Distributed Mini-Batch SDCA

**Martin Takáč**\(^1\), **Peter Richtárik**\(^2\), and **Nathan Srebro**\(^3\)

\(^1\)Lehigh University \(^2\)University of Edinburgh \(^3\)Toyota Technological Institute at Chicago

ISE Technical Report 15T-007
Abstract

We present an improved analysis of mini-batched stochastic dual coordinate ascent for regularized empirical loss minimization (i.e. SVM and SVM-type objectives). Our analysis allows for flexible sampling schemes, including where data is distribute across machines, and combines a dependence on the smoothness of the loss and/or the data spread (measured through the spectral norm).
linear parallelization speedups, even when the loss is non-smooth \[2, 27\]. This parallels the SGD mini-batch analysis for non-smooth loss. But can mini-batching also be beneficial for SDCA with smooth losses and without a data-spread (spectral norm) assumptions, as with SGD? What mini-batch sizes allow for parallel speedups? In this paper we answer these questions and show that as with SGD, when the loss function is smooth, using mini-batches with SDCA yields a linear reduction in the number of iterations and thus allows for linear parallelization speedups, up to similar polynomial limits on the mini-batch size. Furthermore, we provide an analysis that combines the benefits of smoothness with the data-dependent benefits of a low spectral norm, and thus allows for even large mini-batch sizes when the loss is smooth and the data is well-spread.

Another issue that we address is the way mini-batches are sampled. Straight-forward mini-batch analysis, including previous analysis of mini-batch SDCA \[27\], assume that at each iteration we pick a mini-batch of size \(d\) uniformly at random from among all subsets of \(b\) training examples. In practice, though, data is often partitioned between \(C \leq b\) machines, and at each iteration \(b/C\) points are samples from each machine, yielding a mini-batch that is not uniformly distributed among all possible subsets (e.g. we have zero probability of using \(b\) points from the same machine as a mini-batch). Other architectural restrictions might lead to different sampling schemes. The analysis we present can be easily applied to different sampling schemes, and in particular we consider distributed sampling as described above and show that essentially the same guarantees (with minor modification) hold also for this more realistic sampling scheme.

Finally, we compare our optimization guarantees to those recently established for CoCoA+ \[14\]. CoCoA+ is an alternative dual-based distributed optimization approach, which can be viewed as including mini-batch SDCA as a special case, and going beyond SDCA to potentially more powerful optimization. At each iteration of CoCoA+, several groups of dual variables are updated. We focus on CoCoA+SDCA, where each group is updated using some number of SDCA iterations. When each group consists of a single variable, this reduces exactly to mini-batch SDCA. Allowing for multiple SDCA iterations on larger groups of variables yields a method that is more computationally demanding that mini-batch SDCA, and intuitively should be better than SDCA (and does appear better in practice). However, we show that our mini-batch SDCA analysis strictly dominates the CoCoA+ analysis: that is, with the same number of total dual variables updated per iteration, and thus less computation, our mini-batch SDCA guarantees are strictly better than those obtained for CoCoA+. Mini-batch SDCA is thus a simpler, computationally cheaper method, with better guarantees than those established for CoCoA+.

Although SDCA is a dual-method, improving the dual at each iteration, following the analysis methodology of \[23\], all our guarantees are on the duality gap, and thus on the primal sub-optimality, that is on the actual regularized error we care about.

## 2 Setup and Preliminaries

We consider the problem of minimizing the regularized empirical loss

\[
\min_{w \in \mathbb{R}^d} \mathcal{P}(w) := \frac{1}{n} \sum_{i=1}^{n} \phi_i(w^T x_i) + \frac{\lambda}{2} \|w\|^2, \tag{P}
\]

where \(x_1, \ldots, x_n \in \mathbb{R}^d\) are given training examples, \(\lambda > 0\) is a given regularization parameter and \(\phi_i : \mathbb{R} \to \mathbb{R}\) are given non-negative convex loss functions that already incorporate the labels (e.g. \(\phi_i(z) = \phi(y_i z)\) where \(y_i \in \{\pm 1\}\) are given labels). Instead of solving \((P)\), we solve the dual \[23\]

\[
\max_{\alpha \in \mathbb{R}^n} D(\alpha) := -\frac{1}{2} \sum_{i=1}^{n} \phi_i^*(\langle -\alpha_i \rangle) - \frac{\lambda}{2} \|X^T \alpha\|^2, \tag{D}
\]

where \(\phi_i^*(u) : \mathbb{R} \to \mathbb{R}\) is the convex conjugate of \(\phi_i\) defined in the standard way as \(\phi_i^*(u) = \max_z (zu - \phi(z))\) and \(X = [x_1, \ldots, x_n]^T \in \mathbb{R}^{n \times d}\) is the data matrix, where each row corresponds to one sample and each column corresponds to one feature. If \(\alpha^*\) is a dual-optimum of \((D)\) then \(w^* = \frac{1}{\lambda n} X^T \alpha^*\) is a primal-optimum of \((P)\). We therefore consider the mapping \(w_\alpha = \frac{1}{\lambda n} X^T \alpha\) and define the duality gap of a feasible \(\alpha \in \text{dom}(D)\) as:

\[
\mathcal{G}(\alpha) := \mathcal{P}(w_\alpha) - D(\alpha). \tag{G}
\]

### Stochastic Dual Coordinate Ascent (SDCA)

SDCA is a coordinate ascent algorithm optimizing the dual \((D)\). At \(t\)-th iteration of SDCA a coordinate \(i \in \{n\} := \{1, 2, \ldots, n\}\) is chosen at random...
and then a new iteration is obtained by updating only the \(i\)-th coordinate and keeping all other coordinates of \(\alpha\) unchanged i.e. \(\alpha^{(t+1)} = \alpha^{(t)} + \Delta\alpha_i^{(t)} e_i\), where
\[
\Delta\alpha_i^{(t)} = \arg \min_{\delta \in \mathbb{R}} D(\alpha^{(t)} + \delta e_i).
\] (1)

Assumptions on Loss Function We analyze mini-batched SDCA under one of two different assumptions on the loss functions: that they are \(L\)-Lipschitz continuous (but potentially non-smooth), or that they are \((1/\gamma)\)-smooth. Formally: i) \(L\)-Lipschitz continuous loss: \(\forall i, \forall a, b \in \mathbb{R}\) we have \(|\phi_i(a) - \phi_i(b)| \leq L |a - b|\), ii) \((1/\gamma)\)-smooth loss: Each loss function \(\phi_i\) if differentiable and its derivative is \((1/\gamma)\)-Lipschitz continuous, i.e. \(\forall a, b \in \mathbb{R}\) we have \(|\phi_i'(a) - \phi_i'(b)| \leq \frac{1}{\gamma}|a - b|\); iii) We also assume \(\phi_i\) are non-negative and that \(\phi_i(0) \leq 1\) for all \(i\).

For a positive vector \(v = (v_1, \ldots, v_n)^T > 0\) we define a weighted Euclidean norm \(\|\alpha\|^2_w = \sum_{i=1}^n v_i \alpha_i^2\). Instead of assuming the data is uniformly bounded, we will frequently refer to the weighted norm on \(\mathbb{R}^n\) with weights proportional to the squared magnitudes, i.e. \(v_i \sim \|x_i\|^2\).

