Journal of Machine Learning Research 1 (2008) 8888 Submitted 8; Published 8

A Fast Hybrid Algorithm for Large Scale ℓ_1-Regularized Logistic Regression

Jianing Shi js2615@columbia.edu
Department of Biomedical Engineering
Columbia University
New York, NY 10025, USA

Wotao Yin wotao.yin@rice.du
Department of Computational and Applied Mathematics
Rice University
Houston, TX 77005, USA

Stanley Osher sjo@math.ucla.edu
Department of Mathematics
University of California Los Angeles
Los Angeles, CA 90095, USA

Paul Sajda ps629@columbia.edu
Department of Biomedical Engineering
Columbia University
New York, NY 10025, USA

Editor:

Abstract

ℓ_1-regularized logistic regression, also known as sparse logistic regression, is widely used in machine learning, computer vision, data mining, bioinformatics and neural signal processing. The use of ℓ_1-regularization attributes attractive properties to the classifier, such as feature selection, robustness to noise, and as a result, classifier generality in the context of supervised learning. When a sparse logistic regression problem has large-scale data in high dimensions, it is computationally expensive to minimize the non-differentiable ℓ_1-norm in the objective function. Motivated by recent work (Hale et al., 2008; Koh et al., 2007), we propose a novel hybrid algorithm based on combining two types of optimization iterations: one being very fast and memory friendly while the other being slower but more accurate. Called hybrid iterative shrinkage (HIS), the resulting algorithm is comprised of a fixed point continuation phase and an interior point phase. The first phase is based completely on memory efficient operations such as matrix-vector multiplications, while the second phase is based on a truncated Newton's method. Furthermore, we show that various optimization techniques, including line search and continuation, can significantly accelerate convergence. The algorithm has global convergence at a geometric rate (a Q-linear rate in optimization terminology). We present a numerical comparison with several existing algorithms, using benchmark data from the UCI machine learning repository, and show our algorithm is the most computationally efficient without loss of accuracy.

Keywords: logistic regression, ℓ_1 regularization, fixed point continuation, supervised learning, large scale.

1. Introduction

Logistic regression is an important linear classifier in machine learning and has been widely used in computer vision (Bishop, 2007), bioinformatics (Tsuruoka et al., 2007), gene classification (Liao and Chin, 2007), and neural signal processing (Parra et al., 2005; Gerson et al., 2005; Philiastides and Sajda, 2006). ℓ_1-regularized logistic regression or so-called sparse logistic regression (Tibshirani, 1996), where the weight vector of the classifier has a small number of nonzero values, has been shown to have attractive properties such as feature selection and robustness to noise. For supervised learning with many features but limited training samples, overfitting to the training data can be a problem in the absence of proper regularization (Vapnik, 1982, 1988). To reduce overfitting and obtain a robust classifier, one must find a sparse solution.

Minimizing or limiting the ℓ_1-norm of an unknown variable (the weight vector in logistic regression) has long been recognized as a practical avenue for obtaining a sparse solution. The use of ℓ_1 minimization is based on the assumption that the classifier parameters have, *a priori*, a Laplace distribution, and can be implemented using maximum-a-posteriori (MAP). The ℓ_2-norm is a result of penalizing the mean of a Gaussian prior, while a ℓ_1-norm models a Laplace prior, a distribution with heavier tails, and penalizes on its median. Such an assumption attributes important properties to ℓ_1 regularized logistic regression in that it tolerates outliers and, therefore, is robust to irrelevant features and noise in the data. Because the solution is sparse, the nonzero components in the solution correspond to useful features for classification; therefore, ℓ_1-minimization also performs feature selection (Littlestone, 1988; Ng, 1998), important for data mining and biomedical data analysis.

1.1 Logistic Regression

The basic form of logistic regression seeks a hyperplane that separates data belonging to two classes. The inputs are a set of training data $X = [x_1, \cdots, x_m]^\top \in \mathbb{R}^{m\times n}$, where each row of X is a sample and samples of either class are assumed to be independently identically distributed, and class labels $b \in \mathbb{R}^m$ of $-1/+1$ elements. A linear classifier is a hyperplane $\{x : w^\top x + v = 0\}$, where $w \in \mathbb{R}^n$ is a set of weights and $v \in \mathbb{R}$ the intercept. The conditional probability for the classifier label b based on the data, according to the logistic model, takes the following form,

$$p(b_i|x_i) = \frac{\exp\big((w^T x_i + v)b_i\big)}{1 + \exp\big((w^T x_i + v)b_i\big)}, \quad i = 1, ..., m. \tag{1}$$

The average logistic loss function can be derived from the empirical logistic loss, computed from the negative log-likelihood of the logistic model associated with all the samples, divided by number of samples m,

$$l_{\text{avg}}(w, v) = \frac{1}{m}\sum_{i=1}^{m} \theta\big((w^T x_i + v)b_i\big), \tag{2}$$

where θ is the logistic transfer function: $\theta(z) := \log(1+\exp(-z))$. The classifier parameters w and v can be determined by minimizing the average logistic loss function,

$$\arg\min_{w,v} \; l_{\text{avg}}(w, v). \tag{3}$$

Such an optimization can also be interpreted as a MAP estimate for classifier weights w and intercept v.

1.2 ℓ_1-Regularized Logistic Regression

The so-called *sparse logistic regression* has emerged as a popular linear decoder in the field of machine learning, adding the ℓ_1-penalty on the weights w:

$$\arg\min_{w,v} \; l_{\text{avg}}(w,v) + \lambda\|w\|_1, \tag{4}$$

where λ is a regularization parameter. It is well-known that ℓ_1 minimization tends to give sparse solutions. The ℓ_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, ℓ_1 regularization also has appealing asymptotic sample-consistency for feature selection (Zhao and Yu, 2007).

Signals arising in the natural world tend to be sparse (Parra et al., 2001). Sparsity also arises in signals represented in a certain basis, such as the wavelet transform, the Krylov subspace, etc. Exploiting sparsity in a signal is therefore a natural constraint to employ in algorithm development. An exact form of sparsity can be sought using the ℓ_0 regularization, which explicitly penalizes the number of nonzero components,

$$\arg\min_{w,v} \; l_{\text{avg}}(w,v) + \lambda\|w\|_0. \tag{5}$$

Although theoretically attractive, problem (5) is in general NP-hard (Natarajan, 1995), requiring an exhaustive search. Due to this computational complexity, ℓ_1 regularization has become a popular alternative, and is subtly different than ℓ_0 regularization, in that the ℓ_1 norm penalizes large coefficients/parameters more than small coefficients/parameters.

The idea of adopting the ℓ_1 regularization for seeking sparse solutions to optimization problems has a long history. As early as the 1970's, Claerbout and Muir first proposed to use ℓ_1 to deconvolve seismic traces (Claerbout and Muir, 1973), where a sparse reflection function was sought from bandlimited data (Taylor et al., 1979). In the 1980's, Donoho et al quantified the ability of ℓ_1 to recover sparse reflectivity functions (Donoho and Stark, 1989; Donoho and Logan, 1992). After the 1990s', there was a dramatic rise of applications using the sparsity-promoting property of the ℓ_1 norm. Sparse model selection was proposed in statistics using LASSO (Tibshirani, 1996), wherein the proposed soft thresholding is related to that in wavelet thresholding (Donoho et al., 1995). Basis pursuit, which aims to extract sparse signal representation from overcomplete dictionaries, also underwent great development during this time (Donoho and Stark, 1989; Donoho and Logan, 1992; Chen et al., 1998; Donoho and Huo, 2001; Donoho and Elad, 2003; Donoho, 2006b). In recent years, minimization of the ℓ_1 norm has appeared as a key element in the emerging field of compressive sensing (Candés et al., 2006; Candés and Tao, 2006; Donoho, 2006a; Figueiredo et al., 2007; Hale et al., 2008). ℓ_1 minimization also has far reaching impact on various applications such as portfolio optimization (Lobo et al., 2007), sparse principle component analysis (d'Aspremont et al., 2005; Zou et al., 2006), sparse interconnect wiring design (Vandenberghe et al., 1997, 1998), sparse control system design (Hassibi et al., 1999), and optimization of well-connected sparse graphs (Ghosh and Boyd, 2006). Research on total variation based image processing (Rudin et al., 1992) also shows that minimizing the ℓ_1-norm of the intensity gradient can effectively remove random noise. In the realm of machine learning, ℓ_1 regularization exists in various forms of classifiers, including ℓ_1-regularized logistic regression, ℓ_1-regularized probit regression (Figueiredo and Jain, 2001; Figueiredo, 2003), ℓ_1-regularized support vector machines (Zhu et al., 2004), and ℓ_1-regularized multinomial logistic regression (Krishnapuram et al., 2005).

1.3 Existing Algorithms for ℓ_1-Regularized Logistic Regression

The ℓ_1-regularized logistic regression problem (4) is a convex and non-differentiable problem. A solution always exists but can be non-unique. These characteristics postulate some difficulties in solving the problem. Generic methods for nondifferentiable convex optimization, such as the ellipsoid method and various sub-gradient methods (Shor, 1985; Polyak, 1987), are not designed to handle instances of (4) with data of very large scale. There has been very active development on numerical algorithms for solving the ℓ_1-regularized logistic regression, including LASSO (Tibshirani, 1996), Gl1ce (Lokhorst, 1999), Grafting (Perkins and Theiler, 2003), GenLASSO (Roth, 2004), and SCGIS (Goodman, 2004). The IRLS-LARS (iteratively reweighted least squares least angle regression) algorithm uses a quadratic approximation for the average logistic loss function, which is consequently solved by the LARS (least angle regression) method (Efron et al., 2004; Lee et al., 2006). The BBR (Bayesian logistic regression) algorithm, described in (Genkin et al., 2004; Madigan et al., 2005; Eyheramendy et al., 2003), uses a cyclic coordinate descent method for the Bayesian logistic regression. Glmpath, a solver for ℓ_1-regularized generalized linear models using path following methods, can also handle the logistic regression problem (Park and Hastie, 2007). MOSEK is a general purpose primal-dual interior point solver, which can solve the ℓ_1-regularized logistic regression by formulating the dual problem, or treating it as a geometric program (Boyd et al., 2007). SMLR, algorithms for various sparse linear classifiers, can also solve sparse logistic regression (Krishnapuram et al., 2005). Recently, Koh, Kim, and Boyd proposed an interior-point method (Koh et al., 2007) for solving (4). Their algorithm takes truncated Newton steps and uses preconditioned conjugated gradient iterations. This interior-point solver is efficient and provides a high-precision solution. The truncated Newton method has fast convergence, but forming and solving the underlying Newton systems require excessive amounts of memory for large-scale problems, making solving such large-scale problems prohibitive. A comparison of several of these different algorithms can be found in (Schmidt et al., 2007).

