Mach Learn (2015) 101:163–186
DOI 10.1007/s10994-014-5455-y

Feature selection in machine learning: an exact penalty approach using a Difference of Convex function Algorithm

Hoai An Le Thi · Hoai Minh Le · Tao Pham Dinh

Received: 31 December 2013 / Accepted: 2 June 2014 / Published online: 4 July 2014

Abstract We develop an exact penalty approach for feature selection in machine learning via the zero-norm ℓ_0-regularization problem. Using a new result on exact penalty techniques we reformulate equivalently the original problem as a Difference of Convex (DC) functions program. This approach permits us to consider all the existing convex and nonconvex approximation approaches to treat the zero-norm in a unified view within DC programming and DCA framework. An efficient DCA scheme is investigated for the resulting DC program. The algorithm is implemented for feature selection in SVM, that requires solving one linear program at each iteration and enjoys interesting convergence properties. We perform an empirical comparison with some nonconvex approximation approaches, and show using several datasets from the UCI database/Challenging NIPS 2003 that the proposed algorithm is efficient in both feature selection and classification.

Keywords Zero-norm · Feature selection · Exact penalty · DC programming · DCA

Editors: Vadim Strijov, Richard Weber, Gerhard-Wilhelm Weber, and Süreyya Ozogur Akyüz.

H. A. Le Thi (✉) · H. M. Le
Laboratory of Theoretical and Applied Computer Science (LITA EA 3097),
UFR MIM, University of Lorraine, Ile du Saulcy, 57045 Metz, France
e-mail: hoai-an.le-thi@univ-lorraine.fr

H. M. Le
e-mail: minh.le@univ-lorraine.fr

T. Pham Dinh
Laboratory of Mathematics, National Institute for Applied Sciences - Rouen, University of Normandie,
Avenue de l'Université, 76801 Saint-Etienne-du-Rouvray cedex, France
e-mail: pham@insa-rouen.fr

1 Introduction

Feature (or variable) selection, which consists of choosing a subset of available features that capture the relevant properties of the data, is one of fundamental problems in machine learning. Feature selection can help enhance accuracy in many machine learning problems, it can also improve the efficiency of training. Features can be divided into three categories: relevant, redundant and irrelevant features. An irrelevant feature does not apport any useful information while a redundant feature adds no new information to learning procedures (i.e. information already carried by other features). Ideally, the learning process should discard irrelevant/redundant features and use only a subset of relevant features that leads to the best performance.

Machine learning methods for feature selection can be divided into three classes (Rinaldi 2000): wrapper, filter, and embedded methods. Wrapper methods exploit a machine learning algorithm to evaluate the usefulness of features. Filter methods rank the features according to some discrimination measure and select features having higher ranks without using any learning algorithm (it utilizes the underlying characteristics of the training data to evaluate the relevance of the features or feature set by some independent measures such as distance measure, correlation measures, consistency measures (Chen et al. 2006). The wrapper approach is generally considered to produce better feature subsets but runs much more slowly than a filter. Embedded methods do not separate the learning from the feature selection part. It integrates the selection of features in the model building.

This paper concerns with an embedded approach for feature selection in machine learning. For a vector $x \in \mathbb{R}^n$, the support of x, denoted $supp(x)$, is the set of the indices of the non-zero components of x, say

$$supp(x) = \{i \in \{1, \ldots, n\} : \ x_i \neq 0\},$$

and the zero-norm of x, denoted ℓ_0-norm is defined as

$$\|x\|_0 := \text{cardinality of } supp(x).$$

The useful notation $|.|_0$ denoting the ℓ_0-norm on $\mathbb{R}$, also called the step function ($|x|_0 = 1$ if $x \neq 0$, 0 otherwise) allows for expressing the separability of $\|.\|_0$ on $\mathbb{R}^n$

$$\|x\|_0 = \sum_{i=1}^{n} |x_i|_0 . \tag{1}$$

Given a training data $\{\vartheta_i, \delta_i\}_{i=1,\ldots,m}$ where each $\vartheta_i \in \mathbb{R}^n$ corresponds to the observed value δ_i. Formally, a learning task can be defined as the following structural risk minimization problem (for a given $\lambda > 0$)

$$\min_{x,y} \sum_{i=1}^{m} L(\vartheta_i, \delta_i; x, \mu) + \lambda \Omega(x), \tag{2}$$

where L is a loss function defined on $\mathbb{R}^n \times \mathbb{R}^p$ and $\Omega(\cdot)$ is the regularizer (or penalty) term. The loss function L is the data fitting term measuring the discrepancy for all training examples $\{\vartheta_i, \delta_i\}$ between the predicted value and the observed value. The regularizer term $\Omega(\cdot)$ is a penalty providing regularization and controlling generalization ability through model complexity.

A natural way to deal with feature selection in machine learning is using ℓ_0-norm in the regularization term that leads to the following problem:

$$\min \left\{ \sum_{i=1}^{m} L(\vartheta_i, \delta_i; x, \mu) + \lambda \, \|x\|_0 : (x, \mu) \in \mathbb{R}^n \times \mathbb{R}^p \right\}. \tag{3}$$

Let K be a bounded polyhedral convex set in $\mathbb{R}^n \times \mathbb{R}^p$. We consider in this paper a so called ℓ_0-regularizer problem that takes the form

$$\min \left\{ f(x, \mu) + \lambda \, \|x\|_0 : (x, \mu) \in K \right\}, \tag{4}$$

where the function f corresponding to a given convex criterion is assumed to be convex and the regularization parameter λ makes the trade-off between the criterion f and the sparsity of x. Here μ is the variable that does not deal with the sparsity. This problem is a common model that can be used in several learning contexts including feature selection in classification, feature selection in linear regression, sparse Fisher linear discriminant analysis, feature selection in learning to rank with sparse SVM, etc.

The function ℓ_0, apparently very simple, is lower-semicontinuous on $\mathbb{R}^n$, but its discontinuity at the origin makes nonconvex programs involving $\|.\|_0$ challenging, they are known to be NP-hard (Amaldi and Kann 1998; Natarajan 1995) and actually intractable. To circumvent the discontinuity, continuous approaches are developed since many years.

During the last two decades, research is very active in models and methods optimization involving the zero-norm. Works can be divided into three categories according to the way to treat the zero-norm: convex approximation, nonconvex approximation, and nonconvex exact reformulation.

In the machine learning community, one of the best known approaches, belonging to the group "convex approximation", is the ℓ_1 regularization approach proposed in Tibshirani (1996) in the context of linear regression, called least absolute shrinkage and selection operator (LASSO), which consists in replacing the ℓ_0 term $\|x\|_0$ by $\|x\|_1$, the ℓ_1-norm of the vector x. In Gribonval and Nielsen (2003), the authors have proven that, under suitable assumptions, a solution of the ℓ_0- regularizer problem over a polyhedral set can be obtained by solving the ℓ_1- regularizer problem. However, these assumptions may not be satisfied in many cases. Since its introduction, several works have been developed to study the ℓ_1-regularization technique, from the theoretical point of view to efficient computational methods (see Hastie et al. 2009, Chap. 18 for more discussions on ℓ_1-regularized methods). Among the best approaches, it is worth to citing the *Elastic net* proposed by Zou and Hastie (2005) for variable selection in regression which is a combination between the ridge (ℓ_2 norm) and the LASSO penalty. It has been shown that the elastic net not only dominates the LASSO in terms of prediction accuracy but also is a better variable selection procedure than the LASSO. The LASSO penalty has been shown to be, in certain cases, inconsistent for variable selection and biased (Zou 2006). Hence, the Adaptive LASSO is introduced in Zou (2006) in which adaptive weights are used for penalizing different coefficients in the ℓ_1-penalty.

At the same time, nonconvex continuous approaches, belonging to the second group "nonconvex approximation" (the ℓ_0 term $\|x\|_0$ is approximated by a nonconvex continuous function) were extensively developed. The first was concave exponential approximation with successive linear approximation (SLA) algorithm proposed in Bradley and Mangasarian (1998) for feature selection in SVM. Later, with the same approximation, an efficient Difference of Convex functions (DC) algorithm (DCA) was developed in Le Thi et al. (2008). Various other nonconvex regularizations have been developed in several works in different contexts, most of them are for feature selection in SVM or feature selection in regression. For example, hard-thresholding methods and/or DCA using the Smoothly Clipped Absolute Deviation (SCAD) (Fan and Li 2001; Kim et al. 2008; Le Thi et al. 2009; Ong and Le Thi 2013; Zou and Li 2008), the log penalty method (Candes et al. 2008) via the logarithmic approximation of Weston et al. (2003), the ℓ_q, $0 < q < 1$ regularization with reweighted ℓ_2 and/or DCA (Chartrand and Yin 2008; Fu 1998; Gasso et al. 2009; Huang et al. 2008; Knight and Fu 2000; Chen et al. 2010; Guan and Gray 2013; Gorodnitsky and Rao 1997; Rao and Kreutz-Delgado 1999; Rao et al. 2003). The common properties of these approaches are that the nonconvex regularization used for approximating the ℓ_0 norm is a DC function, and the resulting optimization problem is a DC program for which DCA, an efficient approach in nonconvex programming framework (see e.g. Le Thi and Pham Dinh 2005; Pham Dinh and Le Thi 1998) has been investigated (Le Thi et al. 2008, 2009; Neumann et al. 2005; Ong and Le Thi 2013; Collober et al. 2006; Candes et al. 2008; Gasso et al. 2009; Guan and Gray 2013) (note also that the SLA (Bradley and Mangasarian 1998), the adaptive LASSO (Zou 2006) are special cases of DCA). These DCA based algorithms solve iteratively a sequence of convex programs (linear or quadratic programs in many cases) until the convergence and can be viewed as sequences of reweighted ℓ_1 (see for example Candes et al. 2008) or reweighted ℓ_2. For instance, Focal Underdetermined System Solver (FOCUSS) (Gorodnitsky and Rao 1997; Rao and Kreutz-Delgado 1999; Rao et al. 2003), Iteratively reweighted least squares (IRLS) (Chartrand and Yin 2008) introduced in the context of compressed sensing, via the ℓ_q-regularizer with $0 < q < 1$, can be viewed as a reweighted ℓ_2 procedure. The Local Quadratic Approximation (LQA) algorithm in Fan and Li (2001) and Zhang et al. (2006) can be also regarded as reweighted ℓ_2 applied on SCAD penalty. Overall, we can say that most of existing methods in nonconvex approximation approaches are DCA based algorithms. Besides, the relaxed Lasso, a generalization of both soft and hard thresholding, introducing a two-stage procedure has been proposed in Meinshausen (2007).