3 Mini-Batched SDCA

At each iteration of mini-batched SDCA, a subset \(S \subseteq \langle n \rangle\) of the coordinates is chosen at random (see below for a discussion of the sampling distribution) and a new dual iterate is obtained by independently updating only the chosen coordinates. Since each coordinate is updated independently, mini-batch SDCA is amenable to parallelization.

The naïve approach is to use the same update rule for each coordinate as in serial case: the update is then given by \(\alpha^{(t+1)} = \alpha^{(t)} + \sum_{i \in S} \Delta\alpha_i^{(t)}\), where \(\Delta\alpha_i^{(t)}\) is given by (1). Such a naïve approach could be fine if the mini-batch size is very small and the data is “spread-out” enough [2]. However, it is easy to construct an example with just two examples where a naïve mini-batch approach will actually be much worse. In particular, it could be fine if the mini-batch size is very small and the data is “spread-out” enough [2]. However, for a positive vector \(v = (v_1, \ldots, v_n)^T > 0\) we define a weighted Euclidean norm \(\|\alpha\|^2_w = \sum_{i=1}^n v_i \alpha_i^2\). Instead of assuming the data is uniformly bounded, we will frequently refer to the weighted norm on \(\mathbb{R}^n\) with weights proportional to the squared magnitudes, i.e. \(v_i \sim \|x_i\|^2\).

An alternative that avoids this problem is to average the updates instead of adding them up, \(\alpha^{(t+1)} = \alpha^{(t)} + \frac{1}{|S|} \sum_{i \in S} \Delta\alpha_i^{(t)}\), but such an update is overly conservative: it is not any better than just updating a single dual variable, and cannot lead to parallelization speedups. Following [27], the approach we consider here is to use a summed update \(\alpha^{(t+1)} = \alpha^{(t)} + \sum_{i \in S} \Delta\alpha_i^{(t)}\), where the independent updates \(\Delta\alpha_i^{(t)}\) are derived from a relaxation of the dual:
\[
\Delta\alpha_i^{(t)} = \arg \max_{\delta} -\phi_i'(-\alpha_i - \delta) + \frac{1}{2\sqrt{n}} v_i \delta^2 - w^T x_i \delta
\] (2)

When \(v_i = \|x_i\|^2\), the update exactly agrees with the dual-optimizing update (1). But as we shall see, when larger mini-batches are used, larger values of \(v_i\) are required, resulting in smaller steps. The update (2) generalizes [27] where a single parameter \(v_i = v\) was used—here we allow \(v_i\) to vary between dual variables, accommodating differences in \(\|x_i\|\).

To summarize, the mini-batch SDCA algorithm we consider takes as input data \(X\), loss functions \(\phi_i\), a distribution over subsets \(S \subseteq \langle n \rangle\), which we will refer to as the random sampling \(\hat{S}\), and a weight vector \(v\), and proceeds as shown on Algorithm [1].

We will refer to several sampling distributions \(\hat{S}\), yielding different variants of mini-batch SDCA:

**Serial SDCA.** \(\hat{S}\) is a uniform distribution over singletons. That is, \(\hat{S}_i\) contains a single coordinate chosen uniformly at random. Setting \(v_i = \|x_i\|^2\) yields standard SDCA.

**Standard Mini-batch SDCA.** \(\hat{S}\) is a uniform distribution over subsets of size \(b\). Distributed SDCA. Consider a setting with \(C\) machines, \(n\) total data points and a mini-batch size \(b\), where for simplicity \(n\) and \(b\) are both integer multiples of \(C\). For a partition of the \(n\) coordinates into \(C\) equal sized subsets \(\{P_c\}_{c=1}^C\), consider the following sampling distribution \(\hat{S}\); for each \(c = 1..C\), choose a
The main question we now need to address is what weights \( C \) nodes in a cluster \([19, 16]\). When \( n \) is large, we then take their union. We refer to such a sample as a \( S \) subset.

Algorithm 1 mSDCA: minibatch Stochastic Dual Coordinate Ascent

1: \textbf{Input:} \( X, y, \hat{S}, v \)
2: set \( \alpha^{(0)} = 0 \in \mathbb{R}^n \)
3: \textbf{for} \( t = 0, 1, 2, \ldots \) \textbf{do}
4: choose \( S_t \) according the distribution \( \hat{S} \)
5: set \( \alpha^{(t+1)} = \alpha^{(t)}; w_\alpha = \frac{1}{\lambda n} X^T \alpha^{(t)} \)
6: \textbf{for} \( i \in S_t \) \textbf{in parallel do}
7: \( \Delta \alpha_i^{(t)} = \arg \max_{\delta} -\phi_i^{(t)}(-\alpha_i - \delta) - \frac{1}{2 \lambda n} v_i \delta^2 - w_\alpha^T x_i \delta \)
8: \( \alpha_i^{(t+1)} = \alpha_i^{(t)} + \Delta \alpha_i^{(t)} \)
9: \textbf{end for}
10: \textbf{end for}

subset \( S^c \subset P_c \) uniformly and independently at random among all such subsets of size \( b/C \), and then take their union. We refer to such a sample as a \( (C, b) \)-distributed sampling. Such a sampling is suitable in a distributed environment when \( n \) samples are equally partitioned over \( C \) computational nodes in a cluster \([19, 16]\). When \( C = 1 \) we obtain the Standard Mini-batch sampling.

The main question we now need to address is what weights \( v_i \) are suitable for use with each of the above sampling schemes, and what optimization guarantee to they yield. To answer this question, in the next Section we will introduce the notion of Expected Separable Overapproximations.

4 Expected Separable Overapproximation

In this Section we will make use of the Expected Separable Overapproximation (ESO) theory introduced in \([20]\) and further extended e.g. in \([16, 19, 18]\).

4.1 Motivation

Consider the \( t \)-th iteration of mini-batch SDCA. Our current iterate is \( \alpha^{(t)} \) and we have chosen a set \( S_t \) of coordinates which we will update in current iteration. We need to compute the updates to those coordinates, i.e. \( \forall i \in S_t \) we need to compute \( \Delta \alpha_i^{(t)} \). Maybe the natural way how to define the updates would be to define them such that \( D(\alpha^{(t+1)}) \) is as large as possible, i.e. that we maximize \( D(\alpha^{(t)}) + \sum_{i \in S_t} \Delta \alpha_i^{(t)} e_i \). However, this e.g. for hinge loss would lead to a QP, hence the computation cost would be substantial. The main disadvantage of this approach is the fact that the updates for different coordinates are dependent on each other, i.e. the value of \( \Delta \alpha_i^{(t)} \) depends on all coordinates in \( S_t \). This make it hard to parallelize. Considering the fact that \( S_t \) is a random set, maybe one would like to define the updates so that the updates doesn’t depend on current choice of \( S_t \) and that they maximize the expected value of \( D \) at next iteration. In this case we are facing following maximization problem