1.4 Our Hybrid Algorithm

In this paper, we propose a hybrid algorithm that is comprised of two phases: the first phase is based on a new algorithm called iterative shrinkage, inspired by a fixed point continuation (FPC) (Hale et al., 2008), which is computationally fast and memory friendly; the second phase is a customized interior point method, devised by (Koh et al., 2007).

Fig. 1 shows a diagram of the hybrid algorithm, termed Hybrid Iterative Shrinkage (HIS) algorithm. Our algorithm requires less memory and, on mid/large-scale problems, runs faster than the interior point method. The iterative shrinkage phase only performs

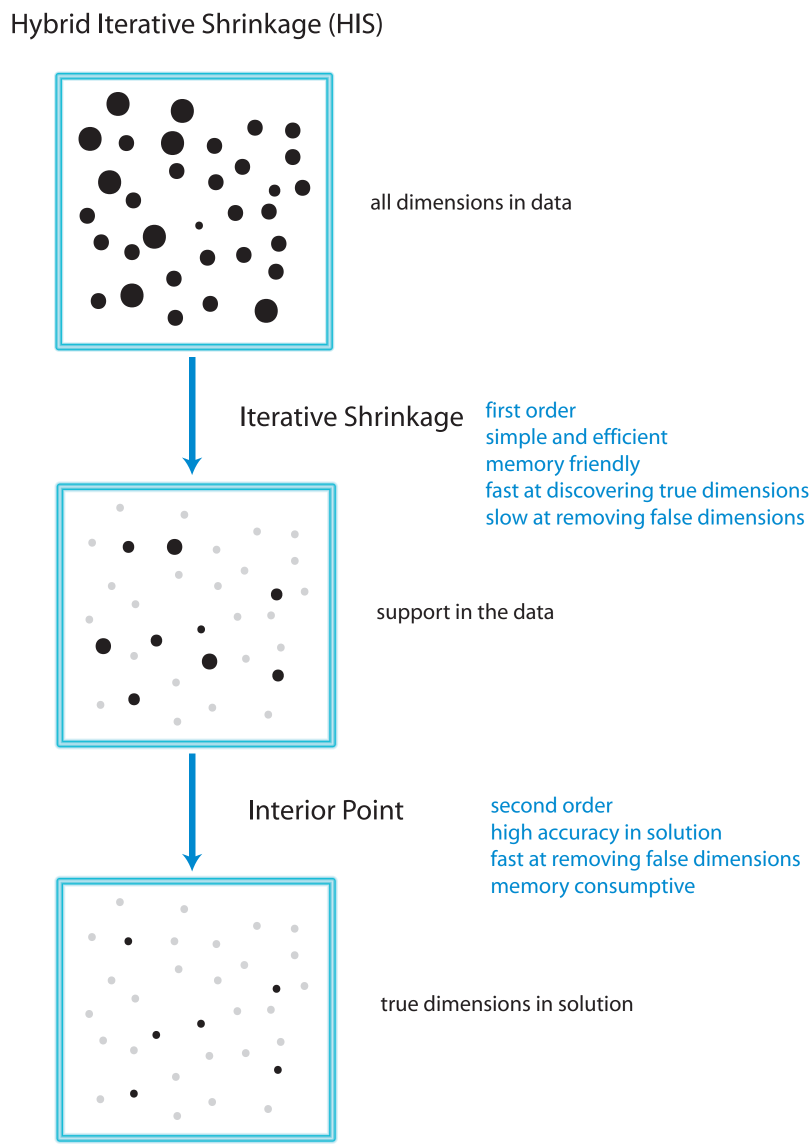

Figure 1: A diagram of our proposed hybrid iterative shrinkage (HIS) algorithm. The HIS algorithm is comprised of two phases: the iterative shrinkage phase and the interior point phase. The iterative shrinkage is inspired by a fixed point continuation method (Hale et al., 2008), which is computationally fast and memory friendly. The interior point method is based on a second-order truncated Newton method, devised by (Koh et al., 2007). Our hybrid approach will take advantage of different computational strengths of the two methods and use them for optimal speedup while attaining high accuracy. Black dots indicate the nonzero dimensions, gray dots indicate dimensions that are get rid of, and size of the dots show the error that each dimension contributes to the final solution. Note that the final solution is sparse with an overall small error.

matrix-vector multiplications in size of X, as well as a very simple *shrinkage* operation (see (16) below), and therefore requires minimal memory consumption. By extending the results in (Hale et al., 2008), we prove Q-linear convergence and show that the signs of w_{opt} (hence, the indices of nonzero elements) are obtained in a finite number of steps, typically much earlier than convergence. Based on the latter result, we propose a hybrid algorithm that is even faster and results in high accurate solutions. Specifically, our algorithm predicts the sign changes in future shrinkage iterations, and when the signs of w^k are likely to be stable, switches to the interior point method and operates on a reduced problem that is much smaller than the original. The interior point method achieves high accuracy in the solution, making our hybrid algorithm equally accurate, as will be shown in the Section 4.

There are several novel aspects of our hybrid approach. The rationale of the hybrid approach is based on the observation that the iterative shrinkage phase reduces the algorithm to gradient projection after a finite number of iterations, which will be described in Section 3.1. We build on this observation a hybrid approach to take advantage of the two phases of the computation by using two types of methods. In the first phase, inspired by the FPC by Hale et al (Hale et al., 2008), we customize the iterative shrinkage algorithm for the sparse logistic regression, whose objective function is not quadratic. In particular, the step length in the iterative shrinkage algorithm is not constant, unlike the compressive sensing problem. Therefore, we resort to a line search strategy to avoid computing the Hessian matrix (required for finding the step length for stability). In addition, the ℓ_1 regularization is only applied to the w component and not v in sparse logistic regression. This change in the model requires a different shrinkage step, as well as a careful treatment in the line search strategy.

The remainder of the paper is organized as follows. In Section 2, we present the iterative shrinkage algorithm for sparse logistic regression, and prove its global convergence and Q-linear convergence. In Section 3, we give the rationale for the hybrid approach, together with a description of the hybrid algorithm. Numerical results are presented in Section 4. We conclude the paper in Section 5.

2. Sparse Logistic Regression using Iterative Shrinkage

The iterative shrinkage algorithm used in the first phase is inspired by a fixed point continuation algorithm by (Hale et al., 2008).

2.1 Notation

For simplicity, we define $\|\cdot\| := \|\cdot\|_2$, as the Euclidean norm. The *support* of $x \in \mathbb{R}^n$ is denoted by $\text{supp}(x) := \{i : x_i \neq 0\}$. We use g to denote the gradient of f, i.e., $g(x) = \nabla f(x)$, $\forall x$. For any index set $I \subseteq \{1, \ldots, n\}$ (later, we will use index sets E and L), $|I|$ is the cardinality of I and x_I is defined as the sub-vector of x of length $|I|$, consisting only of components x_i, $i \in I$. Similarly, for any vector-value mapping h, $h_I(x)$ denotes the sub-vector of $h(x)$ consisting of $h_i(x)$, $i \in I$.

To express the subdifferential of $\|\cdot\|_1$ we will use the signum function and multi-function (i.e., set-valued mapping). The signum function of $t \in \mathbb{R}$ is

$$\operatorname{sgn}(t) := \begin{cases} +1 & t > 0, \\ 0 & t = 0, \\ -1 & t < 0; \end{cases}$$

while the signum multi-function of $t \in \mathbb{R}$ is

$$\operatorname{SGN}(t) := \partial|t| = \begin{cases} \{+1\} & t > 0, \\ [-1, 1] & t = 0, \\ \{-1\} & t < 0, \end{cases}$$

which is also the subdifferential of $|t|$.

For $x \in \mathbb{R}^n$, we define $\operatorname{sgn}(x) \in \mathbb{R}^n$ and $\operatorname{SGN}(x) \subset \mathbb{R}^n$ component-wise as $(\operatorname{sgn}(x))_i := \operatorname{sgn}(x_i)$ and $(\operatorname{SGN}(x))_i := \operatorname{SGN}(x_i)$, $i = 1, 2, \cdots, n$, respectively. Furthermore, vector operators such as $|x|$ and $\max\{x, y\}$ are defined to operate component-wise, analogous with the definitions of sgn and SGN above. For $x, y \in \mathbb{R}^n$, let $x \odot y \in \mathbb{R}^n$ denote the component-wise product of x and y, i.e., $(x \odot y)_i = x_i y_i$. Finally, we let X^* denote the set of all optimal solutions of problem (6).

2.2 Review of Fixed Point Continuation for ℓ_1-minimization

A fixed-point continuation algorithm was proposed in (Hale et al., 2008) as a fast algorithm for large-scale ℓ_1-regularized convex optimization problems. The authors considered the following large-scale ℓ_1-regularized minimization problem,

$$\min_{x \in \mathbb{R}^n} f(x) + \lambda \|x\|_1, \tag{6}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable and convex, but not necessarily strictly convex, and $\lambda > 0$. They devised a fixed-point iterative algorithm and proved its global convergence, including finite convergence for some quantities, and a Q-linear (Quotient-linear) convergence rate without assuming strict convexity of f or solution uniqueness. Numerically they demonstrated Q-linear convergence in the quadratic case $f(x) = \|Ax - b\|_2^2$, where A is completely dense, and applied their algorithm to ℓ_1-regularized compressed sensing problems. As we will adopt this algorithm for solving our problem (4), we review some important and useful results here and develop some new insights in the context of ℓ_1-regularized logistic regression.

The rationale for FPC is based on the idea of operator splitting. It is well-known in convex analysis that minimizing a function in the form of $\phi(x) = \phi_1(x) + \phi_2(x)$, where both ϕ_1 and ϕ_2 are convex, is equivalent to finding a zero of the subdifferential $\partial\phi(x)$, i.e., seeking x satisfying $\mathbf{0} \in T_1(x) + T_2(x)$ for $T_1 := \partial\phi_1$ and $T_2 := \partial\phi_2$. We say $(I + \tau T_1)$ is invertible if $y = x + \tau T_1(x)$ has a unique solution x for any given y. For $\tau > 0$, if $(I + \tau T_1)$ is invertible and T_2 is single-valued, then

$$\begin{aligned} \mathbf{0} \in T_1(x) + T_2(x) &\iff \mathbf{0} \in (x + \tau T_1(x)) - (x - \tau T_2(x)) \\ &\iff (I - \tau T_2)x \in (I + \tau T_1)x \\ &\iff x = (I + \tau T_1)^{-1}(I - \tau T_2)x \end{aligned} \tag{7}$$

This gives rise to the forward-backward splitting algorithm in the form of a fixed-point iteration,

$$x^{k+1} := (I + \tau T_1)^{-1}(I - \tau T_2)x^k. \tag{8}$$

Applying (7) to problem (4), where $\phi_1(x) := \lambda\|x\|_1$ and $\phi_2(x) := f(x)$, the authors of Hale et al. (2008) obtained the following optimality condition of x^*:

$$x^* \in X^* \iff \mathbf{0} \in \lambda\,\mathrm{SGN}(x^*) + g(x^*) \iff x^* = (I + \tau T_1)^{-1}(I - \tau T_2)x^*, \tag{9}$$

where $T_2(\cdot) = g(\cdot)$, the gradient of $f(\cdot)$, and $(I + \tau T_1)^{-1}(\cdot)$ is the shrinkage operator. Therefore, the fixed-point iteration (8) for solving (6) becomes

$$x^{k+1} = s \circ h(x^k), \tag{10}$$

which is a composition of two mappings s and h from $\mathbb{R}^n$ to $\mathbb{R}^n$ where

$$h(\cdot) \;\; := \;\; I(\cdot) - \tau\nabla f(\cdot). \tag{11}$$

In each iteration, the gradient descent step h reduces $f(x)$ by moving along the negative gradient direction of $f(x^k)$ and the shrinkage step s reduces the ℓ_1-norm by "shrinking" the magnitude of each nonzero component in the input vector.

