Optim Lett (2015) 9:57–74
DOI 10.1007/s11590-014-0757-3

ORIGINAL PAPER

Solving ℓ_0-penalized problems with simple constraints via the Frank–Wolfe reduced dimension method

Giampaolo Liuzzi · Francesco Rinaldi

Received: 18 April 2013 / Accepted: 19 May 2014 / Published online: 6 June 2014

Abstract ℓ_0-penalized problems arise in a number of applications in engineering, machine learning and statistics, and, in the last decades, the design of algorithms for these problems has attracted the interest of many researchers. In this paper, we are concerned with the definition of a first-order method for the solution of ℓ_0-penalized problems with simple constraints. We use a reduced dimension Frank–Wolfe algorithm Rinaldi (Optim Methods Softw, 26, 2011) and show that the subproblem related to the computation of the Frank–Wolfe direction can be solved analytically at least for some sets of simple constraints. This gives us a very easy to implement and quite general tool for dealing with ℓ_0-penalized problems. The proposed method is then applied to the numerical solution of two practical optimization problems, namely, the Sparse Principal Component Analysis and the Sparse Reconstruction of Noisy Signals. In both cases, the reported numerical performances and comparisons with state-of-the-art solvers show the efficiency of the proposed method.

Keywords Sparse problems · Frank–Wolfe method · SPCA · Noisy signals

G. Liuzzi
Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti", Consiglio Nazionale delle Ricerche,
Viale Manzoni 30, 00185 Rome, Italy
e-mail: giampaolo.liuzzi@iasi.cnr.it

F. Rinaldi (✉)
Dipartimento di Matematica, Università degli Studi di Padova,
Via Trieste 63, 35121 Padua, Italy
e-mail: rinaldi@math.unipd.it

1 Introduction

In this paper we consider the following class of problems

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} g(x)+\lambda\|x\|_0 \\ &s.t. \quad x\in C, \end{aligned} \tag{1}$$

where λ is a positive parameter, $g(x)$ is a continuously differentiable function, C is a compact convex set and $\|x\|_0$ is the zero-norm of x, that is,

$$\|x\|_0 = card(\{i : x_i \neq 0\}).$$

In what follows, we require the set C to satisfy the assumption below.

Assumption 1 Set C is described by "simple" constraints. In particular:

(a) a spherical constraint, i.e. $C = \{x \in \mathbb{R}^n : \|x\|^2 \leq 1\}$;
(b) bound constaints on the variables, i.e. $C = \{x \in \mathbb{R}^n : -Me \leq x \leq Me\}$,

where $e \in \mathbb{R}^n$ is the vector of all ones, M is a sufficiently large positive value and the vector inequalities $-Me \leq x \leq Me$ are intended component-wise (namely, $-M \leq x_i \leq M$, for all $i = 1, \ldots, n$).

As shown in [4,24,25], because of the discontinuity and nonconvexity of the zero-norm function, finding a solution of Problem (1) can be a very difficult task.

Despite our assumption on set C, Problem (1) is still sufficiently general to encompass different practical optimization problems such as

- the Sparse Principal Component Analysis (SPCA) [12,19,33] and
- the Sparse Reconstruction of Noisy Signals (SRNS) [1,2,13,15].

In the paper, following the ideas proposed in [22,26,31], we choose to deal with Problem (1) by replacing the zero-norm $\|x\|_0$ with a suitable smooth concave approximating function $h(y) : \mathbb{R}^n \to \mathbb{R}$, thus obtaining the following problem.

$$\begin{aligned} &\min_{x,y\in\mathbb{R}^n} f(x) = g(x)+\lambda h(y) \\ &s.t. \quad x\in C, \\ &\qquad -y \leq x \leq y. \end{aligned} \tag{2}$$

As proposed in [22,26,31], possible expressions for $h(y)$ are the following:

$$h(y) = \sum_{i=1}^{n}(1-e^{-\alpha y_i}), \quad \alpha > 0; \tag{3}$$

$$h(y) = \sum_{i=1}^{n}\ln(\epsilon + y_i), \quad \epsilon > 0; \tag{4}$$

$$h(y) = \sum_{i=1}^{n}(y_i+\epsilon)^p, \quad \epsilon > 0,\ 0 < p < 1; \tag{5}$$

$$h(y) = \sum_{i=1}^{n} -(y_i + \epsilon)^{-p}, \quad \epsilon > 0, \ p \geq 1. \tag{6}$$

We then solve Problem (7) employing the Frank–Wolfe (FW) algorithm proposed in [25]. To this aim, under Assumption 1, we show that the FW direction can be computed analytically thus making the algorithms very efficient and competitive.

Hence, the aim of the paper is to adapt the method proposed in [25] for solving two well-known and widely studied ℓ_0-penalized problems with simple constraints, namely SPCA and SRNS. As we will see later, the resulting algorithm shows good performance when compared to special-purpose codes for the considered problems. Furthermore it is very easy to implement (just a few lines of Matlab code).

