Stochastic DCA for the Large-sum of Non-convex Functions Problem and its Application to Group Variable Selection in Classification

Hoai An Le Thi [1] Hoai Minh Le [1] Duy Nhat Phan [1] Bach Tran [1]

Abstract

In this paper, we present a stochastic version of DCA (Difference of Convex functions Algorithm) to solve a class of optimization problems whose objective function is a large sum of non-convex functions and a regularization term. We consider the $\ell_{2,0}$ regularization to deal with the group variables selection. By exploiting the special structure of the problem, we propose an efficient DC decomposition for which the corresponding stochastic DCA scheme is very inexpensive: it only requires the projection of points onto balls that is explicitly computed. As an application, we applied our algorithm for the group variables selection in multiclass logistic regression. Numerical experiments on several benchmark datasets and synthetic datasets illustrate the efficiency of our algorithm and its superiority over well-known methods, with respect to classification accuracy, sparsity of solution as well as running time.

1. Introduction

We consider the following optimization problem

$$\min \left\{ f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \lambda p(x) \right\}, \qquad (1)$$

whose objective function f is a large sum of non-convex functions $f_i(x)$ and a regularization term $p(x)$, where $f_i(x)$ corresponds to a criteria to optimize and $\lambda \geq 0$ is a trade-off parameter between the two terms. This model covers a very vast class of problems arising from several fields such as machine learning, signal processing, etc. For instance, least-squares regression, logistic regression problem, etc can be expressed in the form of (1).

Nowadays, the growth of technologies leads to exponential augmentation of large-scale data where the number of both variables and samples are huge. Thus, optimization methods for solving the problem (1) are faced with a great challenge that is the number of samples n can be extremely large. Among existing methods for this problem, stochastic programming has been proved to be suitable thanks to its ability to exploit the advantage of the sum structure of the problem. In (Schmidt et al., 2015), the authors considered a special case of the large-sum problem (1) where f_i are convex and smooth functions and p corresponds to the ℓ_2 regularization. Stochastic Average Gradient was developed to solve the resulting problem. Reddi et al. (Reddi et al., 2016) developed Proximal Stochastic Gradient method for the case where f_i are smooth (can be non-convex) and p is convex, non-smooth function. Motivated by its success, we will study stochastic programming for solving (1) in order to deal with data having an extremely large number of samples.

On the other hand, in real-world applications such as image processing, microarray analysis, etc. datasets contain a very large number of variables. In such of cases, we are often to face with the problem of redundant and irrelevant variables. Redundant variables contain information already presented by other variables while irrelevant variables do not contain useful information. Variables selection methods that consist of selecting important variables for a considered task, are a popular and efficient way to deal with redundant and irrelevant variables. In this direction, a natural idea is to formulate the variables selection problem as a minimization of the ℓ_0-norm (or $\|.\|_0$). The sparse optimization has been extensively studied on both theoretical and practical aspects. The readers can refer to Le Thi et al. (Le Thi et al., 2015) for an extensive overview of existing approaches for the minimization of ℓ_0-norm.

Nevertheless, when the data possesses certain group structures, we are naturally interested in selecting important groups of variables rather than individual ones. For instance, in multi-factor analysis of variance, a factor with several levels may be expressed through a group of dummy variables. In genomic data analysis, the correlations between genes sharing the biological pathway can

[1] Laboratory of Theoretical and Applied Computer Science, University of Lorraine, France. Correspondence to: Hoai Minh Le <minh.le@univ-lorraine.fr>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

be high. Hence these genes should be considered as a group. Recently, the mixed-norm regularization has been developed for the group variable selection. It consists in using the $\ell_{2,0}$ regularization term. Assume that $x = (x_1, ..., x_m) \in \mathbb{R}^m$ is partitioned into J non-overlapping groups $x_{(1)}, ..., x_{(J)}$, then the $\ell_{2,0}$-norm of x is defined by $\|x\|_{2,0} = |\{j \in \{1, ..., J\} : \|x_{(j)}\|_2 \neq 0\}|$. Clearly, $\ell_{2,0}$-norm is non-convex that makes the optimization problem involving $\ell_{2,0}$ challenging. Several works have been developed to solve the problem of mixed-norm regularization $\ell_{2,0}$. The first approach, named the group Lasso ($\ell_{2,1}$-norm) (Yuan & Lin, 2006), is closely connected to the Lasso (ℓ_1-norm) - an approximation of the ℓ_0-norm (Tibshirani, 1994). This approach was widely used for selecting groups of variables in multi-task learning (Obozinski et al., 2006), multiclass support vector machine (Blondel et al., 2013), principal component analysis (Khan et al., 2015), linear discriminant analysis (Gu et al., 2011), and compressed sensing (Sun et al., 2009), etc. The second approach consists in replacing the $\ell_{2,0}$-norm by a DC (Difference of Convex functions) approximation. In (Wang et al., 2007), the authors used the smoothly clipped absolute deviation (SCAD) approximation and developed a group coordinate descent based algorithm for the sparse linear regression. Later, Huang et al. (Huang et al., 2012) used the minimax concave penalty (MCP) for the same problem. In (Lee et al., 2016), the authors considered both above approximations and developed DC programming and DCA (DC algorithm) based method for the resulting problems. Recently, Phan et al. (Phan et al., 2017) proposed DCA based algorithms for bi-level variable selection using the combination of the ℓ_0-norm and $\ell_{q,0}$-norm.

Paper's contribution: In this paper, we aim at developing efficient methods to solve the problem (1) where n is extremely large and $p(x)$ corresponds to $\ell_{2,0}$ regularization (in order to deal with the group variables selection). The large-sum optimization (1) becomes

$$\min\left\{f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \lambda\|x\|_{2,0}\right\}. \qquad (2)$$

We assume that $f_i(x)$ is differentiable with L-Lipschitz gradient. This assumption is broad enough to cover several applications. Various important problems in machine learning such as Multi-task feature selection, Sparse logistic regression, Minimizing an expected loss in stochastic programming, etc. can be expressed in the form of (2). As we have mentioned above, the $\ell_{2,0}$-norm can be approximated by a convex (e.g. $\ell_{2,1}$-norm) or non-convex function. Using a non-convex approximation will lead to a "harder" optimization problem but it has been proved that non-convex approximations perform better than convex approximations in terms of sparsity (Le Thi et al., 2015). The resulting problem is then reformulated as a DC program and DCA based algorithm will be developed to solve it. We exploit the special structure of the problem to propose an efficient DC decomposition for which the corresponding DCA scheme is very inexpensive: it only requires the projection of points onto balls that is explicitly computed. On the other hand, in order to deal with data having a large number of samples, we present stochastic version DCA. The convergence properties of the proposed algorithm is rigorously studied to show that the convergence is guaranteed with probability one.