2.3 Iterative Shrinkage for Sparse Logistic Regression

Recall that in the sparse logistic regression problem (4), the ℓ_1 regularization is only applied to w, not to v. Therefore, we propose a slightly different fixed point iteration. For simplicity of notation, we define column vectors $u = (w; v) \in \mathbb{R}^{n+1}$ and $c_i = (a_i; b_i) \in \mathbb{R}^{n+1}$, where $a_i = bx_i$, for $i = 1, 2, ..., m$. This reduces (4) to

$$\min_u \; \frac{1}{m}\sum_{i=1}^{m} \theta(c_i^\top u) + \lambda\|u_{1:n}\|_1, \tag{12}$$

where $l_{\text{avg}} = \frac{1}{m}\sum_{i=1}^{m} \theta(c_i^T u)$, and θ denotes the logistic transfer function $\theta(z) = \log(1 + \exp(-z))$.

The gradient and Hessian of l_{avg} with respect to u is given by

$$\begin{aligned} g(u) &\equiv \nabla l_{\text{avg}}(u) = \frac{1}{m}\sum_{i=1}^{m} \theta'(c_i^\top u)c_i, \\ H(u) &\equiv \nabla^2 l_{\text{avg}}(u) = \frac{1}{m}\sum_{i=1}^{m} \theta''(c_i^\top u)c_i c_i^\top, \end{aligned} \tag{13}$$

where $\theta'(z) = -(1 + e^z)^{-1}$ and $\theta''(z) = (2 + e^{-z} + e^z)^{-1}$. To guarantee convergence, we require the step to be bounded by $2(\max_u \lambda_{\max} H(u))^{-1}$.

The iterative shrinkage algorithm for sparse logistic regression is

$$\begin{aligned} u_{1:n} &= s \circ h_{1:n}(u), \text{ for } w \text{ component} \\ u_{n+1} &= h_{n+1}(u), \text{ for } v \text{ component} \end{aligned} \tag{14}$$

which is a composition of two mappings h and s from $\mathbb{R}^n$ to $\mathbb{R}^n$. The gradient descent operator is defined as

$$h(\cdot) = \cdot - \tau g(\cdot) = \cdot - \tau \nabla l_{\text{avg}}(\cdot). \tag{15}$$

The shrinkage operator, on the other hand, can be written as

$$s(\cdot) = \text{sgn}(\cdot) \odot \max\{|\cdot| - \nu, 0\}, \tag{16}$$

where $\nu = \lambda\tau$. Shrinkage is also referred to soft-thresholding in the language of wavelet analysis:

$$(s(y))_i = \begin{cases} y_i - \nu, & y_i > \nu, \\ 0, & y_i \in [-\nu, \nu], \\ y_i + \nu, & y_i < -\nu. \end{cases} \tag{17}$$

While the authors in Hale et al. (2008) use a constant stepsize satisfying

$$0 < \tau < 2/\lambda_{\max}\{H_{EE}(u) : u \in \Omega\}, \tag{18}$$

we employ line search to avoid the expensive calculation of maximum eigenvalues. We will present the convergence of the iterative shrinkage algorithm in Section 2.4. The details of the line search algorithm will be discussed in Section 2.5.

2.4 Convergence

Global convergence and finite convergence on certain quantities were proved in (Hale et al., 2008) when the following conditions are met: (i) the optimal solution set X^* is non-empty, (ii) $f \in C^2$ and its Hessian $H = \nabla^2 f$ is positive semi-definite in $\Omega = \{x : \|x - x^*\| \leq \rho\} \subset \mathbb{R}^n$ for $\rho > 0$, and (iii) the maximum eigenvalue of H is bounded on Ω by a constant $\hat{\lambda}_{\max}$ and the step length τ is uniformly less than $2/\hat{\lambda}_{\max}$. These conditions are sufficient for the forward operator $h(\cdot)$ to be non-expansive.

Assumption. Assume problem (4) has an optimal solution set $X^* \neq \emptyset$, and there exists a set

$$\Omega = \{x : \|x - x^*\| \leq \rho\} \subset \mathbb{R}^n$$

for some $x^* \in X^*$ and $\rho > 0$ such that $f \in C^2(\Omega)$, $H(x) := \nabla^2 f(x) \succeq 0$ for $x \in \Omega$ and

$$\hat{\lambda}_{\max} := \sup_{x \in \Omega} \lambda_{\max}(H(x)) < \infty. \tag{19}$$

For simplicity for the analysis, we choose a constant time step τ in the fixed-point iterations (14): $x^{k+1} = s(x^k - \tau g(x^k))$, where $\nu = \tau\lambda$, and

$$\tau \in \left(0, 2/\hat{\lambda}_{\max}\right), \tag{20}$$

which guarantees that $h(\cdot) = I(\cdot) - \tau g(\cdot)$ is non-expansive in Ω.

Theorem. Under Assumption, the sequence $\{u^k\}$ generated by the fixed-point iterations (14) applied to problem (4) from any starting point $x^0 \in \Omega$ converges to some $u^* \in U^* \cap \Omega$. In addition, for all but finitely many iterations, we have

$$u_i^k = u_i^* = 0, \qquad \forall i \in L = \{i : |g_i^*| < \lambda, 1 \leq i \leq n\}, \tag{21}$$

$$\text{sgn}(h_i(u^k)) = \text{sgn}(h_i(u^*)) = -\frac{1}{\lambda} g_i^*, \qquad \forall i \in E = \{i : |g_i^*| = \lambda, 1 \leq i \leq n\}, \tag{22}$$

where as long as

$$\omega := \min\{\nu(1 - \frac{|g_i^*|}{\lambda}) : i \in L\} > 0. \tag{23}$$

The numbers of iterations not satisfying (21) and (22) do not exceed $\|u^0 - u^*\|^2/\omega^2$ and $\|u^0 - u^*\|^2/\nu^2$, respectively.

Proof. We sketch the proof here. First, the iteration (14) is shown to be non-expansive in ℓ_2, i.e., $\|u^k - u^*\|$ does not increase in k with the assumption on the stepsize τ. Specifically, in Assumption, the stepsize τ is chosen small enough to guarantee that $\|h(u^k) - h(u^*)\| \leq \|u^k - u^*\|$ (in practice, τ is determined, for example, by line search.) On the other hand, through a component-wise analysis, one can show that no matter what τ is, the shrinkage operator $s(\cdot)$ is always non-expansive, i.e., $\|s(h_{1:n}(u^k)) - s(h_{1:n}(u^*))\| \leq \|h_{1:n}(u^k) - h_{1:n}(u^*)\|$. Therefore, from the definition of u^{k+1} in (14), we have

$$\|u^{k+1} - u^*\| \leq \|u^k - u^*\|, \tag{24}$$

using the fact that u^* is optimal if and only if u^* is a fixed point with respect to (14). However, this non-expansiveness of (14) does not directly give convergence.

Next, $\{u^k\}$ is shown to have a limit point $\bar{x}$, i.e., a subsequence converging to $\bar{u}$, due to the compactness of Ω and (24). (14) can be proven to converge globally to $\bar{u}$. To show this, we first get

$$\|[s \circ h_{1:n}(\bar{u}); h_{n+1}(\bar{u})] - [s \circ h_{1:n}(u^*); h_{n+1}(u^*)]\| = \|\bar{u} - u^*\|,$$

from the fact that $\bar{u}$ is a limit point, and then use this equation to show that $\bar{u} = [s \circ h_{1:n}(\bar{u}); h_{n+1}(\bar{u})]$, i.e., $\bar{u}$ is a fixed point with respect to (14), and thus an optimal solution. Repeating the first step above we have $\|u^{k+1} - \bar{u}\| \leq \|u^k - \bar{u}\|$, which extends $\bar{u}$ from being the limit of a subsequence to one of the entire sequence.

Finally, to obtain the finite convergence result, we need to take a closer look at the shrinkage operator $s(\cdot)$. When (21) does not hold for some iteration k at component i, we have $|u_i^{k+1} - u_i^*|^2 \leq |u_i^k - u_i^*|^2 - \omega^2$, and for (22), we have $|u_i^{k+1} - u_i^*|^2 \leq |u_i^k - u_i^*|^2 - \nu^2$. Obviously, there can be only a finite number of iterations k in which either (21) or (22) does not hold, and the total such numbers do not exceed $\|u^0 - u^*\|^2/\omega^2$ and $\|u^0 - u^*\|^2/\nu^2$, respectively. **Q.E.D.**

A linear convergence result with a certain convergence rate can also be obtained. As long as $H_{EE}(x^*) := [H_{i,j}(x^*)]_{i,j \in E}$ has full rank *or* $f(x)$ is convex quadratic in x, the sequence $\{x^k\}$ converges to x^* R-linearly, and $\{\|x^k\|_1 + \mu f(x^k)\}$ converges to $\|x^*\|_1 + \mu f(x^*)$ Q-linearly. Furthermore, if $H_{EE}(x^*)$ has the full rank, then R-linear convergence can be strengthened to Q-linear convergence by using the fact that the minimal eigenvalue of H_{EE} at x^* is strictly greater than 0.

2.5 Line Search

An important element of the iterative shrinkage algorithm is the step length τ at each iteration. To ensure the stability of the algorithm, we require the step length to satisfy $0 < \tau < 2/\lambda_{\max}\{H_{EE}(u) : u \in \Omega\}$. In compressive sensing, where the smooth part of the objective function is quadratic, the time step is constant. In sparse logistic regression, however, the Hessian matrix changes at each iteration. If one has to dynamically compute

Algorithm 1 Fixed-Point Continuation Algorithm

Require: $A = [c_1^\top; c_2^\top; \cdots; c_m^\top] \in \mathbb{R}^{m \times n+1}$, $u = (w; v) \in \mathbb{R}^{n+1}$, $f(u) = m^{-1}\phi(Au)$, task: $\min_u l_{\text{avg}}(u) + \lambda\|u\|_1$

Initialize u^0

while "not converge" **do**

  Armijo-like line search algorithm (Algorithm 2)

  $k = k + 1$

end while

the time step length at each iteration, this requires an expensive computation for the Hessian matrix. Therefore, we resort to a line search algorithm to avoid such a computational burden. For large scale problems, a line search method, if used appropriately, can save tremendous CPU time and memory.