\[
\max_{t \in \mathbb{R}^n} \mathbb{E}[D(\alpha^{(t)} + t_{[S_t]})], \tag{3}
\]

where \( t_{[S_t]} \) is a masking operator setting all coordinates of \( t \) which are not in set \( S_t \) to zero, i.e. \( (t_{[S_t]})_i = t_i \) if \( i \in S_t \) and \( (t_{[S_t]})_i = 0 \) otherwise. The expectation in \( (3) \) is considered over the distribution \( \hat{S} \). After we get the optimal solution \( t^* \) of \( (3) \) we can define \( \Delta \alpha_i^{(t)} = t^*_i \) for all \( i \in S_t \). Therefore \( \alpha^{(t+1)} = \alpha^{(t)} + \sum_{i \in S_t} \Delta \alpha_i^{(t)} = \alpha^{(t)} + t^*_{[S_t]} \). However, now the problem \( (3) \) is even more complicated. The remedy is to replace \( \mathbb{E}[D(\alpha^{(t)} + t_{[S_t]})] \) by its separable lowerbound. Then due to the fact that it will be separable, the update for any coordinate \( i \) will be independent on the other coordinates in \( S_t \) and moreover, the updates will be obtained by solving 1D problem.

4.2 Lower-bound

Let us first state the definition of ESO.
**Definition 1** (Expected Separable Overapproximation [20]). Assume that sampling $\hat{S}$ has uniform marginals. Then we say that function $f$ admits $v$-ESO with respect to the sampling $\hat{S}$ if $\forall x, t \in \mathbb{R}^n$ we have

$$
\mathbb{E}[f(\alpha + t|\hat{S})] \leq f(\alpha) + \frac{\mathbb{E}[\hat{S}]}{n}((\nabla f(\alpha), t) + \frac{1}{2} \|t\|_2^2).
$$

(4)

Let us now just assume that we can find such a vector $v$ such that [4] holds (we show how to find $v$ in Section 4.3) and we now show how to derive the lowerbound of $\mathbb{E}[D(\alpha(t) + t|\hat{S})]$. If we write [4] for a particular choice of $f$, namely for $f(\alpha) = \|\frac{1}{\alpha n}X^T\alpha\|^2_2$ we obtain

$$
\mathbb{E}[(\|\frac{1}{\alpha n}X^T(\alpha + t|\hat{S})\|^2_2) \mathbb{E}]
\geq \|w_{\alpha}\|^2 + \frac{\mathbb{E}[\hat{S}]}{n}((\|\frac{1}{\alpha n}t\|_2^2 + \frac{2}{\alpha n}t^T X w_{\alpha}).
$$

(5)

Now we can derive the expected lowerbound of $D$ as follows

$$
\mathbb{E}[D(\alpha + t|\hat{S})] \geq \mathbb{E}[-\frac{1}{n}\sum_{i=1}^n \phi_i^*(-(\alpha + t|\hat{S}), t_i)] - \mathbb{E}[(\frac{1}{2} \|\frac{1}{\alpha n}X^T(\alpha + t|\hat{S})\|^2_2)]
$$

$$
\geq -\frac{1}{n}\|w_{\alpha}\|^2 - \frac{\mathbb{E}[\hat{S}]}{n}\sum_{i=1}^n \phi_i^*(-\alpha_i - t_i) - (1 - \frac{\mathbb{E}[\hat{S}]}{n})\frac{1}{n}\sum_{i=1}^n \phi_i^*(-\alpha_i)
$$

$$
- \frac{\mathbb{E}[\hat{S}]}{n}\lambda \|t\|_2^2 + \frac{2}{\alpha n}t^T X w_{\alpha},
$$

(6)

where in the first inequality for the first part we have used the fact that the function is separable (see Theorem 4 in [20]). If we define

$$
\mathcal{H}(t, \alpha) := -\frac{1}{n}\sum_{i=1}^n \phi_i^*(-(\alpha_i + t_i)) - \frac{1}{2}\|w_{\alpha}\|^2 - \frac{1}{2}\|\frac{1}{\alpha n}t\|_2^2 - \frac{1}{n}t^T X w_{\alpha},
$$

(7)

then it is easy to see that we can find a separable (in $t$) expected lower approximation of $D$, i.e. it holds $\forall \alpha, t \in \mathbb{R}^n$ that $\mathbb{E}[D(\alpha + t|\hat{S})] \geq \frac{b}{n} \mathcal{H}(t, \alpha) + (1 - \frac{b}{n}) D(\alpha)$, where $b := \mathbb{E}[\hat{S}]$ is the average number of mini-batch. Now let us note again that it is very hard to maximize $\mathbb{E}[D(\alpha + t|\hat{S})]$ in $t$, but maximize of $\mathcal{H}(t, \alpha)$ in $t$ is very simple, because this function is simple and separable in $t$. It is also easy to verify that the steps in Algorithm 1 are maximizing $\mathcal{H}$.

### 4.3 Computing ESO Parameter

In previous Section we have shown that using ESO we can find a separable lowerbound of $\mathbb{E}[D(\alpha + t|\hat{S})]$. However, we haven’t explained how the ESO parameter ($v$) can be obtained.

In this Section we present some of the results obtained in literature [20, 6, 16, 5] for formulas for computing vector $v$ for samplings described in Section 3. Let us mention that all formulas are data dependent. Some of them involves the spectral radius of following matrix $D^{-\frac{1}{2}} X X^T D^{-\frac{1}{2}}$, where $D = \text{diag}(X^T)$ which we will denote by $\sigma^2$, hence $\sigma^2 := \\max_{x \in \mathbb{R}^n: \|x\|_2 = 1} \|XX^T D^{-\frac{1}{2}} x\|^2$. Note that this can be in practice impossible to compute (we can estimate is using e.g. power method) or we can use an upper-bound (derived in Lemma 5.4 [5]) by $\omega = \max_{i \in \{1, n\}} \frac{1}{n} \sum_{j=1}^n \|x_i \|_0 (\sigma^2 e_j)^2$, where by $\|x_i\|_0$ we have denoted a number of non-zero elements of $i$-th data point.

**Serial SDCA.** In this simplest case we can define $v_i = \|x_i\|^2$.

**Standard Mini-batch SDCA.** In standard mini-batch we can choose $v_i = (1 + \frac{(b-1)(\sigma^2-1)}{\max_{i \in \{1, n\}} \|x_i\|_0 - 1})\|x_i\|^2$. If the data matrix $X$ is sparse, we can define $v_i = \sum_{j=1}^d (x_i^T e_j)^2 (1 + \frac{(b-1)(\|x_i\|_0 - 1)}{\max_{j \in \{1, d\}} \|x_i\|_0 - 1})$. If the data matrix $X$ is sparse, we can define $v_i = \sum_{j=1}^d (x_i^T e_j)^2 (1 + \frac{(b-1)(\|x_i\|_0 - 1)}{\max_{j \in \{1, d\}} \|x_i\|_0 - 1})$.

