i = 1/n$ and $v_i = \|A_i\|^2$, whereby we obtain one

of the $O(n+\kappa)\log(1/\epsilon)$ rates mentioned above. The other rate can be recovered using importance sampling.

The advantage of a dual-free analysis comes from the fact that it guarantees convergence even for *non-convex* loss functions, as long as the average loss is convex. This is a step toward understanding non-convex models. In particular, we show that for non-convex loss functions, our method enjoys the rate (Theorem 1)

$$\max_i \left(\frac{1}{p_i} + \frac{L_i^2 v_i}{\lambda^2 p_i n}\right) \log\left(\frac{(L+\lambda)D^{(0)}}{\lambda\epsilon}\right),$$

where $D^{(0)}$ is a constant depending on the starting point.

Finally, we illustrate through experiments with "chunking"—a simple load balancing technique—the utility of being able to design arbitrary mini-batching schemes.

2 Algorithm

We shall now describe the method (Algorithm 1).

Algorithm 1 dfSDCA: Dual-Free SDCA with Arbitrary Sampling

Parameters: Sampling $\hat{S}$, stepsize θ
Initialization $\alpha_1^{(0)},\dots,\alpha_n^{(0)} \in \mathbb{R}^m$, set $w^{(0)} = \frac{1}{\lambda n}\sum_{i=1}^n A_i\alpha_i^{(0)}$, $p_i = \mathbf{Prob}(i \in \hat{S})$
for $t \geq 1$ **do**
  Sample a set S_t according to $\hat{S}$
  for $i \in S_t$ **do**
    $\alpha_i^{(t)} = \alpha_i^{(t-1)} - \theta p_i^{-1}(\nabla\phi_i(A_i^\top w^{(t-1)}) + \alpha_i^{(t-1)})$
  $w^{(t)} = w^{(t-1)} - \sum_{i\in S_t}\theta(n\lambda p_i)^{-1}A_i(\nabla\phi_i(A_i^\top w^{(t-1)}) + \alpha_i^{(t-1)})$

The method encodes a family of algorithms, depending on the choice of the sampling $\hat{S}$, which encodes a particular *mini-batching scheme*. Formally, a sampling $\hat{S}$ is a set-valued random variable with values being the subsets of $[n]$, i.e., subsets of examples. In this paper, we use the terms "mini-batching scheme" and "sampling" interchangeably. A sampling is defined by the collection of probabilities $\mathbf{Prob}(S)$ assigned to every subset $S \subseteq [n]$ of the examples.

The method maintains n vectors $\alpha_i \in \mathbb{R}^m$ and a vector $w \in \mathbb{R}^d$. At the beginning of step t, we have $\alpha_i^{(t-1)}$ for all i and $w^{(t-1)}$ computed and stored in memory. We then pick a random subset S_t of the examples, according to the mini-batching scheme, and update variables α_i for $i \in S_t$, based on the computation of the gradients $\nabla\phi_i(A_i^\top w^{(t-1)})$ for $i \in S_t$. This is followed by an update of the vector w, which is performed so as to maintain the relation

$$w^{(t)} = \frac{1}{\lambda n}\sum_i A_i\alpha_i^{(t)}. \tag{4}$$

This relation is maintained for the following reason. If w^* is the optimal solution to (1), then

$$0 = \nabla P(w^*) = \frac{1}{n}\sum_{i=1}^n A_i\nabla\phi_i(A_i^\top w^*) + \lambda w^*, \tag{5}$$

and hence $w^* = \frac{1}{\lambda n}\sum_{i=1}^n A_i\alpha_i^*$, where $\alpha_i^* := -\nabla\phi_i(A_i^\top w^*)$. So, if we believe that the variables α_i converge to $-\nabla\phi_i(A_i^\top w^*)$, it indeed does make sense to maintain (4). Why should we believe this? This is where the specific update of the "dual variables" α_i comes from: α_i is set a convex combination of its previous value and our best estimate so far of $-\nabla\phi_i(A_i^\top w^*)$, namely, $-\nabla\phi_i(A_i^\top w^{(t-1)})$. Indeed, the update can be written as

$$\alpha_i^{(t)} = (1-\theta p_i^{-1})\alpha_i^{(t-1)} + \theta p_i^{-1}(-\nabla\phi_i(A_i^\top w^{(t-1)})).$$

Why does *this* make sense? Because we believe that $w^{(t-1)}$ converges to w^*. Admittedly, this reasoning is somewhat "circular". However, a better word to describe this reasoning would be: "iterative".