Algorithm 2 Armijo-like Line Search Algorithm

Compute heuristic step length α_0

Gradient step: $u^{k-} = u^k - \alpha_0 \nabla l_{\text{avg}}(u^k)$

Shrinkage step: $u^{k+} = s_{1:n}(u^{k-}, \lambda\alpha_0)$

Obtain search direction: $p^k = u^{k+} - u^k$

while "j < max line search attempts" **do**

  if Armijo-like condition is met **then**

    Accept line search step, update $u^{k+1} = u^k + \alpha_j p^k$

  else

    Keep backtracking $\alpha_j = \mu\alpha_{j-1}$

  end if

  $j = j + 1$

end while

Denote the objective function for the ℓ_1-regularized logistic regression as $\phi(w, v)$ for convenience

$$\phi(w, v) = l_{\text{avg}}(w, v) + \lambda\|w\|_1, \tag{25}$$

where $l_{\text{avg}}(w, v) = \frac{1}{m}\sum_{i=1}^m \theta\big((w^T x_i + v)b_i\big)$ and θ is the logistic transfer function.

A line search method, at each iteration, computes the step length and the search direction p^j

$$u^{k+1} = u^k + \alpha_k p^k, \tag{26}$$

where the search direction is a combination of negative steepest gradient and the shrinkage step. For our sparse logistic regression, a sequence of step length candidates are identified, and a decision is made to accept one when certain conditions are satisfied. We compute a heuristic step length and gradually decrease it until a sufficient decrease condition is met.

Let's define the heuristic step length as α_0. Ideally the choice of step length α_0, would be a global minimizer of the smooth part of the objective function, motivated by a similar approach in GPSR (Figueiredo et al., 2007):

$$\varphi(\alpha) = l_{\text{avg}}(u^k + \alpha p^k), \ \ \alpha > 0, \tag{27}$$

which is too expensive to evaluate, unlike the quadratic case in compressive sensing. Therefore, an inexact line search strategy is usually performed in practice to identify a step length that achieves sufficient decrease in $\varphi(\alpha)$ at minimal cost. This is computed by finding a minimizer of the quadratic approximation

$$l_{\text{avg}}(u^k - \alpha\nabla l_{\text{avg}}(u^k)) \approx l_{\text{avg}}(u^k) - \alpha\nabla l_{\text{avg}}(u^k)^T \nabla l_{\text{avg}}(u^k) + 0.5\alpha^2 \nabla l_{\text{avg}}(u^k)^T H(u^k) \nabla l_{\text{avg}}(u^k).$$

Differentiating the right-hand side with respect to α and setting the derivative to zero, we obtain

$$\alpha_0 = \frac{\nabla l_{\text{avg}}(\bar{u}^k)^T \nabla l_{\text{avg}}(\bar{u}^k)}{\nabla l_{\text{avg}}(\bar{u}^k)^T H(\bar{u}^k) \nabla l_{\text{avg}}(\bar{u}^k)}, \tag{28}$$

where $\bar{u}_i^k = 0$, if $u_i = 0$ or $|g_i| < \lambda$ and $\bar{u}^k = u^k$ otherwise. Computationally a very useful trick is not to compute the Hessian matrix directly, since we only utilize the vector-matrix product between the gradient vector $l_{\text{avg}}(\bar{u}^k)$ and the Hessian matrix $H(\bar{u}^k)$.

The search direction p^k is a combination of the gradient descent step and the shrinkage step:

$$\begin{aligned} u^{k-} &= u^k - \nabla l_{\text{avg}}(u^k) \\ u^{k+} &= s_{1:n}(u^{k-}, \lambda\alpha_0) \\ p^k &= u^{k+} - u^k. \end{aligned} \tag{29}$$

It is easy to verify that $s_\nu(y)$ is the solution to the non-smooth unconstrained minimization problem $\min \frac{1}{2}\|x-y\|_2^2 + \lambda\|x\|_1$. This minimization problem is equivalent to the following smooth optimization constrained problem

$$\min \frac{1}{2}\|x-y\|_2^2 + \nu z, \text{ subject to } (x,z) \in \Omega := \{(x,z) \mid \|x\|_1 \le z\},$$

whose optimality condition is

$$(s(x,\nu) - x)^T (y - s(x,\nu) + \nu(z - \|s(x,\nu)\|_1) \ge 0, \tag{30}$$

for all $x \in \mathbb{R}^n$, $(y,z) \in \Omega$ and $\nu > 0$. Once we substitute $u - \tau g$ for x, u for y, $\|u_{1:n}\|_1$ for z and set $\nu = \lambda\tau$, the optimality condition becomes

$$(s_{1:n}(u-\tau g, \lambda\tau) - (u-\tau g))^T (u - s_{1:n}(u-\tau g, \lambda\tau)) + \lambda\tau(\|u_{1:n}\|_1 - \|s_{1:n}(u-\tau g, \lambda\tau)\|_1) \ge 0$$

Using the fact $u^+ = s_{1:n}(u-\tau g, \lambda\tau)$, $p = u^+ - u$, we get

$$g^T p + \lambda(\|u^+_{1:n}\|_1 - \|u_{1:n}\|_1) \le -p^T p/\tau, \tag{31}$$

which means

$$\nabla l_{\text{avg}}(u^k)^T p^k + \lambda\|u^{k+}_{1:n}\|_1 - \lambda\|u^k_{1:n}\|_1 \le 0 \tag{32}$$

We then geometrically backtrack the step lengths, letting $\alpha_j = \alpha_0,\ \mu\alpha_0,\ \mu^2\alpha_0, \ldots$, until the following Armijo-like condition is satisfied:

$$\phi(u^k + \alpha_j p^k) \le C_k + \alpha_j \Delta_k. \tag{33}$$

Notice that this is the Armijo-like condition for line search, stipulating that the step length α_j in the search direction p^k should produce a sufficient decrease of the objective function $\phi(u)$. C_k is a reference value with respect to the previous objective values, while the decrease in the objective function is described as

$$\Delta_k := \nabla l_{\text{avg}}(u^k)^T p^k + \lambda \|u_{1:n}^{k+}\|_1 - \lambda \|u_{1:n}^k\|_1 \leq 0. \tag{34}$$

There are two types of Armijo-like conditions depending on the choice of C_k. One can choose $C_k = \phi(u^k)$, which makes the line search monotone. One can also derive a non-monotone line search, where C_k is a convex combination of the previous value C_{k-1} and the function value $\phi(u^k)$. We refer interested readers to (Wen et al., 2008, 2009) for more details.

Fig. 2 illustrates the computational speed-up using line search. The top panel shows the evolution of the objective function as a function of iterations. Tested on the benchmark data from the UCI repository, we see that our algorithm results in a speed-up by a factor of 40 (6000 iterations without line search vs. 150 iterations with line search). The bottom panels shows the step length used in the algorithm. In absence of line search, we require the step length satisfying $\tau < 2/\hat{\lambda}_{\max}$. For the Armijo-like line search, we illustrate both the heuristic step length α_l (black curve) and the actual step length after backtracing (red curve). Red asterisk labels the transition point on the continuation path, a concept we will discuss in the next section. Note that the step lengths can be on the order of 100 times larger for line search vs. no line search.

2.6 Continuation Path

A continuation strategy is adopted in our algorithm, by designing a regularization path similar to that is used in (Hale et al., 2008),

$$\lambda_0 > \lambda_1 > ... > \lambda_{L-1} = \bar{\lambda}.$$

This idea is closely related to the homotopy algorithm in statistics, and has been successfully applied in the ℓ_1-regularized quadratic case, where the fidelity term $f(x) = \|Ax - b\|_2^2$. The rationale of using such a continuation strategy is due to a fast rate of convergence for large λ. Therefore, by taking advantage of different convergence rate for a family of regularization parameter λ, if stopped appropriately, we can speed up the convergence rate of the full path. An intriguing discussion regarding the convergence rate of fixed-point algorithm with λ and ω, the spectral properties of Hessian, was presented in (Hale et al., 2008). In the case of logistic regression, we have decided to use the geometric progression for the continuation path. We define

$$\lambda_i = \lambda_0 \beta^{i-1}, for\ i = 0, ..., L-1,$$

where λ_0 can be calculated from the ultimate $\bar{\lambda}$ we are interested in and the continuation path length L, i.e., $\lambda_0 = \bar{\lambda}/\beta^{L-1}$.

As mentioned earlier, the goal of a continuation strategy is to construct a path with different rate of convergence, with which we can speed up the whole algorithm. The solution obtained from a previous subpath associated with λ_{i-1} is used as the initial condition for the