In the third category that we call nonconvex exact reformulation approaches, the ℓ_0-regularized problem is reformulated as a continuous nonconvex program. There are few works in this category. In Mangasarian (1996), the author reformulated the problem (4) in the context of feature selection in SVM as a linear program with equilibrium constraints (LPEC). However, this reformulation is generally intractable for large-scale datasets. In Thiao et al. (2010) an exact penalty technique is used for Sparse Eigenvalue problem with ℓ_0-norm in constraint functions

$$\max\{x^T Ax : x^T x = 1, \|x\|_0 \le k\}, \tag{5}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and k an integer, and a DCA based algorithm was investigated for the resulting problem.

Convex regularization approaches involve convex optimization problems which are so far "easy" to solve, but they does not attain the solution of the ℓ_0-regularizer problem. Nonconvex approximations are, in general, deeper than convex relaxations, and then can produce good sparsity, but the resulting optimization problems are still difficult since they are nonconvex and there are many local minima which are not global. Moreover, the consistency between the approximate problems and the original problem is an open question, i.e. it can not be guaranteed. The exact reformulation approaches can overcome these drawbacks if efficient methods for the reformulation problem are available.

Besides the three above categories, heuristic methods are developed to tackle directly the original problem (4) by greedy based algorithms, e.g. matching pursuit, orthogonal matching pursuit (Mallat and Zhang 1993; Bach et al. 2012), etc.

Our contributions. The above arguments suggest us to develop in this paper an exact reformulation approach for solving the original problem (4). Our main motivation is to exploit the efficiency of DCA to solve this hard problem in an *equivalent* formulation. A new result on exact penalty techniques recently developed in Le Thi et al. (2012) supports this idea. The ℓ_0-regularization problem is first equivalently formulated as a combinatorial optimization problem by using the binary variables $u_i = 0$ if $x_i = 0$ and $u_i = 1$ if $x_i \neq 0$, and then the last problem is reformulated as a DC program via an exact penalty technique (Le Thi et al. 2012). These combinatorial and continuous formulations of (4) permit us to consider all the above convex and nonconvex approaches to treat the zero-norm in a unified view within DC programming and DCA framework. More precisely, we show that the ℓ_1-approach is nothing else the linear relaxation of our combinatorial formulation of (4) while some nonconvex approximations can be regarded as our exact penalty reformulation with suitable parameters. This study is very useful to justify nonconvex approximation approaches. As an application of the proposed approach, we consider the problem of feature selection in SVM. We perform an empirical comparison with some nonconvex approximation approaches, and show using several datasets from the UCI database that the proposed algorithm is efficient in both feature selection and classification.

The rest of the paper is organized as follows. In the next section we present our exact penalty technique to equivalently reformulate the problem (4) as a DC program and discuss about the links between our approach with convex and/or nonconvex approximation approaches. The solution methods based on DC programming and DCA are developed in Sect. 3 while the implementation of the algorithm for feature selection in SVM and numerical experiments are presented in Sect. 4. Finally, some conclusions are provided in Sect. 5. In the appendix we describe the comparative DCA schemes considered in our experiments.

2 Exact penalty techniques related to the ℓ_0-norm

In this section, we first consider the two following problems (K being a bounded polyhedral convex set in $\mathbb{R}^n \times \mathbb{R}^p$, λ a positive parameter and k a positive integer)

$$\alpha := \inf\{f(x,\mu) + \lambda\|x\|_0 : (x,\mu) \in K\}, \tag{6}$$

$$\alpha := \inf\{f(x,\mu) : (x,\mu) \in K, \}, \|x\|_0 \leq k\} \tag{7}$$

whose feasible sets are assumed to be nonempty.

2.1 Continuous reformulation via exact penalty techniques

We will present some main results concerning penalty techniques related to ℓ_0-norm allowing for reformulation of (6) and (7) as nonconvex programs in the continuous framework, especially DC programs, that can be treated by DC programming and DCA.

Denote by e the vector of ones in the appropriate vector space. We suppose that K is bounded in the variable x,i.e. $K \subset \Pi_{i=1}^n [a_i, b_i] \times \mathbb{R}^m$ where $a_i, b_i \in \mathbb{R}$ such that $a_i \leq 0 < b_i$ for $i = 1, \ldots, n$. Let $c_i := \max\{|x_i| : x_i \in [a_i, b_i]\} = \max\{|a_i|, |b_i|\}$ for $i = 1, \ldots, n$. Define the binary variable $u_i \in \{0, 1\}$ as

$$u_i = |x_i|_0 = \begin{cases} 1 \text{ if } x_i \neq 0 \\ 0 \text{ if } x_i = 0, \end{cases} \qquad \forall i = 1 \ldots n. \tag{8}$$

Then (6) and (7) can be reformulated as

$$\alpha := \inf\{f(x,\mu) + \lambda e^T u : (x,\mu) \in K, u \in \{0,1\}^n, |x_i| \le c_i u_i,\ i = 1,\dots,n\}, \tag{9}$$

and

$$\alpha := \inf\{f(x,\mu) : (x,\mu) \in K, u \in \{0,1\}^n, |x_i| \le c_i u_i, i = 1,\dots,n,\ e^T u \le k\}, \tag{10}$$

respectively.

Let $p(u)$ be the penalty function defined by

$$p(u) := \sum_{i=1}^{n} \min\{u_i, 1 - u_i\}\ . \tag{11}$$

Then (6) and (7) can be rewritten respectively as

$$\alpha = \inf\{f(x,\mu) + \lambda e^T u : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le c_i u_i,\ i = 1,\dots,n,\ p(u) \le 0\}, \tag{12}$$

and

$$\alpha := \inf\{f(x,\mu) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le c_i u_i, i = 1,\dots,n, e^T u \le k, p(u) \le 0\}. \tag{13}$$

It leads to the corresponding penalized problems (τ being the positive penalty parameter)

$$\alpha(\tau) := \inf\{f(x,\mu) + \lambda e^T u + \tau p(u) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le c_i u_i,\ i = 1,\dots,n\}, \tag{14}$$

and

$$\alpha(\tau) := \inf\{f(x,\mu) + \tau p(u) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le c_i u_i, i = 1,\dots,n, e^T u \le k\}. \tag{15}$$

Proposition 1 *There is $\tau_0 \ge 0$ such that for every $\tau > \tau_0$ problems* (6) *(resp.* (7)*) and* (14) *(resp.* (15)*) are equivalent, in the sense that they have the same optimal value and $(x^*, \mu^*) \in K$ is a solution of* (6) *(resp.* (7)*) iff there is $u^* \in \{0,1\}^n$ such that (x^*, μ^*, u^*) is a solution of* (14) *(resp.* (15)*).*

Proof Direct consequences of Theorem 8 in Le Thi et al. (2012).

It is clear that (14) and (15) are DC programs if the function $f(x, y)$ is a DC function on K.

Note that, in general, the minimal penalty parameter τ_0, if any, is not computable. In practice, upper bounds for τ_0 can be calculated in some cases, e.g. sparse eigenvalue problems (Thiao et al. 2010).

In the sequel, we will focus on the ℓ_0-regularizer problem (6) and its penalized problem (14).

2.2 Link between (9) and the ℓ_1-regularization problem

It is easy to see that the linear relaxation of Problem (9) is a ℓ_1-regularization problem. Indeed, the linear relaxation of Problem (9) (which is in fact the penalized problem (14) when $\tau = 0$) takes the form

$$\inf\{f(x,\mu) + \lambda e^T u : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le c_i u_i, i = 1,\dots,n\}. \tag{16}$$

Let $M = \max\{c_i : i = 1, \dots, n\}$, problem (16) becomes

$$\inf\{f(x,\mu) + \lambda e^T u : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le M u_i,\ i = 1,\dots,n\}$$

which can be rewritten as

$$\inf\left\{f(x,\mu) + \lambda e^T u\ : (x,\mu) \in K, u \in [0,1]^n, \frac{|x_i|}{M} \le u_i \le 1,\ i = 1,\dots,n\right\}$$

or again

$$\inf\left\{f(x,\mu) + \frac{\lambda}{M}\sum_{i=1}^{n}|x_i| : (x,\mu) \in K\right\} = \inf\left\{f(x,\mu) + \frac{\lambda}{M}\ \|x\|_1 : (x,y) \in K\right\}.$$

2.3 Link between (14) and a nonconvex approximate problem

Most existing approximations of ℓ_0-norm are its DC minorants. We consider here the last (in chronological order) introduced approximation (Peleg and Meir 2008) which is among the best approximations of ℓ_0-norm (Ong and Le Thi 2013). It is defined by:

$$\Psi_\theta(x) := \sum_{i=1}^{n}\psi_\theta(x_i), \forall x = (x_i) \in \mathbb{R}^n,\ \text{ with } \psi_\theta(t) := \min\{\theta\,|t|\,, 1\},\ t \in \mathbb{R}. \tag{17}$$

Since $\psi_\theta(t) = \theta\,|t| + 1 - \max\{\theta\,|t|\,, 1\}$ is a polyhedral DC function, we name this approximation as *polyhedral DC approximation*. We will show that the resulting approximate problem of (4), namely

$$\beta(\theta) := \inf\left\{f(x,\mu) + \lambda\sum_{i=1}^{n}\psi_\theta\,(x_i) : (x,\mu) \in K\right\} \tag{18}$$

is equivalent to the penalized problem (14) with suitable values of parameters λ, τ and θ.

