Linearized Bregman for $\ell_1$-regularized Logistic Regression

Jianing V. Shi  
Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA  
jianing.shi@rice.edu

Wotao Yin  
Department of Computational and Applied Mathematics, Rice University, Houston, TX, USA  
wotao.yin@rice.edu

Stanley J. Osher  
Department of Mathematics, UCLA, Los Angeles, CA, USA  
sjo@math.ucla.edu

Abstract

Sparse logistic regression is an important linear classifier in statistical learning, providing an attractive route for feature selection. A popular approach is based on minimizing an $\ell_1$-regularization term with a regularization parameter $\lambda$ that affects the solution sparsity. To determine an appropriate value for the regularization parameter, one can apply the grid search method or the Bayesian approach. The grid search method requires constructing a regularization path, by solving a sequence of minimization problems with varying values of the regularization parameter, which is typically time consuming. In this paper, we introduce a fast procedure that generates a new regularization path without tuning the regularization parameter. We first derive the direct Bregman method by replacing the $\ell_1$-norm by Bregman divergence, and contrast it with the grid search method. For faster path computation, we further derive the linearized Bregman algorithm, which is algebraically simple and computationally efficient. Finally we demonstrate some empirical results for the linearized Bregman algorithm on benchmark data and study feature selection as an inverse problem. Compared with the grid search method, the linearized Bregman algorithm generates a different regularization path with comparable classification performance, in a much more computationally efficient manner.

1. Introduction

1.1. $\ell_1$-regularized Logistic Regression

The $\ell_1$-regularized logistic regression (Tibshirani, 1996) is a popular linear decoder in the field of machine learning. The inputs are a set of training data $X = [x_1, \ldots, x_m]^T \in \mathbb{R}^{m \times n}$, where each row of $X$ is a sample and samples of either class are assumed to be independently identically distributed, and class labels $y \in \mathbb{R}^m$ are of $-1/+1$ elements. We seek a hyperplane $\{x : w^T x + v = 0\}$ that separates the data belonging to two classes, where $w \in \mathbb{R}^n$ is a set of weights and $v \in \mathbb{R}$ is the intercept. The $\ell_1$-regularized logistic regression promotes sparsity in the weights $w$:

$$\arg \min_{w,v} \lambda J(w) + l_{\text{avg}}(w,v),$$

where $J(w) = ||w||_1$, and $\lambda > 0$ is a regularization parameter. The empirical loss function penalizes average misclassification error $l_{\text{avg}}(w,v) = \frac{1}{m} \sum_{i=1}^{m} \theta((w^T x_i + v) y_i)$, where the logistic transfer function takes the form $\theta(z) = \log(1 + \exp(-z))$.

It is well-known that $\ell_1$ minimization tends to give sparse solutions. The $\ell_1$ regularization results in logarithmic sample complexity bounds (number of training samples required to learn a function), making it an effective learner even under an exponential number of irrelevant features (Ng, 1998; 2004). Furthermore, $\ell_1$ regularization also has appealing asymptotic sample consistency for feature selection (Zhao & Yu, 2007).

Various algorithms exist for solving the $\ell_1$-regularized logistic regression, including Grafting (Perkins & Theiler, 2003), Gauss-Seidel method (Shevade & Keerthi, 2003), GenLASSO (Roth, 2004), and SCGIS (Goodman, 2004), IRLS-LARS (Efron et al., 2004; Lee et al., 2006), BBR (Genkin et al., 2004), bound optimiza-
tion (Krishnapuram et al., 2005), Glmpath (Park & Hastie, 2007), interior point method (Koh et al., 2007), fixed point continuation (FPC) (Hale et al., 2008), sparse reconstruction by separable approximation (SpaRSA) (Wright et al., 2009), hybrid iterative shrinkage (HIS) (Shi et al., 2010), and accelerated block-coordinate relaxation (Wright, 2012).