The paper is organized as follows. In Sect. 2 we present the general Frank–Wolfe method that we use to solve the approximating problem (2). In Sect. 3, we show how to analytically compute the FW direction for problems satisfying Assumption 1. In Sect. 4 we briefly present the SPCA problem. Section 5 is devoted to the SRNS problem. In Sects. 6 and 7 we report some numerical results of the proposed method and comparison with some other well-known codes for SPCA (namely GPower [19] and SpaSM [17]) and SRNS (namely FPC-AS [15], SALSA [1,2] and SpaRSA [13]), respectively, which show the viability and efficiency of the proposed approach. Finally, in Sect. 8 we draw some conclusions.

2 The Frank–Wolfe reduced dimension algorithm

The Frank–Wolfe algorithm is a well-known and widely-used method in operations research. It was originally proposed by Marguerite Frank and Phil Wolfe in 1956 as a procedure for solving quadratic programming problems with linear constraints [14]. At each step of the algorithm the objective function is linearized and a step is taken along a feasible descent direction. Recently, the approach has been successfully used for finding sparse solutions to problems with convex constraints (see e.g. [21,22,25,26,31]).

In this section, we describe an efficient version of the Frank–Wolfe algorithm for solving problem (2) and recall some theoretical results about its global convergence (see [25] for further details and proofs). The motivation for using this algorithm as a local minimizer is twofold:

(1) The optimization problem to be solved at each step of the algorithm has a linear objective function and a closed convex feasible set described by simple constraints such that its solution can be computed analytically.
(2) It is possibile to reduce the problem dimension at each step of the algorithm, thus obtaining significant savings in terms of computational time.

In order to ease the description of the algorithm we restate problem (2) as follows:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f(x) = g(x) + \lambda h(x) = g(x) + \lambda \sum_{i=1}^{n} h_i(x_i) \\ &s.t. \quad x \in C, \\ &\qquad x_i \geq 0, \ i \in I \subseteq \{1, \ldots, n\}, \end{aligned} \tag{7}$$

we denote by Ω the feasible set of the above problem.

Below we report the outline of the algorithm. As we can easily see, at each iteration the problem (8) is equivalent to a problem of dimension $n - |I^k|$ and $I^k \subseteq I^{k+1}$, then the problems to be solved are of nonincreasing dimensions. This yields obvious advantages in terms of computational time. We state here the main result about the global convergence of the FW-RD Algorithm to a stationary point [25].

Algorithm 1 Frank–Wolfe - Reduced Dimension (FW-RD) Algorithm

Require: $x^0 \in \Omega$.
for $k = 0, 1, \ldots,$ **do**
Set $I^{x^k} = \{i \in I : x_i^k = 0\}$ and $\Omega^{x^k} = \{x \in \Omega : x_i = 0, \forall\, i \in I^{x^k}\}$
Obtain solution $\overline{x}^k$ by solving the following problem:

$$\overline{x}^k = \arg \min_{x \in \Omega^{x^k}} \nabla f(x^k)^T (x - x^k) \tag{8}$$

if $\nabla f(x^k)^T (\overline{x}^k - x^k) = 0$ **then** STOP
else define a feasible descent direction

$$d^k = \overline{x}^k - x^k$$

and generate a new feasible vector

$$x^{k+1} = x^k + \alpha^k d^k$$

with $\alpha^k \in (0, 1]$ a suitably chosen stepsize.
end if
end for

Proposition 1 *Let $\{x^k\}$ be a sequence generated by the FW-RD Algorithm*

$$x^{k+1} = x^k + \alpha^k d^k.$$

Assume that

1. *the method used for choosing stepsize α^k satisfies the following conditions:*
   (i) $f(x^{k+1}) < f(x^k)$, with $\nabla f(x^k) \neq 0$;
   (ii) if $\nabla f(x^k) \neq 0 \ \forall\, k$, then we have

$$\lim_{k \to \infty} \nabla f(x^k)^T d^k = 0\,;$$

2. *there exists a value S such that $h_i'(0) \geq S$, for all $i \in I$.*

Then every limit point $\bar{x}$ of $\{x^k\}$ is a stationary point. □

We notice that assumption 2 of Proposition 1 holds for suitable values of the parameters of the smooth concave approximating functions (3)–(6); so that Algorithm FW-RD can be applied. In practice, the parameters usually need to guarantee that the slope of the graph related to the function h gets reasonably large when we are close to 0.

We report some of the most popular rules for choosing the stepsize α^k:

1. *Minimization rule:* Here α^k is the value obtained by minimizing the function along the direction d^k,

$$f(x^k + \alpha^k d^k) = \min f(x^k + \alpha d^k).$$

   Minimization rule is typically implemented by means of line search algorithms. In practice, the stepsize is not computed exactly, but it is replaced by a stepsize α^k satisfying some termination criteria.
2. *Armijo rule:* In this case, fixed scalars $\triangle^k$, δ and γ, with $\delta \in (0, 1)$ and $\gamma \in (0, 1/2)$, are chosen, and $\alpha^k = \delta^{m^k} \triangle^k$, where m^k is the first nonnegative integer m for which

$$f(x^k + \delta^m \triangle^k d^k) \leq f(x^k) + \gamma \delta^m \triangle^k \nabla f(x^k)^T d^k.$$

   The stepsizes $\delta^m \triangle^k$, $m = 1, 2, \ldots$, are tried successively until the above inequality is satisfied for $m = m^k$.
3. *Constant stepsize:* According to this choice, a fixed stepsize

$$\alpha^k = 1, \quad k = 0, 1, \ldots$$

   is used. This rule can be adopted when the objective function has some particular properties (e.g. concavity). Anyway, if we rescale or redefine appropriately the direction d^k, we can always use a constant stepsize.