As an application of our algorithm, we consider the group variables selection in multiclass logistic regression. We perform an empirical comparison of stochastic DCA with DCA and standard methods on very large synthetic and real-world datasets, and show that the stochastic DCA is efficient in group variable selection ability and classification accuracy as well as running time.

The remainder of the paper is organized as follows. Solution method based on Stochastic DCA for solving (2) is developed in Section 2. In Section 3, we apply the proposed algorithm to the group variables selection in multiclass logistic regression. Finally Section 4 concludes the paper.

2. Solution method via stochastic DCA

2.1. Outline of DC programming and DCA

DC programming and DCA constitute the backbone of smooth/non-smooth non-convex programming and global optimization (Pham Dinh & Le Thi, 1997; 1998; Le Thi & Pham Dinh, 2005). They address the problem of minimizing a DC function on the whole space $\mathbb{R}^n$ or on a closed convex set $\Omega \subset \mathbb{R}^n$. Generally speaking, a standard DC program takes the form:

$$\alpha = \inf\{F(x) := G(x) - H(x) \,|\, x \in \mathbb{R}^n\} \quad (P_{dc}),$$

where G, H are lower semi-continuous proper convex functions on $\mathbb{R}^n$. Such a function F is called a DC function, and $G - H$ is a DC decomposition of F while G and H are the DC components of F. A DC program with convex constraint $x \in \Omega$ can be equivalently expressed as (P_{dc}) by adding the indicator function χ_Ω ($\chi_\Omega(x) = 0$ if $x \in \Omega$ and $+\infty$ otherwise) to the first DC component G.

The modulus of strong convexity of θ on Ω, denoted by $\mu(\theta, \Omega)$ or $\mu(\theta)$ if $\Omega = \mathbb{R}^n$, is given by

$$\mu(\theta, \Omega) = \sup\{\mu \geq 0 : \theta - (\mu/2)\|.\|^2 \text{ is convex on } \Omega\}$$

One says that θ is *strongly convex* on Ω if $\mu(\theta, \Omega) > 0$.

For a convex function θ, the subdifferential of θ at $x_0 \in$ dom$\theta := \{x \in \mathbb{R}^n : \theta(x_0) < +\infty\}$, denoted by $\partial\theta(x_0)$, is defined by

$$\partial\theta(x_0) := \quad \{y \in \mathbb{R}^n : \theta(x) \geq \theta(x_0) + \langle x - x_0, y\rangle, \\ \forall x \in \mathbb{R}^n\}.$$

The subdifferential $\partial\theta(x_0)$ generalizes the derivative in the sense that θ is differentiable at x_0 if and only if $\partial\theta(x_0) \equiv \{\nabla_x \theta(x_0)\}$.

A point x^* is called a *critical point* of $G - H$, or a generalized Karush-Kuhn-Tucker point (KKT) of (P$_{dc}$)) if $\partial H(x^*) \cap \partial G(x^*) \neq \emptyset$.

The main idea of DCA is simple: each iteration l of DCA approximates the concave part $-H$ by its affine majorization (that corresponds to taking $v^l \in \partial H(x^l)$) and then computes x^{l+1} by solving the resulting convex problem.

$$\min_{x\in\mathbb{R}^n} \{G(x) - \langle v^l, x\rangle\}.$$

The sequence $\{x^l\}$ generated by DCA enjoys the following properties (Pham Dinh & Le Thi, 1997; 1998; Le Thi & Pham Dinh, 2005):

(i) The sequence $\{F(x^l)\}$ is decreasing;

(ii) If $F(x^{l+1}) = F(x^l)$, then x^l is a critical point of (P_{dc}) and DCA terminates at l-th iteration.

(iii) If $\mu(G) + \mu(H) > 0$ then the series $\{\|x^{k+1} - x^k\|^2$ converges.

(iv) If the optimal value α of (P_{dc}) is finite and the infinite sequence $\{x^l\}$ is bounded then every limit point of the sequence $\{x^l\}$ is a critical point of $G - H$.

2.2. Stochastic DCA for solving the problem (2)

In this section, we introduce a stochastic version of DCA for solving (2) that exploits the structure of objective function f. We consider a family of DC approximations $\tilde{p}(x)$ of $\ell_{2,0}$-norm, defined by

$$\tilde{p}(x) = \sum_{j=1}^{J} \eta(\|x_{(j)}\|_2),$$

where η is a non-convex penalty function which includes SCAD, MCP, Capped-ℓ_1, exponential function, ℓ_{p+} with $0 < p < 1$, ℓ_{p-} with $p < 0$ (see (Le Thi et al., 2015) for more details). $\tilde{p}(x)$ can be expressed as $\tilde{p}(x) = \tilde{g}(x) - \tilde{h}(x)$, where

$$\tilde{g}(x) = \alpha \sum_{j=1}^{J} \|x_{(j)}\|_2 \text{ and } \tilde{h}(x) = \alpha \sum_{j=1}^{J} \|x_{(j)}\|_2 - \tilde{p}(x).$$

Hence, the approximate problem of (2) can be written as

$$\min_{x\in\mathbb{R}^m} \left\{ f(x) = \frac{1}{n} \sum_{i=1}^{n} \left[f_i(x) + \lambda\tilde{g}(x) - \lambda\tilde{h}(x) \right] \right\}. \quad (3)$$

Each function $f_i(x)$ can be rewritten as

$$f_i(x) = \frac{\rho}{2}\|x\|^2 - \left[\frac{\rho}{2}\|x\|^2 - f_i(x)\right].$$

