Journal of Machine Learning Research 12 (2011) 1865-1892 Submitted 12/10; Revised 5/11; Published 6/11

Stochastic Methods for ℓ_1-regularized Loss Minimization

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
School of Computer Science and Engineering
The Hebrew University of Jerusalem
Givat Ram, Jerusalem 91904, Israel

Ambuj Tewari AMBUJ@CS.UTEXAS.EDU
Computer Science Department
The University of Texas at Austin
Austin, TX 78701, USA

Editor: Leon Bottou

Abstract

We describe and analyze two stochastic methods for ℓ_1 regularized loss minimization problems, such as the Lasso. The first method updates the weight of a single feature at each iteration while the second method updates the entire weight vector but only uses a single training example at each iteration. In both methods, the choice of feature or example is uniformly at random. Our theoretical runtime analysis suggests that the stochastic methods should outperform state-of-the-art deterministic approaches, including their deterministic counterparts, when the size of the problem is large. We demonstrate the advantage of stochastic methods by experimenting with synthetic and natural data sets.[1]

Keywords: L1 regularization, optimization, coordinate descent, mirror descent, sparsity

1. Introduction

We present optimization procedures for solving problems of the form:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{m}\sum_{i=1}^{m} L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) + \lambda\|\mathbf{w}\|_1 \quad , \tag{1}$$

where $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m) \in ([-1,+1]^d \times \mathcal{Y})^m$ is a sequence of training examples, $L : \mathbb{R}^d \times \mathcal{Y} \to [0,\infty)$ is a non-negative loss function, and $\lambda > 0$ is a regularization parameter. This generic problem includes as special cases the Lasso (Tibshirani, 1996), in which $L(a,y) = \frac{1}{2}(a-y)^2$, and logistic regression, in which $L(a,y) = \log(1+\exp(-ya))$.

Our methods can also be adapted to deal with additional boxed constraints of the form $w_i \in [a_i, b_i]$, which enables us to use them for solving the dual problem of Support Vector Machine (Cristianini and Shawe-Taylor, 2000). For concreteness, we focus on the formulation given in (1).

Throughout the paper, we assume that L is convex in its first argument. This implies that (1) is a convex optimization problem, and therefore can be solved using standard optimization techniques, such as interior point methods. However, standard methods scale poorly with the size of the problem (i.e., m and d). In recent years, machine learning methods are proliferating in data-laden domains such as text and web processing in which data sets of millions of training examples or features are

1. An initial version of this work (Shalev-Shwartz and Tewari, 2009) appeared in ICML 2009.

not uncommon. Since traditional methods for solving (1) generally scale very poorly with the size of the problem, their usage is inappropriate for data-laden domains. In this paper, we discuss how to overcome this difficulty using stochastic methods. We describe and analyze two practical methods for solving (1) even when the size of the problem is very large.

The first method we propose is a stochastic version of the familiar coordinate descent approach. The coordinate descent approach for solving ℓ_1 regularized problems is not new (as we survey below in Section 1.1). At each iteration of coordinate descent, a single element of $\mathbf{w}$ is updated. The only twist we propose here regarding the way one should choose the next feature to update. We suggest to choose features uniformly at random from the set $[d] = \{1, \ldots, d\}$. This simple modification enables us to show that the runtime required to achieve ϵ (expected) accuracy is upper bounded by

$$\frac{m d \beta \|\mathbf{w}^\star\|_2^2}{\epsilon} , \tag{2}$$

where β is a constant which only depends on the loss function (e.g., $\beta = 1$ for the quadratic loss function) and $\mathbf{w}^\star$ is the optimal solution. This bound tells us that the runtime grows only linearly with the size of the problem. Furthermore, the stochastic method we propose is parameter free and very simple to implement.

Another well known stochastic method that has been successfully applied for loss minimization problems, is stochastic gradient descent (e.g., Bottou and LeCunn, 2005; Shalev-Shwartz et al., 2007). In stochastic gradient descent, at each iteration, we pick one example from the training set, uniformly at random, and update the weight vector based on the chosen example. The attractiveness of stochastic gradient descent methods is that their runtime do not depend at all on the number of examples, and can even sometime decrease with the number of examples (see Bottou and Bousquet, 2008; Shalev-Shwartz and Srebro, 2008). Unfortunately, the stochastic gradient descent method fails to produce sparse solutions, which makes the algorithm both slower and less attractive as sparsity is one of the major reasons to use ℓ_1 regularization. To overcome this problem, two variants were recently proposed. First, Duchi et al. (2008) suggested to replace the ℓ_1 regularization term with a constraint of the form $\|\mathbf{w}\|_1 \leq B$, and then to use stochastic gradient projection procedure. Another solution, which uses the regularization form given in (1), has been proposed by Langford et al. (2009) and is called truncated gradient descent. In this approach, the elements of $\mathbf{w}$ that cross 0 after the stochastic gradient step are truncated to 0, hence sparsity is achieved. The disadvantage of both Duchi et al. (2008) and Langford et al. (2009) methods is that, in some situations, their runtime might grow quadratically with the dimension d, even if the optimal predictor $\mathbf{w}^\star$ is very sparse (see Section 1.1 below for details). This quadratic dependence on d can be avoided if one uses mirror descent updates (Beck and Teboulle, 2003) such as the exponentiated gradient approach (Littlestone, 1988; Kivinen and Warmuth, 1997; Beck and Teboulle, 2003). However, this approach again fails to produce sparse solutions. In this paper, we combine the idea of truncating the gradient (Langford et al., 2009) with another variant of stochastic mirror descent, which is based on p-norm updates (Grove et al., 2001; Gentile, 2003). The resulting algorithm both produces sparse solutions and has $\tilde{O}(d)$ dependence on the dimension. We call the algorithm SMIDAS for "Stochastic MIrror Descent Algorithm made Sparse".

We provide runtime guarantees for SMIDAS as well. In particular, for the logistic-loss and the squared-loss we obtain the following upper bound on the runtime to achieving ϵ expected accuracy:

$$O\left(\frac{d \log(d) \|\mathbf{w}^\star\|_1^2}{\epsilon^2}\right) . \tag{3}$$

Comparing the above with the runtime bound of the stochastic coordinate descent method given in (2) we note three major differences. First, while the bound in (2) depends on the number of examples, m, the runtime of SMIDAS does not depend on m at all. On the flip side, the dependence of stochastic coordinate descent on the dimension is better both because the lack of the term $\log(d)$ and because $\|\mathbf{w}^\star\|_2^2$ is always smaller than $\|\mathbf{w}^\star\|_1^2$ (the ratio is at most d). Last, the dependence on $\frac{1}{\epsilon}$ is linear in (2) and quadratic in (3). If ϵ is the same order as the objective value at $\mathbf{w}^\star$, it is possible to improve the dependence on $1/\epsilon$ (Proposition 4). Finally, we would like to point out that while the stochastic coordinate descent method is parameter free, the success of SMIDAS and of the method of Langford et al. (2009), depends on a careful tuning of a learning rate parameter.

1.1 Related Work

We now survey several existing methods and in particular show how our stochastic twist enables us to give superior runtime guarantees.

1.1.1 COORDINATE DESCENT METHODS FOR ℓ_1 REGULARIZATION

Following the Gauss-Siedel approach of Zhang and Oles (2001), Genkin et al. (2007) described a coordinate descent method (called BBR) for minimizing ℓ_1 regularized objectives. This approach is similar to our method, with three main differences. First, and most important, at each iteration we choose a coordinate uniformly at random. This allows us to provide theoretical runtime guarantees. We note that no theoretical guarantees are provided by Zhang and Oles (2001) and Genkin et al. (2007). Second, we solely use gradient information which makes our algorithm parameters-free and extremely simple to implement. In contrast, the Gauss-Siedel approach is more complicated and involves second order information, or a line search procedure, or a trusted region Newton step. Last, the generality of our derivation allows us to tackle a more general problem. For example, it is easy to deal with additional boxed constraints. Friedman et al. (2010) generalized the approach of Genkin et al. (2007) to include the case of elastic-net regularization. In a series of experiments, they observed that cyclic coordinate descent outperforms many alternative popular methods such as LARS (Efron et al., 2004), an interior point method called `l1lognet` (Koh et al., 2007), and the Lasso Penalized Logistic (LPL) program (Wu and Lange, 2008). However, no theoretical guarantees are provided in Friedman et al. (2010) as well. Our analysis can partially explain the experimental result of Friedman et al. (2010) since updating the coordinates in a cyclic order can in practice be very similar to stochastic updates.

Luo and Tseng (1992) established a linear convergence result for coordinate descent algorithms. This convergence result tells us that after an unspecified number of iterations, the algorithm converges very fast to the optimal solution. However, this analysis is useless in data laden domains as it can be shown that the initial unspecified number of iterations depends at least quadratically on the number of training examples. In an attempt to improve the dependence on the size of the problem, Tseng and Yun (2009) recently studied other variants of block coordinate descent for optimizing 'smooth plus separable' objectives. In particular, ℓ_1 regularized loss minimization (1) is of this form, provided that the loss function is smooth. The algorithm proposed by Tseng and Yun (2009) is not stochastic. Translated to our notation, the runtime bound given in Tseng and Yun (2009) is

of order[2] $\frac{md^2\beta\|\mathbf{w}^\star\|_2^2}{\varepsilon}$. This bound is inferior to our runtime bound for stochastic coordinate descent given in (2) by a factor of the dimension d.

1.1.2 COORDINATE DESCENT METHODS FOR ℓ_1 DOMAIN CONSTRAINTS

A different, but related, optimization problem is to minimize the loss, $\frac{1}{m}\sum_i L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)$, subject to a domain constraint of the form $\|\mathbf{w}\|_1 \leq B$. Many authors presented a forward greedy selection algorithm (a.k.a. Boosting) for this problem. We refer the reader to Frank and Wolfe (1956), Zhang (2003), Clarkson (2008) and Shalev-Shwartz et al. (2010). These authors derived the upper bound $O\left(\beta\|\mathbf{w}^\star\|_1^2/\varepsilon\right)$ on the number of iterations required by this algorithm to find an ε-accurate solution. Since at each iteration of the algorithm, one needs to calculate the gradient of the loss at $\mathbf{w}$, the runtime of each iteration is md. Therefore, the total runtime becomes $O\left(md\beta\|\mathbf{w}^\star\|_1^2/\varepsilon\right)$. Note that this bound is better than the bound given by Tseng and Yun (2009), since for any vector in $\mathbb{R}^d$ we have $\|\mathbf{w}\|_1 \leq \sqrt{d}\,\|\mathbf{w}\|_2$. However, the boosting bound given above is still inferior to our bound given in (2) since $\|\mathbf{w}^\star\|_1 \geq \|\mathbf{w}^\star\|_2$. Furthermore, in the extreme case we have $\|\mathbf{w}^\star\|_1^2 = d\|\mathbf{w}^\star\|_2^2$, thus our bound can be better than the boosting bound by a factor of d. Lemma 5 in Appendix A shows that the *iteration* bound (not runtime) of *any* algorithm cannot be smaller than $\Omega(\|\mathbf{w}^\star\|_1^2/\varepsilon)$ (see also the lower bounds in Shalev-Shwartz et al., 2010). This seems to imply that *any* deterministic method, which goes over the entire data at each iteration, will induce a runtime which is inferior to the runtime we derive for stochastic coordinate descent.