1.2. Computing the Full Regularization Path

The regularization parameter λ determines the level of sparsity, which is typically unknown a priori. In order to determine an appropriate level of sparsity, one may need to generate a regularization path. In many scenarios, the appropriate level of sparsity can refer to the optimal level sparsity where the corresponding solution gives rise to the best generalization performance for classification. We define two types of regularization paths here:

- **Solution-vs-lambda**: solution path where solution is a function of regularization parameter λ.
- **Solution-vs-sparsity**: solution path where solution is a function of sparsity level.

In the Grid Search method, one typically approximates the solution-vs-sparsity path using the solution-vs-lambda path. More specifically, one constructs a regularization path by varying the regularization parameter λ and solving a sequence of minimization problems corresponding to each λ. The optimal λ can be determined via cross validation. However, one needs to solve each minimization accurately. It is not difficult to see the grid search method is time costly. This is especially true when the cardinality of the true solution support is large, since usually the smaller λ is, the lesser sparse solution is and the longer it takes for an algorithm to converge. In this paper, we devise a methodology to efficiently approximate the solution-vs-sparsity path, using the **Linearized Bregman** algorithm.

Some related work along the line of regularization parameter selection can be found in the Bayesian literature. The Bayesian approach provides an alternative, where the regularization parameter is treated as a parameter of the prior distribution in a hierarchical framework. One strategy is to integrate out the prior parameter to obtain the marginal likelihood, which is used in algorithms such as automatic relevance determination (MacKay, 1992), relevance vector machine (Tipping, 2001). Such a Bayesian approach was used for ℓ1-regularization (Figueiredo, 2003), and ℓ2-regularization (Foo et al., 2009).

1.3. Our Contribution

We propose a new approach for computing the solution-vs-sparsity path, which is much more computationally efficient than the grid search method while achieving comparable solution quality. Our approach is based on the linearized Bregman algorithm, using all of the intermediate solutions to form the path. The linearized Bregman algorithm is based on solving a sequence of minimization subproblems by introducing the Bregman divergence. Each iteration of the algorithm minimizes the sum of certain Bregman divergence of the ℓ1 norm, the linearization of the loss function, and a proximity term. The minimizer of each subproblem can be obtained in closed form, resulting in an algebraically simple and computationally efficient algorithm. Unlike the grid search method which solves for the regularization path of (1), our new regularization path does not require tuning the regularization parameter λ and solving each subproblem corresponding to each parameter value. However, like the regularization path of (1), our new regularization path is roughly monotonic in solution sparsity. The linearized Bregman algorithm creates a new regularization path, starting from a big λ, which results in highly sparse solution; as the algorithm proceeds, the sparsity of solution increases, as well as the classification performance on training data. Empirical results demonstrate that the generalization performance of the Linearized Bregman algorithm is comparable with the Grid Search method.

2. New Regularization Path

2.1. Bregman Divergence

**Definition 2.1** The Bregman divergence (Bregman, 1967), based on a convex functional \( J : \mathbb{R}^n \rightarrow \mathbb{R} \), is formally defined by

\[
D^B_p(u, v) = J(u) - J(v) - \langle u - v, p \rangle, \quad p \in \partial J(v), \tag{2}
\]

where \( p \) denotes an element in the subgradient of \( J(v) \).

For a continuously differentiable functional, such as \( ℓ_2 \) norm, there exists a unique element \( p \) in the subdifferential and consequently a unique Bregman divergence. For a non-differentiable yet convex functional, such as the \( ℓ_1 \) norm, \( p \in \partial J(v) \) is an element in the subgradient of \( J \) at the point \( v \). It is worth noting that the Bregman divergence is not a distance in the usual sense, since in general \( D^B_p(u, v) \neq D^B_p(v, u) \), nor is the triangle inequality satisfied. However, the Bregman divergence measures the closeness between \( u \) and \( v \) in the sense that \( D^B_p(u, v) \geq 0 \) and \( D^B_p(u, v) = D^B_p(w, v) \) for all points \( w \) on the line segment connecting \( u \) and...
v. Moreover, Bregman divergence has the following properties: (a) If \( J \) is convex, \( D^p_f(u, v) \geq 0 \); (b) If \( J \) is strictly convex, \( D^p_f(u, v) > 0 \) for \( u \neq v \); (c) If \( J \) is strongly convex, there exists a constant \( \nu > 0 \) such that \( D^p_f(u, v) \geq \nu \| u - v \|^2_2 \). Furthermore, the subgradient \( p \in \partial J(v) \) is not unique, when \( J(v) = \| v \|_1 \).