3 Main Results

Let $p_i := \mathbb{P}(i \in \hat{S})$. We assume the knowledge of parameters $v_1, \ldots, v_n > 0$ for which

$$\mathbf{E}\left[\left\|\sum_{i\in\hat{S}} A_i h_i\right\|^2\right] \leq \sum_{i=1}^n p_i v_i \|h_i\|^2. \tag{6}$$

Tight and easily computable formulas for such parameters can be found in [19]. For instance, whenever $\mathbf{Prob}(|\hat{S}| \leq \tau) = 1$, inequality (6) holds with $v_i = \tau\|A_i\|^2$.

To simplify the exposure, we will write

$$B^{(t)} \overset{\text{def}}{=} \|w^{(t)} - w^*\|^2, \qquad C_i^{(t)} \overset{\text{def}}{=} \|\alpha_i^{(t)} - \alpha_i^*\|^2, \qquad i = 1, 2, \ldots, n. \tag{7}$$

3.1 Non-convex loss functions

Our result will be expressed in terms of the decay of the potential $D^{(t)} \overset{\text{def}}{=} \frac{\lambda}{2}B^{(t)} + \frac{\lambda}{2n}\sum_{i=1}^n \frac{1}{L_i^2}C_i^{(t)}$, where $B_i^{(t)}$ and $C^{(t)}$ are defined in (7).

Theorem 1. *Assume that the average loss function, $\frac{1}{n}\sum_{i=1}^n \phi_i$, is convex. If (3) holds and we let*

$$\theta \leq \min_i \frac{p_i n\lambda^2}{L_i^2 v_i + n\lambda^2}, \tag{8}$$

then the for $t \geq 0$ the potential $D^{(t)}$ decays exponentially to zero as

$$\mathbf{E}\left[D^{(t)}\right] \leq e^{-\theta t} D^{(0)}. \tag{9}$$

Moreover, if we set θ equal to the upper bound in (8), then

$$T \geq \max_i \left(\frac{1}{p_i} + \frac{L_i^2 v_i}{\lambda^2 p_i n}\right)\log\left(\frac{(L+\lambda)D^{(0)}}{\lambda\epsilon}\right) \quad \Rightarrow \quad \mathbf{E}[P(w^{(T)}) - P(w^*)] \leq \epsilon.$$

3.2 Convex loss functions

Our result will be expressed in terms of the decay of the potential $E^{(t)} \stackrel{\text{def}}{=} \frac{\lambda}{2}B^{(t)} + \frac{1}{2n}\sum_{i=1}^{n}\frac{1}{l_i}C_i^{(t)}$, where $B_i^{(t)}$ and $C^{(t)}$ are defined in (7).

Theorem 2. *Assume that all loss functions $\{\phi_i\}$ are convex and satisfy (2). If we run Algorithm 1 with parameter θ satisfying the inequality*

$$\theta \leq \min_i \frac{p_i n\lambda}{l_i v_i + n\lambda}, \tag{10}$$

then the for $t \geq 0$ the potential $E^{(t)}$ decays exponentially to zero as

$$\mathbf{E}\left[E^{(t)}\right] \leq e^{-\theta t}E^{(0)}. \tag{11}$$

Moreover, if we set θ equal to the upper bound in (10), then

$$T \geq \max_i \left(\frac{1}{p_i} + \frac{l_i v_i}{\lambda p_i n}\right)\log\left(\frac{(L+\lambda)E^{(0)}}{\lambda\epsilon}\right) \quad \Rightarrow \quad \mathbf{E}[P(w^{(T)}) - P(w^*)] \leq \epsilon$$

The rate, θ, precisely matches that of the QUARTZ algorithm [9]. Quartz is the only other method for ERM which has been analyzed for an arbitrary mini-batching scheme. Our algorithm is dual-free, and as we have seen above, allows for an analysis covering the case of non-convex loss functions.

4 Chunking

In this section we illustrate one use of the ability of our method to work with an arbitrary mini-batching scheme. Further examples include the ability to design distributed variants of the method [20], or the use of importance/adaptive sampling to lower the number of iterations [21, 12, 9, 14].

One marked disadvantage of standard mini-batching ("choose a subset of examples, uniformly at random") used in the context of parallel processing on multicore processors is the fact that in a synchronous implementation there is a loss of efficiency due to the fact that the computation time of $\nabla\phi(A_i^\top w)$ may differ through i. This is caused by the data examples having varying degree of sparsity. We hence introduce a new sampling which mitigates this issue.