1.1.3 STOCHASTIC GRADIENT DESCENT AND MIRROR DESCENT

Stochastic gradient descent (SGD) is considered to be one of the best methods for large scale loss minimization, when we measure how fast a method achieves a certain generalization error. This has been observed in experiments (Bottou, Web Page) and also has been analyzed theoretically by Bottou and Bousquet (2008) and Shalev-Shwartz and Srebro (2008).

As mentioned before, one can apply SGD for solving (1). However, SGD fails to produce sparse solutions. Langford et al. (2009) proposed an elegant simple modification of the SGD update rule that yields a variant of SGD with sparse intermediate solutions. They also provide bounds on the runtime of the resulting algorithm. In the general case (i.e., without assuming low objective relative to ε), their analysis implies the following runtime bound

$$O\left(\frac{d\,\|\mathbf{w}^\star\|_2^2 X_2^2}{\varepsilon^2}\right), \tag{4}$$

where $X_2^2 = \frac{1}{m}\sum_i \|\mathbf{x}_i\|_2^2$ is the average squared norm of an instance. Comparing this bound with our bound in (3), we observe that none of the bounds dominates the other, and their relative performance depends on properties of the training set and the optimal solution $\mathbf{w}^\star$. Specifically, if $\mathbf{w}^\star$ has only $k \ll d$ non-zero elements and each $\mathbf{x}_i$ is dense (say $\mathbf{x}_i \in \{-1,+1\}^d$), then the ratio between the above bound of SGD and the bound in (3) becomes $\frac{d}{k\log(d)} \gg 1$. On the other hand, if $\mathbf{x}_i$ has only k non-zeros while $\mathbf{w}^\star$ is dense, then the ratio between the bounds can be $\frac{k}{d\log(d)} \ll 1$. Although the relative performance is data dependent, in most applications if one prefers ℓ_1 regularization over ℓ_2

2. To see this, note that the iterations bound in Equation (21) of Tseng and Yun (2009) is: $\frac{\beta\|\mathbf{w}^\star\|_2^2}{\varepsilon\nu}$, and using Equation (25) in Section 6, we can set the value of ν to be $\nu = 1/d$ (since in our case there are no linear constraints). The complexity bound now follows from the fact that the cost of each iteration is $O(d\,m)$.

regularization, he should also believe that $\mathbf{w}^\star$ is sparse, and thus our runtime bound in (3) is likely to be superior.[3]

The reader familiar with the online learning and mirror descent literature will not be surprised by the above discussion. Bounds that involved $\|\mathbf{w}^\star\|_1$ and $\|\mathbf{x}_i\|_\infty$, as in (3), are well known and the relative performance discussed above was pointed out in the context of additive vs. multiplicative updates (see, e.g., Kivinen and Warmuth, 1997). However, the most popular algorithm for obtaining bounds of the form given in (3) is the EG approach (Kivinen and Warmuth, 1997), which involves the exponential potential, and this algorithm cannot yield intermediate sparse solutions. One of the contributions of this paper is to show that with a different potential, which is called the *p*-norm potential, one can obtain the bound given in (3) while still enjoying sparse intermediate solutions.

1.1.4 RECENT WORKS DEALING WITH STOCHASTIC METHODS FOR LARGE SCALE REGULARIZED LOSS MINIMIZATION

Since the publication of the conference version (Shalev-Shwartz and Tewari, 2009) of this paper, several papers proposing stochastic algorithms for regularized loss minimization have appeared. Of these, we would like to mention a few that are especially connected to the themes pursued in the present paper. Regularized Dual Averaging (RDA) of Xiao (2010) uses a running average of all the past subgradients of the loss function and the regularization term to generate its iterates. He develops a *p*-norm RDA method that is closely related to SMIDAS. The theoretical bounds for SMIDAS and *p*-norm RDA are similar but the latter employs a more aggressive truncation schedule that can potentially lead to sparser iterates.

SMIDAS deals with ℓ_1 regularization. The Composite Objective MIrror Descent (COMID) algorithm of Duchi et al. (2010) generalizes the idea behind SMIDAS to deal with general regularizers provided a certain minimization problem involving a Bregman divergence and the regularizer is efficiently solvable. Viewing the average loss in (1) leads to interesting connections with the area of Stochastic Convex Optimization that deals with minimizing a convex function given access to an oracle that can return unbiased estimates of the gradient of the convex function at any query point. For various classes of convex functions, one can ask: What is the optimal number of queries needed to achieve a certain accuracy (in expectation)? For developments along these lines, please see Lan (2010) and Ghadimi and Lan (2011), especially the latter since it deals with functions that are the sum of a smooth and a non-smooth but "simple" (like ℓ_1-norm) part. Finally, Nesterov (2010) has analyzed randomized versions of coordinate descent for unconstrained and constrained minimization of smooth convex functions.

2. Stochastic Coordinate Descent

To simplify the notation throughout this section, we rewrite the problem in (1) using the notation

$$\min_{\mathbf{w}\in\mathbb{R}^d} \overbrace{\underbrace{\frac{1}{m}\sum_{i=1}^{m} L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)}_{\equiv C(\mathbf{w})} + \lambda\|\mathbf{w}\|_1}^{\equiv P(\mathbf{w})} . \tag{5}$$

3. One important exception is the large scale text processing application described in Langford et al. (2009) where the dimension is so large and ℓ_1 is used simply because we cannot store a dense weight vector in memory.

We are now ready to present the stochastic coordinate descent algorithm. The algorithm initializes $\mathbf{w}$ to be $\mathbf{0}$. At each iteration, we pick a coordinate j uniformly at random from $[d]$. Then, the derivative of $C(\mathbf{w})$ w.r.t. the jth element of $\mathbf{w}$, $g_j = (\nabla C(\mathbf{w}))_j$, is calculated. That is, $g_j = \frac{1}{m}\sum_{i=1}^m L'(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) x_{i,j}$, where L' is the derivative of the loss function with respect to its first argument. Simple calculus yields

$$L'(a,y) = \begin{cases} (a-y) & \text{for squared-loss} \\ \frac{-y}{1+\exp(ay)} & \text{for logistic-loss} \end{cases} . \tag{6}$$

Next, a step size is determined based on the value of g_j and a parameter of the loss function denoted by β. This parameter is an upper bound on the second derivative of the loss. Again, for our running examples we have

$$\beta = \begin{cases} 1 & \text{for squared-loss} \\ 1/4 & \text{for logistic-loss} \end{cases} . \tag{7}$$

If there was no regularization, we would just subtract the step size g_j/β from the current value of w_j. However, to take into account the regularization term, we further add/subtract λ/β from w_j provided we do not cross 0 in the process. If we do, we let the new value of w_j be exactly 0. This is crucial for maintaining sparsity of $\mathbf{w}$. To describe the entire update succinctly, it is convenient to define the following simple "thresholding" operation:

$$s_\tau(w) = \text{sign}(w)(|w| - \tau)_+ = \begin{cases} 0 & w \in [-\tau, \tau] \\ w - \tau & w > \tau \\ w + \tau & w < -\tau \end{cases} .$$

Algorithm 1 Stochastic Coordinate Descent (SCD)

let $\mathbf{w} = \mathbf{0}$
for $t = 1, 2, \ldots$ **do**
  sample j uniformly at random from $\{1, \ldots, d\}$
  let $g_j = (\nabla C(\mathbf{w}))_j$
  $\mathbf{w}_j \leftarrow s_{\lambda/\beta}(w_j - g_j/\beta)$
end for

2.1 Efficient Implementation

We now present an efficient implementation of Algorithm 1. The simple idea is to maintain a vector $\mathbf{z} \in \mathbb{R}^m$ such that $z_i = \langle \mathbf{w}, \mathbf{x}_i \rangle$. Once we have this vector, calculating g_j on average requires $O(sm)$ iterations, where

$$s = \frac{|\{(i,j) : x_{i,j} \neq 0\}|}{md} \tag{8}$$

is the average number of non-zeros in our training set. Concretely, we obtain Algorithm 2 for logistic-loss and squared-loss.

Algorithm 2 SCD for logistic-loss and squared-loss

let $\mathbf{w} = \mathbf{0} \in \mathbb{R}^d$, $\mathbf{z} = \mathbf{0} \in \mathbb{R}^m$
for $t = 1, 2, \ldots$ **do**
  sample j uniformly at random from $\{1, \ldots, d\}$
  let L' and β be as defined in (6) and (7)
  let $g_j = \frac{1}{m} \sum_{i: x_{i,j} \neq 0} L'(z_i, y_i) x_{i,j}$
  if $w_j - g_j/\beta > \lambda/\beta$ **then**
    $w_j \leftarrow w_j - g_j/\beta - \lambda/\beta$
  else if $w_j - g_j/\beta < -\lambda/\beta$ **then**
    $w_j \leftarrow w_j - g_j/\beta + \lambda/\beta$
  else
    $w_j \leftarrow 0$
  end if
  $\forall i$ s.t. $x_{i,j} \neq 0$ let $z_i = z_i + \eta x_{i,j}$
end for

2.2 Runtime Guarantee

The following theorem establishes runtime guarantee for SCD.

Theorem 1 *Let $\mathbf{w}^\star$ be a minimizer of* (5) *where the function $C(\mathbf{w})$ is differentiable and satisfies,*

$$\forall \mathbf{w}, \eta, j,\ C(\mathbf{w} + \eta \mathbf{e}^j) \leq C(\mathbf{w}) + \eta [\nabla C(\mathbf{w})]_j + \frac{\beta}{2} \eta^2 \ . \tag{9}$$

Let $\mathbf{w}_T$ denote the weight vector $\mathbf{w}$ at the end of iteration T of Algorithm 1. Then,

$$\mathbb{E}[P(\mathbf{w}_T)] - P(\mathbf{w}^\star) \ \leq \ \frac{d\, \Psi(0)}{T+1} \ ,$$

where

$$\Psi(\mathbf{w}) = \frac{\beta}{2} \|\mathbf{w}^\star - \mathbf{w}\|_2^2 + P(\mathbf{w})$$

and the expectation is over the algorithm's own randomization.

Proof To simplify the proof, let us rewrite the update as $w_j \leftarrow w_j + \eta_j$ where $\eta_j = s_{\lambda/\beta}(w_j - g_j/\beta) - w_j$. We first show that

$$\eta_j = \operatorname*{argmin}_\eta \left(\eta g_j + \frac{\beta}{2} \eta^2 + \lambda |w_{t-1,j} + \eta| \right) \ . \tag{10}$$

Indeed, if η is a solution of the above then by optimality conditions, we must have,

$$0 = g_j + \beta \eta + \lambda \rho_j \ ,$$

where $\rho_j \in \partial |w_{t-1,j} + \eta|$, the sub-differential of the absolute value function at $w_{t-1,j} + \eta$. Since $\rho_j = \text{sign}(w_{t-1,j} + \eta)$ if $w_{t-1,j} + \eta \neq 0$ and otherwise $\rho_j \in [-1, 1]$, we obtain that:

$$\begin{aligned}
&\text{If } \eta > -w_{t-1,j} \ \Rightarrow \ \rho_j = 1 \ \Rightarrow \ \eta = \tfrac{-g_j - \lambda}{\beta} > -w_{t-1,j} \\
&\text{If } \eta < -w_{t-1,j} \ \Rightarrow \ \rho_j = -1 \ \Rightarrow \ \eta = \tfrac{-g_j + \lambda}{\beta} < -w_{t-1,j} \\
&\text{Else } \eta = -w_{t-1,j} \ .
\end{aligned}$$

But, this is equivalent to the definition of η_j and therefore (10) holds.