### 2.2. Prior Art Using Bregman Divergence

Some earlier work concerning Bregman divergence, logistic regression and Adaboost can be found in (Collins et al., 2002). Such a Bregman divergence framework was extended to \( \ell_1 \)-regularized logistic regression in (Gupta & Huang, 2008). Both applied the Bregman divergence to the logistic loss term. Bregman divergence was also used in clustering in the context of unsupervised learning (Banerjee et al., 2005; Liu et al., 2012).

On the other hand, application of Bregman divergence to the \( \ell_1 \) term was introduced in (Osher et al., 2005). Later, the Bregman regularization was applied to the following \( \ell_1 \)-regularized basis pursuit problem (Yin et al., 2008), which arises in compressive sensing. A linearized Bregman algorithm was derived in (Yin et al., 2008) for compressive sensing, yielding a simple and fast algorithm. However, prior art only considered finding solution with small \( \lambda \), instead of the full solution path, using the linearized Bregman algorithm.

### 2.3. New Regularization Path

We first introduce a new regularization path using the Bregman divergence, resulting in the direct Bregman method. The difference between the grid search method and the direct Bregman method is illustrated below.

- **Grid Search Method**
  
  Given a sequence of regularization parameters \( \lambda_0 > \lambda_1 > \ldots > \lambda_k > \ldots > \lambda_n \), we solve a sequence of minimization subproblems corresponding to each \( \lambda_k \).
  
  \[
  (w^k, v^k) \leftarrow \arg \min_{w,v} \lambda_k J(w) + l_{avg}(w, v), \quad (3)
  \]
  
  for \( k = 0, 1, \ldots, n \).

- **Direct Bregman Method**
  
  Given a fixed regularization parameter \( \lambda_0 \), which is large enough so that \( w^k \) will be sufficiently sparse, we solve a sequence of minimization subproblems,
  
  \[
  (w^{k+1}, v^{k+1}) \leftarrow \arg \min_{w,v} \lambda_0 D^p_f(w, w^k) + l_{avg}(w, v^k),
  \]
  
  for \( k = 0, 1, \ldots, n-1 \), with initial conditions \( w^0 = 0 \), \( v^0 = 0 \), and \( p^0 = 0 \).

In the above direct Bregman method, \( \{(w^k, v^k)\} \) is a sequence of solutions, and \( p^k \in \partial J(w^k) \) is the subgradient of \( J(w^k) \), where \( J(w^k) = \| w^k \|_1 \). By substituting the definition of Bregman divergence, we arrive at

\[
\min_{w,v} \lambda_0 (J(w) - J(w^k) - \langle w - w^k, p^k \rangle) + l_{avg}(w, v).
\]

One can further simply this expression to

\[
\min_{w,v} \lambda_0 J(w) - \lambda_0 \langle w, p^k \rangle + l_{avg}(w, v) \tag{5}
\]

![Figure 1](image-url.com)  

**Figure 1.** Comparison between the grid search method and the direct Bregman method. These two methods generate different regularization paths. In the grid search method, a sequence of decreasing lambdas are generated as the input to the algorithm, and each subproblems solves the original optimization problem with each lambda. In the direct Bregman method, lambda is fixed, and each subproblem solves the Bregman regularized optimization problem.

Fig. 1 illustrates the difference between these two methods. Note in Eqn. (5), the first two terms \( \lambda_0 J(w) - \lambda_0 \langle w, p^k \rangle \) together determine the contribution of the regularization term. Roughly speaking, the amount of regularization decreases along the new regularization path.