Consider the problem (14) in the form

$$\alpha(\tau) := \inf\{f(x,\mu) + \lambda e^T u + \tau p(u) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \le M u_i,\ i = 1,\dots,n\}. \tag{19}$$

Let $r : \mathbb{R} \to \mathbb{R}$ be the function defined by $r(t) = \min\{t, 1-t\}$. Then $p(u) = \sum_{i=1}^{n} r(u_i)$ and the problem (19) can be rewritten as

$$\alpha(\tau) := \inf\left\{f(x,\mu) + \lambda\sum_{i=1}^{n}\left(u_i + \frac{\tau}{\lambda}r(u_i)\right) : (x,\mu) \in K, \frac{|x_i|}{M} \le u_i \le 1,\ i = 1,\dots,n\right\}, \tag{20}$$

or again

$$\alpha(\tau) := \inf\left\{f(x,\mu) + \lambda\sum_{i=1}^{n}\pi\,(u_i) : (x,\mu) \in K, \frac{|x_i|}{M} \le u_i \le 1,\ i = 1,\dots,n\right\} \tag{21}$$

where $\pi : \mathbb{R} \to \mathbb{R}$ be the function defined by $\pi(t) := t + \frac{\tau}{\lambda}r(t)$.

Proposition 2 *Let $\theta := \frac{\tau+\lambda}{\lambda M}$. For all $\tau \ge \lambda$ problems (21) and (18) are equivalent in the following sense:*

(x^*, μ^*) is an optimal solution of (18) iff (x^*, μ^*, u^*) is an optimal solution of (21), where $u_i^* \in \left\{\frac{|x_i^*|}{M}, 1\right\}$ such that $\pi(u_i^*) = \psi_\theta(x_i^*)$ for $i = 1,\dots,n$..

Moreover, $\alpha(\tau) = \beta(\theta)$.

Proof If (x^*, μ^*, u^*) is an optimal solution of (21), then u_i^* is an optimal solution of the following problem, for every $i = 1, \dots, n$

$$\min \left\{ \pi(u_i) : \frac{|x_i^*|}{M} \le u_i \le 1 \right\}. \tag{22}$$

Since r is concave function, so is π. Consequently

$$\begin{aligned} \min \left\{ \pi(u_i) : \tfrac{|x_i^*|}{M} \le u_i \le 1 \right\} &= \min \left\{ \pi\left(\tfrac{|x_i^*|}{M}\right), \pi(1) \right\} \\ &= \min \left\{ \left(1 + \tfrac{\tau}{\lambda}\right) \tfrac{|x_i^*|}{M}, \tfrac{\tau}{\lambda} + \left(1 - \tfrac{\tau}{\lambda}\right) \tfrac{|x_i^*|}{M}, 1 \right\}. \end{aligned}$$

If $\tau \ge \lambda$, then for any $|t| \le M$ there holds

$$\frac{\tau}{\lambda} + \left(1 - \frac{\tau}{\lambda}\right) \frac{|t|}{M} \ge 1.$$

Thus, for $\tau \ge \lambda$ and $|x_i^*| \le M$, we have

$$\pi(u_i^*) = \min \left\{ \pi(u_i) : \frac{|x_i^*|}{M} \le u_i \le 1 \right\} = \min \left\{ \left(1 + \frac{\tau}{\lambda}\right) \frac{|x_i^*|}{M}, 1 \right\} = \psi_\theta(x_i^*). \tag{23}$$

For an arbitrary $(x, \mu) \in K$, we will show that

$$f(x^*, \mu^*) + \lambda \sum_{i=1}^n \psi_\theta(x_i^*) \le f(x, \mu) + \lambda \sum_{i=1}^n \psi_\theta(x_i). \tag{24}$$

By the assumption that (x^*, μ^*, u^*) is an optimal solution of (21), we have

$$f(x^*, \mu^*) + \lambda \sum_{i=1}^n \pi(u_i^*) \le f(x, \mu) + \lambda \sum_{i=1}^n \pi(u_i) \tag{25}$$

for any feasible solution (x, μ, u) of (21). Let

$$u_i^x \in \arg\min \left\{ \pi(\xi) \ : \xi \in \left\{ \frac{|x_i|}{M}, 1 \right\} \right\} \subset \arg\min \left\{ \pi(\xi) : \frac{|x_i|}{M} \le \xi \le 1 \right\},$$

for all $i = 1, \dots, n$. Then (x, μ, u^x) is a feasible solution of (19) and

$$\pi(u_i^x) = \min \left\{ \pi(\xi) : \frac{|x_i|}{M} \le \xi \le 1 \right\} = \psi_\theta(x_i), \quad \forall i = 1, \dots, n.$$

Combining (25) in which u_i is replaced by u_i^x and the last equation we get (24), which implies that (x^*, μ^*) is an optimal solution of (18).

Conversely, if (x^*, μ^*) is a solution of (18), and let $u_i^* \in \left\{ \frac{|x_i^*|}{M}, 1 \right\}$ such that $\pi(u_i^*) := u_i^* + \frac{\tau}{\lambda} r(u_i^*) = \psi_\theta(x_i^*)$ for $i = 1, \dots, n$. Then (x^*, μ^*, u^*) is a feasible solution of (21) and for an arbitrary feasible solution (x, μ, u) of (21), we have

$$\begin{aligned} f(x, \mu) + \lambda \textstyle\sum_{i=1}^n \left(u_i + \frac{\tau}{\lambda} r(u_i)\right) &\ge f(x, \mu) + \lambda \textstyle\sum_{i=1}^n \psi_\theta(x_i) \ge f(x^*, \mu^*) \\ &\quad + \lambda \textstyle\sum_{i=1}^n \psi_\theta(x_i^*) \\ &= f(x^*, \mu^*) + \lambda \textstyle\sum_{i=1}^n \left(u_i^* + \frac{\tau}{\lambda} r(u_i^*)\right). \end{aligned}$$

Thus, (x^*, μ^*, u^*) is an optimal solution of (21). The equality $\alpha(\tau) = \beta(\theta)$ is immediately deduced from the equality $\pi(u_i^*) = \psi_\theta(x_i^*)$.

From the two previous propositions we see that for $\tau > \max\{\lambda, \tau_0\}$ and $\theta = \frac{\tau+\lambda}{\lambda M}$, the approximate problem (18) is equivalent to the original problem (6). This result evidences the advantage of this polyhedral DC approximation of the zero-norm. It opens the door to other nonconvex approximation approaches which are consistent with the original problem.

We are going now to show how to solve the continuous exact reformulation of the ℓ_0-regularization (6), say the penalized problem (14), by DC programming and DCA.

3 Solving the continuous exact reformulation problem by DCA

For the reader's convenience we first give an brief introduction of DC programming and DCA.

3.1 Outline of DC programming and DCA

DC programming and DCA constitute the backbone of smooth/nonsmooth nonconvex programming and global optimization. They address general DC programs of the form:

$$\alpha = \inf\{f(x) := g(x) - h(x) : x \in \mathbb{R}^n\} \qquad (P_{dc})$$

where $g, h \in \Gamma_0(\mathbb{R}^n)$, the convex cone of all lower semicontinuous proper convex functions defined on $\mathbb{R}^n$ and taking values in $\mathbb{R} \cup \{+\infty\}$. Such a function f is called a DC function, and $g - h$ a DC decomposition of f while g and h are the DC components of f. The convex constraint $x \in C$ can be incorporated in the objective function of (P_{dc}) by using the indicator function of C denoted by χ_C which is defined by $\chi_C(x) = 0$ if $x \in C$, and $+\infty$ otherwise :

$$\inf\{f(x) := g(x) - h(x) : x \in C\} = \inf\{\chi_C(x) + g(x) - h(x) : x \in \mathbb{R}^n\}.$$

Polyhedral DC program is a DC program in which at least one of the functions g and h is polyhedral convex. The function φ is polyhedral convex if it is a pointwise supremum of a finite collection of affine functions. Polyhedral DC programming, which plays a central role in nonconvex optimization and global optimization and is the foundation of DC programming and DCA, has interesting properties (from both a theoretical and an algorithmic point of view) on local optimality conditions and the finiteness of DCA's convergence.

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

$$\partial\varphi(x_0) := \{y \in \mathbb{R}^n : \varphi(x) \geq \varphi(x_0) + \langle x - x_0, y\rangle, \forall x \in \mathbb{R}^n\}. \qquad (26)$$

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

The complexity of DC programs resides, of course, in the lack of practical optimal globality conditions. Local optimality conditions are then useful in DC programming.

A point x^* is said to be *a local minimizer* of $g - h$ if $g(x^*) - h(x^*)$ is finite and there exists a neighborhood $\mathcal{U}$ of x^* such that

$$g(x^*) - h(x^*) \leq g(x) - h(x), \quad \forall x \in \mathcal{U}. \qquad (27)$$

The necessary local optimality condition for (primal) DC program (P_{dc}) is given by

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*). \qquad (28)$$

The condition (28) is also sufficient (for local optimality) in many important classes of DC programs, for instance when (P_{dc}) is a polyhedral DC program with h being a convex polyhedral function (see Le Thi and Pham Dinh 1997; 2005).

A point x^* is said to be *a critical point* of $g - h$ if

$$\partial h(x^*) \cap \partial g(x^*) \neq \emptyset. \tag{29}$$

The relation (29) is in fact the generalized Karush-Kuhn-Tucker (KTT) condition for (P_{dc}) and x^* is also called a generalized KKT point.

Philosophy of DCA: DCA is based on local optimality conditions and duality in DC programming. 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 $y^l \in \partial h(x^l)$) and minimizes the resulting convex function.

The generic DCA scheme can be described as follows:

DCA scheme

Initialization: Let $x^0 \in \mathbb{R}^n$ be a guess, set l:=0.

Repeat

- Calculate some $y^l \in \partial h(x^l)$
- Calculate $x^{l+1} \in \arg\min\{g(x) - [h(x^l) + \langle x - x^l, y^l\rangle] : x \in \mathbb{R}^n\}$ (P_l)
- Increase l by 1

Until convergence of $\{x^l\}$.