Define the potential,

$$\Phi(\mathbf{w}_t) = \tfrac{1}{2}\|\mathbf{w}^\star - \mathbf{w}_t\|_2^2 \,,$$

and let $\Delta_{t,j} = \Phi(\mathbf{w}_{t-1}) - \Phi(\mathbf{w}_{t-1} + \eta_j \mathbf{e}^j)$ be the change in the potential assuming we update $\mathbf{w}_{t-1}$ using coordinate j. Since $0 = g_j + \beta\eta_j + \lambda\rho_j$, we have that,

$$\begin{aligned}
\Delta_{t,j} &= \tfrac{1}{2}\|\mathbf{w}^\star - \mathbf{w}_{t-1}\|_2^2 - \tfrac{1}{2}\|\mathbf{w}^\star - \mathbf{w}_{t-1} - \eta_j \mathbf{e}^j\|_2^2 \\
&= \tfrac{1}{2}(w_j^\star - w_{t-1,j})^2 - \tfrac{1}{2}(w_j^\star - w_{t-1,j} - \eta_j)^2 \\
&= \tfrac{1}{2}\eta_j^2 - \eta_j(w_{t-1,j} + \eta_j - w_j^\star) \\
&= \tfrac{1}{2}\eta_j^2 + \frac{g_j}{\beta}(w_{t-1,j} + \eta_j - w_j^\star) + \frac{\lambda\rho_j}{\beta}(w_{t-1,j} + \eta_j - w_j^\star).
\end{aligned}$$

Next, we note that

$$\rho_j(w_{t-1,j} + \eta_j - w_j^\star) \geq |w_{t-1,j} + \eta_j| - |w_j^\star| \,,$$

which yields

$$\Delta_{t,j} \geq \tfrac{1}{2}\eta_j^2 + \frac{g_j}{\beta}(w_{t-1,j} + \eta_j - w_j^\star) + \frac{\lambda}{\beta}(|w_{t-1,j} + \eta_j| - |w_j^\star|) \,.$$

By (9), we have,

$$C(\mathbf{w}_{t-1} + \eta_j \mathbf{e}^j) - C(\mathbf{w}_{t-1}) \leq g_j\eta_j + \frac{\beta}{2}\eta_j^2 \,,$$

and thus

$$\Delta_{t,j} \geq \frac{1}{\beta}(C(\mathbf{w}_{t-1} + \eta_j \mathbf{e}^j) - C(\mathbf{w}_{t-1})) + \frac{g_j}{\beta}(w_{t-1,j} - w_j^\star) + \frac{\lambda}{\beta}(|w_{t-1,j} + \eta_j| - |w_j^\star|) \,.$$

Taking expectations (with respect to the choice of j and conditional on $\mathbf{w}_{t-1}$) on both sides, we get,

$$\begin{aligned}
&\mathbb{E}[\Phi(\mathbf{w}_{t-1}) - \Phi(\mathbf{w}_t) \,|\, \mathbf{w}_{t-1}] \;=\; \frac{1}{d}\sum_{k=1}^d \Delta_{t,k} \\
&\geq \frac{1}{\beta d}\left[\sum_{k=1}^d (C(\mathbf{w}_{t-1} + \eta_k \mathbf{e}^k) - C(\mathbf{w}_{t-1})) + \sum_{k=1}^d g_k(w_{t-1,k} - w_k^\star) + \lambda\sum_{k=1}^d (|w_{t-1,k} + \eta_k| - |w_k^\star|)\right] \\
&= \frac{1}{\beta d}\left[\sum_{k=1}^d (C(\mathbf{w}_{t-1} + \eta_k \mathbf{e}^k) - C(\mathbf{w}_{t-1})) + \langle \nabla C(\mathbf{w}_{t-1}), \mathbf{w}_{t-1} - \mathbf{w}^\star\rangle + \lambda\sum_{k=1}^d (|w_{t-1,k} + \eta_k| - |w_k^\star|)\right] \\
&\geq \frac{1}{\beta d}\left[\sum_{k=1}^d (C(\mathbf{w}_{t-1} + \eta_k \mathbf{e}^k) - C(\mathbf{w}_{t-1})) + C(\mathbf{w}_{t-1}) - C(\mathbf{w}^\star) + \lambda\sum_{k=1}^d (|w_{t-1,k} + \eta_k| - |w_k^\star|)\right] \\
&= \frac{1}{\beta}\left[\mathbb{E}[C(\mathbf{w}_t) \,|\, \mathbf{w}_{t-1}] - C(\mathbf{w}_{t-1}) + \frac{C(\mathbf{w}_{t-1}) - C(\mathbf{w}^\star)}{d} + \frac{\lambda}{d}\sum_{k=1}^d |w_{t-1,k} + \eta_k| - \frac{\lambda\|\mathbf{w}^\star\|_1}{d}\right] \,,
\end{aligned}$$

where the second inequality follows from the convexity of C. Note that, we have,

$$\begin{aligned}\mathbb{E}[\|\mathbf{w}_t\|_1 \,|\, \mathbf{w}_{t-1}] &= \frac{1}{d}\sum_{k=1}^{d} \|\mathbf{w}_{t-1} + \eta_k \mathbf{e}^k\|_1 \\ &= \frac{1}{d}\sum_{k=1}^{d} (\|\mathbf{w}_{t-1}\|_1 - |w_{t-1,k}| + |w_{t-1,k} + \eta_k|) \\ &= \|\mathbf{w}_{t-1}\|_1 - \frac{1}{d}\|\mathbf{w}_{t-1}\|_1 + \frac{1}{d}\sum_{k=1}^{d} |w_{t-1,k} + \eta_k| \ .\end{aligned}$$

Plugging this above gives us,

$$\begin{aligned}&\beta\mathbb{E}[\Phi(\mathbf{w}_{t-1}) - \Phi(\mathbf{w}_t) \,|\, \mathbf{w}_{t-1}] \\ &\geq \mathbb{E}[C(\mathbf{w}_t) + \lambda\|\mathbf{w}_t\|_1 \,|\, \mathbf{w}_{t-1}] - C(\mathbf{w}_{t-1}) - \lambda\|\mathbf{w}_{t-1}\|_1 + \frac{C(\mathbf{w}_{t-1}) + \lambda\|\mathbf{w}_{t-1}\|_1 - C(\mathbf{w}^\star) - \lambda\|\mathbf{w}^\star\|_1}{d} \\ &= \mathbb{E}[P(\mathbf{w}_t) \,|\, \mathbf{w}_{t-1}] - P(\mathbf{w}_{t-1}) + \frac{P(\mathbf{w}_{t-1}) - P(\mathbf{w}^\star)}{d} \ .\end{aligned}$$

This is equivalent to,

$$\mathbb{E}[\beta\Phi(\mathbf{w}_{t-1}) + P(\mathbf{w}_{t-1}) - \beta\Phi(\mathbf{w}_t) - P(\mathbf{w}_t) \,|\, \mathbf{w}_{t-1}] \geq \frac{P(\mathbf{w}_{t-1}) - P(\mathbf{w}^\star)}{d} \ .$$

Thus, defining the composite potential,

$$\Psi(\mathbf{w}) = \beta\Phi(\mathbf{w}) + P(\mathbf{w}) \ ,$$

and taking full expectations, we get,

$$\mathbb{E}[\Psi(\mathbf{w}_{t-1}) - \Psi(\mathbf{w}_t)] \geq \frac{1}{d}\mathbb{E}[P(\mathbf{w}_{t-1}) - P(\mathbf{w}^\star)] \ .$$

Summing over $t = 1, \ldots, T+1$ and realizing that $P(\mathbf{w}_t)$ monotonically decreases gives,

$$\begin{aligned}\mathbb{E}\left[\tfrac{T+1}{d}(P(\mathbf{w}_T) - P(\mathbf{w}^\star))\right] &\leq \mathbb{E}\left[\tfrac{1}{d}\sum_{t=1}^{T+1}(P(\mathbf{w}_{t-1}) - P(\mathbf{w}^\star))\right] \\ &\leq \mathbb{E}\left[\sum_{t=1}^{T+1}(\Psi(\mathbf{w}_{t-1}) - \Psi(\mathbf{w}_t))\right] \\ &= \mathbb{E}\left[\Psi(\mathbf{w}_0) - \Psi(\mathbf{w}_{T+1})\right] \ \leq \ \mathbb{E}\left[\Psi(\mathbf{w}_0)\right] \ = \ \Psi(0) \ .\end{aligned}$$

■

The above theorem bounds the expected performance of SCD. We next give bounds that hold with high probability.

Theorem 2 *Assume that the conditions of Theorem 1 holds. Then, with probability of at least* $1/2$ *we have that*

$$P(\mathbf{w}_T) - P(\mathbf{w}^\star) \ \leq \ \frac{2d\Psi(0)}{T+1} \ .$$

Furthermore, for any $\delta \in (0,1)$*, suppose we run SCD* $r = \lceil \log_2(1/\delta) \rceil$ *times, each time* T *iterations, and let* $\mathbf{w}$ *be the best solution out of the* r *obtained solutions, then with probability of at least* $1-\delta$*,*

$$P(\mathbf{w}) - P(\mathbf{w}^\star) \;\leq\; \frac{2d\,\Psi(0)}{T+1}\,.$$

Proof The random variable $P(\mathbf{w}_T) - P(\mathbf{w}^\star)$ is non-negative and therefore the first inequality follows from Markov's inequality using Theorem 1. To prove the second result, note that the probability that on all r rounds it holds that $P(\mathbf{w}_T) - P(\mathbf{w}^\star) > \frac{2d\Psi(0)}{T+1}$ is at most $2^{-r} \leq \delta$, which concludes our proof. ■

Next, we specify the runtime bound for the case of ℓ_1 regularized logistic-regression and squared-loss. First, Lemma 6 in Appendix B shows that for C as defined in (5), if the second derivative of L is bounded by β then the condition on C given in Theorem 1 holds. Additionally, for the logistic-loss we have $C(0) \leq 1$. Therefore, for logistic-loss, after performing

$$\frac{d\,(\frac{1}{4}\,\|\mathbf{w}^\star\|_2^2 + 2)}{\varepsilon}$$

iterations of Algorithm 2 we achieve (expected) ε-accuracy in the objective P. Since the average cost of each iteration is sm, where s is as defined in (8), we end up with the total runtime

$$\frac{smd\,(\frac{1}{4}\,\|\mathbf{w}^\star\|_2^2 + 2)}{\varepsilon}\,.$$

The above is the runtime required to achieve expected ε-accuracy. Using Theorem 2 the required runtime to achieve ε-accuracy with a probability of at least $1-\delta$ is

$$smd\left(\frac{(\frac{1}{2}\,\|\mathbf{w}^\star\|_2^2 + 4)}{\varepsilon} + \lceil \log(1/\delta) \rceil\right)\,.$$

For the squared-loss we have $C(0) = \frac{1}{m}\sum_i y_i^2$. Assuming that the targets are normalized so that $C(0) \leq 1$, and using similar derivation we obtain the total runtime bound

$$smd\left(\frac{(2\|\mathbf{w}^\star\|_2^2 + 4)}{\varepsilon} + \lceil \log(1/\delta) \rceil\right)\,.$$

3. Stochastic Mirror Descent Made Sparse

In this section, we describe our mirror descent approach for ℓ_1 regularized loss minimization that maintains intermediate sparse solutions. Recall that we rewrite the problem in (1) using the notation

$$\min_{\mathbf{w}\in\mathbb{R}^d} \; \overbrace{\underbrace{\frac{1}{m}\sum_{i=1}^m L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)}_{\equiv C(\mathbf{w})} + \lambda\|\mathbf{w}\|_1}^{\equiv P(\mathbf{w})}\,. \tag{11}$$

Mirror descent algorithms (Nemirovski and Yudin, 1978, Chapter 3) maintain two weight vectors: primal $\mathbf{w}$ and dual θ. The connection between the two vectors is via a link function $\theta = f(\mathbf{w})$, where $f : \mathbb{R}^d \to \mathbb{R}^d$. The link function is always taken to be the gradient map ∇F of some strictly convex function F and is therefore invertible. We can thus also write $\mathbf{w} = f^{-1}(\theta)$. In our mirror descent variant, we use the p-norm link function. That is, the jth element of f is

$$f_j(\mathbf{w}) \;=\; \frac{\operatorname{sign}(\mathbf{w}_j)\,|\mathbf{w}_j|^{q-1}}{\|\mathbf{w}\|_q^{q-2}}\;,$$

where $\|\mathbf{w}\|_q = (\sum_j |w_j|^q)^{1/q}$. Note that f is simply the gradient of the function $\frac{1}{2}\|\mathbf{w}\|_q^2$. The inverse function is (see, e.g., Gentile, 2003)

$$f_j^{-1}(\theta) \;=\; \frac{\operatorname{sign}(\theta_j)\,|\theta_j|^{p-1}}{\|\theta\|_p^{p-2}}\;, \tag{12}$$

where $p = q/(q-1)$.