**Distributed SDCA.** In distributed case we can choose $v_i = \frac{b}{C} (1 + \frac{(b-C)(\sigma^2-1)}{\max_{i \in \{1, n\}} \|x_i\|_0 - C}) \|x_i\|^2$, provided that $b \geq 2C$ and $v_i = (1 + b\sigma^2)\|x_i\|^2$ if $b = C$. A simple upper-bound valid for any $b$ can be derived as follows $v_i = 2(1 + b\sigma^2)\|x_i\|^2$.

### 5 Convergence Guarantees

We are now ready to present optimization guarantees for Algorithm 1 based on the ESO parameters studied in the previous Section. These theorems extends the serial case of [23] to mini-batch setting. The Theorems are based on weights $v$ are chosen such that $f(\alpha) = \|\frac{1}{\alpha n}X^T\alpha\|^2_2$ admits $v$-ESO for a sampling $\hat{S}$ used in the Algorithm 1. Proofs are provided in the supplemental material.
Theorem 2 ((1/γ)-Smooth Loss). If the losses are (1/γ)-smooth and \( f(\alpha) = \|X^T\alpha\|^2 \) admits \( v \)-ESO for the sampling \( \hat{S} \), then for a desired duality gap \( \epsilon_G > 0 \), using Algorithm \([4]\) if we choose
\[
T \geq \frac{\|v\|_\infty}{\beta} \left( \frac{1}{\lambda_X} + \frac{n}{\|v\|_\infty} \right) \log \left( \frac{\|v\|_\infty}{\beta} \left( \frac{1}{\lambda_X} + \frac{n}{\|v\|_\infty} \right) \frac{1}{\epsilon_G} \right)
\]
we have that \( \mathbb{E}[\mathcal{P}(w_T) - D(\alpha_T)] \leq \epsilon_G \). To obtain \( \mathbb{E}[\mathcal{P}(\hat{w}) - D(\hat{\alpha})] \leq \epsilon_G \), it is sufficient to choose
\[
T_0 \geq \frac{\|v\|_\infty}{b \lambda_X} \left( \frac{1}{\lambda_Y} + \frac{n}{\|v\|_\infty} \right) \log \left( \frac{\|v\|_\infty}{b \lambda_Y} \left( \frac{1}{\lambda_X} + \frac{n}{\|v\|_\infty} \right) \frac{1}{(T-T_0)} \right),
\]
where
\[
\hat{\alpha} = \frac{1}{T-T_0} \sum_{t=T_0}^{T-1} \alpha(t).
\]
Moreover, if \( T \geq \frac{\|v\|_\infty + \lambda n \gamma}{b \lambda_X} \log \left( \frac{\|v\|_\infty + \lambda n \gamma}{b \lambda_Y} \frac{1}{\epsilon_G} \right) \) then \( \mathbb{P}(\mathcal{P}(w_T) - D(\alpha_T) \leq \epsilon_G) \geq 1 - \rho \).

Theorem 3 (L-Lipschitz Continuous Loss). If the losses are \( L \)-Lipschitz and \( f(\alpha) = \|X^T\alpha\|^2 \) admits \( v \)-ESO for the sampling \( \hat{S} \), then for a desired duality gap \( \epsilon_G > 0 \), using Algorithm \([4]\) denoting \( G = 4L^2 \sum_{i=1}^{n} \frac{v_i}{n} \), if we choose
\[
T_0 \geq t_0 + \frac{1}{b} \left( \frac{\epsilon_G}{\lambda n \gamma} - 2n \right), \quad T \geq T_0 + \max \{ \frac{n}{b}, \frac{1}{b} \},
\]
\[
t_0 \geq \max(0, \frac{n}{b} \log(2\lambda n \epsilon_G^2 / G)),
\]
we have that \( \mathbb{E}[\mathcal{P}(\hat{w}) - D(\hat{\alpha})] \leq \epsilon_G \), where \( \hat{\alpha} \) is defined in \([2]\). Moreover, when \( t \geq T_0 \), we have dual sub-optimality bound \( \mathbb{E}[D(\alpha^*_t) - D(\hat{\alpha}(t))] \leq \frac{1}{2} \epsilon_G \).

6 Guarantees and Speedups for Specific Sampling Distributions

Theorems\([3, 2]\) are stated in terms of ESO parameter \( v \). Let us now consider the specific sampling distribution of interest. Assume for simplicity \( \|x_i\| \leq 1 \), and define
\[
\beta_{sr} = 1 \quad \beta_{std} = 1 + \frac{(b-1)(n\sigma^2-1)}{\max(1,n-1)} \quad \beta_{dist} = \frac{b}{b-C} \left( 1 + \frac{(b-C)(n\sigma^2-1)}{\max(C,n-C)} \right)
\]
for the serial, standard and distributed sampling schemes respectively, with overall mini-batch size \( b \) and distribution over \( C \) machines. Using the weights \( v_i = \beta \), then we have the following obtain the following iteration complexities:

\( (1/\gamma) \)-Smooth Loss. In this case \([8]\) in Theorem \([2]\) becomes \( T \geq \frac{\beta}{\beta} \left( \frac{1}{\lambda_X} + \frac{n}{\beta} \right) \log \left( \frac{\beta}{\beta} \left( \frac{1}{\lambda_X} + \frac{n}{\beta} \right) \frac{1}{\epsilon_G} \right) \), and hence the iteration complexity is (ignoring logarithmic terms) \( \tilde{O} \left( \frac{n}{\beta} + \frac{\beta}{\beta} \frac{n^2}{\lambda_X \epsilon_G} \right) \).

\( L \)-Lipschitz Continuous Loss. Combining equations \([10, 11]\) and \([12]\), and again ignoring logarithmic factors, we get an iteration complexity of:
\[
\tilde{O} \left( \frac{n}{\beta} + \frac{\beta}{\beta} \frac{n^2}{\lambda_X \epsilon_G} \right).
\]

Plugging in \( \beta_{std} \) into \([13]\) recovers the previous analysis of Lipschitz loss with standard sampling.

Both the Lipschitz and smooth cases involve two terms: the first term, \( \frac{n}{\beta} \), always displays a linear improvement as we increase the mini-batch size. However, in the second term, we also have a dependence on the data-dependent \( 1 \geq \beta \leq b \), which depends on the mini-batch size \( b \). We will have a linear improvement in the second term, i.e. potential for linear speedup, as long as \( \beta = O(1) \). For standard sampling we have that \( \beta \approx 1 + b \sigma^2 \), and so we obtain linear speedups as long as \( b = O(1/\sigma^2) \), as discussed in \([13]\). We can now also quantify the effect of distributed sampling and see that it is quite negligible and yields almost the same speedups and the same maximum allows mini-batch size as with standard sampling. Note that typically we will have \( C \ll b \), as we would like to process multiple example on each machine—otherwise communication costs would overwhelm computational costs \([26]\). The analysis supports this choice as well as the extreme choice \( C = b \).

Focusing on the smooth loss, it is possible to obtain a linear reduction in the iteration complexity (corresponding to linear speedups) for SGD with mini-batch size of up to \( O(\sqrt{n}) \) without any data-dependent assumption, that is regardless of the value of \( \beta \) \([4, 13]\). Is this possible also with SDCA? Indeed, even if we don’t account for the data dependent quantity \( \beta \), since we always have \( \beta \leq b \),
then the iteration complexity of SDCA for mini-batch SDCA with smooth loss is: $\mathcal{O}(1/(\lambda \gamma) + n/b \log(1/\epsilon))$.