Note that (P_l) is a convex optimization problem and is so far "easy" to solve.

Convergence properties of DCA and its theoretical basis can be found in (Le Thi 1997; Le Thi and Pham Dinh 1997, 2005; Pham Dinh and Le Thi 1998). For instance it is important to mention that (for the sake of simplicity we omit here the dual part of DCA).

- DCA is a descent method (the sequence $\{g(x^l) - h(x^l)\}$ is decreasing) without linesearch but with global convergence (i.e. convergence from every starting point).
- If $g(x^{l+1}) - h(x^{l+1}) = g(x^l) - h(x^l)$, then x^l is a critical point of $g - h$. In such a case, DCA terminates at l-th iteration.
- If the optimal value α of problem (P_{dc}) is finite and the infinite sequence $\{x^l\}$is bounded, then every limit point x^* of the sequence $\{x^l\}$ is a critical point of $g - h$.
- DCA has a *linear convergence* for DC programs.
- DCA has a *finite convergence* for polyhedral DC programs.

It is worth to noting that the construction of DCA involves DC components g and h but not the function f itself. Hence, for a DC program, each DC decomposition corresponds to a different version of DCA. Since a DC function f has an infinite number of DC decompositions which have crucial impacts on the qualities (speed of convergence, robustness, efficiency, globality of computed solutions,…) of DCA, the search of a "good" DC decomposition is important from an algorithmic point of view. How to develop an efficient algorithm based on the generic DCA scheme for a practical problem is thus a sensitive question to be studied. Generally, the answer depends on the specific structure of the problem being considered. The solution of a nonconvex program (P_{dc}) by DCA must be composed of two stages: the search of an *appropriate* DC decomposition of f and that of a *good* initial point.

DCA has been successfully used for various nonconvex optimization models, in particular those in machine learning (see the list of references in Le Thi's website and (Krause and Singer (2004); Le Thi et al. (2007); Liu et al. (2005)). It should be noted that

(i) the convex concave procedure (CCCP) for constructing discrete time dynamical systems mentioned in Yuille and Rangarajan (2003) is a special case of DCA applied to smooth optimization;

(ii) the SLA (Successive Linear Approximation) algorithm developed in Bradley and Mangasarian (1998) is a version of DCA for concave minimization;

(iii) the EM algorithm, (Dempster et al. 1997) applied to the log-linear model is a special case of DCA.

Last but not least, with appropriate DC decomposition in DC reformulations, DCA generates most of standard algorithms in convex/nonconvex programming.

For a complete study of DC programming and DCA the reader is referred to (Le Thi 1997; Le Thi and Pham Dinh 1997, 2005; Pham Dinh and Le Thi 1998) and the references therein. We show below how the DCA can be applied on the penalized problem (12).

3.2 DCA for solving the continuous exact reformulation problem (14)

We consider in the sequel the problem with a sufficient large number $\tau > \tau_0$:

$$\alpha(\tau) := \inf\{f(x,\mu) + \lambda e^T u + \tau p(u) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \leq M u_i,\ i = 1,\dots,n\}. \tag{30}$$

Let Δ be the feasible set of Problem (12), i.e. $\Delta := \{(x,\mu,u) : (x,\mu) \in K, u \in [0,1]^n, |x_i| \leq M u_i,\ i = 1,\dots,n\}$. Since f is convex and p is concave, the following DC formulation of (12) seems to be natural:

$$\inf\{g(x,\mu,u) - h(x,\mu,u) : (x,\mu,u) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^n, \tag{31}$$

where

$$g(x,\mu,u) := \chi_\Delta(x,\mu,u) + f(x,\mu), h(x,\mu,u) := -\lambda e^T u - \tau p(u)$$

are clearly convex functions. Moreover, since h is a polyhedral convex function, (31) is a polyhedral DC program.

According to the general DCA scheme described above, applying DCA to (31) amounts to computing two sequences $\{(x^l,\mu^l,u^l)\}$ and $\{(y^l,\upsilon^l,v^l)\}$ in the way that $(z^l,\upsilon^l,v^l) \in \partial h(x^l,\mu^l,u^l)$ and $(x^{l+1},\mu^{l+1},u^{l+1})$ solves the convex program of the form (P_l). Since $(y^l,\upsilon^l,v^l) \in \partial h(x^l,\mu^l,u^l) \Leftrightarrow y^l = 0,\ \upsilon^l = 0$ and

$$v_i^l = \begin{cases} -\lambda + \tau & \text{if } u_i^l \geq 0.5 \\ -\lambda - \tau & \text{if } u_i^l < 0.5 \end{cases}, i = 1,\dots n,$$

the algorithm can be described as follow.

DCAEP (DCA applied on Exact Penalty problem (30))

Initialization: Let $(x^0,\mu^0,u^0) \in \mathbb{R}^n \times \mathbb{R}^p \times [0,1]^n$ be a guess, set $l:=0$.

Repeat

- Set $v^l = (v_i^l)$ with $v_i^l = -\lambda + \tau$ if $u_i^l \geq 0.5$, $-\lambda - \tau$ otherwise, for $i = 1,\dots n$.
- Solve the convex program

$$\min\{f(x,\mu) - \langle u, v^l\rangle : (x,\mu,u) \in \Delta\} \tag{32}$$

  to obtain $(x^{l+1},\mu^{l+1},u^{l+1})$.
- Increase l by 1.

Until convergence of $\{(x^l,\mu^l,u^l)\}$.

Note that (32) is a linear (resp. convex quadratic) program with f is a linear (resp. quadratic) function. Note also that **DCAEP** has a finite convergence because that (30) with the DC decomposition (31) is a polyhedral DC program. The above exact penalty reformulation technique holds with another penalty function p, say $p(u) := \sum_{i=1}^n u_i(1-u_i)$. In this case DCA doesn't have the finite convergence (if f is not convex polyhedral function or Δ is not a polytope) since this function p is not polyhedral.

4 Application to feature selection in classification

4.1 DC formulation via exact penalty technique and DCA based algorithm

Feature selection is often applied to high-dimensional data prior to classification learning. The main goal is to select a subset of features of a given data set while preserving or improving the discriminative ability of the classifier.

Given a training data $\{\vartheta_i, \delta_i\}_{i=1,\ldots,m}$ where each $\vartheta_i \in \mathbb{R}^n$ is labeled by its class $\delta_i \in Y$. The goal of classification learning is to construct a classifier function that discriminates the data points $\{\vartheta_i\}_{i=1,\ldots,m}$ with respect to their classes$\{\delta_i\}_{i=1,\ldots,m}$. The embedded feature selection in classification consists of determining a classifier which uses as few features as possible, that leads to the following optimization problem like (4). In this section we focus on the context of Support Vector Machines (SVMs) learning with two-class linear models (Cristianini and Shawe-Taylor 2000). Generally, the problem can be formulated as follows.

Given two finite point sets $\mathscr{A}$ (with label $+1$) and $\mathscr{B}$ (with label -1) in $\mathbb{R}^n$ represented by the matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{k\times n}$, respectively, we seek to discriminate these sets by a separating plane ($x \in \mathbb{R}^n$, $\gamma \in \mathbb{R}$)

$$P = \{w \in \mathbb{R}^n : w^T x = \gamma\} \tag{33}$$

which uses as few features as possible. We adopt the notations introduced in Bradley and Mangasarian (1998) and consider the optimization problem proposed in Bradley and Mangasarian (1998) that takes the form:

$$\begin{array}{ll} \min_{x,\gamma,\xi,\zeta} & (1-\lambda)(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta) + \lambda \, \|x\|_0 \\ s.t. & -Ax + e\gamma + e \leq \xi, \; Bx - e\gamma + e \leq \zeta, \; \xi \geq 0, \; \zeta \geq 0. \end{array} \tag{34}$$

The nonnegative slack variables ξ_j, $j = 1, \ldots m$ represent the errors of classification of $\vartheta_j \in \mathscr{A}$ while ζ_j, $j = 1, \ldots k$ represent the errors of classification of $\vartheta_j \in \mathscr{B}$. More precisely, each positive value of ξ_j determines the distance between a point $a_j \in \mathscr{A}$ lying on the wrong side of the bounding plane $w^T x = \gamma + 1$ for $\mathscr{A}$. Similarly for ζ_j, $\mathscr{B}$ and $w^T x = \gamma - 1$. The first term of the objective function of (34) is the average error of classification, and the second term is the number of nonzero components of the vector x, each of which corresponds to a representative feature. Further, if an element of x is zero, the corresponding feature is removed from the dataset. Here λ is a control parameter of the trade-off between the training error and the number of features.

Observe that the problem (34) is a special case of (4) where the function f is given by

$$f(x, \gamma, \xi, \zeta) := (1-\lambda)\left(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta\right) \tag{35}$$

and K is a polytope defined by

$$K := \left\{(x, \gamma, \xi, \zeta) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^m_+ \times \mathbb{R}^k_+ : Ax + e\gamma + e \leq \xi, \; Bx - e\gamma + e \leq \zeta\right\}. \tag{36}$$

Applying the results developed in the previous section with f and K defined, respectively, in (35) and (36) we get the following DC formulation of (34):

$$\inf\{g(x, \gamma, \xi, \zeta, u) - h(x, \gamma, \xi, \zeta, u) : (x, \gamma, \xi, \zeta, u) \in \mathbb{R}^n \times \mathbb{R}^{1+m+k} \times \mathbb{R}^n\}, \tag{37}$$

where

$$g(x,\gamma,\xi,\zeta,u) := \chi_\Delta(x,\gamma,\xi,\zeta,u) + (1-\lambda)(\tfrac{1}{m}e^T\xi + \tfrac{1}{k}e^T\zeta),$$
$$h(x,\gamma,\xi,\zeta,u) := -\lambda e^T u - \tau p(u).$$

Since K is a polyhedral convex set, so is Δ, hence χ_Δ is a polyhedral convex function. Therefore (37) is a polyhedral DC program with both polyhedral DC components g and h. In the algorithm **DCAEP**, the convex program (32) becomes now a linear program.