We first describe how mirror descent algorithms can be applied to the objective $C(\mathbf{w})$ without the ℓ_1 regularization term. At each iteration of the algorithm, we first sample a training example i uniformly at random from $\{1,\ldots,m\}$. We then estimate the gradient of $C(\mathbf{w})$ by calculating the vector $\mathbf{v} \;=\; L'(\langle \mathbf{w},\mathbf{x}_i\rangle, y_i)\,\mathbf{x}_i$. Note that the expectation of $\mathbf{v}$ over the random choice of i is $\mathbb{E}[\mathbf{v}] \;=\; \nabla C(\mathbf{w})$. That is, $\mathbf{v}$ is an unbiased estimator of the gradient of $C(\mathbf{w})$. Next, we update the dual vector according to $\theta = \theta - \eta\,\mathbf{v}$. If the link function is the identity mapping, this step is identical to the update of stochastic gradient descent. However, in our case f is not the identity function and it is important to distinguish between θ and $\mathbf{w}$. The above update of θ translates to an update of $\mathbf{w}$ by applying the link function $\mathbf{w} = f^{-1}(\theta)$. So far, we ignored the additional ℓ_1 regularization term. The simplest way to take this term into account is by also subtracting from θ the gradient of the term $\lambda\|\mathbf{w}\|_1$. (More precisely, since the ℓ_1 norm is not differentiable, we will use any subgradient of $\|\mathbf{w}\|_1$ instead, for example, the vector whose jth element is $\operatorname{sign}(w_j)$, where we interpret $\operatorname{sign}(0) = 0$.) Therefore, we could have redefined the update of θ to be $\theta_j = \theta_j - \eta(v_j + \lambda\operatorname{sign}(w_j))$. Unfortunately, as noted in Langford et al. (2009), this update leads to a dense vector θ, which in turn leads to a dense vector $\mathbf{w}$. The solution proposed in Langford et al. (2009) breaks the update into three phases. First, we let $\tilde{\theta} = \theta - \eta\,\mathbf{v}$. Second, we let $\hat{\theta} = \tilde{\theta} - \eta\,\lambda\operatorname{sign}(\tilde{\theta})$. Last, if in the second step we crossed the zero value, that is, $\operatorname{sign}(\hat{\theta}_j) \neq \operatorname{sign}(\tilde{\theta}_j)$, then we truncate the jth element to be zero. Intuitively, the goal of the first step is to decrease the value of $C(\mathbf{w})$ and this is done by a (mirror) gradient step, while the goal of the second and third steps is to decrease the value of $\lambda\|\mathbf{w}\|_1$. So, by truncating θ at zero we make the value of $\lambda\|\mathbf{w}\|_1$ even smaller.

3.1 Runtime Guarantee

We now provide runtime guarantees for Algorithm 3. We introduce two types of assumptions on the loss function:

$$|L'(a,y)| \;\leq\; \rho\;, \tag{13}$$
$$|L'(a,y)|^2 \;\leq\; \rho\,L(a,y)\;. \tag{14}$$

In the above, L' is the derivative w.r.t. the first argument and can also be a sub-gradient of L if L is not differentiable. It is easy to verify that (14) holds for the squared-loss with $\rho = 4$ and that (13)

Algorithm 3 Stochastic Mirror Descent Algorithm mAde Sparse (SMIDAS)

parameter: $\eta > 0$
let $p = 2\ln(d)$ and let f^{-1} be as in (12)
let $\theta = 0, \mathbf{w} = 0$
for $t = 1, 2, \ldots$ **do**
  sample i uniformly at random from $\{1, \ldots, m\}$
  let $\mathbf{v} = L'(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i)\, \mathbf{x}_i$
  (L' is the derivative of L. See, for example, (6))
  let $\tilde{\theta} = \theta - \eta \mathbf{v}$
  let $\forall j, \theta_j = \text{sign}(\tilde{\theta}_j) \max\{0, |\tilde{\theta}_j| - \eta\lambda\}$
  let $\mathbf{w} = f^{-1}(\theta)$
end for

holds for the hinge-loss, $L(a,y) = \max\{0, 1 - ya\}$, with $\rho = 1$. Interestingly, for the logistic-loss, both (13) holds with $\rho = 1$ and (14) holds with $\rho = 1/2$.

Theorem 3 *Let $\mathbf{w}^\star$ be a minimizer of* (11)*. Suppose Algorithm 3 is run for $T-1$ iterations. Denote the value of $\mathbf{w}$ at the end of iteration t by $\mathbf{w}_t$ (with $\mathbf{w}_0 = 0$) and set $\mathbf{w}_o = \mathbf{w}_r$ for r chosen uniformly at random from $0, \ldots, T-1$.*

1. *If L satisfies* (13) *then,*

$$\mathbb{E}[P(\mathbf{w}_o)] - P(\mathbf{w}^\star) \leq \frac{\eta(p-1)\rho^2 e}{2} + \frac{1}{\eta T} \|\mathbf{w}^\star\|_1^2 .$$

   In particular, if we set

$$\eta = \frac{\|\mathbf{w}^\star\|_1}{\rho} \sqrt{\frac{2}{(p-1)\, e\, T}} ,$$

   then we have,

$$\mathbb{E}[P(\mathbf{w}_o)] - P(\mathbf{w}^\star) \leq \rho \|\mathbf{w}^\star\|_1 \sqrt{\tfrac{12 \log(d)}{T}} .$$

2. *If L satisfies* (14) *then,*

$$\mathbb{E}[P(\mathbf{w}_o)] - P(\mathbf{w}^\star) \leq \left(\frac{1}{1 - \frac{\eta(p-1)\rho e}{2}} - 1 \right) P(0) + \frac{\|\mathbf{w}^\star\|_1^2}{\eta\, T\, (1 - \frac{\eta(p-1)\rho e}{2})} .$$

   In particular, if we set

$$\eta = \frac{\|\mathbf{w}^\star\|_1^2}{P(0)\, T} \left(\sqrt{1 + \frac{2 P(0)\, T}{(p-1)\, \rho\, e\, \|\mathbf{w}^\star\|_1^2}} - 1 \right) ,$$

   then we have,

$$\mathbb{E}[P(\mathbf{w}_o)] - P(\mathbf{w}^\star) \leq 4 \|\mathbf{w}^\star\|_1 \sqrt{\frac{6 \rho \log(d)\, P(0)}{2T}} + \frac{12 \rho \log(d)\, \|\mathbf{w}^\star\|_1^2}{T} .$$

In both cases, the expectation is with respect to the algorithm's own randomization.

Proof We first give the proof for the case when (13) holds. Let θ_t be the value of θ at the beginning of iteration t of the algorithm, let $\mathbf{v}_t$ be the value of $\mathbf{v}$, and let $\tilde{\theta}_t = \theta_t - \eta \mathbf{v}_t$. Let $\mathbf{w}_t = f^{-1}(\theta_t)$ and $\tilde{\mathbf{w}}_t = f^{-1}(\tilde{\theta}_t)$ where f^{-1} is as defined in (12). Recall that $f(\mathbf{w}) = \nabla F(\mathbf{w})$ where $F(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_q^2$. Consider the Bregman divergence,

$$\begin{aligned}\Delta_F(\mathbf{w}, \mathbf{w}') &= F(\mathbf{w}) - F(\mathbf{w}') - \langle \nabla F(\mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle \\ &= F(\mathbf{w}) - F(\mathbf{w}') - \langle f(\mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle \ ,\end{aligned}$$

and define the potential,

$$\Psi(\mathbf{w}) = \Delta_F(\mathbf{w}^\star, \mathbf{w}) \ .$$

We first rewrite the change in potential as

$$\Psi(\mathbf{w}_t) - \Psi(\mathbf{w}_{t+1}) = (\Psi(\mathbf{w}_t) - \Psi(\tilde{\mathbf{w}}_t)) + (\Psi(\tilde{\mathbf{w}}_t) - \Psi(\mathbf{w}_{t+1})) \ , \tag{15}$$

and bound each of the two summands separately.

Definitions of Δ_F, Ψ and simple algebra yield,

$$\begin{aligned}\Psi(\mathbf{w}_t) - \Psi(\tilde{\mathbf{w}}_t) &= \Delta_F(\mathbf{w}^\star, \mathbf{w}_t) - \Delta_F(\mathbf{w}^\star, \tilde{\mathbf{w}}_t) \\ &= F(\tilde{\mathbf{w}}_t) - F(\mathbf{w}_t) - \langle f(\mathbf{w}_t) - f(\tilde{\mathbf{w}}_t), \mathbf{w}^\star \rangle + \langle f(\mathbf{w}_t), \mathbf{w}_t \rangle - \langle f(\tilde{\mathbf{w}}_t), \tilde{\mathbf{w}}_t \rangle \\ &= \Delta_F(\tilde{\mathbf{w}}_t, \mathbf{w}_t) + \langle f(\mathbf{w}_t) - f(\tilde{\mathbf{w}}_t), \tilde{\mathbf{w}}_t - \mathbf{w}^\star \rangle \qquad (16) \\ &= \Delta_F(\tilde{\mathbf{w}}_t, \mathbf{w}_t) + \langle \theta_t - \tilde{\theta}_t, \tilde{\mathbf{w}}_t - \mathbf{w}^\star \rangle \\ &= \Delta_F(\tilde{\mathbf{w}}_t, \mathbf{w}_t) + \langle \eta \mathbf{v}_t, \tilde{\mathbf{w}}_t - \mathbf{w}^\star \rangle \\ &= \Delta_F(\tilde{\mathbf{w}}_t, \mathbf{w}_t) + \langle \eta \mathbf{v}_t, \mathbf{w}_t - \mathbf{w}^\star \rangle + \langle \eta \mathbf{v}_t, \tilde{\mathbf{w}}_t - \mathbf{w}_t \rangle \ . \qquad (17)\end{aligned}$$

By strong convexity of F with respect to the q-norm (see, e.g., Section A.4 of Shalev-Shwartz, 2007), we have

$$\Delta_F(\tilde{\mathbf{w}}_t, \mathbf{w}_t) \geq \tfrac{q-1}{2}\|\tilde{\mathbf{w}}_t - \mathbf{w}_t\|_q^2 \ .$$