Compared to the CoCoA+ guarantee (ignoring log factors): $\tilde{d} = \frac{\text{Data (and hence dual variables) are partitioned among } C \text{ machines (as in our distributed sampling), defining } C \text{ subproblems, one for each machine. At each iteration, the set of dual variables of each of the } C \text{ machines are updated independently, and then communicated and aggregated across machines. Different local updates can be used, and the CoCoA+ analysis depends on how well the update improves the local subproblem. Here we will consider using local SDCA updates in conjunction with CoCoA+: at each iteration, on each of the } C \text{ machines, } b/C \text{ dual variables are selected (as in our distributed sampling), and } H \text{ iterations of SDCA are performed sequentially on these } b/C \text{ points (in parallel on each of the } C \text{ machines, and while considering all other dual variables, including all variables on other machines, as fixed).}

We will consider for simplicity 1-smooth loss functions and compare the CoCoA+ guarantees on the number of required iterations to the SDCA guarantees we present here, noting also the differences in the amount of computation per iterations. In all our comparisons, the required communication in each iteration of SDCA and CoCoA+ is identical and amounts to a single distributed averaging of vectors in $\mathbb{R}^d$.

**Setting** $b = C$ and $H = 1$, we exactly recover mini-batch SDCA with a minibatch of size $b$, and so we would expect the CoCoA+ analysis to yield the same guarantee. However, our guarantee on the number of required iterations in this case is (ignoring log factors) $\tilde{O} \left( \frac{n}{b} + \frac{1}{b\lambda} + \frac{\sigma^2}{\lambda \gamma} \right)$ compared to the CoCoA+ guarantee (ignoring log factors): $\tilde{O} \left( \frac{n}{b} + \frac{n\sigma^2}{b\lambda} + \frac{1}{b} + \frac{\sigma^2}{\lambda \gamma} \right)$. where $\sigma^2 = \max_{i} \max_{x} \| \sum_{i \in P_c} \alpha_i x_i \|_{2}^{2} = 1 \left( \frac{C}{n} \| \sum_{i \in P_c} \alpha_i x_i \|_{2} \right)$, and so $\sigma^2 \geq 1/n$. Our guarantee therefore dominates that of CoCoA+: the second term is worse by a factor of $n\tilde{\sigma}^2 > 1$, the third by a factor of $1/\sigma^2 < 1$ and the fourth term in the CoCoA+ bound, can be particularly bad when $\lambda$ is small (e.g. when $\lambda \propto 1/n$).

**Setting** $b > C$ and $H = b/C$, both minibatch SDCA and CoCoA+ perform the same number of SDCA updates (same amount of computation) at each iteration, but while minibatch SDCA’s updates are entirely independent, each group of $H$ CoCoA+ updates (the $H$ updates on the same machine) are performed sequentially. We would therefore expect CoCoA+’s updates to be better, and therefore require less iterations. Unfortunately, the CoCoA+ analysis does not show this.

To see the deficiency in the CoCoA+ analysis at another extreme, consider the case where $b = n, 1 < C < n$ and $H \rightarrow \infty$. In this case, each iteration of mini-batch SDCA is actually a full batch of parallel updates (updating each coordinate independently), while for CoCoA+ this corresponds to fully optimizing each group of $n/C$ dual variables using many SDCA updates (and thus much more computation). Still, the CoCoA+ iteration bound here would be $\tilde{O} \left( 1 + \frac{\sigma^2}{\lambda} \right)$, where $\sigma^2 = \max_{i} \frac{1}{C} \sum_{x_i} \| X^T \alpha \|_{2} / \| \sum_{i \in P_c} x_i \alpha_i \|$, and so $\sigma^2 \geq \tilde{\sigma}^2$, compared to the better mini-batch SDCA bound $\tilde{O} \left( 1 + \frac{\sigma^2}{\lambda} \right)$.

7. **Comparison with CoCoA+**

CoCoA+ [14] is a recently presented framework and analysis for distributed optimization of the dual (\textbf{D}). Data (and hence dual variables) are partitioned among \( C \) machines (as in our distributed sampling), defining \( C \) subproblems, one for each machine. At each iteration, the set of dual variables of each of the \( C \) machines are updated independently, and then communicated and aggregated across machines.
Table 1: Basic characteristics of datasets; obtained from libsvm collection [12].

<table>
<thead>
<tr>
<th>name</th>
<th># train. samples</th>
<th># test samples</th>
<th># features</th>
<th>Sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>epsilon</td>
<td>400,000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rcv1</td>
<td>20,242</td>
<td>677,399</td>
<td>2,000</td>
<td>100%</td>
</tr>
<tr>
<td>news20</td>
<td>15,000</td>
<td>4,996</td>
<td>1,355,191</td>
<td>0.03%</td>
</tr>
<tr>
<td>real-sim</td>
<td>72,309</td>
<td></td>
<td>20,958</td>
<td>0.24%</td>
</tr>
</tbody>
</table>

Figure 1: TOP ROW: Number of iterations needed to get an approximate solution is almost the same for standard SDCA and distributed SDCA for $C \in \{1, 2, 4, 8, 16\}$. BOTTOM ROW: Comparison of mSDCA and CoCoA+ when solving the SVM dual problem on $C = 4$ computers (left) and $C = 16$ computers (right).

And so, even though CoCoA+ with SDCA updates should be a more powerful algorithm, its analysis [14] fails to show benefits over the simpler mini-batch SDCA, and out analysis here of mini-batch SDCA even dominates the CoCoA+ analysis. The reason for this is that CoCoA+ aims to be a more general framework capable of including arbitrary local solvers. Hence, necessarily, the analysis must be more conservative.

8 Numerical Experiments

In this Section we show that the cost of distribution is negligible (in terms of # iterations) when compared to standard mSDCA. We also show that if $b \gg 1$, then CoCoA+ is faster than mSDCA in practice. We have run experiments on 4 datasets (see Table 1). Note that most of the datasets are sparse (e.g. news20: an average tsample depends on 385 features out of 1.3M).

Standard vs. Distributed SDCA. Figure 1 (top row) compares standard and distributed SDCA. Recall that distributed sampling with $C = 1$ and standard mini-batch sampling coincide. On the $x$-axis is the parameter $b$ and on the $y$-axis we plot how much more data-accesses we have to as $b$ or $C$ grow, to get achieve the same accuracy. We see that the lines are almost identical for various choices of $C$, which implies that the cost of using distributed mSDCA does not affect the number of iterations significantly. This is also supported by the theory (notice that in [12] we have $\beta_{\text{dist}}/\beta_{\text{std}} \approx 1$). Also note that, for news20 for instance, increasing $b$ to $10^4$ implies that the number of data-accesses (epochs) will increase by a factor of 11, which implies that # iterations will decrease almost by 1,000 for $b = 10^4$ when compared with $b = 1$.

mSDCA vs. CoCoA+. In Figure 1 (bottom row) we compare the mSDCA with CoCoA+ with SDCA as a local solver. We plot the duality gap as a function of epochs (if communication is negligible then the main cost is in computation) or iterations (if the communication cost is significant than this is the correct measure of performance). As the results suggest, is the communication cost it negligible then the mSDCA with small $b$ is the best (as expected), however, if communications cost is significant, then CoCoA+ with large values of $H$ significantly outperforms mSDCA.
References


A Technical Results

Lemma 4 (Lemma 2 in [23]). For all \( \alpha \in \mathbb{R}^n \):
\[
D(\alpha) \leq P(w^*) \leq P(0) \leq 1.
\]
Moreover \( D(0) \geq 0 \).