Since $f_i(x)$ is differentiable with L-Lipschitz gradient, $\left[\frac{\rho}{2}\|x\|^2 - f_i(x)\right]$ is strongly convex with $\rho > L$. Hence, $f_i(x)$ is a DC function. Consequently, $f(x)$ is a DC function with the following DC decomposition

$$f(x) = g(x) - h(x), \quad (4)$$

where $g(x)$ and $h(x)$ are convex functions defined by

$$g(x) = \tfrac{\rho}{2}\|x\|^2 + \lambda\tilde{g}(x), \\ h(x) = \tfrac{1}{n}\sum_{i=1}^{n} h_i(x); h_i(x) = \tfrac{\rho}{2}\|x\|^2 - f_i(x) + \lambda\tilde{h}(x).$$

DCA for solving (3) amounts to computing two sequences $\{x^l\}$ and $\{v^l\}$ such that $v^l \in \partial h(x^l)$ and x^{l+1} is an optimal solution of the following convex problem

$$\min \left\{ g(x) - \langle v^l, x\rangle \right\}. \quad (5)$$

The computation of subgradients of h requires the one of all components h_i. This can be expensive when n is very large. Hence we propose a stochastic version of DCA in which we only compute the subgradients of a small subset of components h_i. Precisely, at each iteration l, we compute $v_i^l \in \partial h_i(x^l)$ for $i \in s_l$ and keep $v_i^l = v_i^{l-1}$ for $i \notin s_l$, where $s_l \subset \{1, ..., n\}$ is a randomly chosen set of index.

The computation of $v_i^l \in \partial h_i(x^l)$ can be given as $v_i^l = \rho x^l - \nabla f_i(x^l) + y^l$, where $y^l \in \lambda\partial\tilde{h}(x^l)$ for all $i \in s_l$. The convex problem (5) take the form

$$\min \left\{ \lambda\alpha \sum_{j=1}^{J} \|x_{(j)}\|_2 + \frac{\rho}{2}\|x\|^2 - \langle \frac{1}{n}\sum_{i=1}^{n} v_i^l, x\rangle \right\}. \quad (6)$$

We observe that the objective of (6) is separable in groups of x, then the solution to this problem can be computed by solving J independent sub-problems of the same form:

$$\min \left\{ \lambda\alpha\|x_{(j)}\|_2 + \frac{\rho}{2}\|x_{(j)}\|^2 - \langle v_{(j)}^l, x_{(j)}\rangle \right\}, \quad (7)$$

where $v_{(j)}^l = \frac{1}{\rho n}\sum_{i=1}^{n} (v_i^l)_{(j)}$ for $j = 1, ..., J$. The solution of (7) can be explicitly computed by

$$x_{(j)}^{l+1} = \left(\|v_{(j)}^l\|_2 - \lambda\alpha/\rho\right)_+ \frac{v_{(j)}^l}{\|v_{(j)}^l\|_2}, \quad (8)$$

Thus, the stochastic DCA (SDCA) for solving the problem (3) is described in Algorithm 1.

Now we will prove that the convergence properties of SDCA are guaranteed with probability one.

Algorithm 1 SDCA for solving the problem (3)

```
Initialization: Choose $x^0 \in \mathbb{R}^m$, $\rho > L$ and $s_0 = \{1,...,n\}$, $l \leftarrow 0$.
Repeat
    1. Compute $v_i^l \in \partial h_i(x^l)$ for $i \in s_l$ and keep $v_i^l = v_i^{l-1}$ for $i \notin s_l$.
    2. Compute $x^{l+1}$ by using (8).
    3. Set $l \leftarrow l+1$ and randomly choose a small subset $s_l \subset \{1,...,n\}$.
Until Stopping criterion.
```

Theorem 1. *If $\alpha^* = \inf f(x) > -\infty$ and $|s_l| = b$ for all $l \geq 1$, then SDCA generates the sequence $\{x^l\}$ such that*

a) $\{f(x^l)\}$ is the almost sure convergent sequence.

b) $\sum_{l=1}^{\infty} \|x^l - x^{l-1}\|^2$ is almost surely finite and $\lim_{l\to\infty} \|x^l - x^{l-1}\| = 0$ almost surely.

c) Every limit point of $\{x^l\}$ is a critical point of f with probability one.

Proof. a) Let $x_i^l = x^l$ and $y_i^l = y^l$ for $i \in s_l$, $x_i^l = x_i^{l-1}$ and $y_i^l = y^{l-1}$ for $i \notin s_l$. We denote T_i^l the function given by

$$T_i^l(x) = \lambda\tilde{g}(x) + \frac{\rho}{2}\|x - x_i^l\|^2 - \langle x - x_i^l, y_i^l - \nabla f_i(x_i^l)\rangle + f_i(x_i^l) - \tilde{h}(x_i^l),$$

and $T^l(x) = \frac{1}{n}\sum_{i=1}^n T_i^l(x)$. From the step 2 in Algorithm 1, it follows that $x^{l+1} = \arg\min T^l(x)$. Hence, we have

$$\begin{aligned} T^l(x^{l+1}) \leq T^l(x^l) = & T^{l-1}(x^l) + \frac{1}{n}\sum_{i\in s_l}[f_i(x^l) \\ & + \lambda\tilde{p}(x^l) - T_i^{l-1}(x^l)]. \end{aligned} \tag{9}$$

Let $\mathcal{F}_l$ denote the σ-algebra generated by the entire history of SDCA up to the iteration l, i.e., $\mathcal{F}_0 = \sigma(x^0)$ and $\mathcal{F}_l = \sigma(x^0, ..., x^l, s_0, ..., s_{l-1})$ for all $l \geq 1$. By taking the expectation of the inequality (9) conditioned on $\mathcal{F}_l$, we have

$$\mathbb{E}\left[T^l(x^{l+1})|\mathcal{F}_l\right] \leq T^{l-1}(x^l) - \frac{b}{n}\left[T^{l-1}(x^l) - f(x^l)\right].$$

By the supermartingale convergence theorem, we can conclude that the sequence $\{T^{l-1}(x^l) - \alpha^*\}$ converges almost surely. Moreover,

$$\sum_{l=1}^{\infty}\left[T^{l-1}(x^l) - f(x^l)\right] < \infty, \tag{10}$$

almost surely and hence $\{f(x^l)\}$ converges almost surely.