Moreover, using Fenchel-Young inequality with the conjugate functions $g(\mathbf{x}) = \frac{q-1}{2}\|\mathbf{x}\|_q^2$ and $g^\star(\mathbf{x}) = \frac{1}{2(q-1)}\|\mathbf{x}\|_p^2$ we have

$$|\langle \eta \mathbf{v}_t, \tilde{\mathbf{w}}_t - \mathbf{w}^\star \rangle| \leq \tfrac{\eta^2}{2(q-1)}\|\mathbf{v}_t\|_p^2 + \tfrac{q-1}{2}\|\tilde{\mathbf{w}}_t - \mathbf{w}^\star\|_q^2 \ .$$

Plugging these into (17), we get

$$\begin{aligned}\Psi(\mathbf{w}_t) - \Psi(\tilde{\mathbf{w}}_t) &\geq \eta \langle \mathbf{v}_t, \mathbf{w}_t - \mathbf{w}^\star \rangle - \tfrac{\eta^2}{2(q-1)}\|\mathbf{v}_t\|_p^2 \\ &= \eta \langle \mathbf{v}_t, \mathbf{w}_t - \mathbf{w}^\star \rangle - \tfrac{\eta^2(p-1)}{2}\|\mathbf{v}_t\|_p^2 \ .\end{aligned}$$

By convexity of L, we have,

$$\langle \mathbf{v}_t, \mathbf{w}_t - \mathbf{w}^\star \rangle \geq L(\langle \mathbf{w}_t, \mathbf{x}_i \rangle, y_i) - L(\langle \mathbf{w}^\star, \mathbf{x}_i \rangle, y_i) \ ,$$

and therefore

$$\Psi(\mathbf{w}_t) - \Psi(\tilde{\mathbf{w}}_t) \geq \eta(L(\langle \mathbf{w}_t, \mathbf{x}_i \rangle, y_i) - L(\langle \mathbf{w}^\star, \mathbf{x}_i \rangle, y_i)) - \tfrac{\eta^2(p-1)}{2}\|\mathbf{v}_t\|_p^2 \ .$$

From (13), we obtain that

$$\|\mathbf{v}_t\|_p^2 \leq \left(\|\mathbf{v}_t\|_\infty d^{1/p}\right)^2 \leq \rho^2 d^{2/p} = \rho^2 e \ . \tag{18}$$

Thus,

$$\Psi(\mathbf{w}_t) - \Psi(\tilde{\mathbf{w}}_t) \geq \eta(L(\langle \mathbf{w}_t, \mathbf{x}_i\rangle, y_i) - L(\langle \mathbf{w}^\star, \mathbf{x}_i\rangle, y_i)) - \tfrac{\eta^2(p-1)\rho^2 e}{2} \ . \tag{19}$$

So far, our analysis has followed the standard analysis of mirror descent (see, e.g., Beck and Teboulle, 2003). It is a bit more tricky to show that

$$\Psi(\tilde{\mathbf{w}}_t) - \Psi(\mathbf{w}_{t+1}) \geq \eta\lambda(\|\mathbf{w}_{t+1}\|_1 - \|\mathbf{w}^\star\|_1) \ . \tag{20}$$

To show this, we begin the same way as we did to obtain (16),

$$\begin{aligned}
\Psi(\tilde{\mathbf{w}}_t) - \Psi(\mathbf{w}_{t+1}) &= \Delta_F(\mathbf{w}_{t+1}, \tilde{\mathbf{w}}_t) + \langle f(\tilde{\mathbf{w}}_t) - f(\mathbf{w}_{t+1}), \mathbf{w}_{t+1} - \mathbf{w}^\star\rangle \\
&= \Delta_F(\mathbf{w}_{t+1}, \tilde{\mathbf{w}}_t) + \langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}_{t+1} - \mathbf{w}^\star\rangle \\
&\geq \langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}_{t+1} - \mathbf{w}^\star\rangle \\
&= \langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}_{t+1}\rangle - \langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}^\star\rangle \ .
\end{aligned} \tag{21}$$

Note that $\text{sign}(w_{t+1,j}) = \text{sign}(\theta_{t+1,j})$. Moreover, when $\theta_{t+1,j} \neq 0$ then,

$$\tilde{\theta}_{t,j} - \theta_{t+1,j} = \eta\lambda\,\text{sign}(\theta_{t+1,j}) \ .$$

Thus, we have,

$$\begin{aligned}
\langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}_{t+1}\rangle &= \sum_{j: w_{t+1,j} \neq 0} (\tilde{\theta}_{t,j} - \theta_{t+1,j}) w_{t+1,j} \\
&= \sum_{j: w_{t+1,j} \neq 0} \eta\lambda\,\text{sign}(\theta_{t+1,j}) w_{t+1,j} \\
&= \eta\lambda \sum_{j: w_{t+1,j} \neq 0} \text{sign}(w_{t+1,j}) w_{t+1,j} \\
&= \eta\lambda \|\mathbf{w}_{t+1}\|_1 \ .
\end{aligned}$$

Note that this equality is crucial and does not hold for the Bregman potential corresponding to the exponentiated gradient algorithm. Plugging the above equality, along with the inequality,

$$|\langle \tilde{\theta}_t - \theta_{t+1}, \mathbf{w}^\star\rangle| \leq \|\tilde{\theta}_t - \theta_{t+1}\|_\infty \|\mathbf{w}^\star\|_1 = \eta\,\lambda\,\|\mathbf{w}^\star\|_1$$

into (21), we get (20).

Combining the lower bounds (19) and (20) and plugging them into (15), we get,

$$\begin{aligned}
\Psi(\mathbf{w}_t) - \Psi(\mathbf{w}_{t+1}) \geq \eta(L(\langle \mathbf{w}_t, \mathbf{x}_i\rangle, y_i) - L(\langle \mathbf{w}^\star, \mathbf{x}_i\rangle, y_i)) \\
- \tfrac{\eta^2(p-1)\rho^2 e}{2} + \eta\lambda(\|\mathbf{w}_{t+1}\|_1 - \|\mathbf{w}^\star\|_1) \ .
\end{aligned}$$

Taking expectation with respect to i drawn uniformly at random from $\{1, \ldots, m\}$, we get,

$$\begin{aligned}
\mathbb{E}[\Psi(\mathbf{w}_t) - \Psi(\mathbf{w}_{t+1})] &\geq \eta\mathbb{E}[C(\mathbf{w}_t) - C(\mathbf{w}^\star)] - \tfrac{\eta^2(p-1)\rho^2 e}{2} + \eta\lambda\mathbb{E}[\|\mathbf{w}_{t+1}\|_1 - \|\mathbf{w}^\star\|_1] \\
&= \eta\mathbb{E}[P(\mathbf{w}_t) - P(\mathbf{w}^\star)] - \tfrac{\eta^2(p-1)\rho^2 e}{2} + \eta\lambda\mathbb{E}[\|\mathbf{w}_{t+1}\|_1 - \|\mathbf{w}_t\|_1] \ .
\end{aligned}$$

Summing over $t = 0, \ldots, T-1$, dividing by ηT, and rearranging gives,

$$\begin{aligned}
\tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)] - P(\mathbf{w}^\star) &\leq \tfrac{\eta(p-1)\rho^2 e}{2} + \tfrac{\lambda}{T}\,\mathbb{E}\left[\|\mathbf{w}_0\|_1 - \|\mathbf{w}_T\|_1\right] + \tfrac{1}{\eta T}\,\mathbb{E}\left[\Psi(\mathbf{w}_0) - \Psi(\mathbf{w}_T)\right] \\
&\leq \tfrac{\eta(p-1)\rho^2 e}{2} + 0 + \tfrac{1}{\eta T}\Delta_F(\mathbf{w}^\star, 0) \\
&= \tfrac{\eta(p-1)\rho^2 e}{2} + \tfrac{1}{\eta T}\|\mathbf{w}^\star\|_q^2 \\
&\leq \tfrac{\eta(p-1)\rho^2 e}{2} + \tfrac{1}{\eta T}\|\mathbf{w}^\star\|_1^2 \,. \qquad (22)
\end{aligned}$$

Now, optimizing over η gives

$$\tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)] - P(\mathbf{w}^\star) \leq \rho\|\mathbf{w}^\star\|_1\sqrt{\frac{2(p-1)e}{T}}$$

and this concludes our proof for the case when (13) holds, since for a random r we have $\mathbb{E}[P(\mathbf{w}_r)] = \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)]$.

When (14) holds, instead of the bound (18), we have,

$$\|\mathbf{v}_t\|_p^2 \leq \left(\|\mathbf{v}_t\|_\infty d^{1/p}\right)^2 \leq \rho\, L(\langle \mathbf{w}_t, \mathbf{x}_i\rangle, y_i)\, d^{2/p} = \rho\, L(\langle \mathbf{w}_t, \mathbf{x}_i\rangle, y_i)\, e \,.$$

As a result, the final bound (22) now becomes,

$$\begin{aligned}
\tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)] - P(\mathbf{w}^\star) &\leq \tfrac{\eta(p-1)\rho e}{2T}\sum_{t=0}^{T-1}\mathbb{E}[C(\mathbf{w}_t)] + \tfrac{1}{\eta T}\|\mathbf{w}^\star\|_1^2 \\
&\leq \tfrac{\eta(p-1)\rho e}{2T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)] + \tfrac{1}{\eta T}\|\mathbf{w}^\star\|_1^2 \,.
\end{aligned}$$

For the sake of brevity, let $a = (p-1)\rho e/2$ and $b = \|\mathbf{w}^\star\|_1^2$, so that the above bound can be written as,

$$\begin{aligned}
\tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[P(\mathbf{w}_t)] - P(\mathbf{w}^\star) &\leq \left(\frac{1}{1-a\eta} - 1\right)P(\mathbf{w}^\star) + \frac{b/(\eta T)}{1-a\eta} \qquad (23) \\
&\leq \left(\frac{1}{1-a\eta} - 1\right)P(0) + \frac{b/(\eta T)}{1-a\eta} \,.
\end{aligned}$$

At this stage, we need to minimize the expression on the right hand side as a function of η. This somewhat tedious but straightforward minimization is done in Lemma 7 in Appendix B. Using Lemma 7 (with $P = P(0)$), we see that the right hand side is minimized by setting

$$\eta = \frac{\|\mathbf{w}^\star\|_1^2}{P(0)\,T}\left(\sqrt{1 + \frac{2P(0)\,T}{(p-1)\,\rho\, e\,\|\mathbf{w}^\star\|_1^2}} - 1\right) \,,$$

and the minimum value is upper bounded by

$$4\|\mathbf{w}^\star\|_1\sqrt{\frac{(p-1)\,\rho\, e P(0)}{2T}} + \frac{2\,(p-1)\,\rho\, e\,\|\mathbf{w}^\star\|_1^2}{T} \,.$$

This concludes the proof for the case when (14) holds. ■

The bound in the above theorem can be improved if (14) holds and the desired accuracy is the same order as $P(\mathbf{w}^\star)$. This is the content of the next proposition.