DCAEP-SVM (DCA applied on Exact Penalty problem (37))

Initialization: Let $(x^0,\gamma^0,\xi^0,\zeta^0,u^0) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^m_+ \times \mathbb{R}^k_+ \times [0,1]^n$ be a guess, let $\epsilon > 0$ be sufficiently small, set $l{:=}0$.

Repeat

- Set $v^l = (v^l_i)$ with $v^l_i = -\lambda + \tau$ if $u^l_i \geq 0.5$, $-\lambda - \tau$ otherwise, for $i = 1,\ldots n$.
- Solve the linear program

$$\min\{(1-\lambda)(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta) - \langle u, v^l\rangle : (x,\gamma,\xi,\zeta,u) \in \Delta\} \tag{38}$$

  to obtain $(x^{l+1},\gamma^{l+1},\xi^{l+1},\zeta^{l+1},u^{l+1})$
- Increase l by 1

Until $\left\|(x^l,\gamma^l,\xi^l,\zeta^l,u^l) - (x^{l-1},\gamma^{l-1},\xi^{l-1},\zeta^{l-1},u^{l-1})\right\| \leq \epsilon \left\|(x^l,\gamma^l,\xi^l,\zeta^{l-1},u^l)\right\|$.

Thanks to the very special structure of (37) (f is a linear function and Δ is a polytope), **DCAEP-SVM** enjoys interesting convergence properties.

Theorem 1 (Convergence properties of DCAEP-SVM)

(i) ***DCAEP-SVM*** *generates a sequence* $\{(x^l,\gamma^l,\xi^l,\zeta^l,u^l)\}$ *contained in* $V(\Delta)$ *such that the sequence* $\{f(x^l,\gamma^l,\xi^l,\zeta^l) + \tau p(u^l)\}$ *is decreasing.*

(ii) For a number τ *sufficiently large, if at an iteration* q *we have* $u^q \in \{0,1\}^n$*, then* $u^l \in \{0,1\}^n$ *for all* $l \geq q$.

(iii) The sequence $\{(x^l,\gamma^l,\xi^l,\zeta^l,u^l)\}$ *converges to* $\{(x^*,\gamma^*,\xi^*,\zeta^*,u^*)\} \in V(\Delta)$ *after a finite number of iterations. The point* $(x^*,\gamma^*,\xi^*,\zeta^*,u^*)$ *is a critical point of Problem (37). Moreover if* $u^*_i \neq \frac{1}{2}$ *for all* $i = 1\ldots n$*, then* $\{(x^*,\gamma^*,\xi^*,\zeta^*,u^*)\}$ *is a local solution to (37).*

Proof (i) is consequence of DCA's convergence Theorem for a general DC program.

(ii) Let $\tau > \tau_1 := \max\left\{\frac{f(x,\gamma,\xi,\zeta)+\lambda e^T u-\eta}{\delta} : (x,\gamma,\xi,\zeta,u) \in V(\Delta),\, p(u) \leq 0\right\}$ where $\eta := \min\{f(x,\gamma,\xi,\zeta) + \lambda e^T u : (x,\gamma,\xi,\zeta,u) \in V(\Delta)\}$ and $\delta := \min\{p(u) : (x,\gamma,\xi,\zeta,u) \in V(\Delta)\}$. Let $\{(x^l,\gamma^l,\xi^l,\zeta^l,u^l)\} \subset V(\Delta)$ $(l \geq 1)$ be generated by **DCAEP-SVM**. If $V(\Delta) \subset \{\Delta \cap u \in \{0,1\}^n\}$, then the assertion is trivial. Otherwise, let $(x^l,\gamma^l,\xi^l,\zeta^l,u^l) \in \{\Delta \cap u \in \{0,1\}^n\}$ and $(x^{l+1},\gamma^{l+1},\xi^{l+1},\zeta^{l+1},u^{l+1}) \in V(\Delta)$ be an optimal solution of the linear program (38). Then from (i) of this theorem we have

$$f(x^{l+1},\gamma^{l+1},\xi^{l+1},\zeta^{l+1}) + \lambda e^T u^{l+1} + tp(u^{l+1}) \leq f(x^l,\gamma^l,\xi^l,\zeta^l) + \lambda e^T u^l + tp(u^l).$$

Since $p(u^l) = 0$, it follows

$$\begin{aligned}\tau p(u^{l+1}) &\leq f(x^l,\gamma^l,\xi^l,\zeta^l) + \lambda e^T u^l - f(x^{l+1},\gamma^{l+1},\xi^{l+1},\zeta^{l+1}) - \lambda e^T u^{l+1}\\ &\leq f(x^l,\gamma^l,\xi^l,\zeta^l) + \lambda e^T u^l - \eta.\end{aligned}$$

If $p(u^{l+1}) > 0$, then

$$\tau \leq \frac{f(x^l,\gamma^l,\xi^l,\zeta^l) + \lambda e^T u^l - \eta}{p(u^{l+1})} \leq \frac{f(x^l,\gamma^l,\xi^l,\zeta^l) + \lambda e^T u^l - \eta}{\delta} \leq \tau_1$$

which contradicts the fact that $\tau > \tau_1$. Therefore we have $p(u^{l+1}) = 0$.

(iii) Since (37) is a polyhedral DC program, **DCAEP-SVM** has a finite convergence, say, the sequence $\{(x^l, \gamma^l, \xi^l, \zeta^l, u^l)\}$ converges to a critical point $(x^*, \gamma^*, \xi^*, \zeta^*, u^*) \in V(\Delta)$ after a finite number of iterations. If $u_j^* \neq 1/2, \forall j \in 1..n$, then the function h is differentiable at $(x^*, \gamma^*, \xi^*, \zeta^*, u^*)$ and then the necessary local condition

$$\partial h(x^*, \gamma^*, \xi^*, \zeta^*, u^*) \subset \partial g(x^*, \gamma^*, \xi^*, \zeta^*, u^*)$$

holds. Since h is a polyhedral convex function, this subdifferential inclusion is also a sufficient local optimality condition, i.e. $(x^*, \gamma^*, \xi^*, \zeta^*, u^*)$ is a local minimizer of (37). The proof is then complete.

□

4.2 Computational experiments

To study the performances of our approach, we perform it on several datasets. Our experiment is composed of two parts. In the first one we consider the synthetic data and in the second we test on a collection of real-world datasets.

4.2.1 Datasets

Synthetic datasets

We generate the datasets such that among n features, there exists a subset of n_i features that define a subspace in which two classes can be discriminated (i.e. only n_i of n features are informative while the others are irrelevant). Thus we are available to evaluate the performance of the algorithms in terms of feature selection, not only on the sparsity but also on the correctness of the selected features. The data are generated in a similar way given in Rakotomamonjy et al. (2011). First, we randomly drawn a mean vector $\nu \in \{-1, 1\}^{n_i}$ and a $n_i \times n_i$ covariance matrix Σ from Wishart distribution. Then, the n_i informative features are generated from a multivariate Gaussian distribution $N(\nu, \Sigma)$ and $N(-\nu, \Sigma)$, respectively, for class $+1$ and -1. The $n - n_i$ remaining features (irrelevant features) follow an i.i.d Gaussian distribution $N(0, 1)$.

Real-world datasets Real-word datasets are taken from well-known UCI data repository and from challenging feature-selection problems of the NIPS 2003 datasets. Datasets from UCI repository include several problems of gene selection for cancer classification with standard public microarray gene expression datasets. Challenging NIPS 2003 datasets are known to be difficult and are designed to test various feature-selection methods using an unbiased testing procedure without revealing the labels of the test set. They contain a huge number of features while the number of examples in both training sets and test sets is small. In Table 1, the number of features, the number of points in training and test set of each dataset are given. The full description of each dataset can be found on the web site of UCI repository and NIPS 2003.

4.2.2 Set up experiments and Parameters

All algorithms were implemented in the Visual C++ 2008, and performed on a PC Intel i5 CPU650, 3.2 GHz of 4GB RAM. CPLEX 12.2 was used for solving linear/quadratic programs. We stop all algorithms with the tolerance $\epsilon = 10^{-4}$. The non-zero elements of x are determined according to whether $|x_i|$ exceeds a small threshold (10^{-5}).

Table 1 Datasets

Data	# features	# points in training set	# points in test set
Ionosphere	34	234	117
WPBC (24 months)	32	104	51
WPBC (60 months)	32	380	189
Breast cancer	24,481	78	19
Leukemia	7,129	38	34
Arcene	10,000	100	100
Gisette	5,000	6,000	10,000
Prostate	12,600	102	21

We used the following set of candidate values for the parameter λ in our experiments $\{0.001, 0.002, 0.003, 0.004, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$.

Concerning the parameter τ, as τ_0 is not computable, we take a quite large value τ_0 at the beginning and use an adaptive procedure described in Pham Dinh and Le Thi (2014) for updating τ during the DCA scheme.

We compare the performance of algorithms in terms of the following three criteria: the percentage of well classified objects (PWCO), the number and percentage of selected features and CPU Time in seconds. POWC$_1$ (resp. POWC$_2$) denotes the POWC on training set (resp. test set). In addition, for the synthetic data we examine how the algorithms retrieve the informative features.