b) Since $y_i^{l-1} \in \lambda\partial\tilde{h}(x_i^{l-1})$, we have

$$\lambda\tilde{h}(x) \geq \lambda\tilde{h}(x_i^{l-1}) + \langle x - x_i^{l-1}, y_i^{l-1}\rangle. \tag{11}$$

Since $f_i(x)$ is a differentiable function with L-Lipschitz gradient, we have

$$f_i(x) \leq f_i(x_i^{l-1}) + \langle x - x_i^{l-1}, \nabla f_i(x_i^{l-1})\rangle + \frac{L}{2}\|x - x_i^{l-1}\|^2.$$

Thus, we get that

$$f_i(x) + \lambda\tilde{p}(x) \leq T_i^{l-1}(x) + \frac{L-\rho}{2}\|x - x_i^{l-1}\|^2. \tag{12}$$

From (9) and (12), we have

$$T^l(x^{l+1}) \leq T^{l-1}(x^l) - \frac{\rho - L}{2n}\sum_{i\in s_l}\|x^l - x_i^{l-1}\|^2. \tag{13}$$

By taking the expectation of the inequality (13) conditioned on $\mathcal{F}_l$, we have

$$\mathbb{E}\left[T^l(x^{l+1})|\mathcal{F}_l\right] \leq T^{l-1}(x^l) - \frac{b(\rho - L)}{2n^2}\sum_{i=1}^n\|x^l - x_i^{l-1}\|^2.$$

By the supermartingale convergence theorem, we conclude that

$$\sum_{l=1}^{\infty}\sum_{i=1}^{n}\|x^l - x_i^{l-1}\|^2 < \infty, \tag{14}$$

is almost surely satisfied. In particular, we have

$$\sum_{l=1}^{\infty}\|x^l - x^{l-1}\|^2 < \infty, \tag{15}$$

almost surely and hence $\lim_{l\to\infty}\|x^l - x^{l-1}\| = 0$ almost surely.

c) Assume that there exists a sub-sequence $\{x^{l_k}\}$ of $\{x^l\}$ such that $x^{l_k} \to x^*$ almost surely. From (14) and (15), we have $\|x^{l_k+1} - x_i^{l_k}\| \to 0$ almost surely. Without loss of generality, we can suppose that the sub-sequences $y_i^{l_k} \to y^*$ almost surely. We note that $y_i^{l_k} \in \lambda\partial\tilde{h}(x_i^{l_k})$ and by the closed property of the subdifferential mapping $\partial\tilde{h}$, we have $y^* \in \lambda\partial\tilde{h}(x^*)$ with probability one. It follows from $x^{l_k+1} \in \arg\min T^{l_k}(x)$ that $T^{l_k}(x^{l_k+1}) \leq T^{l_k}(x)$. Taking $k \to \infty$, we get that

$$\lambda\tilde{g}(x^*) \leq \lambda\tilde{g}(x) + \frac{\rho}{2}\|x - x^*\|^2 - \langle x - x^*, y^* - \frac{1}{n}\sum_{i=1}^n\nabla f_i(x^*)\rangle,$$

is almost surely satisfied for all $x \in \mathbb{R}^m$. Thus, we have

$$y^* - \frac{1}{n}\sum_{i=1}^n\nabla f_i(x^*) \in \partial\lambda\tilde{g}(x^*), \tag{16}$$

with probability one. Therefore,

$$y^* \in \left[\nabla\frac{1}{n}\sum_{i=1}^n f_i(x^*) + \partial\lambda\tilde{g}(x^*)\right] \cap \partial\lambda\tilde{h}(x^*), \tag{17}$$

with probability one. This implies that x^* is a critical point of f with probability one and the proof is complete. □

3. Application to Group Variables Selection in Multiclass Logistic Regression

Logistic regression, introduced by D. Cox in 1958 (Cox, 1958), is a popular method in supervised learning. Logistic regression has been successfully applied in various real-life problems such as cancer detection (Kim et al., 2008), medical (Bagley et al., 2001; Subasi & Erçelebi, 2005), social science (King & Zeng, 2001), etc. Especially, logistic regression combined with feature selection has been proved to be suitable for high dimensional problems, for instance, document classification (Genkin et al., 2007) and microarray classification (Liao & Chin, 2007; Kim et al., 2008).

We describe the multiclass logistic regression problem as follows. Let W be a $d \times Q$ matrix, where d and Q are the number of features and number of classes, respectively. We denote the i-th column of W by $W_{:,i}$ and $b = (b_1, ..., b_Q) \in \mathbb{R}^Q$. In the multiclass logistic classification problem, a new instance x^* is classified to class y^* by using the rule $y^* = \arg\max_k p(Y = k|X = x^*)$, where $p(Y = y|X = x)$ is the conditional probability defined by

$$p(Y = y|X = x) = \frac{\exp(b_y + W_{:,y}^T x)}{\sum\limits_{k=1}^{Q} \exp(b_k + W_{:,k}^T x)}. \tag{18}$$

Given a training set containing n instances x_i and their corresponding labels $y_i \in \{1, ..., Q\}$, we aim to find (W, b) for which the total probability of the training instances x_i belonging to its correct classes y_i is maximized. To estimate (W, b), we maximize the log-likelihood function defined as

$$\mathcal{L}(W, b) := -\frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i, W, b) \tag{19}$$

where $\ell(x_i, y_i, W, b) = -\log p(Y = y_i|X = x_i)$. As mentioned above, to deal with irrelevant and/or redundant variables in high-dimensional data, we use variables selection method. Note that a variable j is to be removed if and only if all components in the row j of W are zero. Therefore, we can consider each row of W as a group. Denote by $W_{j,:}$ the j-th row of the matrix W. The $\ell_{2,0}$-norm of W, i.e., the number of non-zero rows of W, is defined by

$$\|W\|_{2,0} = |\{j \in \{1, ..., d\} : \|W_{j,:}\|_2 \neq 0\}|.$$

Hence, the $\ell_{2,0}$ regularized multiclass logistic regression problem is formulated as

$$\min_{W,b} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i, W, b) + \lambda \|W\|_{2,0} \right\}. \tag{20}$$