Proposition 4 *Let $\mathbf{w}^\star$ be a minimizer of* (11)*. Suppose Algorithm 3 is run for $T-1$ iterations. Denote the value of $\mathbf{w}$ at the beginning of iteration t by $\mathbf{w}_t$ and set $\mathbf{w}_o = \mathbf{w}_r$ for r chosen uniformly at random from $0,\ldots,T-1$. If L satisfies* (14) *and we set*

$$\eta = \frac{2}{(p-1)\rho e} \cdot \frac{K}{1+K} \,,$$

for some arbitrary $K > 0$, then we have,

$$\mathbb{E}[P(\mathbf{w}_o)] \leq (1+K)P(\mathbf{w}^\star) + \frac{(1+K)^2}{K} \cdot \frac{3\rho \log(d) \|\mathbf{w}^\star\|_1^2}{T} \,.$$

Proof Plugging $\eta = K/a(1+K)$ in (23) gives,

$$\tfrac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[P(\mathbf{w}_t)] - P(\mathbf{w}^\star) \leq K P(\mathbf{w}^\star) + \frac{(1+K)^2}{K} \cdot \frac{ab}{T} \,.$$

Recalling that $p = 2\log(d)$, $a = (p-1)\rho e/2$ and $b = \|\mathbf{w}^\star\|_1^2$ concludes our proof. ■

4. Experiments

In this section, we provide experimental results for our algorithms on 4 data sets. We begin with a description of the data sets following by a description of the algorithms we ran on them.

4.1 Data Sets

We consider 4 binary classification data sets for our experiments: DUKE, ARCENE, MAGIC04S, and MAGIC04D.

DUKE is a breast cancer data set from West et al. (2001). It has 44 examples with 7,129 features with a density level of 100%. ARCENE is a data set from the UCI Machine Learning repository where the task is to distinguish cancer patterns from normal ones based on 10,000 mass-spectrometric features. Out of these, 3,000 features are synthetic features as this data set was designed for the NIPS 2003 variable selection workshop. There are 100 examples in this data set and the example matrix contains 5.4×10^5 non-zero entries corresponding to a density level of 54%. The data sets MAGIC04S and MAGIC04D were obtained by adding 1,000 random features to the MAGIC Gamma Telescope data set from the UCI Machine Learning repository. The original data set has 19,020 examples with 10 features. This is also a binary classification data set and the task is to distinguish high-energy gamma particles from background using a gamma telescope. Following the experimental setup of Langford et al. (2009), we added 1,000 random features, each of which takes value 0 with probability 0.95 or 1 with probability 0.05, to create a sparse data set, MAGIC04S. We also created a dense data set, MAGIC04D, in which the random features took value -1 or $+1$, each with probability 0.5. MAGIC04S and MAGIC04D have density levels of 5.81% and 100% respectively.

4.2 Algorithms

We ran 4 algorithms on these data sets: SCD, GCD, SMIDAS, and TRUNCGRAD. SCD is the stochastic coordinate descent algorithm given in Section 2 above. GCD is the corresponding deterministic and "greedy" version of the same algorithm. The coordinate to be updated at each iteration is *greedily* chosen in a deterministic manner to maximize a lower bound on the guaranteed decrease in the objective function. This type of deterministic criterion for choosing features is common in Boosting approaches. Since choosing a coordinate (or feature in our case) in a deterministic manner involves significant computation in case of large data sets, we expect that the deterministic algorithm will converge much slower than the stochastic algorithm. We also tried the *cyclic* version of coordinate descent that just cycles through the coordinates. We found its performance to be indistinguishable from that of SCD and hence we do not report it here. SMIDAS is the mirror descent algorithm given in Section 3 above. TRUNCGRAD is the truncated gradient algorithm of Langford et al. (2009) (In fact, Langford et al., 2009 suggests another way to truncate the gradient. Here, we refer to the variant corresponding to SMIDAS.) Of these 4, the first two are parameter-free algorithms while the latter two require a parameter η. In our experiments, we ran SMIDAS and TRUNCGRAD for a range of different values of η and chose the one that yielded the minimum value of the objective function (i.e., the regularized loss). We chose to minimize the (regularized) logistic loss in all our experiments.

4.3 Results

For each data set, we show two plots. One plot shows the regularized objective function plotted against the number of *data accesses*, that is, the number of times the algorithm accesses the data matrix $(x_{i,j})$. We choose to use this as opposed to, say CPU time, as this is an implementation independent quantity. Moreover, the actual time taken by these algorithms will be roughly proportional to this quantity provided computing features is time consuming. The second plot shows the density (or ℓ_0-norm, the number of non-zero entries) of the iterate plotted against the number of data accesses. In the next subsection, we use mild regularization ($\lambda = 10^{-6}$). Later on, we will show results for stronger regularization ($\lambda = 10^{-2}$).

4.3.1 LESS REGULARIZATION

Figure 1 is for the DUKE data set. It is clear that GCD does much worse than the other three algorithms. GCD is much slower because, as we mentioned above, it spends a lot of time in finding the best coordinate to update. The two algorithms having a tunable parameter η have roughly the same performance as SCD. However, SCD has a definite edge if we add up the time to perform several runs of these algorithms for tuning η. Note, however, that SMIDAS has better sparsity properties as compared to TRUNCGRAD and SCD even though their performance measured in the objective is similar.

Figure 2 is for the ARCENE data set. The results are quite similar to those for the DUKE data set. SMIDAS is slow for a short while early on but quickly catches up. Again, it displays good sparsity properties.

For the MAGIC data sets, SMIDAS does much better than TRUNCGRAD for the MAGIC04D data set (where the example vectors are dense). TRUNCGRAD is slightly better for the MAGIC04S data set (where the example vectors are sparse). This is illustrated in Figure 3. Note that this behavior is consistent with the bounds (3) and (4) given above. These bounds suggest that if the

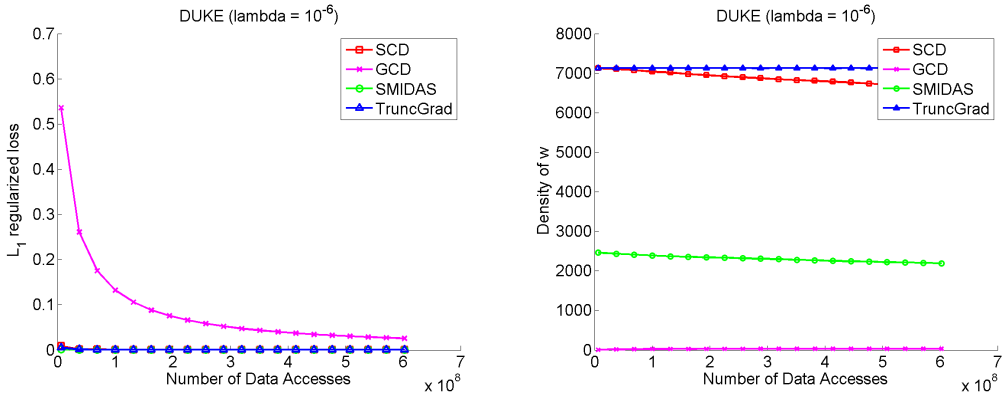

Figure 1: DUKE data set; less regularization

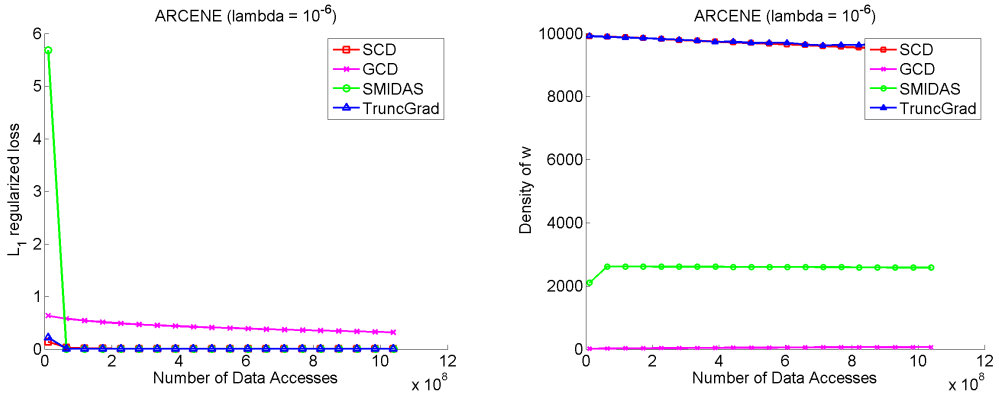

Figure 2: ARCENE data set; less regularization

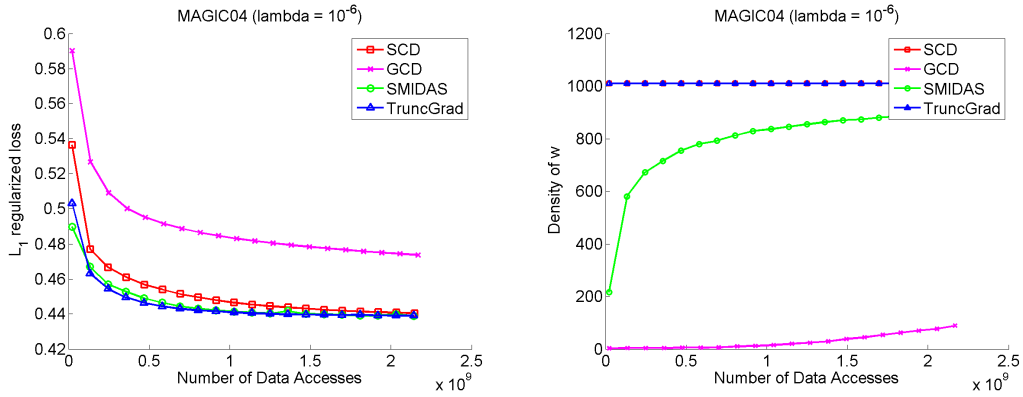

Figure 3: MAGIC04S data set; less regularization

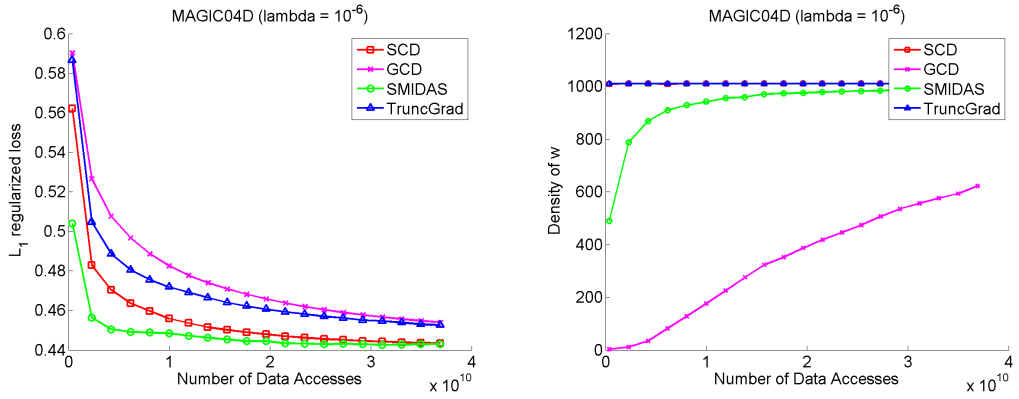

Figure 4: MAGIC04D data set; less regularization

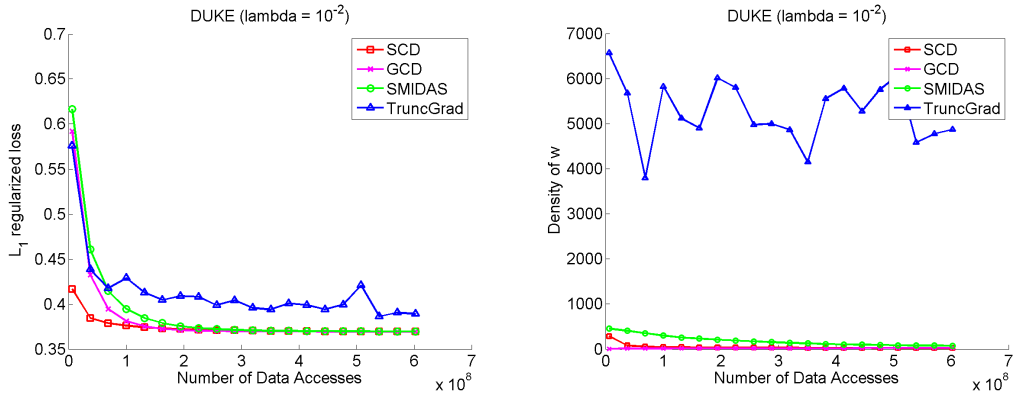

Figure 5: DUKE data set; more regularization

true solution has low ℓ_1 norm, SMIDAS will require fewer iterations than TRUNCGRAD when the examples are dense. The parameter-free SCD algorithm is slightly worse than the parameter-based algorithms TRUNCGRAD and SMIDAS on MAGIC04S. For MAGIC04D, its performance is better than TRUNCGRAD, but slightly worse than SMIDAS. On both data sets, SMIDAS seems to be doing quite well in terms of sparsity.