Following Lemma is a minibatch extension of Lemma 1 in [23].

Lemma 5 (Expected increase of dual objective). Assume that \( \phi_i^* \) is \( \gamma \)-strongly convex (\( \gamma \) can be also zero). Then, for any \( t \) and any \( s \in [0, 1] \) we have
\[
\mathbb{E}[D(\alpha^{(t+1)}) - D(\alpha^{(t)})] \geq b\left(\frac{\gamma}{n} G(\alpha^{(t)}) - \left(\frac{\gamma}{n}\right)^2 \frac{1}{2\lambda} G^{(t)}(x)\right),
\]
where
\[
G^{(t)} = \frac{1}{n} \left(\frac{\|u(t) - \alpha^{(t)}\|^2_v}{v} - \frac{2\gamma \lambda n(1-s)x}{\lambda} \left(\frac{1}{2}\right) \left(\|u^{(t)}(1-s)x - \alpha^{(t)}\|^2\right)\right)
\]
\[
= \frac{1}{n} \sum_{i=1}^{n} (v_i - \frac{\gamma \lambda n(1-s)x}{\lambda} \left(\|u^{(t)}(1-s)x - \alpha^{(t)}\|^2\right)) (u_i^{(t)} - \alpha_i^{(t)})^2,
\]
\[
u = (u_1^{(t)}, \ldots, u_n^{(t)})^T
\]
and \( -u_i^{(t)} \in \partial \phi_i(w^T(1-s)x) \).

Lemma 6 (Lemma 3 in [23]). Let \( \phi : \mathbb{R} \rightarrow \mathbb{R} \) be an \( L \)-Lipschitz continuous. Then for any \( |\alpha| > L \) we have that \( \phi^*(\alpha) = \infty \).

Following lemma is a small extension of Lemma 4 in [23] to obtain more tight bounds in case each sample has different norm or when ESO bound is used. For example, in serial case we will have that \( \forall t : G_t \leq 4L^2 \sum_{i=1}^{n} \|x_i\|^2 \).

Lemma 7 (Bound on \( G^{(t)} \)). Suppose that for all \( i, \phi_i \) is \( L \)-Lipschitz continuous. Then
\[
\forall t : G^{(t)} \leq 4L^2 \sum_{i=1}^{n} \|x_i\|^2.
\]

Proof. Indeed,
\[
G^{(t)} \leq \frac{1}{n} \sum_{i=1}^{n} (v_i - \frac{\gamma \lambda n(1-s)x}{\lambda} \left(\|u^{(t)}(1-s)x - \alpha^{(t)}\|^2\right)) (\alpha_i^{(t)})^2 \leq \frac{1}{n} \sum_{i=1}^{n} (v_i)(2L)^2.
\]

Lemma 8 (Theorem 1 in [21]). Fix \( x_0 \in \mathbb{R}^N \) and let \( \{x_k\}_{k \geq 0} \) be a sequence of random vectors in \( \mathbb{R}^N \) with \( x_{k+1} \) depending on \( x_k \). Let \( \phi : \mathbb{R}^N \rightarrow \mathbb{R} \) be a nonnegative function and define \( \xi_k = \phi(x_k) \). Lastly, choose accuracy level \( 0 < \epsilon < \xi_0 \), confidence level \( 0 \leq \rho \leq 1 \), and assume that the sequence of random variables \( \{\xi_k\}_{k \geq 0} \) is nonincreasing and has one of the following properties:

(i) \( \mathbb{E}[\xi_{k+1} | x_k] \leq (1 - \frac{\xi_k}{c_1}) \xi_k \), for all \( k \), where \( c_1 > \epsilon \) is a constant,

(ii) \( \mathbb{E}[\xi_{k+1} | x_k] \leq (1 - \frac{\xi_k}{c_2}) \xi_k \), for all \( k \) such that \( \xi_k \geq \epsilon \), where \( c_2 > 1 \) is a constant.

If property (i) holds and we choose \( K \geq 2 + \frac{\log(1 - \frac{\xi_0}{\epsilon} + \log(\frac{1}{\rho}))}{\epsilon} \), or if property (ii) holds, and we choose \( K \geq c_2 \log(\frac{\xi_0}{c_0}) \), then \( \mathbb{P}(\xi_K \leq \epsilon) \geq 1 - \rho \).

B Proofs

B.1 Proof of Lemma 5

Let us define \( T_\alpha \) as an unique maximizer of a function \( \mathcal{H}(t, \alpha) \) defined in (7), i.e.
\[
T_\alpha := \arg \max_t \mathcal{H}(t, \alpha).
\]
Let us now state some basic properties about function \( \mathcal{H} \). We have that \( \forall t, \alpha \in \mathbb{R}^n \) and sampling \( \hat{S} \):

- \( \mathcal{H}(0, \alpha) = D(\alpha) \),
from ESO we have
\[ E[D(\alpha + t_i S_i)] \geq (1 - \frac{\lambda}{n})D(\alpha) + \frac{\lambda}{n} H(t, \alpha), \]
(19)

\[ H(t, \alpha) \leq H(T_{i}, \alpha). \]

Convex conjugate maximal property implies that
\[ \phi_i^*(-u_i^{(t)}) = -u_i^{(t)} w_i^{T} x_i - \phi_i(w_i^{T} x_i). \]
(20)

Let us estimate the expected change of dual objective.
\[ \frac{n}{b} E[D(\alpha^{(t)}) - D(\alpha^{(t+1)})] \leq \frac{n}{b} E[D(\alpha^{(t)}) - D(\alpha^{(t)}) + (T_{i}, \alpha)] \leq D(\alpha^{(t)}) - H(T_{i}, \alpha^{(t)}) \]
\[ = \frac{1}{n} \sum_{i=1}^{n} \left( \phi_i^*(-(\alpha_i + (T_{i}, \alpha^{(t)}))) - \phi_i^*(-\alpha_i^{(t)}) \right) \]
\[ + \frac{\lambda}{2} \left( \frac{1}{\lambda n} T_{i} \alpha^{(t)} \right)^{2} + 2 \left( \frac{1}{\lambda n} T_{i} \alpha^{(t)} \right)^{T} X w_{\alpha} \]
\[ \leq \frac{1}{n} \sum_{i=1}^{n} \left( \phi_i^*(-(\alpha_i^{(t)} + s(u_i - \alpha_i^{(t)})) - \phi_i^*(-\alpha_i^{(t)}) \right) \]
\[ + \frac{\lambda}{2} \left( \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{2} + 2 \left( \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{T} X w_{\alpha} \].