Chunks: Choose sets $G_1, \ldots, G_k \subset [n]$, such that $\cup_{i=1}^{k} G_i = [n]$ and $G_i \cap G_j = \emptyset\ \forall i, j$ and $\psi(i) := \sum_{j\in G_i} \text{nnz}(A_j)$ is similar for every i, i.e. $\psi(1) \approx \cdots \approx \psi(k)$. Instead of sampling τ coordinates we propose a new sampling, which on each iteration t samples τ sets $G_{(1)}^{(t)}, \ldots, G_{(\tau)}^{(t)}$ out of $G_1, \ldots, G_k$ and uses coordinates $i \in \cup_{i=1}^{\tau} G_{(i)}^{(t)}$ as the sampled set. We assign each core one of the sets $G_{(i)}^{(t)}$ for parallel computation. The advantage of this sampling lies in the fact, that the load of computing $\nabla\phi(A_i^\top w)$ for all $i \in G_j$ is similar for all $j \in [k]$. Hence, using this sampling we minimize the waiting time of processors.

Algorithm 2 Naive Chunks

Parameters: vector of nnz u
Initialization $n = \text{length}(u)$; Empty vector g and s of length n; $m = \max(u)$
$g[1] = 1, \quad s[1] = u[1], \quad i = 1$
for $t = 2 : n$ **do**
  if $g[i] + u[t] \leq m$ **then**
    $g[i] = g[i] + 1$, $s[i] = s[i] + u[t]$
  else
    $i = i + 1$, $g[i] = 1$, $s[i] = u[t]$

How to choose $G_1, \ldots, G_k$: We introduce the following algorithm:

The algorithms returns the partition of $[n]$ into $G_1, \ldots, G_k$ in a sense, that the first $g[1]$ coordinates belong to G_1, next $g[2]$ coordinates belong to G_2 and so on. The main advantage of this approach is, that it makes a preprocessing step on the dataset which takes just one pass through the data. On Figure 1a through Figure 1f we show the impact of Algorithm 2 on the probability of the waiting time of a single core, which we measure by the difference

$$\max_{i \in S_t} \{\text{nnz}(A_i)\} - \frac{1}{\tau} \sum_{i \in S_t} \text{nnz}(A_i)$$

and

$$\max_{i \in [\tau]} \{\text{nnz}(G_{(i)}^{(t)})\} - \frac{1}{\tau} \sum_{i=1}^{\tau} \text{nnz}(G_{(i)}^{(t)})$$

for the initial and preprocessed dataset respectively. We can observe, that the waiting time is smaller using the preprocessing.