4.3.2 MORE REGULARIZATION

As mentioned before, we now present results with a large value of the regularization parameter ($\lambda = 10^{-2}$).

From Figures 5 and 6, we see that SCD outperforms all other algorithms on the DUKE and ARCENE data sets. Not only does it get to the minimum objective faster, but it also gets best sparsity. On both data sets, SMIDAS does better than TRUNCGRAD.

For the MAGIC data sets (Figures 7 and 8), we see a previous phenomenon repeated: SMIDAS does better when the features are dense. The coordinate descent algorithms SCD and GCD are quicker to reach the minimum on MAGIC04S. The edge, however, seems to be lost on the

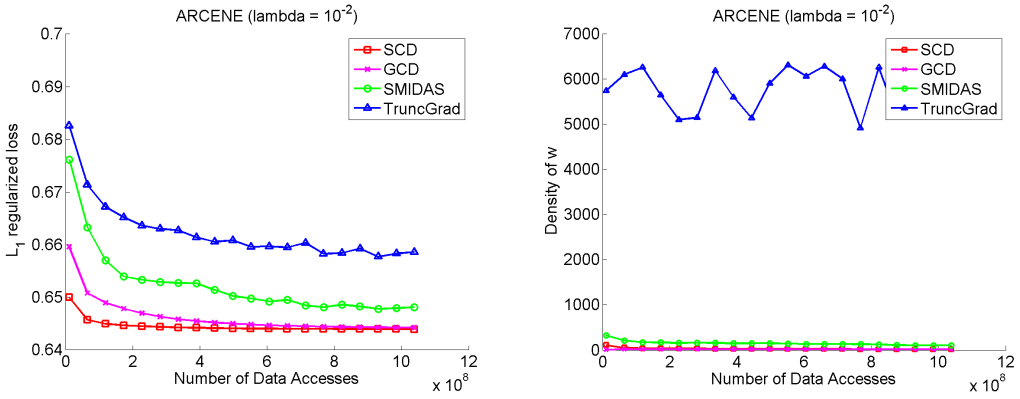

Figure 6: ARCENE data set; more regularization

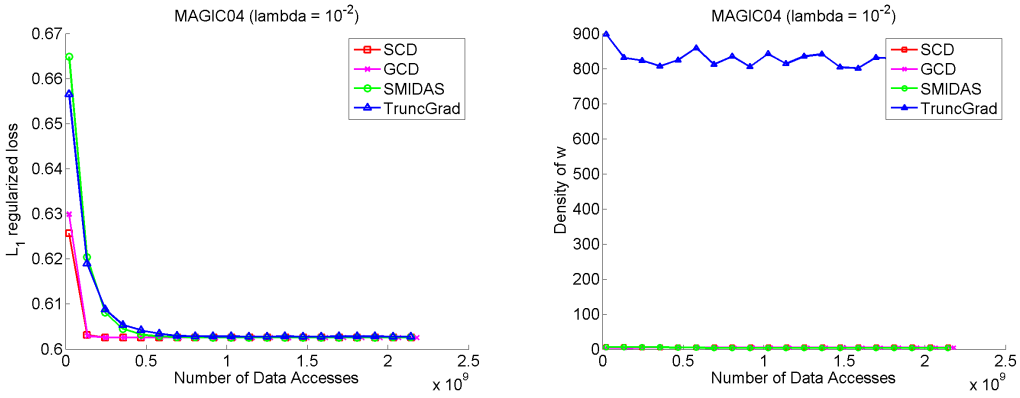

Figure 7: MAGIC04S data set; more regularization

MAGIC04D data set. In all cases, TRUNCGRAD seems to be unable to keep sparsity of the iterates as it progresses. This effect is particularly stark in Figure 8, where all the other algorithms have density levels of a few tens while TRUNCGRAD has almost no sparsity whatsoever.

4.3.3 PARAMETER VALUES AND SOURCE CODE

Two of the algorithms we used above, namely TRUNCGRAD and SMIDAS, have a step-size parameter η. In the interest of reproducible research, we report the values of η used in our experiments in Table 1. The parameter p of SMIDAS was always $\lceil 2\ln(d) \rceil$, where d is the total number of features (including non-relevant features). The source code for a C++ implementation of SCD and SMIDAS can be found at `http://mloss.org` (by searching for either "SCD" or "SMIDAS").

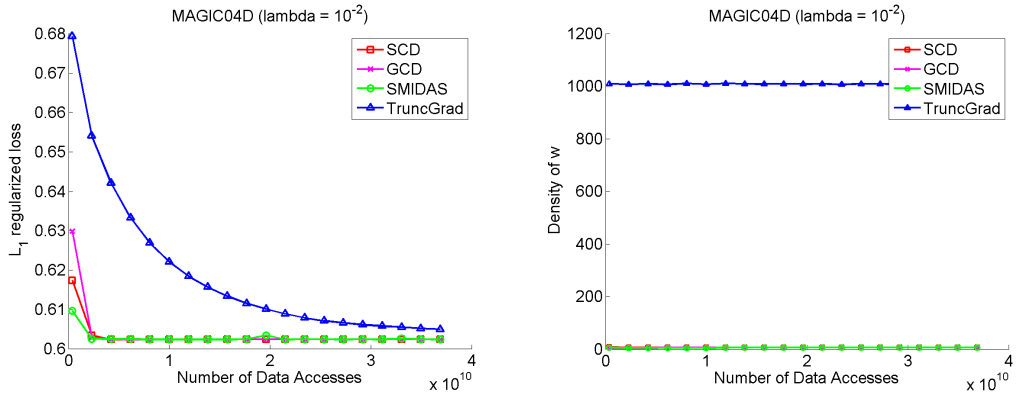

Figure 8: MAGIC04D data set; more regularization

Data Set, λ	SMIDAS	TRUNCGRAD
DUKE, 10^{-6}	50	10^{-1}
DUKE, 10^{-2}	10^{-1}	10^{-2}
ARCENE, 10^{-6}	50	10^{-1}
ARCENE, 10^{-2}	10^{-2}	5×10^{-5}
MAGIC04S, 10^{-6}	10^{-2}	5×10^{-4}
MAGIC04S, 10^{-2}	10^{-4}	5×10^{-5}
MAGIC04D, 10^{-6}	5×10^{-3}	10^{-4}
MAGIC04D, 10^{-2}	10^{-3}	10^{-5}

Table 1: Values of the step-size parameter η used in the experiments for SMIDAS and TRUNCGRAD

Acknowledgments

We thank Jonathan Chang for pointing out some errors in the preliminary conference version of this work. Most of the research reported in this paper was done while the authors were at the Toyota Technological Institute at Chicago (TTIC). We are grateful to TTIC for providing a friendly and stimulating environment to work in.

Appendix A.

Lemma 5 *Let* $\varepsilon \leq 0.12$. *There exists an optimization problem of the form:*

$$\min_{\mathbf{w}\in\mathbb{R}^d \,:\, \|\mathbf{w}\|_1 \leq B} \quad \frac{1}{m}\sum_{i=1}^{m} L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) \ ,$$

where L is the smooth loss function $L(a,b) = (a-b)^2$, such that any algorithm which initializes $\mathbf{w} = 0$ and updates a single element of the vector $\mathbf{w}$ at each iteration, must perform at least $B^2/16\varepsilon$ iterations to achieve an ε accurate solution.

Proof We denote the number of non-zeros elements of a vector $\mathbf{w}$ by $\|\mathbf{w}\|_0$. Recall that we denote the average loss by $C(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)$. We show an optimization problem of the form given above, for which the optimal solution, denoted $\mathbf{w}^\star$, is dense (i.e., $\|\mathbf{w}^\star\|_0 = d$), while any $\mathbf{w}$ for which

$$C(\mathbf{w}) \leq C(\mathbf{w}^\star) + \varepsilon$$

must satisfy $\|\mathbf{w}\|_0 \geq \Omega(B^2/\varepsilon)$. This implies the statement given in the lemma since an iteration bound for the type of algorithms we consider is immediately translated into an upper bound on $\|\mathbf{w}\|_0$.

Let $L(a,b) = (a-b)^2$ and consider the following joint distribution over random variables (X,Y). First, each Y is chosen at random according to $\mathbb{P}[Y=1] = \mathbb{P}[Y=-1] = \frac{1}{2}$. Next, each element j of X is chosen i.i.d. from $\{+1,-1\}$ according to $\mathbb{P}[X_j = y|y] = \frac{1}{2} + \frac{1}{2B}$. This definition implies that:

$$\mathbb{E}_{X_j|Y=y}[X_j] = \tfrac{1}{B}y$$

and

$$\mathrm{Var}_{X_j|Y=y}[X_j] = 1 - \tfrac{1}{B^2} \ .$$

Consider the vector $\mathbf{w}_0 = (\frac{B}{d}, \ldots, \frac{B}{d})$. We have

$$\begin{aligned}
\mathbb{E}\left[(\langle \mathbf{w}_0, X\rangle - Y)^2\right] &= \mathbb{E}_Y \mathbb{E}_{X|Y=y}\left[(\langle \mathbf{w}_0, X\rangle - y)^2\right] \\
&= \mathbb{E}_Y \mathrm{Var}_{X|Y=y}\left[\langle \mathbf{w}_0, X\rangle\right] \\
&= \mathbb{E}_Y \frac{B^2}{d}\mathrm{Var}_{X_1|Y=y}\left[X_1\right] \\
&= \mathbb{E}_Y \frac{B^2}{d}\left(1 - \tfrac{1}{B^2}\right) \\
&= \frac{B^2-1}{d} \ .
\end{aligned}$$

Now fix some $\mathbf{w}$ with $\|\mathbf{w}\|_0 \leq s$. We have

$$\mu_y := \mathbb{E}_{X|Y=y}[\langle \mathbf{w}, X\rangle] = \tfrac{y}{B}\sum_j w_j \ ,$$

and

$$\begin{aligned}\mathbb{E}\left[(\langle \mathbf{w}, X\rangle - Y)^2\right] &= \mathbb{E}_Y \mathbb{E}_{X|Y=y}\left[(\langle \mathbf{w}, X\rangle - y)^2\right] \\ &= \mathbb{E}_Y \mathrm{Var}_{X|Y=y}[\langle \mathbf{w}, X\rangle] + (\mu_y - y)^2\end{aligned}$$