Using $\gamma$-strong convexity of $\phi_i^*$ we have that
\[ \phi_i^*(-(\alpha_i^{(t)} + s(u_i - \alpha_i^{(t)})) \leq s \phi_i^*(-u_i) + (1 - s) \phi_i^*(-\alpha_i^{(t)}) - \frac{\gamma}{2} (1 - s) s(u_i - \alpha_i^{(t)})^2. \]
(21)

Therefore,
\[ \frac{n}{b} E[D(\alpha^{(t)}) - D(\alpha^{(t+1)})] \leq \frac{1}{n} \sum_{i=1}^{n} \left( s \phi_i^*(-u_i) + s u_i x_i^T w_{\alpha} - s \phi_i^*(-\alpha_i^{(t)}) - \frac{\gamma}{2} (1 - s) s(u_i - \alpha_i^{(t)})^2 \right) \]
\[ + \frac{\lambda}{2} \left( \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{2} + 2 \left( \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{T} X w_{\alpha} \]
\[ \leq \frac{s}{n} \sum_{i=1}^{n} \left( -u_i w_{\alpha}^T x_i - \phi_i(w_{\alpha}^T x_i) + u_i x_i^T w_{\alpha} - \phi_i(-\alpha_i^{(t)}) \right) \]
\[ + \frac{\lambda}{2} \left( -\frac{\gamma}{\lambda n} (1 - s) s \|u - \alpha\|^2 + \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{2} + 2 \left( \frac{1}{\lambda n} s(-\alpha^{(t)}) \right)^{T} X w_{\alpha} \].

Substituting the definition of duality gap $G$ we obtain
\[ \frac{n}{b} E[D(\alpha^{(t)}) - D(\alpha^{(t+1)})] \leq \frac{s}{n} \sum_{i=1}^{n} \left( -\phi_i(w_{\alpha}^T x_i) - \phi_i(-\alpha_i^{(t)}) - \alpha_i^{(t)} w_{\alpha}^T x_i \right) \]
\[ + \frac{\lambda}{2} \left( -\frac{\gamma}{\lambda n} (1 - s) s \|u - \alpha\|^2 + \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{2} \]
\[ = -sG(\alpha^{(t)}) + \frac{\lambda}{2} \left( \frac{1}{\lambda n} s(u - \alpha^{(t)}) \right)^{2} - \frac{\gamma}{\lambda n} (1 - s) s \|u - \alpha\|^2 \]
\[ = -sG(\alpha^{(t)}) + \frac{1}{2\lambda} \left( \frac{1}{s} \right)^{2} \left( \|u - \alpha^{(t)}\|^2 - \frac{\gamma n(1 - s)}{s} \|u - \alpha\|^2 \right). \]

Multiplying both sides by $-\frac{b}{n}$ we obtain [15].
B.2 Proof of Theorem 3

At first let us estimate expected change of dual feasibility.

\[ E[\epsilon_D^{(t+1)}] \leq E[D(\alpha^*) - D(\alpha^{(t+1)})] = E[D(\alpha^*) - D(\alpha^{(t+1)}) + D(\alpha^{(t)}) - D(\alpha^{(t)})] \]

\[ \leq -b \frac{s}{n} E[\epsilon_D^{(t)}] + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} G + E[\epsilon_D^{(t)}] = (1 - b \frac{s}{n}) E[\epsilon_D^{(t)}] + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} G. \]

From the above follows that

\[ E[\epsilon_D^{(t)}] \leq (1 - b \frac{s}{n})^t \epsilon_D^0 + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} t (1 - b \frac{s}{n})^i \leq (1 - b \frac{s}{n})^t \epsilon_D^0 + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} t. \]

Choice of \( s = 1 \) and \( t = t_0 := \max \{ 0, \left\lceil \frac{\epsilon_D^0}{2\lambda(2\lambda + b(t - t_0))} \right\rceil \} \) will lead to

\[ E[\epsilon_D^{(t)}] \leq (1 - b \frac{s}{n})^t \epsilon_D^0 + \frac{s G}{2\lambda} n \leq \frac{G}{2\lambda} \frac{\epsilon_D^0}{n} + \frac{1}{2\lambda} G = \frac{G}{\lambda n}. \]

Following the proof in [23] we are now going to show that

\[ \forall t \geq t_0 : E[\epsilon_D^{(t)}] \leq \frac{2G}{\lambda(2n + b(t - t_0))}. \]

Clearly, (24) implies that (25) holds for \( t = t_0 \). Now imagine that it holds for any \( t \geq t_0 \) then we show that it also has to hold for \( t + 1 \). Indeed, using \( s = \frac{2n}{2n + b(t - t_0)} \) we obtain

\[ E[\epsilon_D^{(t+1)}] \leq (1 - b \frac{s}{n}) E[\epsilon_D^{(t)}] + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} G \]

\[ \leq (1 - b \frac{s}{n}) \frac{2G}{\lambda(2n + b(t - t_0))} + b \left( \frac{s}{n} \right)^2 \frac{1}{2\lambda} G \]

\[ = \frac{2G}{\lambda(2n + b(t - t_0))} \left( \frac{1}{b} + b \right) \left( \frac{2n + b(t - t_0) + b}{2n + b(t - t_0)} \right) \]

\[ \leq \frac{2G}{\lambda(2n + b(t - t_0)) + b} \cdot \frac{2n + b(t - t_0) + b}{2n + b(t - t_0)} \]

\[ \leq \frac{2G}{\lambda(2n + b(t - t_0)) + b}. \]

In the last inequality we have used the fact that geometric mean is less or equal to arithmetic mean.

If \( \alpha \) is defined as (9) then we obtain that

\[ E[G(\alpha)] = E \left[ G \left( \sum_{t=T_0}^{T_0-1} \alpha^{(t)} \right) \right] \leq \frac{1}{T - T_0} \sum_{t=T_0}^{T-1} E \left[ \frac{1}{T - T_0} \alpha^{(t)} \right] \]

\[ \leq \frac{1}{T - T_0} E \left[ \sum_{t=T_0}^{T-1} \left( -n \frac{1}{s} E[D(\alpha^{(t)}) - D(\alpha^{(t+1)})] + \left( \frac{s}{n} \right) \frac{1}{2\lambda} \left( \frac{1}{n} \left\| u_t - \alpha^{(t)} \right\|_2 \right)^2 \right) \right] \]

\[ \leq n \frac{1}{s} \frac{1}{b} \frac{1}{T - T_0} \left( E[D(\alpha^{(T)})] - E[D(\alpha^{(T_0)})] \right) + \frac{s G}{n 2\lambda} \]

\[ \leq n \frac{1}{s} \frac{1}{b} \frac{1}{T - T_0} \left( D(\alpha^*) - E[D(\alpha^{(T_0)})] \right) + \frac{s G}{n 2\lambda}. \]