4.2.3 Comparative algorithms

We will compare our exact approach with some algorithms in convex and nonconvex approximation approaches. In convex regularization approaches we consider the well known ℓ_1-regularization (Tibshirani 1996) and Elastic Net (Zou and Hastie 2005) (ℓ_1-regularization and Elastic Net for SVM have been proposed, respectively in Bradley and Mangasarian (1998) (Appendix A) and Wang et al. (2006) (Appendix B)). Among usual sparse inducing functions in nonconvex approximation approaches the capped ℓ_1 (Peleg and Meir 2008) (the polyhedral DC approximation discussed in Sect. 2), the piecewise concave exponential (Bradley and Mangasarian 1998) and SCAD (Fan and Li 2001) approximations have been proved to be the most efficient in several works of various authors. As we have proved in Sect. 2 that the capped ℓ_1 is equivalent to our exact formulation with suitable parameters, we exclude it from our comparison and focus on the piecewise concave exponential and SCAD approximations. The first algorithm based on the piecewise concave exponential approximation is the SLA (Successive Linear Approximation) (Bradley and Mangasarian 1998) (which is in fact a version of DCA). The DCA based algorithm using this approximation (but with another DC decomposition) in Le Thi et al. (2008) (see Appendix C) has been shown to be more efficient than SLA, hence we consider it in our comparative experiments. Likewise, for the SCAD approximation we consider the DCA based algorithm developed in Le Thi et al. (2009) (Appendix D) which is less expensive than the LQA (Local Quadratic Approximation) algorithm proposed in Fan and Li (2001) and used for feature selection in SVM in Zhang et al. (2006) (subproblems are quadratic programs).

The comparative algorithms are named as follows:

- ℓ_1**-SVM**: SVM with ℓ_1 regularization (Bradley and Mangasarian 1998);

- **ElasticNet-SVM**: SVM with Elastic net regularization (Wang et al. 2006);
- **DCA-PiE-SVM**: DCA for piecewise exponential approximation (Le Thi et al. 2008);
- **DCA-SCAD-SVM**: DCA for SCAD approximation (Le Thi et al. 2009);
- **DCAEP-SVM**: the algorithm proposed in this paper.

4.2.4 Experiment on synthetic data

We set the sample sizes of training and test set to 500 and 10, 000, respectively. For each experimental setting (n, n_i) (number total of features, number of informative features), 50 training sets and 1 test set are generated. For each training set, we performed 5-folds cross-validation to choose the best parameters of each algorithm. Then, for each experimental setting (n, n_i), we summarize in Table 2 the average of accuracy, the average of number of selected features,the average of CPU time as well as the percentage of success of 50 runs over 50 training sets. A success means the considered algorithm retrieves exactly the n_i informative features and suppresses all irrelevant features.

We observe from the Table 2 that

- In terms of feature selection, **DCAEP-SVM** and **DCA-PiE-SVM** give the best results on all three experimental settings (n, n_i) (**DCA-PiE-SVM** is slightly better on the last dataset ($n = 50, n_i = 10$). Moreover, DCA based algorithms are more succes than the convex regularization approaches when retrieving the informative features. The percentage of success of DCA based algorithm varies from 84 to 94%, while that of ℓ_1**-SVM** (resp. **ElasticNet-SVM**) goes from 72 % to 81 % (resp. 68 % to 85 %).
- As for the accuracy of classification, the results are comparable. All five algorithms furnish quite good accuracy, more than 85 % correctness.

4.2.5 Experiment on real-world data

For each algorithm, we first use a 10-folds cross-validation to determine the best set of parameter values. Afterward, we perform, with these parameter values a 5-folds cross-validation and report the average and the standard deviation of each evaluation criterion. The comparative results are given in Table 3.

Comments on numerical results:

- Concerning the sparsity of solution (the number of selected feature), as above, **DCAEP-SVM** and **DCA-PiE-SVM** are the best: averagely, only 5 % and 4.6 % of features are selected, respectively. All DCA based algorithms perform better than ℓ_1**-SVM** and **ElasticNet-SVM**, especially on *Gisette* and *Breast*. All DCA based algorithms suppress considerably the number of features (up to 99 % on large datasets such as Arcene and Leukemia) while the correctness of classification is quite good (from 77 % to 100 %). For WPBC(60) and Prostate, **DCAEP-SVM** suppresses more features than the other algorithms while furnishing a better classification accuracy. On other datasets, **DCAEP-SVM** selects slightly more features than **DCA-PiE-SVM** (1 or 2 features, except for *Gisette*). Overall, **DCAEP-SVM** realizes a better trade-off between accuracy and sparsity than other algorithms.
- As for the accuracy of classification, **DCAEP-SVM** is the best for 6 out of 8 training sets. The gain is important on 2 datasets: *WPBC(24)* 10, 4 % and *Gisette* 12, 1 %. The same conclusion goes for test sets, **DCAEP-SVM** is better on 6 datasets (with a gain up to 17.1 % on *Gisette* dataset). **ElasticNet-SVM** is slightly better than DCA based algorithms

Table 2 Comparative results on synthetic datasets

Dataset		ℓ_1-SVM	ElasticNet -SVM	DCA-SCAD -SVM	DCA-PiE -SVM	DCAEP-SVM
SynData1	% of success	74	68	84	**92**	**92**
($n = 10, n_i = 2$)	POWC$_1$	94.2 ±1.3	95.2 ±1.5	96.2 ±1.5	96.5 ±1.2	**96.8 ±1.1**
	POWC$_2$	93.3 ±0.9	94.5 ±1.2	93.3 ±1.1	**94.3 ±1.1**	93.1 ±1.1
	FS	3.2 (32.0 %)	3 (30.0 %)	2.4 (24.0 %)	**2.3 (23.0 %)**	**2.3 (23.0 %)**
	CPU	**0.01**	0.02	0.02	0.02	0.02
SynData2	% of success	81	85	93	**94**	**94**
($n = 20, n_i = 5$)	POWC$_1$	90.2 ±1.5	93.1 ±1.4	**94.2 ±1.2**	93.4 ±1.2	93.9 ±1.2
	POWC$_2$	**92.3 ±1.1**	91.3 ±1.1	90.3 ±1.4	91.4 ±1.2	92.0 ±1.3
	FS	7.6 (38.0 %)	7.3 (36.5 %)	5.2 (26.0 %)	**4.9 (24.5 %)**	**4.9 (24.5 %)**
	CPU	**0.01**	0.02	0.02	0.02	0.02
SynData3	% of success	72	82	88	**92**	90
($n = 50, n_i = 10$)	POWC$_1$	85.6 ±1.2	**91.3 ±1.2**	90.2 ±1.7	91.2 ±1.2	**91.3 ±1.2**
	POWC$_2$	82.3 ±1.1	84.3 ±1.2	**85.3 ±1.4**	84.4 ±1.6	84.3 ±1.5
	FS	15.4 (30.8 %)	14.7 (29.4 %)	12.3 (24.6 %)	**11.3 (22.6 %)**	11.6 (23.2 %)
	CPU	**0.02**	0.03	0.04	0.04	0.04

Table 3 Comparative results on real-world datasets

Dataset	ℓ_1-SVM	ElasticNet -SVM	DCA-SCAD-SVM	DCA-PiE-SVM	DCAEP-SVM
Ionosphere					
POWC$_1$	85.6 ±1.2	**86.3 ±1.6**	81.5 ±1.4	82.1 ±3.1	85.2 ±1.2
POWC$_2$	81.2 ±1.1	80.3 ±1.4	73.5 ±1.2	82.3 ±2.2	**83.4 ±1.5**
FS	10.9 (32.1 %)	9.4 (27.6 %)	3.1 (9.1 %)	**2.3 (6.8 %)**	3.1 (9.1 %)
Time	**0.01**	0.04	0.3	0.3	0.2
WPBC(24)					
POWC$_1$	74.8±1.2	76.5±1.2	77.8 ±1.3	78.3 ±1.5	**85.2 ±1.2**
POWC$_2$	77.1±1.5	78.9±1.3	79.2 ±1.2	82.3 ±1.2	**83.5 ±1.2**
FS	9.1 (28.4 %)	8.3 (25.9 %)	4.1 (12.8 %)	**3 (9.4 %)**	4.1 (12.8 %)
Time	**0.03**	0.07	0.2	0.2	0.2
WPBC(60)					
POWC$_1$	89.2 ±1.6	87.5 ±1.1	84.3 ±1.3	90.3 ±1.2	**92.5 ±1.5**
POWC$_2$	82.5 ±1.1	87.4 ±1.3	89.4 ±1.4	92.3 ±1.4	**94.2 ±1.2**
FS	8.3 (27.7 %)	7.9 (26.3 %)	5.2 (17.3 %)	5.2 (17.3 %)	**4.2 (14.0 %)**
Time	**0.02**	0.05	0.4	0.4	0.3
Breast					
POWC$_1$	94.2±1.2	**95.6±1.3**	91.4±1.4	91.5±1.5	93.8±1.1
POWC$_2$	68.1±1.3	**71.9±1.2**	69.4±1.6	70.3±1.4	71.5±1.7
FS	142.3 (0.6 %)	102.4 (0.4 %)	37.0 (0.2 %)	**17.0 (0.1 %)**	18.6 (0.1 %)
Time	**19**	45	31	25	25
Leukemia					
POWC$_1$	92.2±1.4	**100**	98.9±1.1	**100**	**100**
POWC$_2$	96.3±1.4	96.2±1.3	**97.3±3.2**	96.2±2.4	**97.4±1.2**
FS	12.0 (0.2 %)	10.2 (0.1 %)	**8.2 (0.1 %)**	8.3 (0.1 %)	15.3 (0.2 %)
Time	1	5	21	23	24
Arcene					
POWC$_1$	91.4±1.5	90.4±2.1	89..4±2.1	88.3±1.7	**96.5±3.7**
POWC$_2$	72.2±1.3	76.2±1.2	78.2±3.7	**78.9±1.4**	78.3±2.6
FS	81.3 (0.81 %)	92.3 (0.92 %)	32.5 (0.33 %)	**32.4 (0.32 %)**	32.8 (0.33 %)
Time	**10**	20	31	34	31
Gisette					
POWC$_1$	78.3±1.2	79.2±1.2	87.3±1.5	83.3±2.1	**90.4±1.2**
POWC$_2$	68.5±1.2	80.1±1.3	82.2±1.4	82.2±1.2	**85.6±1.3**
FS	1276.5 (25.5 %)	1034.4 (20.7 %)	**140.1 (2.8 %)**	**140.1 (2.8 %)**	165.5 (3.3 %)
Time	**45**	192	81	98	74
Prostate					
POWC$_1$	92.4±1.4	91.8±1.3	91.5±1.9	91.6±1.8	**94.4±1.3**
POWC$_2$	**100**	**100**	**100**	**100**	**100**
FS	56.4 (0.45 %)	45.4 (0.36 %)	34.2 (0.27 %)	32.1 (0.25 %)	**30.8 (0.24 %)**
Time	**14**	16	16	16	18

Table 3 continued

Dataset	ℓ_1-SVM	ElasticNet -SVM	DCA-SCAD-SVM	DCA-PiE-SVM	DCAEP-SVM
Average					
POWC$_1$	87.2	88.4	87.7	88.1	**92.2**
POWC$_2$	80.7	83.9	83.3	85.8	**86.6**
FS(%)	14.4 %	12.8 %	5.5 %	4.6 %	5.0 %
Time	**11.1**	34.7	22.6	24.4	21.5

(1.1 % and 1.8 % on two datasets *Breast* and *Ionosphere*. This can be explained by the fact that **ElasticNet-SVM** selects 6 (resp. 4) times more features than DCA based algorithms on *Breast* (resp. *Ionosphere*) dataset.