Given that \((w^{k+1}, v^{k+1})\) satisfies the first-order optimality condition of problem (4),

\[
0 \in \partial \left( \lambda_0 J(w^{k+1}) - \lambda_0 \langle w^{k+1}, p^k \rangle + l_{avg}(w^{k+1}, v^{k+1}) \right)
\]

\[
0 \in \lambda_0 p^{k+1} - \lambda_0 p^k + \nabla_w l_{avg}(w^{k+1}, v^{k+1}),
\]
where \( p^k \) is the subgradient of \( J(w^k) \), not single-valued due to the non-differentiability of \( \ell_1 \) norm. Hence we arrive at the iterate for updating \( p^{k+1} \),
\[
p^{k+1} = p^k - \frac{1}{\lambda_0} \nabla_w l_{\text{avg}}(w^{k+1}, v^{k+1}).
\]  
(6)

So far, (4) and (6) constitute the direct Bregman method. We summarize it in Algorithm 1.

**Algorithm 1 Direct Bregman Method**

**Input:** data \( X \in \mathbb{R}^{m \times n} \) and label \( y \in \mathbb{R}^m \).

Initialize \( k = 0, w^0 = 0, v = 0, p^0 = 0, \lambda_0 > 0 \).

while stopping criterion not satisfied do

\( (w^{k+1}, v^{k+1}) \leftarrow \arg \min_{w, v} \lambda_0 J(w) - \lambda_0 \langle w, p^k \rangle + l_{\text{avg}}(w, v) \)

\( p^{k+1} \leftarrow p^k - \frac{1}{\lambda_0} \nabla_w l_{\text{avg}}(w^{k+1}, v^{k+1}) \)

\( k \leftarrow k + 1 \)

end while

The direct Bregman procedure generates a regularization path. Compared to the grid search method, where a sequence \( \{\lambda_k\} \) controls the regularization path, the direct Bregman procedure generates a regularization path without tuning the regularization parameter \( \lambda \). This is the key insight of our algorithm, illustrated in Fig. 1. Note the grid search method and direct Bregman method generate two different regularization paths.

Since each subproblem (5) needs to be solved accurately, the direct Bregman method turns out to be computationally expensive. Therefore we derive a simpler and more efficient algorithm called linearized Bregman in the next section.

### 2.4. Linearized Bregman Algorithm

Recall our goal is to improve the regularization path, especially in terms of computational efficiency. During the direct Bregman procedure, each subproblem needs to be solved accurately. Suppose each subproblem can be solved easily, then our goal is achieved: we do so via linearization. By linearizing the loss function, we can obtain a close-form solution for each subproblem (5). This will result in the linearized Bregman algorithm, as well as a different regularization path.

The linearized version of the direct Bregman algorithm can be derived by approximating the loss function \( l_{\text{avg}}(w, v) \) using first-order Taylor expansion in the \( w \) component at \( (w^k, v^k) \), which is \( l_{\text{avg}}(w^k, v^k) + \langle \nabla_w l_{\text{avg}}(w^k, v^k), w \rangle \), and adding a proximal term \( \|w - w^k\|^2/(2\alpha) \) to the objective function. We simply perform gradient descent in the \( v \) component.

\[
\min_w \lambda_0 D^2_f(w, w^k) + \langle w, \nabla_w l_{\text{avg}}(w^k, v^k) \rangle + \frac{1}{2\alpha} \|w - w^k\|^2, \\
\min_v l_{\text{avg}}(w^k, v).
\]

Now we can further group \( w \) from the last two terms and get

\[
\min_w \lambda_0 D^2_f(w, w^k) + \frac{1}{2\alpha} \|w - (w^k - \alpha \nabla_w l_{\text{avg}}(w^k, v^k))\|^2.
\]

(7)

In order to derive the update rule for \( p^{k+1} \), we use the optimality condition for the objective function of the linearized Bregman procedure (7), which leads to

\[
p^{k+1} = p^k - \frac{1}{\lambda_0} \nabla_w l_{\text{avg}}(w^k, v^k) - \frac{1}{\lambda_0} (w^{k+1} - w^k).
\]

(8)

Hence the iterates (7) together with (8) constitute the linearized Bregman iterative algorithm. We further simplify the linearized Bregman iterative algorithm below and derive the Linearized Bregman Algorithm.