Now, if \( T \geq \left\lceil \frac{1}{b} \right\rceil + T_0 \) such that \( T_0 \geq t_0 \) we obtain

\[ E[G(\alpha)] \leq \frac{n}{s} \frac{1}{b} \frac{1}{T - T_0} \left( \frac{2G}{\lambda(2n + b(T_0 - t_0))} \right) + \frac{s G}{n 2\lambda} \]

\[ = \frac{G}{\lambda} \frac{n}{s} \frac{1}{b(T - T_0)} \left( \frac{2}{(2n + b(T_0 - t_0))} \right) + \frac{s}{2n}. \]
Using \( s = \frac{n}{b(T-T_0)} \) we obtain that
\[
\mathbb{E}[G(\bar{\alpha})] \leq \frac{G}{b\lambda} \left( \frac{2}{2\overline{\beta} + (T_0 - t_0)} + \frac{1}{2(T - T_0)} \right).
\]

To have this quantity \( \leq \epsilon_G \) we obtain that \( T, t_0, T_0 \) has to satisfy (10). The fact that \( T_0 \geq t_0 + \frac{\beta}{\mathcal{E}} \left( \frac{4\mathcal{E}}{\lambda n\gamma} - 2n \right) \) implies that right-hand site of (26) is \( \leq \epsilon_G \).

### B.3 Proof of Theorem 2

If function \( \phi \) is \( (1/\gamma) \)-smooth then \( \phi^*_s \) is \( \gamma \)-strongly convex. If we plug \( s = \bar{s} = \frac{\lambda n\gamma}{\|v\|_\infty + \lambda n\gamma} \in (0, 1) \) into (16) we obtain that \( \forall t : G(t) \leq 0 \). Hence (15) will read as follows
\[
\mathbb{E}[D(\alpha^{(t+1)}) - D(\alpha^{(t)})] \geq b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \mathcal{G}(\alpha^{(t)}) \geq b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} (D(\alpha^{*}) - D(\alpha^{(t)})).
\]

Using the fact that \( \mathbb{E}[D(\alpha^{(t+1)}) - D(\alpha^{(t)})] = \mathbb{E}[D(\alpha^{(t+1)})] - D(\alpha^{*}) - D(\alpha^{(t)}) \) we have
\[
\mathbb{E}[D(\alpha^{*}) - D(\alpha^{(t+1)})] \leq \left( 1 - b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \right) (D(\alpha^{*}) - D(\alpha^{(t)})).
\]

Therefore if we denote by \( \epsilon_D^{(t)} = D(\alpha^{*}) - D(\alpha^{(t)}) \) we have that
\[
\mathbb{E}[\epsilon_D^{(t)}] \leq \left( 1 - b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \right) \epsilon_D^{(0)} \leq \left( 1 - b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \right) \leq \exp \left( -bt \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \right).
\]

Right hand site will be smaller than some \( \epsilon_D \) if
\[
t \geq \frac{\|v\|_\infty}{b} \left( \frac{1}{\lambda \gamma} + \frac{n}{\|v\|_\infty} \right),
\]

Moreover, to bound the duality gap we have
\[
b \frac{\lambda 2n}{\|v\|_\infty + \lambda n\gamma} \mathcal{G}(\alpha^{(t)}) \leq \mathbb{E}[\epsilon_D^{(t)} - \epsilon_D^{(t+1)}] \leq \mathbb{E}[\epsilon_D^{(t)}].
\]

Therefore \( \mathcal{G}(\alpha^{(t)}) \leq \frac{\|v\|_\infty + \lambda n\gamma}{b \lambda \gamma} \mathbb{E}[\epsilon_D^{(t)}] \). Hence if \( \epsilon_D \leq \frac{b \lambda 2n}{\|v\|_\infty + \lambda n\gamma} \epsilon_G \) then \( \mathcal{G}(\alpha^{(t)}) \leq \epsilon_G \). Therefore after
\[
t \geq \frac{\|v\|_\infty}{b} \left( \frac{1}{\lambda \gamma} + \frac{n}{\|v\|_\infty} \right),
\]

iterations we have duality gap less than \( \epsilon_G \) and the first part of the proof is done. To show the second part of Theorem let us sum (30) over \( t = T_0, \ldots, T - 1 \) to obtain
\[
\mathbb{E} \left[ \frac{1}{T - T_0} \sum_{t=T_0}^{T-1} \mathcal{G}(\alpha^{(t)}) \right] \leq \frac{\|v\|_\infty + \lambda n\gamma}{b \lambda \gamma} \frac{1}{T - T_0} \mathbb{E}[D(\alpha^{(T)}) - D(\alpha^{(T_0)})].
\]

Now, if we choose \( \bar{w}, \bar{\alpha} \) to be either average vectors or a randomly chosen vector over \( t \in \{T_0 + 1, \ldots, T \} \), then we have
\[
\mathbb{E}[\mathcal{G}(\bar{\alpha})] \leq \frac{\|v\|_\infty + \lambda n\gamma}{b \lambda \gamma} \frac{1}{T - T_0} \mathbb{E}[D(\alpha^{(T)}) - D(\alpha^{(T_0)})] \leq \frac{\|v\|_\infty + \lambda n\gamma}{b \lambda \gamma} \frac{1}{T - T_0} \mathbb{E}[D(\alpha^{*}) - D(\alpha^{(T_0)})].
\]

Hence to have \( \mathbb{E}[\mathcal{G}(\bar{\alpha})] \leq \epsilon_G \) it is sufficient to choose
\[
\mathbb{E}[\epsilon_D^{(T_0)}] \leq \frac{b \lambda 2n}{\|v\|_\infty + \lambda n\gamma} (T - T_0) \epsilon_G.
\]

Therefore we need \( T_0 \) to satisfy
\[
T_0 \geq \frac{\|v\|_\infty}{b} \left( \frac{1}{\lambda \gamma} + \frac{n}{\|v\|_\infty} \right) \log \left( \frac{\|v\|_\infty}{b} \left( \frac{1}{\lambda \gamma} + \frac{n}{\|v\|_\infty} \right) \frac{1}{(T - T_0) \epsilon_G} \right).
\]
To get a high probability result we use Lemma 8 with \( \xi^{(t)} = D(\alpha^*) - D(\alpha^{(t)}) \), \( c_2 = \frac{\|v\|_{\infty} + \lambda n \gamma}{b \lambda \gamma} \) (see (29)) and \( \epsilon = \frac{b \lambda \gamma}{\|v\|_{\infty} + \lambda n \gamma} \epsilon_G \) to obtain that after

\[ \tilde{T} = c_2 \log \left( \frac{\xi^{(0)}}{\epsilon \rho} \right) \leq \text{Lemma 4} \left( \|v\|_{\infty} + \lambda n \gamma \right) \log \left( \frac{1}{\epsilon \rho} \right) \]

\[ 1 - \rho \leq P \left( D(\alpha^*) - D(\alpha^{(T)}) \leq \epsilon \right) \leq P \left( \frac{b \lambda \gamma}{\|v\|_{\infty} + \lambda n \gamma} G(\alpha^{(t)}) \leq \epsilon \right) = P \left( G(\alpha^{(t)}) \leq \epsilon_G \right). \]