- In terms of CPU time, not surprisingly, ℓ_1**-SVM** is the fastest algorithm, with an average of CPU time 11, 1 s, since it only requires solving one linear program. The CPU time of DCA based algorithms is quite small, less than 101 s for the largest dataset (Gisette). *DCAEP-SVM* is somehow slightly faster with an average of CPU time 21, 5 s while that of *DCA- PiE-SVM* (resp. **DCA-SCAD-SVM**) is 24, 6 (resp. 22, 6) s.

5 Conclusion

We have proposed an exact reformulation approach based on DC programming and DCA for minimizing a class of functions involving the zero-norm and its application on feature selection in classification. Using a recent result on exact penalty in DC programming we show that the original problem (4) can be equivalently reformulated as a continuous optimization problem which is a DC program. By this result we can unify all nonconvex approaches for treating the zero-norm into the DC programming and DCA. The link between the exact reformulation and convex/nonconvex approximations stated in this paper allows to analyze / justify the performance of these approximations approaches. Numerical experiments on feature selection in SVM show that our algorithm is efficient on both feature selection and classification. The advantage of this approach is that it solves directly an equivalent model of the original problem. Several issues arise from this work. Firstly, the choice of a good exact penalty parameter is still open. Secondly, the link between the exact reformulation and the polyhedral approximation suggests us to study new approximations such that the exact penalty reformulation are equivalent to the approximate problem for which efficient DCA schemes can be investigated. Thirdly, we should extend our exact penalty approaches for larger classes of problems (when f nonconvex for example) as well as other applications including regression, sparse Fisher linear discriminant analysis, feature selection in learning to rank with sparse SVM, compressed sensing, etc. Works in these directions are in progress.

Appendix:

ℓ_1 regularization for SVM (Bradley and Mangasarian (1998))

ℓ_1**-SVM** formulation:

$$\begin{array}{ll} \min_{x,\gamma,\xi,\zeta} & (1-\lambda)(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta) + \lambda\,\|x\|_1 \\ s.t. & -Ax + e\gamma + e \le \xi,\ Bx - e\gamma + e \le \zeta,\ \xi \ge 0,\ \zeta \ge 0. \end{array} \qquad (39)$$

By adding a new variable $z \in \mathbb{R}^n_+$, we obtain the following equivalent linear program:

$$\begin{array}{ll} \min_{x,\gamma,\xi,\zeta,z} & (1-\lambda)\left(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta\right) + \lambda e^T z \\ s.t. & -Ax + e\gamma + e \leq \xi,\ Bx - e\gamma + e \leq \zeta, \\ & -z_i \leq x_i \leq z_i,\ \forall i = 1,\ldots,n, \\ & \xi \geq 0,\ \zeta \geq 0,\ z \geq 0. \end{array} \tag{40}$$

Elastic Net regularization for SVM (Wang et al. (2006))

Elastic Net regularization was introduced by Zou and Hastie (Zou and Hastie 2005) in the context of regression. Later, in Wang et al. (2006), the authors proposed to use *Elastic net* regularization for feature selection in SVM. The problem can be described as follows (Wang et al. 2006)

$$\begin{array}{ll} \min_{x,\gamma,\xi,\zeta} & \frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta + \lambda_1 \|x\|_1 + \frac{\lambda_2}{2}\|x\|_2^2 \\ s.t. & -Ax + e\gamma + e \leq \xi,\ Bx - e\gamma + e \leq \zeta,\ \xi \geq 0,\ \zeta \geq 0. \end{array} \tag{41}$$

with $\lambda_1, \lambda_2 > 0$. Similarly to SVM using ℓ_1, (41) is equivalent to the following convex quadratic problem:

$$\begin{array}{ll} \min_{x,\gamma,\xi,\zeta,t} & \frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta + \lambda_1 e^T z + \frac{\lambda_2}{2}\|x\|_2^2 \\ s.t. & -Ax + e\gamma + e \leq \xi,\ Bx - e\gamma + e \leq \zeta, \\ & -z_i \leq x_i \leq z_i,\ \forall i = 1,\ldots,n, \\ & \xi \geq 0,\ \zeta \geq 0,\ z \geq 0. \end{array} \tag{42}$$

Piecewise concave exponential approximation (Le Thi et al. (2008))

For $y \in \mathbb{R}$, let η_1 be the function defined by

$$\eta_1(t) = \begin{cases} 1 - \varepsilon^{-\alpha t} & \text{if } t \geq 0 \\ 1 - \varepsilon^{\alpha t} & \text{if } t \leq 0 \end{cases} \tag{43}$$

with $\alpha > 0$. Hence, for all $x \in \mathbb{R}^n$, the step vector $|x|_0$ is approximated by $|x|_0 \simeq \eta_1(x_i)$ and the approximation of the zero-norm $\|x\|_0$ is determined as

$$\|x\|_0 \simeq \sum_{i=1}^{n} \eta_1(x_i).$$

This approximation has been proposed for the first time in Bradley and Mangasarian (1998) for feature selection in SVM where the authors developed a SLA (Sucessive Linear Approximation) algorithm for the resulting approximate problem (called FSV algorithm). Later, in Le Thi et al. (2008) , the authors proposed another DC decomposition for which the follwing DCA scheme has been investiaged. Numerical experiments in Le Thi et al. (2008) showed that this new algorithm is better than FSV which is also an instance of DCA.

DCA-PiE-SVM:

Initialization Let ϵ be a tolerance sufficiently small, set $l = 0$.

Choose $(x^0, \gamma^0, \xi^0, \zeta^0) \in \mathbb{R}^{n+1m+k+n}$.

Repeat

- Compute $\bar{x}^l$ as follows

$$\bar{x}_j = \begin{cases} \alpha(1-\varepsilon^{-\alpha x_j}) & \text{if } x_j \geq 0 \\ -\alpha(1-\varepsilon^{\alpha x_j}) & \text{if } x_j < 0 \end{cases} \quad , j = 1, \ldots, n. \tag{44}$$

- Solve the linear program

$$\begin{cases} \min(1-\lambda)\left(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta\right) + \lambda \sum_{i=1}^{n} t_j - \langle \bar{x}^l, x\rangle \\ s.t \quad (x, \gamma, \xi, \gamma) \in K \\ \qquad -\alpha w_j \leq t_j, \alpha w_j \leq t_j, j = 1..n \end{cases} \tag{45}$$

  to obtain $(x^{l+1}, \gamma^{l+1}, \xi^{l+1}, \zeta^{l+1})$
- Increase l by 1

Until

$$\left\|(x^l, \gamma^l, \xi^l, \zeta) - (x^{l-1}, \gamma^{l-1}, \xi^{l-1}, \zeta^{l-1})\right\| \leq \left\|\epsilon(x^l, \gamma^l, \xi^l, \zeta)\right\|.$$

Remember that K is a polytope defined by

$$K := \left\{(x, \gamma, \xi, \zeta) \in \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^m_+ \times \mathbb{R}^k_+ : Ax + e\gamma + e \leq \xi,\; Bx - e\gamma + e \leq \zeta\right\}. \tag{46}$$

For more detail about **DCA-PiE-SVM** and its convergence properties, the reader is referred to Le Thi et al. (2008).

SCAD approximation (Le Thi et al. (2009))

The SCAD (Smoothly Clipped Absolute Deviation) penalty function has been proposed for the first time by J. Fan and R. Li (Fan and Li 2001) in the context of regression and variable selection. The SCAD penalty function is expressed as follow, for $t \in \mathbb{R}$:

$$\omega(|t|) = \begin{cases} \beta|t| & \text{if} \quad |t| \leq \beta, \\ -\frac{|t|^2 - 2\alpha\beta|t| + \beta^2}{2(\alpha-1)} & \text{if} \quad \beta < |t| \leq \alpha\beta, \\ \frac{(\alpha+1)\beta^2}{2} & \text{if} \quad |t| > \alpha\beta, \end{cases} \tag{47}$$

where $\alpha > 2$ and $\beta > 0$ are two tuning parameters.