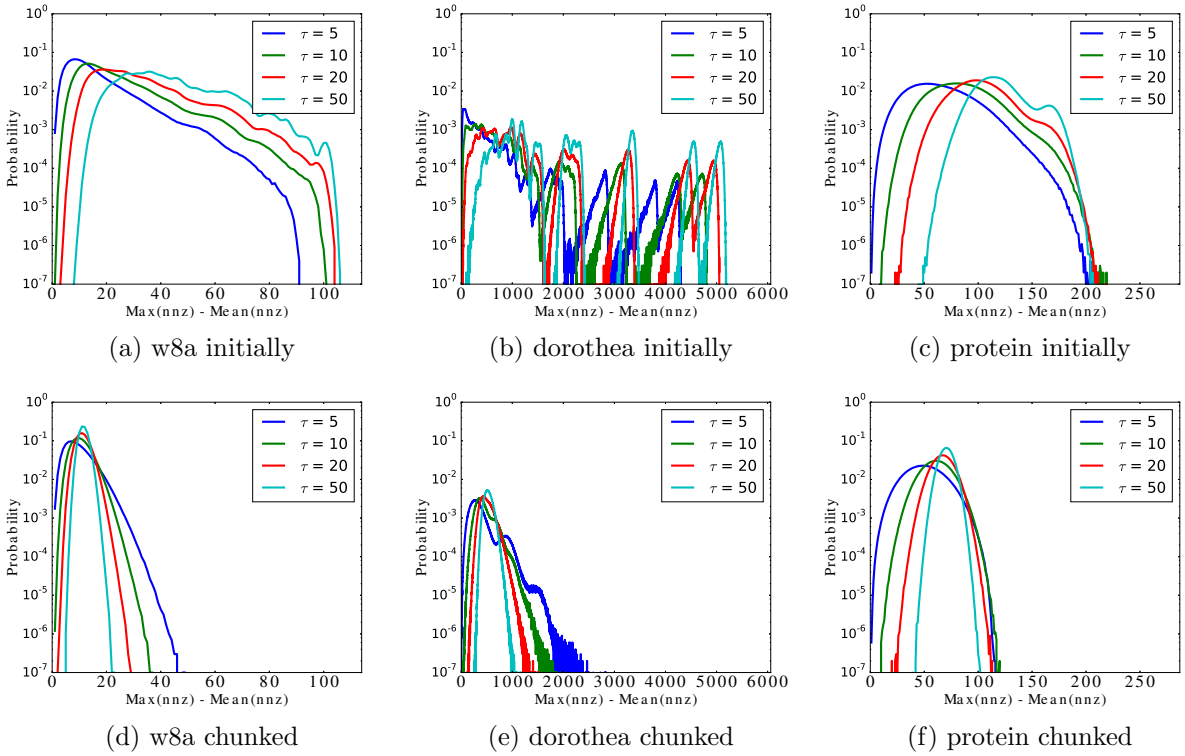
5 Experiments

In all our experiments we used logistic regression. We normalized the datasets so that $\max_i \|A_i\| = 1$, and fixed $\lambda = 1/n$. The datasets used for experiments are summarized in Table 1.

Dataset	#samples	#features	sparsity
w8a	49,749	300	3.8%
dorothea	800	100,000	0.9%
protein	17,766	358	29%
rcv1	20,242	47,237	0.2%
cov1	581,012	54	22%

Table 1: Datasets used in the experiments.

Experiment 1. In Figure 2a we compared the performance of Algorithm 1 with uniform serial sampling against state of the art algorithms such as SGD [1], SAG[2] and S2GD [5] in number of epochs. The real running time of the algorithms was 0.46s for S2GD, 0.79s for SAG, 0.47s for SDCA and 0.58s for SGD. In Figure 2b we show the convergence rate for different regularization parameters λ. In Figure 2c we show convergence rates for different serial samplings: uniform,



(a) w8a initially (b) dorothea initially (c) protein initially

(d) w8a chunked (e) dorothea chunked (f) protein chunked

Figure 1: Distribution of the difference between the maximum number of nonzeros processed by a single core and the mean of all nonzeros processed by each core. This difference shows us, how much time is wasted per core waiting on the slowest core to finish its task, therefore smaller numbers are better. The first row corresponds to the initial distribution while the second row shows the distribution after using Algorithm 2.

importance [12] and also 4 different randomly generated serial samplings. These samplings were generated in a controlled manner, such that *random* c has $(\max_i\ p_i)/(\min_i\ p_i) < c$. All of these samplings have linear convergence as shown in the theory.

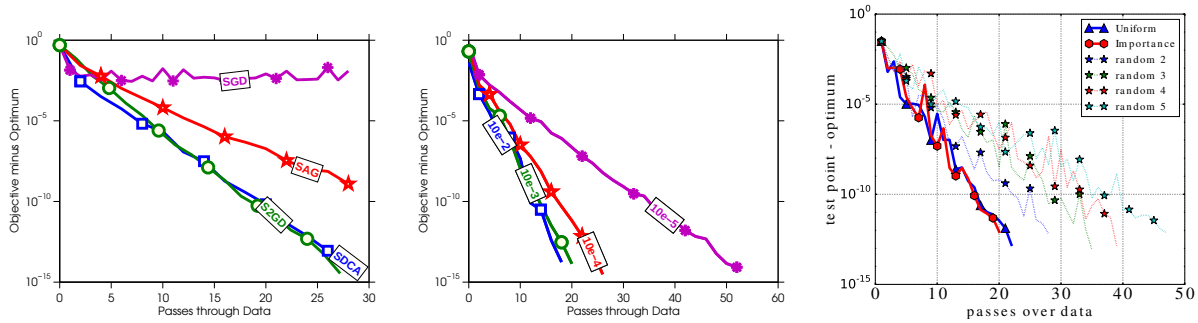
Experiment 2: New sampling vs. old sampling. In Figure 3a through Figure 3l we compare the performance of a standard parallel sampling against sampling of blocks $G_1, \ldots, G_k$ output by Algorithm 2. In each iteration we measure the time by