If $|\sum_j w_j| \leq B/2$ then $(\mu_y - y)^2 \geq 1/4$ and thus we obtain from the above that $\mathbb{E}\left[(\langle \mathbf{w}, X\rangle - Y)^2\right] \geq 1/4$. Otherwise,

$$\sqrt{s\sum_j w_j^2} \geq \sum_j |w_j| \geq |\sum_j w_j| \geq B/2 \ \ ,$$

and thus we have that

$$\begin{aligned}\mathbb{E}[(\langle \mathbf{w}, X\rangle - Y)^2] &\geq \mathbb{E}_Y \mathrm{Var}_{X|Y=y}\left[\langle \mathbf{w}, X\rangle\right] \\ &= \mathbb{E}_Y \sum_{j=1}^d w_j^2 \mathrm{Var}_{X_1|Y=y}[X_1] \\ &= \mathbb{E}_Y \sum_{j=1}^d w_j^2 \left(1 - \tfrac{1}{B^2}\right) \\ &= \left(1 - \tfrac{1}{B^2}\right) \sum_{j=1}^d w_j^2 \\ &\geq \left(1 - \tfrac{1}{B^2}\right) \tfrac{B^2}{4s} = \frac{B^2 - 1}{4s} \ .\end{aligned}$$

Choose $B \geq 2$ and $d = 100(B^2 - 1)$, we have shown that if $\|\mathbf{w}\|_0 \leq s$ then

$$\mathbb{E}[(\langle \mathbf{w}, X\rangle - Y)^2 - (\langle \mathbf{w}_0, X\rangle - Y)^2] \geq \min\left\{0.24, \tfrac{B^2}{8s}\right\} =: \varepsilon' \ .$$

Now, consider the random variable $Z = (\langle \mathbf{w}, X\rangle - Y)^2 - (\langle \mathbf{w}_0, X\rangle - Y)^2$. This is a bounded random variable (because $|\langle \mathbf{w}, \mathbf{x}\rangle| \leq B$) and therefore using Hoeffding inequality we have that with probability of at least $1 - \delta$ over a draw of a training set of m examples we have

$$C(\mathbf{w}) - C(\mathbf{w}_0) \geq \varepsilon' - cB^2\sqrt{\frac{\log(1/\delta)}{m}} \ ,$$

for some universal constant $c > 0$.

This is true if we first fix $\mathbf{w}$ and then draw the m samples. We want to establish an inequality true for any $\mathbf{w}$ in

$$\mathcal{W} := \{\mathbf{w} \in \mathbb{R}^d \ : \ \|\mathbf{w}\|_0 \leq s, \ \|\mathbf{w}\|_1 \leq B\} \ .$$

This set has infinitely many elements so we cannot trivially appeal to a union bound. Instead, we create an $\varepsilon'/16B$-cover of $\mathcal{W}$ in the ℓ_1 metric. This has size $\mathcal{N}_1(\mathcal{W}, \varepsilon'/16B)$ where we have the crude estimate,

$$\forall \varepsilon > 0, \ \mathcal{N}_1(\mathcal{W}, \varepsilon) \leq d^s \left(\frac{2Bd}{\varepsilon}\right)^s \ .$$

Moreover, if $\|\mathbf{w}-\mathbf{w}'\|_1 \leq \epsilon'/16B$ then it is easy to see that $|C(\mathbf{w})-C(\mathbf{w}')| \leq \epsilon'/4$. Therefore, applying a union bound over all vectors in the cover, we obtain that with probability at least $1-\delta$, for *all* such $\mathbf{w} \in \mathcal{W}$ we have

$$C(\mathbf{w})-C(\mathbf{w}_0) \geq \epsilon' - \epsilon'/4 - cB^2\sqrt{\frac{\log \mathcal{N}_1(\mathcal{W},\epsilon'/16B)+\log(1/\delta)}{m}} \; .$$

Taking m large enough, we can guarantee that, with high probability, for all $\mathbf{w} \in \mathcal{W}$,

$$C(\mathbf{w})-C(\mathbf{w}_0) \geq \epsilon'/2 \; .$$

Finally, we clearly have that $C(\mathbf{w}^\star) \leq C(\mathbf{w}_0)$.

Thus, we have proved the following. Given $B \geq 2$ and s, there exist $\{(\mathbf{x}_i, y_i)\}_{i=1}^m$ in some dimension d, such that

$$\min_{\|\mathbf{w}\|_1 \leq B, \|\mathbf{w}\|_0 \leq s} C(\mathbf{w}) - \min_{\|\mathbf{w}\|_1 \leq B} C(\mathbf{w}) \geq \min\left\{0.12, \tfrac{B^2}{16s}\right\} \; .$$

This concludes the proof of the lemma. ■

Appendix B.

Lemma 6 *Let C be as defined in* (5) *and assume that the second derivative of L with respect to its first argument is bounded by β. Then, for any $j \in [d]$,*

$$C(\mathbf{w}+\eta\mathbf{e}^j) \leq C(\mathbf{w}) + \eta(\nabla C(\mathbf{w}))_j + \tfrac{\beta\eta^2}{2} \; .$$

Proof Note that, by assumption on L, for any i, j we have,

$$\begin{aligned} L(\langle \mathbf{w}+\eta\mathbf{e}^j, \mathbf{x}_i\rangle, y_i) &= L(\langle \mathbf{w}, \mathbf{x}_i\rangle + \eta x_{i,j}, y_i) \\ &\leq L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) + \eta L'(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)\, x_{i,j} + \tfrac{\beta\eta^2 x_{i,j}^2}{2} \\ &\leq L(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) + \eta L'(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)\, x_{i,j} + \tfrac{\beta\eta^2}{2} \; , \end{aligned}$$

where the last inequality follows because $x_{i,j} \in [-1,+1]$. Adding the above inequalities for $i = 1,\ldots,m$ and dividing by m, we get

$$\begin{aligned} C(\mathbf{w}+\eta\mathbf{e}^j) &\leq C(\mathbf{w}) + \frac{\eta}{m}\sum_{i=1}^m L'(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)\, x_{i,j} + \tfrac{\beta\eta^2}{2} \\ &= C(\mathbf{w}) + \eta(\nabla C(\mathbf{w}))_j + \tfrac{\beta\eta^2}{2} \; . \end{aligned}$$

■

Lemma 7 *Let $a, b, P, T > 0$. The function $f : (0, 1/a) \to \mathbb{R}$ defined as,*

$$f(\eta) = \left(\frac{1}{1-a\eta} - 1\right) P + \frac{b/(\eta T)}{1-a\eta}$$

is minimized at

$$\eta^\star = \frac{b}{PT}\left(\sqrt{1+\frac{PT}{ab}}-1\right) ,$$

and the minimum value satisfies

$$f(\eta^\star) \leq 4\sqrt{\frac{abP}{T}}+\frac{4ab}{T} .$$

Proof A little rearranging gives,

$$f(\eta) = \frac{1}{\frac{1}{\eta}-a}\left(aP+\frac{b}{\eta^2 T}\right) .$$

This suggests the change of variable $C = 1/\eta$ and we wish to minimize $g : (a,\infty) \to \mathbb{R}$ defined as,

$$g(C) = \frac{1}{C-a}\left(aP+\frac{bC^2}{T}\right) .$$

The expression for the derivative g' is,

$$g'(C) = \frac{b}{T(C-a)^2}\left(C^2-2aC-\frac{aTP}{b}\right) .$$

Setting $g'(C) = 0$ gives a quadratic equation whose roots are,

$$a \pm \sqrt{a^2+\frac{aTP}{b}} .$$

Choosing the larger root (the smaller one is smaller than a) gives us the minimizer,

$$C^\star = a+\sqrt{a^2+\frac{aTP}{b}} .$$

It is easy to see that $g'(C)$ is increasing at $C^\star$ and thus we have a local minima at $C^\star$ (which is also global in this case). The minimizer $\eta^\star$ of $f(\eta)$ is therefore,

$$\eta^\star = \frac{1}{C^\star} = \frac{b}{PT}\left(\sqrt{1+\frac{PT}{ab}}-1\right) .$$

Plugging in the value of $C^\star$ into $g(C)$, we get,

$$\begin{aligned}
g(C^\star) &= \frac{2}{\sqrt{1+\frac{PT}{ab}}}\left(P+\frac{ab}{T}+\sqrt{\frac{a^2b^2}{T^2}+\frac{abP}{T}}\right) \\
&\leq \frac{2}{\sqrt{1+\frac{PT}{ab}}}\left(2P+\frac{2ab}{T}\right) \\
&= 4\sqrt{\frac{abP}{T}+\frac{a^2b^2}{T^2}} \\
&\leq 4\sqrt{\frac{abP}{T}}+\frac{4ab}{T} .
\end{aligned}$$

Since $g(C^\star) = f(\eta^\star)$, this concludes the proof of the lemma. ■

References

A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31:167–175, 2003.

L. Bottou. Stochastic gradient descent examples, Web Page. `http://leon.bottou.org/projects/sgd`.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems 20*, pages 161–168, 2008.

L. Bottou and Y. LeCunn. On-line learning for very large datasets. *Applied Stochastic Models in Business and Industry*, 21(2):137–151, 2005.

K.L. Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 922–931, 2008.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *International Conference on Machine Learning*, pages 272–279, 2008.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In *Proceedings of the 23rd Annual Conference on Learning Theory*, pages 14–26, 2010.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–499, 2004.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

J. Friedman, T. Hastie, and R. Tibshirani. Regularized paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

A. Genkin, D. Lewis, and D. Madigan. Large-scale Bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

C. Gentile. The robustness of the p-norm algorithms. *Machine Learning*, 53(3):265–299, 2003.

S. Ghadimi and G. Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, 2011. available at `http://www.ise.ufl.edu/glan/papers/strongSCOSubmit.pdf`.

A. J. Grove, N. Littlestone, and D. Schuurmans. General convergence results for linear discriminant updates. *Machine Learning*, 43(3):173–210, 2001.

J. Kivinen and M. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–64, January 1997.

K. Koh, S.J. Kim, and S. Boyd. An interior-point method for large-scale ℓ_1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, 2007.

G. Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, pages 1–33, 2010.

J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. In *Advances in Neural Information Processing Systems 21*, pages 905–912, 2009.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

Z. Q. Luo and P. Tseng. On the convergence of coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72:7–35, 1992.

A. Nemirovski and D. Yudin. *Problem complexity and method efficiency in optimization*. Nauka Publishers, Moscow, 1978.

Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. Technical Report CORE Discussion Paper 2010/02, Center for Operations Research and Econometrics, UCL, Belgium, 2010.

S. Shalev-Shwartz. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, The Hebrew University, 2007.

S. Shalev-Shwartz and N. Srebro. SVM optimization: Inverse dependence on training set size. In *International Conference on Machine Learning*, pages 928–935, 2008.

S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1 regularized loss minimization. In *International Conference on Machine Learning*, pages 929–936, 2009.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *International Conference on Machine Learning*, pages 807–814, 2007.

S. Shalev-Shwartz, T. Zhang, and N. Srebro. Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM Journal on Optimization*, 20:2807–2832, 2010.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

P. Tseng and S. Yun. A block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization. *Journal of Optimization Theory and Applications*, 140:513–535, 2009.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J. A. Olson, J. R. Marks, and J. R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Sciences USA*, 98(20):11462–11467, 2001.

T. T. Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

L. Xiao. Dual averaging method for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11:2543–2596, 2010.

T. Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transaction on Information Theory*, 49:682–691, 2003.

T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, 4:5–31, 2001.