Observe that the problem (20) takes the form of (2) where the function $f_i(W, b) = \ell(x_i, y_i, W, b)$. In this application, we use a non-convex approximation of the $\ell_{2,0}$-norm based on the piecewise exponential penalty function. This approximation function has shown its efficiency in several problems, for instance, variables selection in SVM (Bradley & Mangasarian, 1998; Le Thi et al., 2008), semi-supervised support vector machines (Le et al., 2015), sparse multiclass support vector machines (Le Thi & Nguyen, 2017), sparse signal recovery (Le Thi et al., 2013), sparse linear discriminant analysis (Le Thi & Phan, 2016a;b), variables selection in SVM with uncertain data (Le Thi et al., 2014), etc. Using the piecewise exponential penalty function, the corresponding approximate problem of (20) takes the form:

$$\min_{W,b} \left\{ \frac{1}{n} \sum_{i=1}^{n} f_i(W, b) + \lambda \tilde{p}(W) \right\}, \tag{21}$$

where $\tilde{p}(W) = \sum_{j=1}^{d} \eta_\alpha(\|W_{j,:}\|_2)$ with $\eta_\alpha(t) = 1 - \exp(-\alpha|t|)$. The function $\tilde{p}(W)$ can be expressed as a DC function:

$$\tilde{p}(W) = \alpha \sum_{j=1}^{d} \|W_{j,:}\|_2 - \tilde{h}(W),$$

where $\tilde{h}(W) = \sum_{j=1}^{d} [-1 + \alpha|W_{j,:}\|_2 + \exp(-\alpha\|W_{j,:}\|_2)]$.

According to the SDCA scheme in Algorithm 1, at each iteration l, we have to compute $(v_i^l, z_i^l) = \rho(W^l, b^l) - \nabla f_i(W^l, b^l) + (y^l, 0)$ for $i \in s_l$, where

$$\begin{aligned} \nabla_{b_k} f_i(W^l, b^l) &= p_k^l(x_i) - \delta_{ky_i} \\ \nabla_{W_{:,k}} f_i(W^l, b^l) &= \left(p_k^l(x_i) - \delta_{ky_i}\right) x_i \end{aligned} \tag{22}$$

with $p_k^l(x_i) = \frac{\exp(b_k^l + (W_{:,k}^l)^T x_i)}{\sum_{h=1}^{Q} b_h^l + (W_{:,h}^l)^T x_i)}$ and $\delta_{ky_i} = 1$ if $k = y_i$ and 0 otherwise. The computation of y^l is given by

$$y_{j,:}^l = \begin{cases} 0 & \text{if } \|W_{j,:}^l\|_2 = 0 \\ \frac{\lambda \alpha \eta_\alpha(\|W_{j,:}^l\|_2)}{\|W_{j,:}^l\|_2} W_{j,:}^l & \text{otherwise} \end{cases}. \tag{23}$$

SDCA for solving (21) is described in Algorithm 2.

3.1. Numerical Experiment

3.1.1. Datasets

To illustrate the performances of algorithms, we performed numerical tests on real datasets (*aloi*, *covertype*, *madelon* and *sensorless*) and simulated datasets (*sim_1*, *sim_2* and *sim_3*). Dataset *Aloi* is a library of object images [1] while *covertype, madelon, sensorless,* are taken from the well-known UCI data repository.

We used the same way as proposed in (Witten & Tibshirani, 2011) to generate simulated datasets. In *sim_1*, features are independent with different means in each class.

[1] http://aloi.science.uva.nl/

Algorithm 2 SDCA for solving the problem (21)

```
**Initialization:** Choose $W^0 \in \mathbb{R}^{d\times Q}, b^0 \in \mathbb{R}^Q$, $\rho > L$, $s_0 = \{1,...,n\}$, $l \leftarrow 0$.
**Repeat**
    1. Compute $(v_i^l, z_i^l) = \rho(W^l, b^l) - \nabla f_i(W^l, b^l) + (y^l, 0)$ for $i \in s_l$ using (22)-(23) and keep $(v_i^l, z_i^l) = (v_i^{l-1}, z_i^{l-1})$ for $i \notin s_l$.
    2. Compute $(W^{l+1}, b^{l+1})$ by
        $b^{l+1} = \frac{1}{\rho n}\sum_{i=1}^n z_i^l$
        $W_{j,:}^{l+1} = \left(\|v_{j,:}^l\|_2 - \lambda\alpha/\rho\right)_+ \frac{v_{j,:}^l}{\|v_{j:,}^l\|_2}$,     (24)
    where $v_{j,:}^l = \frac{1}{\rho n}\sum_{i=1}^n (v_i^l)_{j,:}$ for $j = 1,...,d$.
    3. Set $l \leftarrow l+1$ and randomly choose a small subset $s_l \subset \{1,...,n\}$.
**Until** Stopping criterion.
```

In *sim_2*, features also have different means in each class, however they are dependent. The dataset *sim_3* has different one-dimensional means in each class with independent features. The procedure for generating simulated datasets is described as follows.

For *sim_1*: this dataset consists of four classes. The class k is sampled from the multivariate normal distribution $\mathcal{N}(\mu_k, I)$, where the mean vector $\mu_k \in \mathbb{R}^{50}$ is given by $\mu_{kj} = 0.5$ if $10(k-1)+1 \leq j \leq 10k$ and 0 otherwise. We generate $25,000$ samples for each class.

For *sim_2*: we generate a dataset with three classes sampled from the multivariate normal distributions $\mathcal{N}(\mu_k, \Sigma)$, $k = 1,2,3$, where $\mu_k \in \mathbb{R}^{50}$ is defined by $\mu_{kj} = 0.4(k-1)$ if $j \leq 40$ and 0 otherwise. We use the block diagonal matrix Σ with five blocks of dimension 10×10 whose element (j,j') is $0.6^{|j-j'|}$. $150,000$ instances are generated.

For *sim_3*: we generate a dataset including four classes as follows: $x_i \in C_k$ then $x_{ij} \sim \mathcal{N}((k-1)/3, 1)$ if $j \leq 100$, $k = 1,2,3,4$ and $x_{ij} \sim \mathcal{N}(0,1)$ otherwise. We generate $250,000$ instances with equal probabilities for each class.