$$\max_{i \in S_t} \{\text{nnz}(A_i)\}$$

and

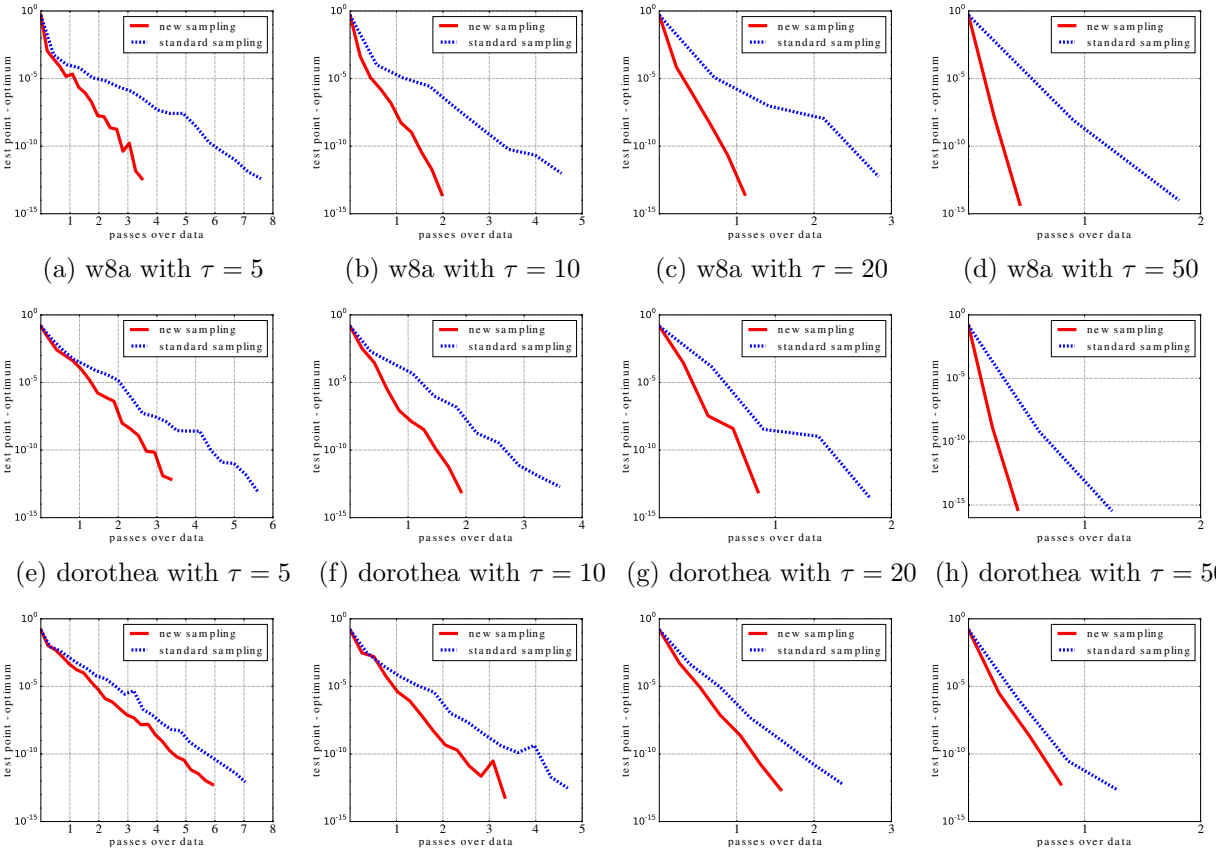
$$\max_{i \in [\tau]} \{\text{nnz}(G_{(i)})\}$$

for the standard and new sampling respectively. This way we measure only the computations done by the core which is going to finish the last in each iteration, and consider the number of multiplications with nonzero entries of the data matrix as a proxy for time.



(a) rcv1, state of the art (b) rcv1, different λ (c) cov1, various samplings

Figure 2: LEFT: Comparison of SDCA with other state of the art methods. MIDDLE: SDCA for various values of λ. RIGHT: SDCA run with various samplings $\hat{S}$.



(a) w8a with $\tau = 5$ (b) w8a with $\tau = 10$ (c) w8a with $\tau = 20$ (d) w8a with $\tau = 50$

(e) dorothea with $\tau = 5$ (f) dorothea with $\tau = 10$ (g) dorothea with $\tau = 20$ (h) dorothea with $\tau = 50$

(i) protein with $\tau = 5$ (j) protein with $\tau = 10$ (k) protein with $\tau = 20$ (l) protein with $\tau = 50$

Figure 3: Logistic regression with $\lambda = 1/n$. Comparison between new and standard sampling with fine-tuned stepsizes for different values of τ.

6 Proofs

As a first approximation, our proof is an extension of the proof of Shalev-Shwartz [18] to accommodate an arbitrary sampling [16, 17, 9, 15]. For all i and t we let $u_i^{(t-1)} = -\nabla\phi_i(A_i^\top w^{(t)})$ and $z_i^{(t-1)} = \alpha_i^{(t-1)} - u_i^{(t-1)}$. We will use the following lemma.