In Zhang et al. (2006), using the SCAD approximation, a local quadratic approximation has been proposed for features selection in SVM. This algorithm can be regarded as a reweighted l_2 procedure which is in fact a version of DCA. Later, in Le Thi et al. (2009), using a suitable DC decompostion of SCAD function the authors developed an efficient DCA based algorithm that requires one linear program at each iteration. The algorithm can be described as follow.
DCA-SCAD-SVM:
Initialization Let τ be a tolerance sufficiently small, set $l = 0$.
Choose $(x^0, \gamma^0, \xi^0, \zeta^0) \in \mathbb{R}^{n+1m+k+n}$.
Repeat

- Compute $\bar{x}^l$ as follows

$$\bar{x}_j = \begin{cases} 0 & \text{if} \quad -\beta \le x_j \le \beta \\ (\alpha-1)^{-1}(x_j-\beta) & \text{if} \quad \beta < x_j \le \alpha\beta \\ (\alpha-1)^{-1}(x_j+\beta) & \text{if} \quad -\alpha\beta < x_j \le -\beta \\ \beta & \text{if} \quad x_j > \alpha\beta \\ -\beta & \text{if} \quad x_j < -\alpha\beta \end{cases} \quad, j = 1, \ldots n, \tag{48}$$

- Solve the linear program

$$\begin{cases} \min(1-\lambda)\left(\frac{1}{m}e^T\xi + \frac{1}{k}e^T\zeta\right) + \lambda \sum_{i=1}^{n} t_j - \langle \bar{x}^l, x\rangle \\ s.t \quad (x, \gamma, \xi, \gamma) \in K \\ \qquad -\alpha w_j \le t_j, \alpha w_j \le t_j, \; j = 1..n \end{cases} \tag{49}$$

  to obtain $(x^{l+1}, \gamma^{l+1}, \xi^{l+1}, \zeta^{l+1})$
- Increase l by 1

Until
$$\left\|(x^l, \gamma^l, \xi^l, \zeta) - (x^{l-1}, \gamma^{l-1}, \xi^{l-1}, \zeta^{l-1})\right\| \le \left\|\epsilon(x^l, \gamma^l, \xi^l, \zeta)\right\|.$$

The reader is referred to Le Thi et al. (2009) for more details.

References

Amaldi, E., & Kann, V. (1998). On the approximability of minimizing non zero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, *209*, 237–260.

Bach, F., Jenatton, R., Mairal, J., & Obzinski, G. (2012). Optimization with sparsity-inducing penalties foundations and trends. *Foundations and Trends in Machine Learning*, *4*(1), 1–106.

Bradley, P. S., & Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. In *Proceeding of international conference on machine learning ICML'98*.

Candes, E., Wakin, M., & Boyd, S. (2008). Enhancing sparsity by reweighted l_1 minimization. *Journal of Mathematical Analysis and Applications*, *14*, 877–905.

Chartrand, R., & Yin, W. (2008). Iteratively reweighted algorithms for compressive sensing. *Acoustics, speech and signal processing, IEEE international conference ICASSP*, *2008*, 3869–3872.

Chen, X., Xu, F. M., & Ye, Y. (2010). Lower bound theory of nonzero entries in solutions of l2-lp minimization. *SIAM Journal on Scientific Computing*, *32*(5), 2832–2852.

Chen, Y., Li, Y., Cheng, X.-Q., & Guo, L. (2006). Survey and taxonomy of feature selection algorithms in intrusion detection system. In *Proceedings of inscrypt, 2006*. LNCS (Vol. 4318, 153–167).

Collober, R., Sinz F., Weston, J., & Bottou, L. (2006). Trading convexity for scalability. In *Proceedings of the 23rd international conference on machine learning ICML 2006* (pp. 201–208). Pittsburgh, PA. ISBN:1-59593-383-2.

Cristianini, N., & Shawe-Taylor, N. (2000). *Introduction to support vector machines*. Cambridge: Cambridge University Press.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1997). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, *39*, 1–38.

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, *96*(456), 1348–1360.

Fu, W. J. (1998). Penalized regression: The bridge versus the lasso. *Journal of Computational and Graphical Statistics*, *7*, 397–416.

Gasso, G., Rakotomamonjy, A., & Canu, S. (2009). Recovering sparse signals with a certain family of nonconvex penalties and dc programming. *IEEE Transactions on Signal Processing*, *57*, 4686–4698.

Gorodnitsky, I. F., & Rao, B. D. (1997). Sparse signal reconstructions from limited data using FOCUSS: A re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, *45*, 600–616.

Guan, W., & Gray, A. (2013). Sparse high-dimensional fractional-norm support vector machine via DC programming. *Computational Statistics and Data Analysis*, *67*, 136–148.

Gribonval, R., & Nielsen, M. (2003). Sparse representation in union of bases. *IEEE Transactions on Information Theory*, *49*, 3320–3325.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning* (2nd ed.). Heidelberg: Springer.

Huang, J., Horowitz, J., & Ma, S. (2008). Asymptotic properties of bridge estimators in sparse high-dimensional regression models. *Annals of Statistics*, *36*, 587–613.

Kim, Y., Choi, H., & Oh, H. S. (2008). Smoothly clipped absolute deviation on high dimensions. *Journal of the American Statistical Association*, *103*(484), 1665–1673.

Knight, K., & Fu, W. (2000). Asymptotics for lasso-type estimators. *Annals of Statistics*, *28*, 1356–1378.

Krause, N., & Singer, Y. (2004). Leveraging the margin more carefully. In *Proceedings of the 21 international conference on Machine learning ICML 2004*. Banff, Alberta, Canada, 63.ISBN:1-58113-828-5.

Le Thi, H.A. DC Programming and DCA. http://lita.sciences.univ-metz.fr/~lethi.

Le Thi, H. A. (1997). *Contribution à l'optimisation non convexe et l'optimisation globale: Théorie*. Algorithmes et Applications: Habilitation à Diriger des Recherches, Université de Rouen.

Le Thi, H. A., & Pham Dinh, T. (1997). Solving a class of linearly constrained indefinite quadratic problems by DC algorithms. *Journal of Global Optimization*, *11*(3), 253–285.

Le Thi, H. A., & Pham Dinh, T. (2005). The DC (difference of convex functions) programming and DCA revisited with DC models of real-world nonconvex optimization problems. *Annals of Operations Research*, *133*, 23–46.

Le Thi, H. A., Belghiti, T., Pham Dinh, T. (2007) A new efficient algorithm based on DC programming and DCA for clustering. *Journal of Global Optimization*, *37*, 593–608.

Le Thi, H. A., Le, H. M. & Pham Dinh, T. (2006). Optimization based DC programming and DCA for hierarchical clustering. *European Journal of Operational Research*, *183*(3), 1067–1085.

Le Thi, H. A., Le, H. M., Nguyen, V. V., & Pham Dinh, T. (2008). A dc programming approach for feature selection in support vector machines learning. *Journal of Advances in Data Analysis and Classification*, *2*, 259–278.

Le Thi, H. A., Nguyen, V. V., & Ouchani, S. (2009). Gene selection for cancer classification using DCA. *Journal of Fonctiers of Computer Science and Technology*, *3*(6), 62–72.

Le Thi, H. A., Huynh, V. N., & Pham Dinh, T. (2012). Exact penalty and error bounds in DC programming. *Journal of Global Optimization dedicated to Reiner Horst*, *52*(3), 509–535.

Liu, Y., Shen, X., & Doss, H. (2005). Multicategory ψ-learning and support vector machine: Computational tools. *Journal of Computational and Graphical Statistics*, *14*, 219–236.

Liu, Y., & Shen, X. (2006). Multicategoryψ-learning. *Journal of the American Statistical Association*, *101*, 500–509.

Mangasarian, O. L. (1996). Machine learning via polyhedral concave minimization. In H. Fischer, B. Riedmueller, & S. Schaeffler (Eds.), *Applied mathematics and parallel computing—Festschrift for Klaus Ritter* (pp. 175–188). Heidelberg: Physica.

Mallat, S., & Zhang, Z. (1993). Matching pursuit in a time-frequency dictionary. *IEEE Transactions on Signal Processing*, *41*(12), 3397–3415.

Meinshausen, N. (2007). Relaxed Lasso. *Computational Statistics and Data Analysis*, *52*(1), 374–393.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, *24*, 227–234.

Neumann, J., Schnörr, C., & Steidl, G. (2005). Combined SVM-based feature selection and classification. *Machine Learning*, *61*(1–3), 129–150.

Ong, C. S., & Le Thi, H. A. (2013). Learning sparse classifiers with Difference of Convex functions algorithms. *Optimization Methods and Software*, *28*(4), 830–854.

Peleg, D., & Meir, R. (2008). A bilinear formulation for vector sparsity optimization. *Signal Processing*, *8*(2), 375–389.

Pham Dinh, T., & Le Thi, H. A. (1998). DC optimization algorithms for solving the trust region subproblem. *SIAM Journal on Optimization*, *8*, 476–505.

Pham Dinh, T., & Le Thi, H. A (2014). Recent advances in DC programming and DCA. Transactions on Computational Collective. *Intelligence.*, *8342*, 1–37.

Rakotomamonjy, A., Flamary, R., Gasso, G., & Canu, S. (2011). $\ell_p - \ell_q$ penalty for sparse linear and sparse multiple kernel multi-task learning. *IEEE Transactions on Neural Networks*, *22*(8), 13071320.

Rao, B. D., & Kreutz-Delgado, K. (1999). An affine scaling methodology for best basis selection. *IEEE Transactions on Signal Processing*, *47*, 187–200.

Rao, B. D., Engan, K., Cotter, S. F., Palmer, J., & KreutzDelgado, K. (2003). Subset selection in noise based on diversity measure minimization. *IEEE Transactions on Signal Processing*, *51*(3), 760–770.

Rinaldi, F. (2000). Mathematical Programming Methods for minimizing the zero-norm over polyhedral sets, PhD thesis, Sapienza, University of Rome (2009)

Thiao, M., Pham Dinh, T., & Le Thi, H. A. (2010). A DC programming approach for sparse eigenvalue problem. *Proceeding of ICML*, *2010*, 1063–1070.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, *46*, 431–439.

Yuille, A. L., & Rangarajan, A. (2003). The convex concave procedure. *Neural Computation*, *15*(4), 915–936.

Wang, L., Zhu, J., & Zou, H. (2006). The doubly regularized support vector machine. *Statistica Sinica*, *16*, 589–615.

Weston, J., Elisseeff, A., Scholkopf, B., & Tipping, M. (2003). Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research.*, *3*, 1439–1461.

Zhang, H. H., Ahn, J., Lin, X., & Park, C. (2006). Gene selection using support vector machines with non-convex penalty. *Bioinformatics*, *2*(1), 88–95.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, *67*, 301–320.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, *101*, 1418–1429.

Zou, H., & Li, R. (2008). One-step sparse estimates in nonconcave penalized likelihood models. *Annals of Statistics*, *36*(4), 1509–1533.