For pre-processing data, we use standardization to scale the data.

3.1.2. Comparative algorithms

We compare our algorithm with two algorithms: *msgl* and *liblinear*. *msgl* (Vincent & Hansen, 2014) is a coordinate gradient descent algorithm for solving the multiclass logistic regression using $\ell_{2,1}$ regularization term, i.e., the convex problem $\min_{W,b}\left\{\frac{1}{n}\sum_{i=1}^n \ell(x_i, y_i, W, b) + \lambda\|W\|_{2,1}\right\}$. `LibLinear` (Fan et al., 2008) is a well-known package for solving large-scale problems by using the coordinate descent algorithm. We use the ℓ_1-regularized logistic regression solver of `LibLinear` to solve the binary logistic regression problem $\min_w \left\{\sum_{i=1}^n log(1+e^{-y_i w^T x_i}) + \lambda\sum_{j=1}^d |w_j|\right\}$, and then the one-vs-the-rest strategy is used for the multiclass case.

3.1.3. Experiment setting

The comparison of algorithms are performed in terms of three criteria: classification accuracy on test set, sparsity of solution and running time. Sparsity is computed as the percentage of selected features, where a feature $j \in \{1,\ldots,d\}$ is considered to be removed if all absolute values of components of row $W_{j,:}$ are smaller than a threshold $\epsilon = 10^{-8}$.

The cross-validation procedure is used for experiments. We randomly take 80% of the whole dataset as a training set and the rest is used as test set (20%). This process is repeated 10 times and we report the mean and standard deviation of each criterion.

We use the early-stopping condition for `SDCA`. This is a well-know technique in machine learning, especially in stochastic learning which permits to avoid the over-fitting problem. After each epoch, we compute the accuracy based on the validation set, then we stop `SDCA` if the accuracy is not improved after $n_{patience} = 5$ epochs. For comparative algorithms, we use their default stopping parameters. We also stop algorithms if they exceed 2 hours of running time in the training process.

For `SDCA`, we set the trade-off parameter $\lambda \in \left\{10^{-4}, 10^{-3}, ...1\right\}$ and the parameter for controlling the tightness of zero-norm approximation $\alpha \in \{0.5, 1, 2, 5\}$. For both `LibLinear` and `msgl`, the trade-off parameter is chosen in interval $\{10^{-3}, \ldots, 10^4\}$.

All experiments are performed on a PC Intel (R) Xeon (R) E5-2630 v2 @2.60 GHz of 32GB RAM.

3.1.4. Experiment 1 : comparison of SDCA and DCA

Firstly, we will study the impact of batch size on the quality of solution and the running time of *SDCA*. The batch size refers to the size of set of index s_l, i.e., the number of components h_i that are used to compute the subgradients of $\bar{h}$ at each iteration (c.f Algorithm 1). In DCA, or full-batch DCA, all components h_i are used, i.e., $s_l \equiv \{1,\ldots,n\}$. The Table 1 reports the accuracy and the running time of *SDCA* as the batch size varies on an arbitrary chosen dataset (*sensorless*).

We observe that the running time is smallest (1.78s) with batch size equals to 10% while giving the second best classification accuracy 85.23%, only 0.07% smaller than the best one. Hence, we choose the batch size as 10% through-

Table 1. Performance of *SDCA* as batch size varies

Batch Size	5%	10%	15%	20%	25%	30%	40%	50%
Time (s)	2.93±0.1	1.78±0.1	2.53±0.1	2.06±0.2	1.54±0.2	2.24±0.3	2.98±0.4	1.94±0.50
Accuracy (%)	84.62±1.1	85.23±0.7	83.25±1.2	85.05±0.9	82.77±1.1	84.54±1.1	85.30±0.7	82.88±1.1

Table 2. Comparative results on both simulated and real datasets.
Bold values correspond to best results for each dataset. NA means that the algorithm fails to furnish a result. n, d and Q is the number of instances, the number of dimensions and the number of classes respectively.

Dataset	Algorithm	Accuracy (%)		Time (s)		Sparsity (%)	
		Mean	STD	Mean	STD	Mean	STD
aloi	DCA (Full-batch)	**85.05**	0.44	2414.72	77.12	97.66	1.35
($n \times d$) = (108,000×128)	SDCA	82.98	0.47	**208.67**	50.03	**63.87**	3.57
Q = 1,000	MSGL	NA	NA	NA	NA	NA	NA
	LibLinear	81.61	0.20	2732.96	46.38	100.00	0.00
covertype	DCA (Full-batch)	71.30	0.09	322.98	0.27	100.00	0.00
($n \times d$) = (581,012×54)	SDCA	**71.58**	0.13	**5.86**	1.14	84.72	6.48
Q = 7	MSGL	71.22	0.02	525.49	1.10	**68.52**	0.00
	LibLinear	71.54	0.19	264.88	26.83	100.00	0.00
madelon	DCA (Full-batch)	61.35	0.39	5.25	0.02	0.80	0.04
($n \times d$) = (2,600×500)	SDCA	**62.60**	1.38	0.12	0.12	**0.43**	0.23
Q = 2	MSGL	60.48	2.37	23.92	0.12	0.67	0.00
	LibLinear	61.54	2.72	**0.08**	0.01	0.58	0.06
sensorless	DCA (Full-batch)	**90.21**	0.41	34.65	1.79	**38.19**	1.20
($n \times d$) = (58,509×48)	SDCA	85.11	0.83	**11.76**	4.60	40.00	4.52
Q = 11	MSGL	85.06	0.31	199.00	41.75	50.00	0.00
	LibLinear	75.55	0.24	216.48	72.05	100.00	0.00
sim_1	DCA (Full-batch)	72.11	0.57	24.67	6.39	84.00	2.00
($n \times d$) = (100,000×50)	SDCA	72.24	0.42	**1.33**	0.23	**80.00**	0.00
Q = 4	MSGL	72.33	0.18	214.83	25.40	82.00	0.00
	LibLinear	**72.62**	0.38	2038.85	5.05	**80.00**	0.00
sim_2	DCA (Full-batch)	65.53	3.70	76.04	1.57	**79.33**	1.15
($n \times d$) = (150,000×50)	SDCA	**68.60**	0.22	1.74	0.27	80.00	0.00
Q = 3	MSGL	68.42	0.03	367.29	53.52	82.00	0.00
	LibLinear	66.92	0.02	2.04	0.23	80.00	0.00
				15			
sim_3	DCA (Full-batch)	99.87	0.02	151.72	4.75	80.53	3.13
($n \times d$) = (250,000×500)	SDCA	**99.93**	0.01	**22.83**	2.55	**80.00**	0.00
Q = 4	MSGL	**99.93**	0.01	1581.44	14.76	80.20	0.00
	LibLinear	99.03	0.00	50.50	2.96	97.16	0.50

out our experiments.