Lemma 3 (Evolution of $C_i^{(t)}$ and $B^{(t)}$)**.** *For a fixed iteration $t$and all i we have:*

$$\mathbf{E}_{\hat{S}}\left[C_i^{(t-1)} - C_i^{(t)}\right] = \theta\left[\|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|u_i^{(t-1)} - \alpha_i^*\|^2 + (1-\theta p_i^{-1})\|z_i^{(t-1)}\|^2\right] \tag{12}$$

$$\mathbf{E}_{\hat{S}}\left[B^{(t-1)} - B^{(t)}\right] \geq \frac{2\theta}{\lambda}(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) - \frac{\theta^2}{n^2\lambda^2}\sum_{i=1}^n \frac{v_i}{p_i}\|z_i^{(t-1)}\|^2. \tag{13}$$

Proof. It follows that for $i \in S_t$ using the definition (7) we have

$$\begin{aligned}
C_i^{(t-1)} - C_i^{(t)} &\overset{(7)}{=} \|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|\alpha_i^{(t)} - \alpha_i^*\|^2 \\
&= \|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|(1-\theta p_i^{-1})(\alpha_i^{(t-1)} - \alpha_i^*) + \theta p_i^{-1}(u_i^{(t-1)} - \alpha_i^*)\|^2 \\
&= \|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - (1-\theta p_i^{-1})\|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \theta p_i^{-1}\|u_i^{(t-1)} - \alpha_i^*\|^2 \\
&\quad +\theta p_i^{-1}(1-\theta p_i^{-1})\|\alpha_i^{(t-1)} - u_i^{(t-1)}\|^2 \\
&= \theta p_i^{-1}\left[\|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|u_i^{(t-1)} - \alpha_i^*\|^2 + (1-\theta p_i^{-1})\|z_i^{(t-1)}\|^2\right]
\end{aligned}$$

and for $i \notin S_t$ we have $C_i^{(t-1)} - C_i^{(t)} = 0$. Taking the expectation over S_t we get the result.

For the second potential we get

$$\begin{aligned}
B^{(t-1)} - B^{(t)} &\overset{(7)}{=} \|w^{(t-1)} - w^*\|^2 - \|w^{(t)} - w^*\|^2 \\
&= \frac{2\theta}{n\lambda}\sum_{i\in S_t} p_i^{-1}(w^{(t-1)} - w^*)^\top A_i z_i^{(t-1)} - \frac{\theta^2}{n^2\lambda^2}\|\sum_{i\in S_t} p_i^{-1} A_i z_i^{(t-1)}\|^2.
\end{aligned}$$

Taking the expectation over S_t, using inequality (6), and noting that

$$\frac{1}{n}\sum_{i=1}^n A_i z_i^{(t-1)} = \frac{1}{n}\sum_{i=1}^n A_i \nabla\phi(A_i^\top w^{(t-1)}) + \lambda w^{(t-1)} = \nabla P(w^{(t-1)}), \tag{14}$$

we get

$$\begin{aligned}
\mathbf{E}\left[B^{(t-1)} - B^{(t)}\right] &= \frac{2\theta}{n\lambda}\sum_{i=1}^n (w^{(t-1)} - w^*)^\top A_i z_i^{(t-1)} - \frac{\theta^2}{n^2\lambda^2}\mathbf{E}\left[\|\sum_{i\in S_t} A_i(p_i^{-1} z_i^{(t-1)})\|^2\right] \\
&\overset{(6)}{\geq} \frac{2\theta}{n\lambda}\sum_{i=1}^n (w^{(t-1)} - w^*)^\top A_i z_i^{(t-1)} - \frac{\theta^2}{n^2\lambda^2}\sum_{i=1}^n p_i v_i \|p_i^{-1} z_i^{(t-1)}\|^2 \\
&\overset{(14)}{=} \frac{2\theta}{\lambda}(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) - \frac{\theta^2}{n^2\lambda^2}\sum_{i=1}^n \frac{v_i}{p_i}\|z_i^{(t-1)}\|^2 \quad \square
\end{aligned}$$

6.1 Proof of Theorem 1 (nonconvex case)

Combining (12) and (13), we obtain

$$\begin{aligned}
\mathbf{E}[D^{(t-1)} - D^{(t)}] &\geq \frac{\theta\lambda}{2n}\sum_{i=1}^{n}\frac{1}{L_i^2}\left[\|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|u_i^{(t-1)} - \alpha_i^*\|^2 + (1-\theta p_i^{-1})\|z_i^{(t-1)}\|^2\right] \\
&\quad + \frac{\lambda}{2}\left[\frac{2\theta}{\lambda}(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) - \frac{\theta^2}{n^2\lambda^2}\sum_{i=1}^{n}\frac{v_i}{p_i}\|z_i^{(t-1)}\|^2\right] \\
&= \frac{\theta}{2n}\sum_{i=1}^{n}\left[\frac{\lambda}{L_i^2}\left(C_i^{(t-1)} - \|u_i^{(t-1)} - \alpha_i^*\|^2\right) + \left(\frac{\lambda(1-\theta p_i^{-1})}{L_i^2} - \frac{\theta v_i}{n\lambda p_i}\right)\|z_i^{(t-1)}\|^2\right] \\
&\quad + \theta(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) \\
&\overset{(8)}{\geq} \frac{\theta}{2n}\sum_{i=1}^{n}\frac{\lambda}{L_i^2}\left(C_i^{(t-1)} - \|u_i^{(t-1)} - \alpha_i^*\|^2\right) + \theta(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}).
\end{aligned}$$