**Theorem 2.2** The original linearized Bregman iterative algorithm solves a sequence of minimizers (7) along with subgradient update (8). Such an algorithm can be further reduced to updating a sequence of \( (w^k, v^k, z^k) \), involving

\[
\begin{align*}
z^{k+1} &= z^k - \frac{1}{\lambda_0} \nabla_w l_{\text{avg}}(w^k, v^k), \\
w^{k+1} &= \lambda_0 \mathcal{S}(z^{k+1}, 1), \\
v^{k+1} &= v^k - \frac{1}{\lambda_0} \nabla_v l_{\text{avg}}(w^k, v^k),
\end{align*}
\]

where \( \mathcal{S} \) is the shrinkage operator.

**Proof.** The objective function for each subproblem of the linearized Bregman iterative algorithm (7) can be reduced to

\[
\min_w \lambda_0 J(w) - \lambda_0 \langle w, p^k \rangle + \frac{1}{2\alpha} \|w - (w^k - \alpha \nabla_w l_{\text{avg}}(w^k, v^k))\|^2.
\]

We hence have the following update,

\[
\min_w \lambda_0 J(w) + \frac{1}{2\alpha} \|w - (w^k + \lambda_0 \alpha p^k - \alpha \nabla_w l_{\text{avg}}(w^k, v^k))\|^2.
\]

The updating rule for the subgradient \( p^k \in \partial J(w^k) \), where \( J(w^k) = \|u^k\|_1 \), can be obtained from the first-order optimality condition of (7),

\[
0 \in \partial (\lambda_0 D^2_f(w, w^k) + \frac{1}{2\alpha} \|w - (w^k - \alpha \nabla_w l_{\text{avg}}(w^k, v^k))\|^2)
\]

\[
0 = \lambda_0 p^{k+1} - \lambda_0 p^k + \frac{1}{\alpha} u^{k+1} - \frac{1}{\alpha} (w^k - \alpha \nabla_w l_{\text{avg}}(w^k, v^k)).
\]
Algorithm 2 Linearized Bregman Algorithm

**Input:** data $X \in \mathbb{R}^{m \times n}$ and label $y \in \mathbb{R}^{m}$.
Set parameters $\lambda_0 > 0$, $\alpha > 0$.
Initialize $k = 0$, $w^0 = 0$, $v^0 = 0$, $p^0 = 0$.

**while** stopping criterion not satisfied **do**

$z^{k+1} \leftarrow z^k - \frac{1}{\lambda_0} \nabla \|w\|_{\text{avg}}(w^k, v^k)$

$w^{k+1} \leftarrow \lambda_0 S(z^{k+1}, 1)$

$v^{k+1} \leftarrow v^k - \frac{1}{\lambda_0} \nabla \|w\|_{\text{avg}}(w^k, v^k)$

$k \leftarrow k + 1$

**end while**

Now we have the following update for subgradient,

$$p^{k+1} + \frac{1}{\lambda_0 \alpha} w^{k+1} = p^k + \frac{1}{\lambda_0 \alpha} w^k - \frac{1}{\lambda_0} \nabla \|w\|_{\text{avg}}(w^k, v^k).$$

(10)

In order to simplify these two equations further, we introducing a new variable $z^k = p^k + \frac{1}{\lambda_0 \alpha} w^k$. Now Eqn. (10) can be rewritten as

$$z^{k+1} = z^k - \frac{1}{\lambda_0} \nabla \|w\|_{\text{avg}}(w^k, v^k).$$

Objective function (7) becomes

$$\min_{w} \lambda_0 \alpha \|w\|_1 + \frac{1}{2} \|w - \lambda_0 \alpha z^{k+1}\|_2^2.$$ 

The above minimization has closed-form solution,

$$w^{k+1} = S(\lambda_0 \alpha z^{k+1}, \lambda_0 \alpha) = \lambda_0 \alpha S(z^{k+1}, 1),$$

where the shrinkage operator is defined as

$$S(z, \alpha) = \text{sgn}(z) \odot \max(|z| - \alpha, 0),$$

(11)

with $\odot$ denoting element-wise product.