To illustrate the potential gain of *SDCA*, we compare it with a DCA for solving the problem (21). From the Table 2, we see that the gain of running time of *SDCA* ranges from 11.6 times (*aloi*) to 55.1 times (*covertype*).

Concerning the classification accuracy, *SDCA* and *DCA* are comparable. *SDCA* gives slightly better accuracy than *DCA* on *covertype, sim_1, sim_3*, with a gain ranges from 0.06% to 0.28%. The gain of *SDCA* is higher on 2 datasets (*madelon, sim_2*), 1.35% and 3.07%. *DCA* furnishes a better result on *aloi* and *sensorless*, especially the gain is up to 4.9% on *sensorless*. The results prove that *SDCA* can greatly improve the running time of *DCA* while archiving a similar accuracy.

3.1.5. Experiment 2 : simulated dataset

For synthetic datasets (*sim_1*, *sim_2* and *sim_3*), we know in advance the informative features that were used to generate the datasets. Hence, the purpose of this experiment is to study the ability of algorithms to select these informative features in order to furnish a good classification accuracy. The comparison is performed with 3 algorithms, `SDCA`, `msgl` and `LibLinear`. We report the results in Table 2, and observe that.

For *sim_1* dataset, `LibLinear` gives a slightly better classification accuracy (72.62%) comparing to `SDCA` (72.24%) and `msgl` (72.33%). However, `SDCA` is by far the fastest algorithm. `SDCA` is 1532 (resp. 136) times faster than `LibLinear` (resp. `msgl`). Furthermore, `SDCA` and `LibLinear` successfully suppress the 20% uninformative features, which it also matches with our procedure of generating this synthetic dataset. `msgl` fails on this purpose by selecting 82% of features.

For *sim_2* dataset, `SDCA` is the best algorithm on both criteria: classification accuracy and running time. Similarly to *sim_1* dataset, only `SDCA` and `LibLinear` can correctly select the informative features (80%).

For *sim_3* dataset, `SDCA` exceeds `LibLinear` and `GLASSO` on all three comparison criteria: classification accuracy, sparsity and speed. `LibLinear` almost selects all the features (97.16% selected) but gives 0.89% accuracy lower than `SDCA` (99.93%), and it is also 2 times slower than `SDCA`. Among the three algorithm, only `SDCA` successfully selects the informative features.

To summarize, for all three synthetic datasets, `SDCA` successfully selects the exact informative features. `LibLinear` selects the exact features on 2 out of 3 datasets while `GLASSO` fails on all three datasets.

3.1.6. Experiment 3 : real-world datasets

In this experiment, we perform the comparative study between `SDCA`, `msgl` and `LibLinear` on real-world datasets . We observe from Table 2 that.

For *aloi* dataset, `SDCA` only selects 63.87% of features for a classification accuracy of 82.98% while `LibLinear` has a worse accuracy with 100% of features used. Moreover, `SDCA` is the 12.6 times faster than `LibLinear` while `msgl` fails to furnish a result after 2 hours of running time.

For *covertype* dataset, `SDCA` furnishes better classification accuracy than `LibLinear` and `msgl`. Moreover, `SDCA` is by far faster than the two others. `SDCA` is `SDCA` is 45 times faster than `LibLinear` and 89 times faster than `msgl`. Concerning the sparsity of solution, `msgl` is the best while `LibLinear` fails to suppress features.

The dataset *madelon* is known to be non-linear. Hence, all three algorithms furnish quite low classification accuracy (62.38% for `SDCA`, 61.54% for `LibLinear` and 60.48% for `msgl`). As for the sparsity, `SDCA` suppresses more features than `LibLinear` and `msgl`.

For *sensorless* dataset, `SDCA` is better than both `LibLinear` and `msgl` on all three aspects: classification accuracy, sparsity and running time. In terms of classification accuracy, the gain of `SDCA` versus `msgl` (resp. `LibLinear`) is 3.89% (resp. 2.26%). Regarding the running time, `SDCA` is 113 times faster than `msgl` and 137 times faster than `LibLinear`. As for the sparsity, `SDCA` selects 10% less features than `msgl` while `LibLinear` fails to suppress features.

Overall, `SDCA` gives the best among the three in term of classification accuracy on all 4 datasets. As for running time, `SDCA` is by far the fastest algorithm. Concerning the sparsity of solution, `SDCA` suppresses more features than the two others on 3 out of 4 datasets.

4. Conclusions

We have rigorously studied the large-sum optimization problem involving $\ell_{2,0}$ regularization. The $\ell_{2,0}$-norm is approximated by a DC function, namely the piecewise exponential function. The resulting problem is then reformulated as a DC program and we developed stochastic DCA to solve it. Exploiting the fact that each component $f_i(x)$ is differentiable with L-Lipschitz gradient, we propose, a stochastic version of DCA that is very inexpensive. At each iteration, the algorithm only requires the computing the subgradients of a small subset of functions and the projection of points onto balls that is explicitly computed. We have also proved that the convergence is guaranteed with probability one. As an application, we applied our algorithm to the group variables selection in multiclass logistic regression problem. Numerical experiments were carefully conducted on both synthetic and real-world datasets. The numerical results show that SDCA greatly improves the running time of DCA while giving similar accuracy. Moreover, our algorithm `SDCA` outperforms standard algorithms (`Liblinear` and `msgl`) on all 3 criteria: classification accuracy, sparsity of solution and running time. Especially, the gain in running time is huge. `SDCA` is up to 210 times faster than `msgl` and 1537 times faster than `LibLinear`. We are convinced that stochastic DCA is a promising approach for handling very large-scale datasets in machine learning.