Using (3) we have

$$\|u_i^{(t-1)} - \alpha_i^*\|^2 = \|\nabla\phi_i(A_i^\top w^{(t-1)}) \ - \ \nabla\phi_i(A_i^\top w^*)\|^2 \leq L_i^2\|w^{(t-1)} \ - \ w^*\|^2.$$

By strong convexity of P,

$$(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) \geq P(w^{(t-1)}) - P(w^*) + \frac{\lambda}{2}\|w^{(t-1)} - w^*\|^2$$

and $P(w^{(t-1)}) - P(w^*) \geq \frac{\lambda}{2}\|w^{(t-1)} - w^*\|^2$, which together yields

$$(w^{(t-1)} - w^*)^\top \nabla P(w^{(t-1)}) \geq \lambda\|w^{(t-1)} - w^*\|^2.$$

Therefore,

$$\mathbf{E}[D^{(t-1)} - D^{(t)}] \geq \theta\left[\frac{1}{n}\sum_{i=1}^{n}\frac{\lambda}{2L_i^2}C_i^{(t-1)} + \left(-\frac{\lambda}{2} + \lambda\right)B^{(t-1)}\right] = \theta D^{(t-1)}.$$

It follows that $\mathbf{E}[D^{(t)}] \leq (1-\theta)D^{(t-1)}$, and repeating this recursively we end up with $\mathbf{E}[D^{(t-1)}] \leq (1-\theta)^t D^{(0)} \leq e^{-\theta t}D^{(0)}$. This concludes the proof of the first part of Theorem 1. The second part of the proof follows by observing that P is $(L+\lambda)$-smooth, which gives $P(w) - P(w^*) \leq \frac{L+\lambda}{2}\|w - w^*\|^2$.

6.2 Convex case

For the next theorem we need an additional lemma:

Lemma 4. *Assume that ϕ_i are L_i-smooth and convex. Then, for every w,*

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{L_i}\|\nabla\phi_i(w) - \nabla\phi_i(w^*)\|^2 \leq 2\left(P(w) - P(w^*) - \frac{\lambda}{2}\|w - w^*\|^2\right) \tag{15}$$

Proof. Let $g_i(x) = \phi_i(x) - \phi_i(A_i^\top w^*) - \nabla\phi_i(A_i^\top w^*)^\top(x - A_i^\top w^*)$. Clearly, g_i is also l_i-smooth. By convexity of ϕ_i we have $g_i(x) \geq 0$ for all x. It follows that g_i satisfies $\|\nabla g_i(x)\|^2 \leq 2l_i g_i(x)$. Using the definition of g_i, we obtain

$$\begin{aligned}\|\nabla\phi_i(A_i^\top w) - \nabla\phi_i(A_i^\top w^*)\|^2 &= \|\nabla g_i(A_i^\top w)\|^2 \\ &\leq 2l_i[\phi_i(A_i^\top w) - \phi_i(A_i^\top w^*) - \nabla\phi_i(A_i^\top w^*)^\top(A_i^\top w - A_i^\top w^*)]. \quad (16)\end{aligned}$$

Summing these terms up weighted by $1/l_i$ and using (5) we get

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \frac{1}{l_i}\|\nabla\phi_i(A_i^\top w) - \nabla\phi_i(A_i^\top w^*)\|^2 &\overset{(16)}{\leq} \sum_{i=1}^n \frac{2}{n}[\phi_i(A_i^\top w) - \phi_i(A_i^\top w^*) - A_i\nabla\phi_i(A_i^\top w^*)^\top(w - w^*)] \\ &\overset{(5)}{=} 2\left[P(w) - \frac{\lambda}{2}\|w\|^2 - P(w^*) + \frac{\lambda}{2}\|w^*\|^2 + \lambda {w^*}^\top(w - w^*)\right] \\ &= 2\left[P(w) - P(w^*) - \frac{\lambda}{2}\|w - w^*\|^2\right]. \quad \square\end{aligned}$$