Therefore we arrive at a three line code, summarized in Algorithm 2.

The linearized Bregman algorithm constructs a well-defined sequence $\{(w^k, v^k, z^k)\}$, essentially a regularization path starting from $w^0 = 0$, $v^0 = 0$, $z^0 = 0$. Following this path, the penalization on the $\ell_1$ regularization is weakened due to linearization $(w, p^k)$, resulting in less sparse solutions, as if the regularization parameter $\lambda$ is reduced in model (1). The linearized Bregman algorithm is very straightforward to implement, and only involves matrix vector multiplication and scalar shrinkage. Again, we note that the regularization paths generated by the grid search method, direct Bregman method, and linearized Bregman method are different. Note the update of subgradient is not computationally explicit in Algorithm 2 due to the introduction of new variable $z^k$. One attractive property of the linearized Bregman algorithm is that it starts from a big $\lambda$ and achieves denser solutions without actually tuning $\lambda$. The solution to the linearized Bregman algorithm, appears to get closer, in the Bregman sense, to the minimizer of the empirical logistic loss function. Such a result was observed and analyzed formally for image denoising, where the fidelity term is quadratic, in (Osher et al., 2005).

3. Numerical results

3.1. Forward Model for Data Generation

Consider two Gaussian classes with $n$ dimensions, both with covariance matrices equal to identify. Means of the two classes are $\mu_1 = [c, 0, ..., c, 0, c, 0, ..., 0]^T$, where only $nnz$ elements are nonzero and the remaining $n - nnz$ elements are zero, and $\mu_2 = -\mu_1$. The ground truth of these data is known, i.e. those $nnz$ dimensions with non-zero means are the informative features for classification. To introduce noise in the data, we add Gaussian noise with zero mean.

3.2. Convergence

We demonstrate some numerical results for the linearized Bregman algorithm concerning algorithm convergence. In this example, $n = 100$, $m = 100$ (50 samples per class).

![Figure 2. Linearized Bregman algorithm. (a) Logistic loss term $l_{\text{avg}}(w^k, v^k)$ as function of iteration $k$. (b) Regularization term $\lambda_0 J(w)$ as function of iteration $k$, where $\lambda_0 = 10$. The data used in this experiment is simulated, with $m = 100$ samples and $n = 100$ dimensions.](image)

Fig. 2(a) shows the evolution of loss function $l_{\text{avg}}(w^k, v^k)$ as function of iteration $k$. This shows that $l_{\text{avg}}(w^k, v^k)$ monotonically decreases and converges to the minimum as $k \to \infty$. Fig. 2(b) shows the evolution of the regularization term $\lambda_0 J(w^k)$ as function of $k$. The solution gets less and less sparse along the regularization path.

Let’s denote residual as $\|w^k - w^*\|$, where $w^*$ is the op-
Fig. 4(a) & (b) show how the cardinality of solution support evolves as function of accumulated computational time. One can see that linearized Bregman is much more efficient in generating the full solution path. Fig. 4(c) & (d) illustrate the evolution of solution path for both algorithms. One can see the solution path generated by the linearized Bregman algorithm is much more finely grained compared with the grid search method.