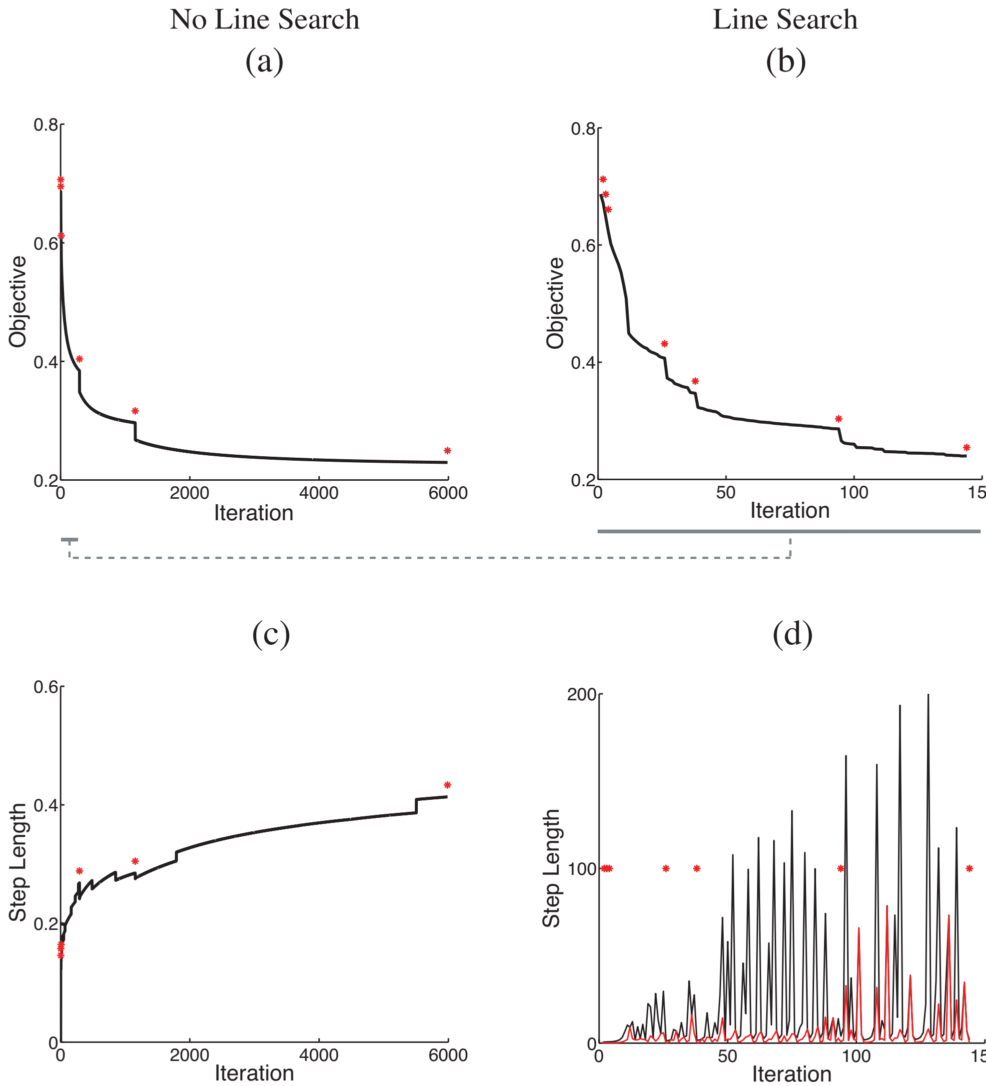

Figure 2: Illustration of the Armijo-like line search, comparing the iterative shrinkage algorithm with (right column) and without (left column) line search. (a) The objective function of the iterative shrinkage algorithm without line search, attaining convergence after 6000 iterations. (b) The objective of the iterative shrinkage algorithm with line search, converging at around 150 iterations. The gray bars under the "Iteration" axes highlight the difference between the number of iterations–the gray bar in (a) represents the same number of iterations as the gray bar in (b). (c) The step length without line search is bounded by $2/\hat{\lambda}_{\max}$ to ensure convergence. (d) The step length used in the Armijo-like line search, (black) heuristic step length α_l (Eqn. (28)), (red) actual time step after backtracking. Data used in this numerical experiment are the ionosphere data from the UCI machine learning data repository (http://archive.ics.uci.edu/ml/datasets/Ionosphere). Parameters used are $utol = 0.001$, $gtol = 0.01$, $\lambda_0 = 0.1$, $\lambda = 0.001$.

next subpath for λ_i. Note that we design the path length L and the geometric progression rate β in such a way that the initial regularization λ_0 is fairly large, leading to a sparse solution for the initial path. Therefore, the initial condition for the whole path, considering the sparsity in solution, is a zero vector.

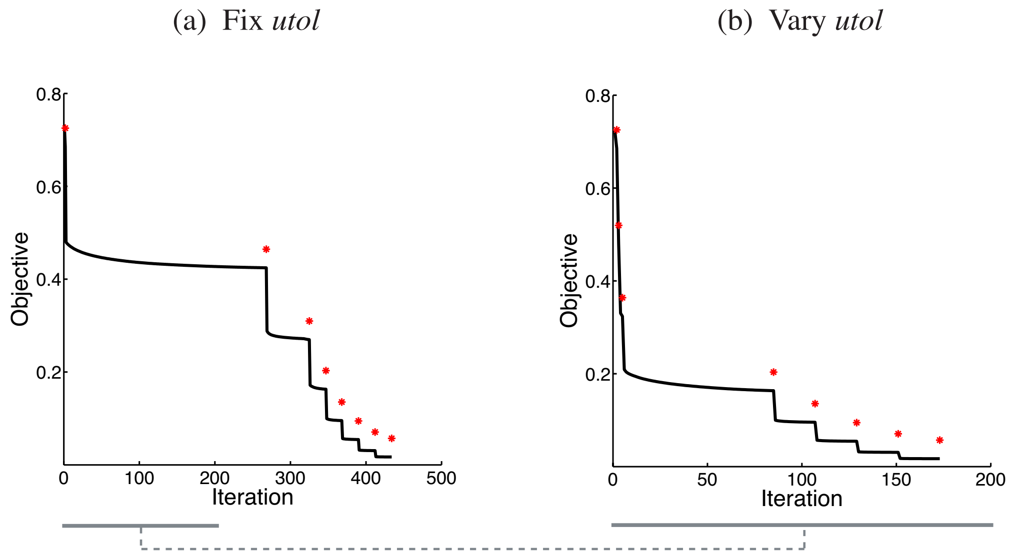

Figure 3: Illustration of the continuation strategy (a) using a fixed $utol = 0.0001$ is used for the stopping criterion, (b) using a varying $utol$ according to geometric progression. Note that a stronger convergence is not necessary in earlier stages on the continuation path. By using a varying $utol$, especially tightening $utol$ as we move along the path, we can accelerate the fixed point continuation algorithm. Data used in this experiment has 10000 dimension and 100 samples. A continuation path of length 8, starting from 0.128 and ending at 0.001.

Another design issue regarding such a continuation strategy is we stop each subpath according to some criteria, in an endeavor to approximate the solution in the next λ as fast as possible. This means that a strong convergence is not required in subpath's except for the final one, and we can vary the stopping criteria to "tighten" such a convergence as we proceed. The following two stopping criteria are used.

$$\frac{\|u^{k+1} - u^k\|}{\max(\|u^k\|, 1)} < utol_i, \tag{35}$$

$$\frac{\|\nabla l_{\text{avg}}(u^k)\|_\infty}{\lambda_i} - 1 < gtol. \tag{36}$$

The first stopping criterion requires that relative change in u is small, while the second one is related to the optimality condition, defined in Eqn. (37). Theoretically, we would like to vary $utol_i$ to attain a seamless Q-linear convergence path. Such a choice seems to be problem dependent, and probably even data dependent in practice. It remains an important, yet difficult research topic to study the properties of different continuation strategies. We have chosen to use a geometric progression for the tolerance value, $utol_i = utol_0 * \gamma^{i-1}$, with $utol_0 = utol/\gamma^{L-1}$. In our numerical simulation, we use $utol = 10^{-4}$ and $gtol = 0.2$.

Fig. 3 shows the continuation path using fixed *utol* and a varying *utol* following geometric progression. Red asterisk labels the transition point on the regularization path. When we use a fixed *utol* to ensure strong convergence each for λ along the path, the solver spends a lot of time evolving slowly. One can see in (a) that the objective function shows a fairly flat reduction at earlier stages of the path. Clearly by relaxing the convergence at earlier stages of the path, we can accelerate the computation, shown in (b).

3. Hybrid Iterative Shrinkage (HIS) Algorithm

3.1 Why A Hybrid Approach?

The hybrid approach is based on an interesting observation for the iterative shrinkage algorithm, regarding some finite convergence properties. The optimality condition for $\min f(x) + \lambda\|x\|_1$ is the following

$$g(x) + \lambda \mathrm{SGN}(x) \in \mathbf{0}, \tag{37}$$

which requires that $|g_i| \leq \lambda$, for $i = 1, ..., n$. We define two index sets

$$L := \{i : |g_i^*| < \lambda\} \text{ and } E := \{i : |g_i^*| = \lambda\}, \tag{38}$$

where $g^* = g(u^*)$ is constant for all $u^* \in X^*$ and $|g_i^*| \leq \lambda$ for all i. Hence, $L \cap E = \emptyset$ and $L \cup E = \{1, \ldots, n\}$. The following holds true for all but a finite number of k:

$$\begin{aligned} u_i^k = u_i^* &= 0, \qquad \forall i \in L, \\ \mathrm{sgn}(h_i(u^k)) = \mathrm{sgn}(h_i(u^*)) &= -\frac{1}{\lambda} g_i^*, \qquad \forall i \in E. \end{aligned}$$

Assume that the underlying problem is nondegenerate, then L and E equal the sets of zero and nonzero components in x^*. According to the above finite convergence result, the iterative shrinkage algorithm obtains L and E, and thus the optimal support and signs of the optimal nonzero components, in a finite number of steps.

Corollary. Under Assumption (page 9), after a finite number of iterations, the fixed-point iteration (14) reduces to gradient projection iterations for minimizing $\phi(u_E)$ over a constraint set O_E, where

$$\phi(u_E) := -(g_E^*)^\top u_E + f((u_E; \mathbf{0})), \text{ and}$$

$$O_E = \{u_E \in \mathbb{R}^{|E|} : -\mathrm{sgn}(g_E^*) \odot u_E \geq 0\}.$$

Specifically, we have $u^{k+1} = (u_E^{k+1}; \mathbf{0})$ in which

$$u_E^{k+1} := P_{O_E}\left(u_E^k - \tau \nabla \phi(u_E^k)\right), \tag{39}$$

where P_{O_E} is the orthogonal projection onto O_E, and $\nabla\phi(u_E) = -g_E^* + g_E((u_E; \mathbf{0}))$. This corollary, see Corollary 4.6 in (Hale et al., 2008), can be directly applied to sparse logistic regression. It implies a very important fact: the fixed point continuation reduces to the gradient projection after a finite number of iterations. The proof of this corollary is in general true for the $u_{1:n}$, i.e. the w component in our problem.

The corollary in Section 3.1 implies an important fact: there are two phases in the fixed point continuation algorithm. In the first phase, the number of nonzero elements in the x evolve rapidly, until after a finite number of iterations, when the support (non-zero elements in a vector) is found. Precisely, it means that for all $k > K$, the nonzero entries in u^k include all true nonzero entries in u^* with the matched signs. However, unless k is large, u^k typically also has extra nonzeros. At this point, the fixed point continuation reduces to the gradient projection, starting the second phase of the algorithm. In the second phase, the zero elements in the vector stay unaltered, while the magnitude of the nonzero elements (support) keeps evolving.

The above observation is a general statement for any f that is convex. Recall the quadratic case, where $f = \|y - Ax\|_2^2$, the second phase is very fast in terms of convergence rate. This is due to the quadratic function, and in an application to compressive sensing, the fixed point continuation algorithm alone has resulted in super-fast performance for large scale problems (Hale et al., 2008). In the case of sparse logistic regression, we have a non-strictly convex f, the average logistic regression. This results in a fairly slow convergence rate when the algorithm reaches the second phase. In view of the continuation strategy we have, this greatly affects the speed of the last subpath, with the regularization parameter $\bar{\lambda}$ of interest. In some sense, we have designed a continuation path that is super-fast until it reaches the second phase of the final subpath. Recall that our fixed point continuation algorithm is based on gradient descent and shrinkage operator. We envision that by switching to a Newton's method, we can accelerate the second phase.