References

Bagley, S. C., White, H., and Golomb, B. A. Logistic regression in the medical literature: Standards for use and reporting, with particular attention to one medical domain. *Journal of Clinical Epidemiology*, 54(10):979–985, 2001.

Blondel, Mathieu, Seki, Kazuhiro, and Uehara, Kuniaki. Block coordinate descent algorithms for large-scale sparse multiclass classification. *Machine Learning*, 93 (1):31–52, 2013.

Bradley, Paul S. and Mangasarian, O. L. Feature Selection via Concave Minimization and Support Vector Machines. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pp. 82–90, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

Cox, David. The regression analysis of binary sequences (with discussion). *J Roy Stat Soc B*, 20:215–242, 1958.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. URL https://cran.r-project.org/web/packages/LiblineaR/index.html.

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

Gu, Quanquan, Li, Zhenhuif, and Han, Jiawei. Linear discriminant dimensionality reduction. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 549–564. Springer, 2011.

Huang, Jian, Wei, Fengrong, and Ma, Shuangge. Semiparametric Regression Pursuit. *Statistica Sinica*, 22(4): 1403–1426, 2012.

Khan, Z., Shafait, F., and Mian, A. Joint Group Sparse PCA for Compressed Hyperspectral Imaging. *IEEE Transactions on Image Processing*, 24(12):4934–4942, 2015.

Kim, Jinseog, Kim, Yuwon, and Kim, Yongdai. A Gradient-Based Optimization Algorithm for LASSO. *Journal of Computational and Graphical Statistics*, 17 (4):994–1009, 2008.

King, Gary and Zeng, Langche. Logistic Regression in Rare Events Data. *Political Analysis*, 9:137–163, 2001.

Le, H. M., Le Thi, H. A., and Nguyen, M. C. Sparse semi-supervised support vector machines by DC programming and DCA. *Neurocomputing*, 153:62–76, 2015.

Le Thi, H. A. and Nguyen, M. C. DCA based algorithms for feature selection in multi-class support vector machine. *Annals of Operations Research*, 249(1):273–300, 2017.

Le Thi, H. A., Nguyen, T. B. T, and Le, H. M. Sparse Signal Recovery by Difference of Convex Functions Algorithms. In *Intelligent Information and Database Systems*, pp. 387–397. Springer, Berlin, Heidelberg, 2013.

Le Thi, H. A., Vo, X. T., and Pham Dinh, T. Feature selection for linear SVMs under uncertain data: Robust optimization based on difference of convex functions algorithms. *Neural Networks*, 59:36–50, 2014.

Le Thi, H. A., Pham Dinh, T., Le, H. M., and Vo, X. T. DC approximation approaches for sparse optimization. *European Journal of Operational Research*, 244(1):26–46, 2015.

Le Thi, Hoai An and Pham Dinh, Tao. The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems. *Annals of Operations Research*, 133(1-4):23–46, 2005.

Le Thi, Hoai An and Phan, Duy Nhat. DC Programming and DCA for Sparse Optimal Scoring Problem. *Neurocomput.*, 186(C):170–181, 2016a.

Le Thi, Hoai An and Phan, Duy Nhat. DC programming and DCA for sparse Fisher linear discriminant analysis. *Neural Computing and Applications*, pp. 1–14, 2016b.

Le Thi, Hoai An, Le, Hoai Minh, Nguyen, Van Vinh, and Pham, Dinh Tao. A DC programming approach for feature selection in support vector machines learning. *Advances in Data Analysis and Classification*, 2(3):259–278, 2008.

Lee, Sangin, Oh, Miae, and Kim, Yongdai. Sparse optimization for nonconvex group penalized estimation. *Journal of Statistical Computation and Simulation*, 86 (3):597–610, 2016.

Liao, J. G. and Chin, Khew-Voon. Logistic regression for disease classification using microarray data: Model selection in a large p and small n case. *Bioinformatics*, 23 (15):1945–1951, 2007.

Obozinski, Guillaume, Taskar, Ben, and Jordan, Michael. Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2, 2006.

Pham Dinh, Tao and Le Thi, Hoai An. Convex analysis approach to dc programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

Pham Dinh, Tao and Le Thi, Hoai An. A D. C. Optimization Algorithm for Solving the Trust-Region Subproblem. *SIAM Journal of Optimization*, 8(2):476–505, 1998.

Phan, Duy Nhat, Le Thi, Hoai An, and Pham Dinh, Tao. Efficient bi-level variable selection and application to estimation of multiple covariance matrices. In *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Proceedings, Part I*, volume 10234, pp. 304–316. Springer International Publishing, 2017.

Reddi, Sashank J., Sra, Suvrit, Poczos, Barnabas, and Smola, Alexander J. Proximal stochastic methods for Nonsmooth Nonconvex Finite-Sum Optimization. In *Advances in Neural Information Processing Systems*, pp. 1145–1153, 2016.

Schmidt, Mark, Le Roux, Nicolas, and Bach, Francis. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112, 2015.

Subasi, Abdulhamit and Erçelebi, Ergun. Classification of EEG signals using neural network and logistic regression. *Computer Methods and Programs in Biomedicine*, 78(2):87–99, 2005.

Sun, Liang, Liu, Jun, Chen, Jianhui, and Ye, Jieping. Efficient Recovery of Jointly Sparse Vectors. In Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems 22*, pp. 1812–1820. Curran Associates, Inc., 2009.

Tibshirani, Robert. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Vincent, Martin and Hansen, Niels Richard. Sparse group lasso and high dimensional multinomial classification. *Comput. Stat. Data Anal.*, 71:771–786, 2014.

Wang, Lifeng, Chen, Guang, and Li, Hongzhe. Group SCAD regression analysis for microarray time course gene expression data. *Bioinformatics*, 23(12):1486–1494, 2007.

Witten, Daniela M. and Tibshirani, Robert. Penalized classification using Fisher's linear discriminant. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(5):753–772, 2011.

Yuan, Ming and Lin, Yi. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.