The linearized Bregman iterative algorithm can solve a problem starting from a big \( \lambda \), and create a new regularization path that “effectively” decreases the amount of solutions generated by the linearized Bregman algorithm and the grid search method are different. The differences are due to the fact that the linearized Bregman algorithm iteratively applies linear relaxations the \( \ell_1 \) regularization term \( J(w) \), and the particular form of \( \ell_1 \)-induced Bregman divergence frees nonzero entries in the last iteration from being penalized in the current iteration, so their values become larger. On the other hand, all entries in the grid search method are penalized equally by \( \ell_1 \) regularization, so the nonzero entries in the solution are smaller due to the \( \ell_1 \) penalty.

### 3.3. Regularization Path Quality and Speed

As mentioned earlier, practical usage of the \( \ell_1 \)-regularized logistic regression for classification problems requires running the algorithm on a regularization path with a sequence of decreasing \( \lambda \), through the grid search approach. In general, the regularization parameter \( \lambda \) affects the number of iterations to converge for most solvers. Since the linearized Bregman algorithm has first-order accuracy, it is fair to compare it with FISTA (Beck & Teboulle, 2009), which is an accelerated version of ISTA (Chambolle et al., 1998; Daubechies et al., 2004; Bect et al., 2009). Note that we employ the continuation strategy (i.e. warm start) when we use FISTA, in that we use the solution from previous \( \lambda^k \) as the initial solution for the next \( \lambda^{k+1} \).

We ran FISTA until either of the two stopping criteria are satisfied:

\[
\left\| w^{k+1} - w^k \right\| / \max(\left\| w^k \right\|_\infty, 1) < 10^{-8} \\
\left\| \nabla_w l_{\text{avg}}(w^k, v^k) \right\| < 10^{-4}. \tag{12}
\]

We compare solution paths generated by the linearized Bregman algorithm and the grid search method. We run these two algorithms independently on the same data, and record accumulated computational time. Fig. 4 contrasts the computational efficiency and quality of solution paths between these two algorithms.
of regularization without tuning \( \lambda \). More importantly, each iterate of the linearized Bregman is exact. This indicates an important advantage of using the linearized Bregman algorithm.

When one runs the linearized Bregman algorithm, the notion of regularization parameter selection becomes obsolete. The algorithm starts from a big regularization parameter \( \lambda \) and converges to the true solution. In the grid search method, one needs to construct a regularization path varying \( \lambda \). Therefore, it requires solving a sequence of minimization subproblems with different \( \lambda \). Such an approach can be extremely time costly. We can thus gain a tremendous amount of speed-up using the linearized Bregman algorithm. Secondly, one can see that linearized Bregman algorithm gives a more finely grained regularization path. In the case of grid search method, one can miss the true solution if one does not fine tune \( \lambda \).

We then compare the computational efficiency between the linearized Bregman algorithm and the grid search method. Fig. 5(a) compares the evolution of support cardinality \( \text{nnz}(w^k) \) as a function of iteration \( k \), and Fig. 5(b) compares the evolution of empirical loss function value \( l_{\text{avg}}(w^k, v^k) \) as a function of iteration \( k \). The computational advantage of linearized Bregman algorithm is quite obvious based on our numerical experiments.

![Figure 5](image)

**Figure 5.** Comparison of solution paths between the Linearized Bregman algorithm and the Grid Search method. The grid search method is solved using FISTA with warm start. (a) evolution of support cardinality \( \text{nnz}(w^k) \) plotted against iteration \( k \), (b) evolution of loss function \( l_{\text{avg}}(w^k, v^k) \) plotted against iteration \( k \).

### 3.4. Algorithm Scaling

In order to test whether the proposed algorithm is scalable for large-scale data, we compare the linearized Bregman algorithm with the grid search method, on data of different dimensions. We note such a comparison is dependent on data, and how the regularization parameters are chosen for the grid search. In order to make a fair comparison, we simulate data under the same distribution (see Section 3.1). For all cases, we run the linearized Bregman algorithm for 1000 time steps, and run the FISTA for 1000 different regularization parameters.