Based on this intuition, we now are in a position to describe a hybrid algorithm : a fixed point continuation plus an interior point truncated Newton method. For the latter part we resort to the customized interior point in (Koh et al., 2007). We modified the source code of the *l1logreg* software (written in C), and built an interface to our MATLAB code. This hybrid approach, based on our observation of the two phases, enables us to attain a good balance of speed and accuracy.

3.2 Interior Point Phase

The second phase of our HIS algorithm used an interior point method developed by (Koh et al., 2007). We directly utilized a well-developed software package *l1logreg* [1] and modified the source code to build an interface to MATLAB. We review some key points for the interior point method here.

In (Koh et al., 2007), the authors overcome the difficulty of non-differentiability of the objective function by transforming the original problem into an equivalent one with linear inequality constraints,

$$\begin{aligned} \min \quad & \frac{1}{m}\sum_{i=1}^{m} l_{\text{avg}}(w^T a_i + v b_i) + \lambda \mathbf{1}^T u \\ \text{s.t.} \quad & -u_i \le w_i \le u_i, \;\; i = 1, ..., n. \end{aligned} \tag{40}$$

1. software can be downloaded at `http://www.stanford.edu/~boyd/papers/l1_logistic_reg.html`

A logarithmic barrier function, smooth and convex, is further constructed for the bound constraints,

$$\rho(w,u) = -\sum_{i=1}^{n} \log(u_i + w_i) - \sum_{i=1}^{n} \log(u_i - w_i), \tag{41}$$

defined on the domain $\{(w,u) \in \mathbb{R}^n \times \mathbb{R}^n \| w_i| < u_i, i = 1,...,n\}$. The following optimization problem can be obtained by augmenting the logarithmic barrier,

$$\psi_t(v,w,u) = tl_{\text{avg}}(v,w) + t\lambda \mathbf{1}^T u + \rho(w,u), \tag{42}$$

where $t > 0$. The resulting objective function is smooth, strictly convex and bounded below, and therefore has a unique minimizer $(v^*(t), w^*(t), u^*(t))$. This defines a curve in $\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n$, parameterized by t, called the central path. The optimal solution is also shown to be dual feasible. In addition, $(v^*(t), w^*(t))$ is $2n/t$-suboptimal.

As a primal interior-point method, the authors computed a sequence of points on the central path, for an increasing sequence of values of t, and minimized $\psi_t(v,w,u)$ for each t using a truncated Newton's method. The interior point method was customized by the authors in several ways: 1) the dual feasible point and the associated duality gap was computed in a cheap fashion, 2) the central path parameter t was updated to achieve a robust convergence when combined with the preconditioned conjugate gradient (PCG) algorithm, 3) an option for solving the Newton's system was given for problems of different scales, where small and medium dense problems were solved by direct methods (Cholesky factorization), while large problems were solved using iterative methods (conjugate gradients). Interested readers are referred to (Koh et al., 2007) for more details.

3.3 The Hybrid Algorithm

The hybrid algorithm takes advantage of the computational strengths of both the iterative shrinkage solver and the interior point solver. In the first phase, we use the iterative shrinkage solver, due to its computational efficiency and memory friendliness. It is especially beneficial to have a memory friendly solver for the initial phase when one is dealing with large-scale datasets. Recall that we use a continuation strategy for the iterative shrinkage phase, where a sequence of λ's is used along a regularization path. In the last subpath where λ is the desired one, we transit to the interior point when the true support of the vector is found. The corollary in Section 3.1 states that iterative shrinkage recovers the true support in a finite number of steps. In addition, iterative shrinkage obtains all true nonzero components long before the true support is obtained. Therefore, as long as the iterative shrinkage seems to stagnate, which can be observed when the objective function evolves very slowly, it is highly likely that all true nonzero components are obtained. This indicates that the algorithm is ready for switching to the interior point.

In practice, we require the following transition condition,

$$\frac{\|u^{k+1} - u^k\|}{\max(\|u^k\|, 1)} < utol_t, \tag{43}$$

and extract the nonzero components in w as the input to the interior point solver. In this sense, we reduce the problem to a subproblem where the dimension is much smaller, and use the subproblem using the interior point method.

The resulting hybrid algorithm achieves high computational speed while attaining the same numerical accuracy as the interior point method, as demonstrated with empirical results in the next section.

Algorithm 3 Hybrid Iterative Shrinkage (**HIS**) Algorithm

Require: $A = [c_1^\top; c_2^\top; \cdots; c_m^\top] \in \mathbb{R}^{m\times n+1}$, $u = (w; v) \in \mathbb{R}^{n+1}$, $f(u) = m^{-1}\phi(Au)$
  task: $\min_u l_{\text{avg}}(u) + \lambda\|u\|_1$
  Initialize u^0
  PHASE 1 : ITERATIVE SHRINKAGE
  Select λ_0 and $utol_0$
  while "not converge" **do**
    if "the last continuation path", $i == (L-1)$ and "transition condition" **then**
      "transit into PHASE 2"
    else
      Update $\lambda_i = \lambda_{i-1}\beta$, $utol_i = utol_{i-1}\gamma$
      Compute heuristic step length α_0
      Gradient descent step: $u^{k-} = u^k - \alpha_0 \nabla l_{\text{avg}}(u^k)$
      Shrinkage step: $u^{k+} = s_{1:n}(u^{k-}, \lambda\alpha_0)$
      Obtain line search direction: $p^k = u^{k+} - u^k$
      while "j < max line search attempts" **do**
        if Armijo-like condition is met **then**
          Accept line search step, update $u^{k+1} = u^k + \alpha_j p^k$
        else
          Keep backtracking $\alpha_j = \mu\alpha_{j-1}$
        end if
        $j = j + 1$
      end while
    end if
  end while
  PHASE 2 : INTERIOR POINT
  Initialize $\tilde{w} = w_{nonzero}$ get subproblem $\min \psi_t(v, \tilde{w}, u)$
  while "not converged" $\eta > \epsilon$ **do**
    Solve the Newton system : $\nabla^2\psi_t^k(v, \tilde{w}, u)[\Delta v, \Delta\tilde{w}, \Delta u] = -\nabla\psi_t^k(v, \tilde{w}, u)$
    Backtracking line search : find the smallest integer $j \geq 0$ that satisfies
      $\psi_t^k(v + \alpha_j\Delta v, \tilde{w} + \alpha_j\Delta\tilde{w}, u + \alpha_j\Delta u) \leq \psi_t^k(v, \tilde{w}, u) + c\alpha_j\nabla\psi_t^k(v, \tilde{w}, u)^T[\Delta v, \Delta\tilde{w}, \Delta u]$
    Update $\psi_t^{k+1}(v, \tilde{w}, u) = \psi_t^k(v, \tilde{w}, u) + \alpha_j(\Delta v, \Delta\tilde{w}, \Delta u)$
    Check dual feasibility
    Evaluate duality gap η
    $k = k + 1$
  end while

4. Numerical Results

4.1 Benchmark

We carried out a numerical comparison of our hybrid iterative shrinkage (HIS) algorithm with several existing algorithms in literature for ℓ_1-regularized logistic regression. Inspired by a comparison study on this topic by Schmidt et al (Schmidt et al., 2007) [2], we compared our algorithm with 10 algorithms, including a generalized version of Gauss-Seidel, Shooting, Grafting, Sub-Gradient, epsL1, Log-Barrier, Log-Norm, SmoothL1, EM, ProjectionL1 and Interior-Point method. In the numerical study, we replaced the interior point solver by the one written by Koh, Kim and Boyd (Koh et al., 2007). Benchmark data were taken from the publicly available UCI machine learning repository [3]. We used 10 data sets of small to median size (internetad1, arrhythmia, glass, horsecolic, indiandiabetes, internetad2, ionosphere, madelon, pageblock, spambase, spectheart, wine).

All of the methods were run until the same convergence criteria was met, where appropriate, for instance the step length, change in function value, negative directional derivative, optimality condition, convergence tolerance is less than 10^{-6}. We treated each algorithm solver as a black box and evaluate both the computation time and the sparsity (measured by cardinality of solution). We set an upper limit of 250 iterations, meaning we stop the solver when the number of iteration exceeds 250. Since different algorithm has different speed for each iterate (usually a Newton step is more expensive than a gradient descent step), we think the computation time is a more appropriate evaluation criterion than number of iterations. The ability of the algorithm to find sparse solution, measured by the cardinality, was also evaluated in this process.

Fig. 4 shows the benchmark result using data from the UCI machine learning repository. All numerical results shown are averaged over a regularization path. The parameters for the regularization path are calculated according to each data set, where the maximal regularization parameter is calculated as follows:

$$\lambda_{max} = \frac{1}{m}\Big\|\frac{m_-}{m}\sum_{b_i=+1} a_i + \frac{m_+}{m}\sum_{b_i=-1} a_i\Big\|_\infty, \tag{44}$$

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). λ_{max} is an upper bound for the useful range of regularization parameter. When $\lambda \geq \lambda_{max}$, the cardinality of the solution will be zero. In this case, we test a regularization path of length 10, i.e. λ_{max}, $0.9\lambda_{max}$, $0.8\lambda_{max}$ ... $0.1\lambda_{max}$. Among all the numerical solvers, our HIS algorithm is the most efficient. We also achieve comparable cardinality in the solution, compared to that of the interior point solver.

We also evaluate the accuracy of the solution by looking at the classification performance using Kfold cross-validation. Table 1 summaries the accuracy of the solution using the HIS algorithm, and compare that to the interior point (IP) algorithm. Clearly, HIS algorithm achieves comparable accuracy compared to IP, an algorithm that is recognized for its high accuracy.