6.3 Proof of Theorem 2

Combining (12) and (13), we obtain

$$\begin{aligned}\mathbf{E}[E^{(t-1)} - E^{(t)}] &\geq \frac{\theta}{n}\sum_{i=1}^n \frac{1}{2l_i}\left[\|\alpha_i^{(t-1)} - \alpha_i^*\|^2 - \|u_i^{(t-1)} - \alpha_i^*\|^2 + (1 - \theta p_i^{-1})\|z_i^{(t-1)}\|^2\right] \\ &\quad + \frac{\lambda}{2}\left[\frac{2\theta}{\lambda}(w^{(t-1)} - w^*)^\top\nabla P(w^{(t-1)}) - \frac{\theta^2}{n^2\lambda^2}\sum_{i=1}^n \frac{v_i}{p_i}\|z_i^{(t-1)}\|^2\right] \\ &= \frac{\theta}{n}\sum_{i=1}^n \left[\frac{1}{2l_i}(C_i^{(t-1)} - \|u_i^{(t-1)} - \alpha_i^*\|^2) + \left(\frac{(1 - \theta p_i^{-1})}{2l_i} - \frac{\theta v_i}{2p_i\lambda n}\right)\right] \\ &\quad + \theta(w^{(t-1)} - w^*)^\top\nabla P(w^{(t-1)}) \\ &\overset{(10)}{\geq} \frac{\theta}{n}\sum_{i=1}^n \left[\frac{1}{2l_i}(C_i^{(t-1)} - \|u_i^{(t-1)} - \alpha_i^*\|^2)\right] + \theta(w^{(t-1)} - w^*)^\top\nabla P(w^{(t-1)})\end{aligned}$$

Using the convexity of P we have $P(w^*) - P(w^{(t-1)}) \geq (w^{(t-1)} - w^*)^\top\nabla P(w^{(t-1)})$ and using Lemma 4, we have

$$\begin{aligned}\mathbf{E}[E^{(t-1)} - E^{(t)}] &\overset{(15)}{\geq} \frac{\theta}{n}\sum_{i=1}^n \frac{1}{2l_i}C_i^{(t-1)} - \theta\left(P(w^{(t-1)}) - P(w^*) - \frac{\lambda}{2}\|w^{(t-1)} - w^*\|^2\right) \\ &\quad + \theta(w^{(t-1)} - w^*)^\top\nabla P(w^{(t-1)}) \\ &\geq \theta\left[\frac{1}{n}\sum_{i=1}^n \frac{1}{2l_i}C_i^{(t-1)} + \frac{\lambda}{2}B^{(t-1)}\right] \quad = \quad \theta E^{(t-1)}.\end{aligned}$$

This gives $\mathbf{E}[E^{(t)}] \leq (1-\theta)E^{(t-1)}$, which concludes the first part of the Theorem 2. The second part follows by observing, that P is $(L + \lambda)$-smooth, which gives $P(w) - P(w^*) \leq \frac{L+\lambda}{2}\|w - w^*\|^2$.

References

[1] Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 09 1951.

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

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

[4] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, 2013.

[5] Jakub Konečný and Peter Richtárik. S2GD: Semi-stochastic gradient descent methods. *arXiv:1312.1666*, 2014.

[6] Jakub Konečný, Jie Lu, Peter Richtárik, and Martin Takáč. mS2GD: Mini-batch semi-stochastic gradient descent in the proximal setting. *arXiv:1410.4744*, 2014.

[7] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014.

[8] Jakub Konečný, Zheng Qu, and Peter Richtárik. Semi-stochastic coordinate descent. *arXiv:1412.6293*, 2014.

[9] Zheng Qu, Peter Richtárik, and Tong Zhang. Randomized Dual Coordinate Ascent with Arbitrary Sampling. *arXiv:1411.5873*, 2014.

[10] Shai Shalev-Shwartz and Tong Zhang. Accelerated mini-batch stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems 26*, pages 378–385. 2013.

[11] Shai Shalev-Shwartz and Tong Zhang. Proximal stochastic dual coordinate ascent. *arXiv:1211.2717*, 2012.

[12] Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling. ICML, 2015.

[13] Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *to appear in Mathematical Programming*, 2015.

[14] Dominik Csiba, Zheng Qu, and Peter Richtárik. Stochastic dual coordinate ascent with adaptive probabilities. *ICML 2015*.

[15] Zheng Qu, Peter Richtárik, Martin Takáč, and Olivier Fercoq. Stochastic Dual Newton Ascent for empirical risk minimization. *arXiv:1502.02268*.

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

[17] Zheng Qu and Peter Richtárik. Coordinate descent methods with arbitrary sampling I: Algorithms and complexity. *arXiv:1412.8060*, 2014.

[18] Shai Shalev-Shwartz. SDCA without duality. *CoRR*, abs/1502.06177, 2015.

[19] Zheng Qu and Peter Richtárik. Coordinate Descent with Arbitrary Sampling II: Expected Separable Overapproximation. *arXiv:1412.8063*, 2014.

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

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