In the grid search method, a geometric progression is used for constructing the sequence of regularization parameters. We compute an upper bound for useful range of regularization parameters via \( \lambda_{\text{max}} = \frac{1}{m_+} \| \frac{1}{m} \sum_{b_i = +1} a_i + \frac{1}{m} \sum_{b_i = -1} a_i \|_{\infty} \), where \( m_- \) is the number of training samples with label \(-1\) and \( m_+ \) is the number of training samples with label \(+1\) (Koh et al., 2007). In the linearized Bregman algorithm, we use \( \lambda_0 \) which gives rise to over-smoothed initial solution and stable solution path.

All computation is done using MATLAB R2010a, on a MacBook Pro laptop with Mac OSX 10.6.5, with 2.66 GHz Intel Core i7, and 8 GB 1067 MHz DDR3 memory. Clearly linearized Bregman algorithm is more computationally efficient compared with the grid search method.

<table>
<thead>
<tr>
<th>n</th>
<th>m</th>
<th>( \text{nnz} )</th>
<th>LB</th>
<th>GS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0.14 s</td>
<td>6.11 s</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>50</td>
<td>0.29 s</td>
<td>12.18 s</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>500</td>
<td>0.87 s</td>
<td>17.99 s</td>
</tr>
<tr>
<td>10000</td>
<td>100</td>
<td>5000</td>
<td>22.58 s</td>
<td>120.13 s</td>
</tr>
<tr>
<td>100000</td>
<td>100</td>
<td>50000</td>
<td>200.23 s</td>
<td>845.19 s</td>
</tr>
</tbody>
</table>

*Table 1.* This table summaries the scaling for Linearized Bregman (LB) and Grid Search (GS). GS is computed using FISTA with warm start. We test the computational efficiency of both algorithms for data with different dimensions. The data used in this experiment is simulated, see Section 3.1 for details. Here \( n = \# \) dimensions, \( m = \# \) samples, and \( \text{nnz} = \# \) nonzeros within the original data dimension.

Table 1 is a summary of computational efficiency. We note linearized Bregman induces almost 100 times speedup, compared with the grid search method. Such a result suggests that linearized Bregman can facilitate computational efficiency for large-scale learning problems.

### 3.5. Generalization Performance

In order to prevent overfitting, the optimal level of sparsity is typically not when the classification performance achieves its maximum on the training data, but on testing data. Therefore, it is important to examine the quality of generalization performance of the algorithm. We thus compare the generalization perfor-
This is done using cross validation traditionally; recently early stopping has been quite popular in large-scale learning. The classification performance can be evaluated using the receiver operating characteristic (ROC) analysis and K-fold cross validation. In this case, we used two-fold cross validation, similar to early stopping technique in the machine learning community. In signal detection theory, a ROC curve plots the true positive rate versus false positive rate, as the discrimination threshold varies, for a binary classifier system (Green & Swets, 1966). The area under the ROC curve is termed $A_z$ value, $A_z \in [0,1]$. The higher the $A_z$ value, the better the classification performance.

Fig. 6 shows the generalization performance for data with varying noise levels. We split the data into training and testing sets, and run both the linearized Bregman algorithm and the grid search method. The generalization performance for the dataset is determined when the testing $A_z$ reaches its maximum. Numerical results show that linearized Bregman always achieves comparable, if not better, testing performance compared with the grid search method. More specifically, the maximum of testing $A_z$ value using linearized Bregman algorithm is the same as, or sometimes slightly higher than, that of the grid search method. Fig. 6(a)-(d) shows the consistency of such a result for increasing noise level.

4. Conclusions

We have presented a Bregman regularized model and an efficient linearized Bregman algorithm to generate a solution path for sparse logistic regression. The resultant algorithm accelerates the computation of the regularization path, compared to that of the traditional grid search method. We have tested both algorithms on feature selection problems. The linearized Bregman algorithm achieves comparable classification accuracy on noise-free data, while giving out better classification performance on noisy data. In conclusion, we have found the linearized Bregman algorithm very attractive for seeking the optimal level of sparsity in feature selection problems.

References


Linearized Bregman