2. source code available at `http://www.cs.wisc.edu/~gfung/GeneralL1`
3. UCI machine learning repository `http://www.ics.uci.edu/~mlearn/MLRepository.html`

(a) Time (b) Cardinality

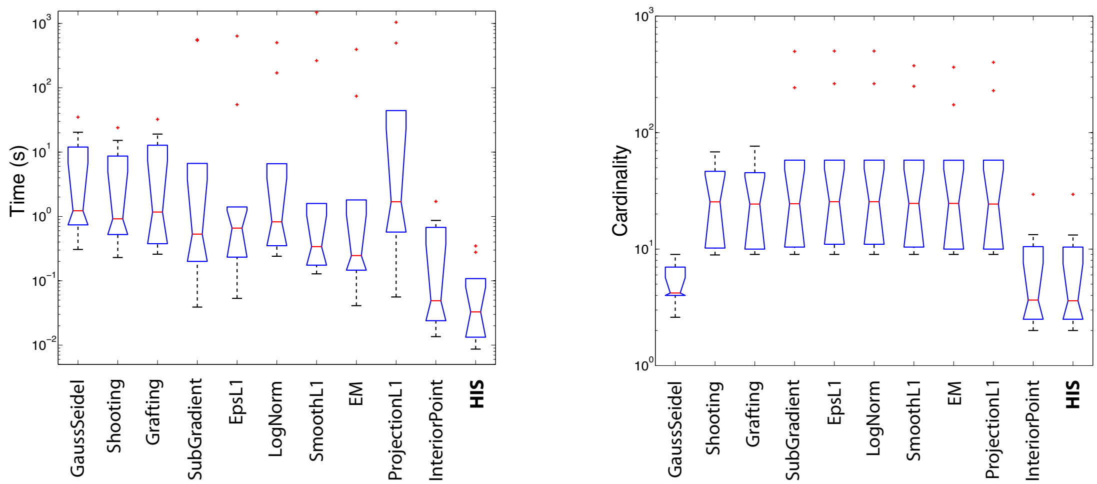

Figure 4: Comparison of our hybrid iterative shrinkage (HIS) method with several other existing methods in literature. Benchmark data were taken from the UCI machine learning repository, including 10 publicly available datasets. (a) Distribution of computation time across 10 data sets, (b) Distribution of cardinality for the solution across 10 data sets, averaged over a regularization path.

Accuracy Comparison
($Az \in [0.5, 1.0]$)

dataname	accuracy(HIS)	accuarcy(IP)
arrhythmia	0.7363	0.7363
glass	0.6102	0.6102
horsecolic	0.5252	0.5252
ionosphere	0.5756	0.5756
madelon	0.6254	0.6254
spectheart	0.5350	0.5350
wine	0.6102	0.6102
internetad	0.8486	0.8486

Table 1: Comparison of solution accuracy for our hybrid iterative shrinkage (HIS) algorithm and the interior point (IP) algorithm. Accuracy of the solution was measured by Az value, resulted from Kfold cross-validation, where Kfold is 10. The datasets were taken from the UCI machine learning repository.

4.2 Scaling Result

Numerical experiments were carried out to study how our algorithm scales with the problem size. For the sake of generality, we used simulated data whose dimension ranges from 64 to 131072. The data is drawn from a Normal distribution, where the mean of the distribution is shifted by a small number for each class (0.1 for samples with label 1, and -0.1 for samples with label -1). The number of samples is the same for both classes and chosen to be smaller than the dimension of the data. Experiments for each dimension were carried out on 100 different sets of random data. We compared the mean and variances of the computation time for each dimension of dataset, and compared our HIS algorithm to the interior point method.

Speed Comparison
(in *second*)

dimension	mean(HIS)	std(HIS)	mean(IP)	std(IP)
64	0.0026	0.00069	0.0043	0.00057
128	0.0025	0.00058	0.0049	0.00037
256	0.0026	0.00075	0.0078	0.00052
512	0.0024	0.00059	0.018	0.0017
1024	0.0023	0.00056	0.029	0.0023
2048	0.0026	0.00064	0.054	0.0026
4096	0.0028	0.00057	0.098	0.0050
8192	0.0030	0.00059	0.19	0.0076
16384	0.0033	0.00055	0.40	0.018
32768	0.0038	0.00055	0.89	0.037
65536	0.0049	0.00054	2.01	0.096
131072	0.0077	0.00056	4.49	0.24

Table 2: Speed results for some random benchmark data, a comparison of our hybrid approach with the interior-point method (Koh et al., 2007). Shown here is the computation speed as a function of dimension. Data used here are generated by sampling from two Gaussian distributions. Note that in the simulation, the continuation path used in the iterative shrinkage may or maybe not be optimal, which means that the speed profile for the HIS algorithm can be essentially accelerated even more.

Table 2 summarizes the computational speed for the HIS algorithm and the interior point algorithm. It is noteworthy that the HIS algorithm improves the efficiency of computation, while maintaining comparable accuracy to the interior point method. Fig. 5 plots the computation result as a function of dimension for better illustration. In (a) one can see that the speedup we gain from the HIS algorithm (red), compared to the interior point method (blue), is obvious. We also show the solution quality in (b), where the weights we get from both solvers are comparable.

(a) Speed

(b) Solution

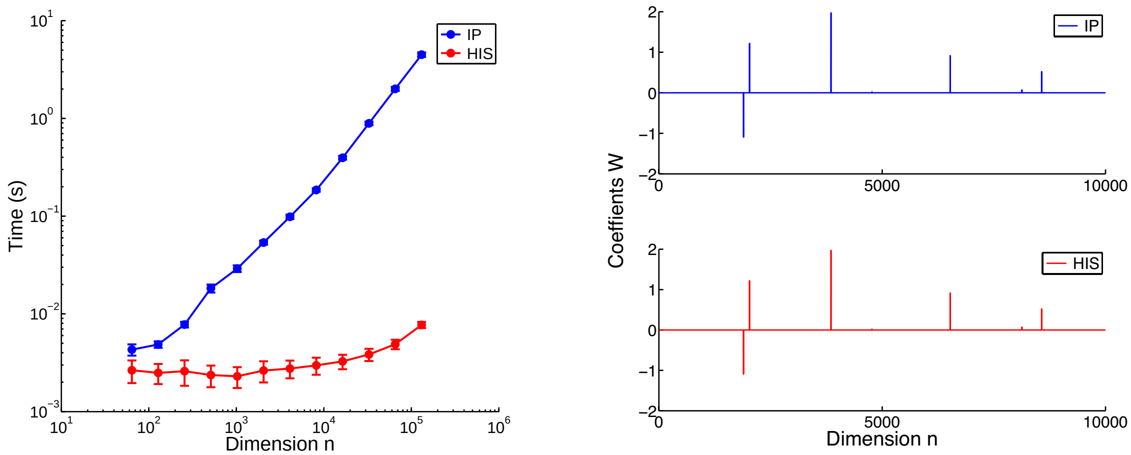

Figure 5: Comparison for the random benchmark data, between our hybrid approach and the interior-point method (Koh et al., 2007). (a) Speed profile for these two approaches. Blue curve shows the speed profile for the interior point method, and curve curve shows the speed profile for our hybrid approach as a function of the data dimension. (b) An example of the solutions using interior point (blue) and HIS algorithm (red).

4.3 Regularization parameter

In general, the regularization parameter λ affects the number of iterations to converge for any solver. It is observed that the smaller the λ, the more time it takes to converge to a solution. As λ becomes smaller, the cardinality of the solution goes up, and the computation time needed for convergence also goes up. Therefore when one seeks a solution with less sparsity (small λ), it is more computationally expensive.

The regularization parameter also affects the cardinality of the solution. In practice, when one carries out classification on a set of data, the optimal regularization parameter is often unknown. Speaking of optimality, we refer to a regularization parameter that results in the best classification result evaluated using Kfold cross-validation. One would also run the algorithm along a regularization path, $\lambda_{max}, ..., \lambda_{min}$, where λ_{max} is computed by (44).

Fig. 6 shows the evolution of solution along the regularization path, using a small dataset (ionosphere) from the UCI machine learning repository. This explores sparsity of different degrees in the solution, and one can determine the optimal sparsity for the data. This is an attractive property of this model, where one can apply to look in the feature space the most informative features about discrimination.

We illustrate the effect of the regularization parameter using real data of large scale. The data concerns a two alternative force choice task for face versus car discrimination. We use a spiking neuron model of primary visual cortex to map the input into cortical space, and decode the resulting spike trains using sparse logistic regression (Shi et al., 2009). The data

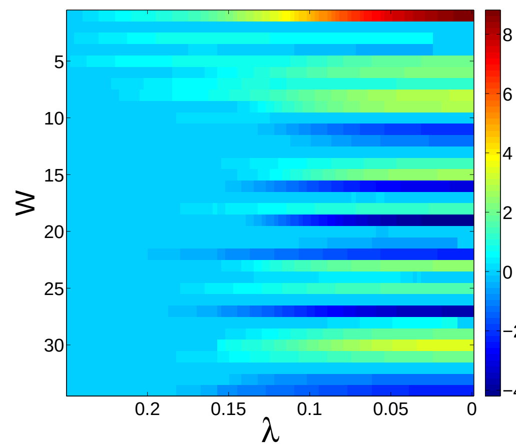

Figure 6: Solution w evolves along a regularization path, following a geometric progression from 10^{-1} to 10^{-4}. Data is ionosphere from UCI machine learning repository. As the λ becomes smaller, the cardinality of the solution goes up.

has dimensions of 40960 and the number of samples is 360 for each of the two classes. Kfold cross-validation is utilized to evaluate the classification performance, where the number of Kfolds is ten in our simulation. We compare the performance of HIS algorithm with IP algorithm.

Fig. 7 shows the results. The speedup of the HIS algorithm compared to the IP algorithm is shown in Fig. 7(a), where blue indicates the computation time of the IP algorithm, while the red shows the HIS algorithm. The HIS algorithm results in approximately a factor of 2 speedup over IP, without loss in accuracy.

Note that there is an issue of model selection when we apply sparse logistic regression model to the data, in a sense there exists an optimal level of sparsity that achieves the best classification result. We run the model with a sequence of regularization parameters, which results in classification result (evaluated by Az value from Kfold cross-validation). Fig. 7(b) illustrates the classification result as a function of the cardinality of the solution. One can see the bell shape in the curve, which provides a route to select the optimal sparsity for the solution.

5. Conclusion

We have presented in this paper a computationally efficient algorithm for the ℓ_1-regularized logistic regression, also called the sparse logistic regression. The sparse logistic regression is a widely used model for binary classification in supervised learning. The ℓ_1 regularizer results in sparsity in the solution, making it a robust classifier for data whose dimensions are larger than the number of samples. Sparsity also provides an attractive avenue for feature selection, useful for various data mining tasks.

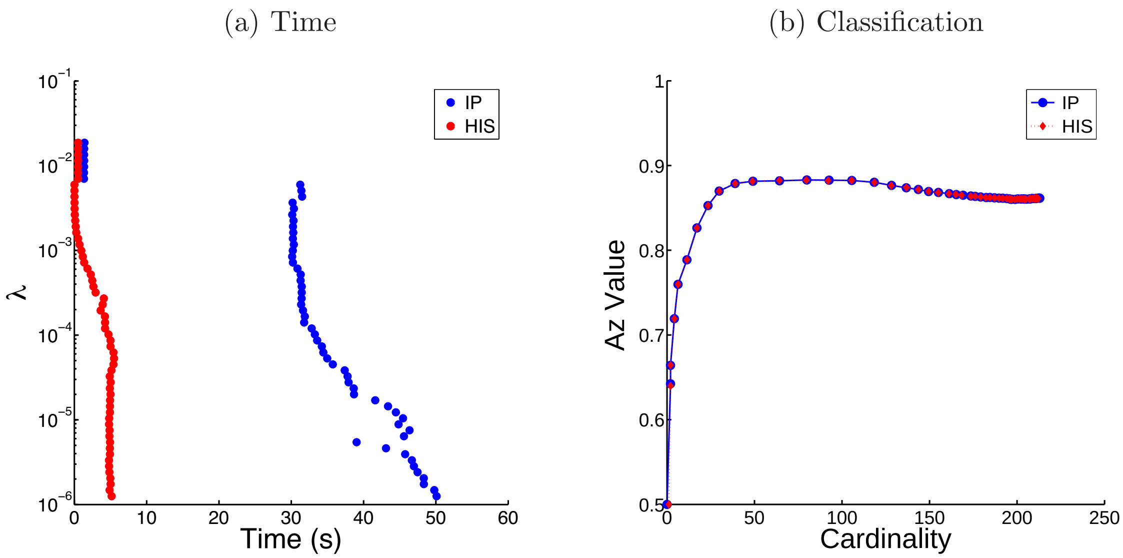

Figure 7: An example using real data of large scale, $n = 40960$, $m = 360$. (a) Computation time along such a regularization path, where the smaller λ requires more computation time. (b) Classification performance derived from ROC analysis based on Kfold cross-validation. Data used in this simulation are neural data for a visual discrimination task (Shi et al., 2009).

Solving large-scale sparse logistic regression usually requires expensive computational resources, depending on the specific solver, memory and/or CPU time. The interior point method is so far the most efficient solver in the literature, but requires expensive memory consumption. We present a hybrid iterative shrinkage (HIS) algorithm, which couples a fast shrinkage method and a slower but more accurate interior point method. The iterative shrinkage algorithm has global convergence with a Q-linear rate. Various techniques such as line search and a continuation strategy are utilized to accelerate the computation. The shrinkage solver only involves the gradient descent and the shrinkage operator, both of which are first-order. Based solely on efficient memory operations such as matrix-vector multiplication, the shrinkage solver serves as the first phase for the algorithm. This reduces the problem to a subspace whose dimension is smaller than the original problem. The HIS algorithm then transits into the second phase, using a more accurate interior point solver. We numerically compare the HIS algorithm with other popular algorithms in the literature, using benchmark data from the UCI machine learning repository. We show that the HIS algorithm is the most computationally efficient, while maintaining high accuracy. The HIS algorithm also scales very well with dimension of the problem, making it attractive for solving large-scale problems.

There are several ways to extend the HIS algorithm. One is to extend it to beyond binary classification, allowing for multiple classes (Krishnapuram and Hartemink, 2005). The other is to further improve regularization. When applying the HIS algorithm, one will usually explore a range of sparsity by constructing a regularization path (λ_{max}, λ_1,

..., λ_{min}). Usually the smaller the λ the more expensive it is to employ the shrinkage algorithm. One can accelerate the computation using the Bregman regularization, inspired by (Yin et al., 2008). The Bregman iterative algorithm essentially boosts the solution by solving a sequence of optimizations, resulting in a different regularization path. Bregman has also been shown to improve solution quality in the presence of noise (Osher et al., 2008). We will discuss such a regularization path in a future paper.

Acknowledgements

We thank Mads Dyrholm (Columbia University) for fruitful discussions. Jianing Shi and Paul Sajda's work was supported by grants from NGA (HM1582-07-1-2002) and NIH (EY015520). Wotao Yin's work was supported by NSF CAREER Award DMS-0748839 and ONR Grant N00014-08-1-1101. Stanley Osher's work was supported by NSF grants DMS-0312222, and ACI-0321917 and NIH G54 RR021813.

References

C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.

S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

E.J. Candés and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inform. Theory*, 52(2):5406–5425, 2006.

E.J. Candés, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1998.

J.F. Claerbout and F. Muir. Robust modeling with erratic data. *Geophysics*, 38(5):826–844, 1973.

A. d'Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. In *Advances in Neural Information Processing Systems*, pages 41–48. MIT Press, 2005.

D.L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, pages 1289–1306, 2006a.

D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306, 2006b.

D.L. Donoho and M. Elad. Optimally sparse representations in general nonorthogonal dictionaries by ℓ_1 minimization. *Proc. Nat'l Academy of Science*, pages 2197–2202, 2003.

D.L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE. Trans. Inform. Theory*, 48(9):2845–2862, 2001.

D.L. Donoho and B.F. Logan. Signal recovery and the large sieve. *SIAM J. Appl. Math.*, 52(2):577–591, 1992.

D.L. Donoho and P.B. Stark. Uncertainty principle and signal recovery. *SIAM J. Appl. Math.*, 49(3):906–931, 1989.

D.L. Donoho, I. Johnstone, G. Kerkyacharian, and D. Picard. Wavelet shrinkage: Asymptopia? *J. Roy. Stat. Soc. B*, 57(2):301–337, 1995.

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

S. Eyheramendy, A. Genkin, W. Ju, D. Lewis, and D. Madigan. Sparse bayesian classifiers for text categorization. Technical report, Department of Statistics, Rutgers University, 2003.

M. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 1150–1159, 2003.

M. Figueiredo and A. Jain. Bayesian learning of sparse classifiers. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 35–41, 2001.

M. Figueiredo, R. Nowak, and S. Wright. Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, 1(4):586–598, 2007.

A. Genkin, D.D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. Technical report, Rutgers University, 2004. `http://www.stat.rutgers.edu/~madigan/BBR/`.

A.D. Gerson, L.C. Parra, and P. Sajda. Cortical origins of response time variability during rapid discrimination of visual objects. *Neuroimage*, 28(2):342–353, 2005.

A. Ghosh and S. Boyd. Growing well-connected graphs. In *45th IEEE Conference on Decision and Control*, pages 6605–6611, 2006.

J. Goodman. Exponential priors for maximum entropy models. In *Proceedings of the Annual Meetings of the Association for Computational Linguistics*, 2004.

E. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for l1-minimization: methodology and convergence. *Submitted to SIAM Optimization*, 0:0, 2008.

A. Hassibi, J. How, and S. Boyd. Low-authority controller design via convex optimization. *AIAA Journal of Guidance, Control and Dynamics*, pages 862–872, 1999.

K. Koh, S.-J. Kim, and S. Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 2007.

B. Krishnapuram and A. Hartemink. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27 (6):957–968, 2005.

B. Krishnapuram, L. Carin, and M. Figueiredo. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 957–968, 2005.

S. Lee, H. Lee, P. Abbeel, and A. Ng. Efficient ℓ_1-regularized logistic regression. In *National Conference on Artificial Intelligence (AAAI)*, 2006.

J.G. Liao and K.V. Chin. Logistic regression for disease classification using microarray data: model selection in a large p and small n cas. *Bioinformatics*, 23(15):1945–51, 2007.

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

M. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):376–394, 2007.

J. Lokhorst. The lasso and generalised linear models. Technical report, Honors Project, Department of Statistics, University of Adelaide, South Australia, Australia, 1999.

D. Madigan, A. Genkin, D. Lewis, and D. Fradkin. Bayesian multinomial logistic regression for author identification. In *AIP*, 2005.

B.K. Natarajan. Sparse approximate solutions to linear system. *SIAM J. Comput.*, 24(2): 227–234, 1995.

A. Ng. Feature selection, l1 vs l2 regularization, and rotational invariance. In *International Conference on Machine Learning (ICML)*, pages 78–85. ACM Press, New York, 2004.

A. Ng. On feature selection: Learning with exponentially many irrelevant features as training examples. In *International Conference on Machine Learning (ICML)*, pages 404–412, 1998.

S. Osher, Y. Mao, B. Dong, and W. Ying. Fast linearized bregman iteration for compressive sensing and sparse denoising. Technical report, UCLA Computational and Applied Mathematics, 2008.

M.Y. Park and T. Hastie. L1 regularized path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69:659–677, 2007.

L.C. Parra, C. Spence, and P. Sajda. Higher-order statistical properties arising from the non-stationarity of natural signals. In *Advances in Neural Information Processing Systems*, volume 13, pages 786–792, 2001.

L.C. Parra, C.D. Spence, A.D. Gerson, and P. Sajda. Recipes for the linear analysis of eeg. *Neuroimage*, pages 326–341, 2005.

S. Perkins and J. Theiler. Online feature selection using grafting. In *Proceedings of the Twenty-First International Conference on Machine Learning (ICML)*, pages 592–599. ACM Press, 2003.

M.G. Philiastides and P. Sajda. Temporal characterization of the neural correlates of perceptual decision making in the human brain. *Cereb Cortex*, pages 509–518, 2006.

B. Polyak. *Introduction to Optimization.* Optimization Software, 1987.

V. Roth. The generalized lasso. *IEEE Tran. Neural Networks*, pages 16–28, 2004.

L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: a comparative study and two new approaches. In *ECML*, pages 286–297, 2007.

J. Shi, J. Wielaard, R.T. Smith, and P. Sajda. Perceptual decision making investigated via sparse decoding of a spiking neuron model of v1. IEEE EMBS Conference on Neural Engineering, 2009.

N.Z. Shor. *Minimization Methods for Non-differentiable functions.* Springer Series in Computational Mathematics. Springer, 1985.

H.L. Taylor, S.C. Banks, and J.F. McCoy. Deconvolution with the ℓ_1 norm. *Geophysics*, 44(1):39–52, 1979.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc. B*, 58(1): 267–288, 1996.

Y. Tsuruoka, J. McNaught, J. Tsujii, and S. Ananiadou. Learning string similarity measures for gene/protein name dictionary look-up using logistic regression. *Bioinformatics*, 23(20): 2768–74, 2007.

L. Vandenberghe, S. Boyd, and A. El Gamal. Optimal wire and transistor sizing for circuits with non-tree topology. In *IEEE/ACM International Conference on Computer Aided Design*, pages 252–259, 1997.

L. Vandenberghe, S. Boyd, and A. El Gamal. Optimizing dominant time constant in rc circuits. *IEEE Trans. Computer-Aided Design*, 1998.

V. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, 1982.

V. Vapnik. *Statistical Learning Theory.* John Wiley & Sons, 1988.

Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang. On the convergence of an active set method for ℓ_1 minimization. Technical report, Dept of IEOR, Columbia University, 2008.

Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation. Technical report, Rice University CAAM TR09-01, 2009.

W. Yin, S. Osher, J. Darbon, and D. Goldfarb. Bregman iterative algorithm for ℓ_1-minimization with applications to compressed sensing. *SIAM J. Imaging Science*, 1(1): 143–168, 2008.

P. Zhao and B. Yu. On model selection consistency of lasso. *J. Machine Learning Research*, 7:2541–2567, 2007.

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems*, volume 16, pages 49–56. MIT Press, 2004.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principle component analysis. *Journal of Computational and Graphical Statistics*, 15(2):262–286, 2006.