As a final remark to this section, we note that in the practical implementation of method FW-RD a sufficiently small threshold ρ is needed for identifying (near) zero components of the solution at iteration k. Hence, the actual definition of set I^{x^k} is $I^{x^k} = \{i \in I : |x_i^k| \leq \rho\}$ for sufficiently small $\rho > 0$ (e.g. $\rho = 10^{-4}$).

3 Analytical solution of the Frank–Wolfe subproblem

In this section, under Assumption 1, we show how to analytically compute the solution of the Frank–Wolfe subproblem.

In the next proposition, we give the analytical solution for the FW subroblem when C satisfies Assumption 1 (a).

Proposition 2 *Let* $C = \{x \in \mathbb{R}^n : \|x\|^2 \leq 1\}$. *The problem*

$$\begin{array}{ll} \min & c_x' x + c_y' y \\ s.t. & x'x \leq 1, \\ & -y \leq x \leq y. \end{array} \tag{9}$$

admits the following solution:

$$x^\star = \begin{cases} 0 & if\ \forall\, i\ |(c_x)_i| \leq (c_y)_i \\ \frac{\tilde{x}}{\|\tilde{x}\|} & otherwise \end{cases} \qquad y^\star = |x^\star|$$

where

$$\tilde{x}_i = \begin{cases} 0 & |(c_x)_i| \leq (c_y)_i \\ sgn[(c_x)_i](c_y)_i - (c_x)_i & |(c_x)_i| > (c_y)_i \end{cases} \quad i = 1, \ldots, n. \tag{10}$$

Proof The KKT conditions for Problem (9) are the following:

$$c_x + 2\mu x + \sigma - \rho = 0, \tag{11a}$$
$$c_y - \sigma - \rho = 0, \tag{11b}$$
$$\mu(\|x\|^2 - 1) = 0, \quad \mu \geq 0, \tag{11c}$$
$$\sigma'(x - y) = 0, \quad \sigma \geq 0, \tag{11d}$$
$$\rho'(x + y) = 0, \quad \rho \geq 0, \tag{11e}$$
$$\|x\|^2 \leq 1, \tag{11f}$$
$$x - y \leq 0, \tag{11g}$$
$$-x - y \leq 0. \tag{11h}$$

We say that $(\bar{x}, \bar{y}) \in \mathbb{R}^{2n}$ is a KKT pair for problem (9) when multipliers $\bar{\mu} \in \mathbb{R}$, $\bar{\sigma} \in \mathbb{R}^n$ and $\bar{\rho} \in \mathbb{R}^n$ exist such that $(\bar{x}, \bar{y}, \bar{\mu}, \bar{\sigma}, \bar{\rho})$ satisfy (11a). Let us consider the two cases:

1. $|(c_x)_i| \leq (c_y)_i$ for all $i = 1, \ldots, n$. It is easy to see that the tuple $(x^\star, y^\star, \mu^\star, \sigma^\star, \rho^\star)$ where

$$\begin{aligned} &x^\star = 0, \quad y^\star = 0, \quad \mu^\star = 0, \\ &\rho_i^\star = \frac{(c_y)_i + (c_x)_i}{2}, \quad \text{for all} i = 1, \ldots, n, \\ &\sigma_i^\star = \frac{(c_y)_i - (c_x)_i}{2}, \quad \text{for all} i = 1, \ldots, n, \end{aligned}$$

   satisfy the KKT conditions.
2. $|(c_x)_i| > (c_y)_i$, for at least an index $i \in \{1, \ldots, n\}$. In this case, it can be seen that the tuple $(x^\star, y^\star, \mu^\star, \sigma^\star, \rho^\star)$ where

$$x^\star = \frac{\tilde{x}}{\|\tilde{x}\|}, \quad y^\star = |x^\star|, \quad \mu^\star = \frac{\|\tilde{x}\|}{2},$$

   with $\tilde{x}$ is given by (10), and

$$\sigma_i^\star = \begin{cases} (c_y)_i & \text{if } (c_x)_i < -(c_y)_i < 0 \\ & \text{if } (c_x)_i > \quad (c_y)_i > 0 \\ \frac{(c_y)_i - (c_x)_i}{2} & |(c_x)_i| \leq (c_y)_i, \end{cases}$$
$$\rho_i^\star = \begin{cases} 0 & \text{if } (c_x)_i < -(c_y)_i < 0 \\ (c_y)_i & \text{if } (c_x)_i > \quad (c_y)_i > 0 \\ \frac{(c_y)_i + (c_x)_i}{2} & |(c_x)_i| \leq (c_y)_i, \end{cases}$$

   satisfy the KKT conditions.

The proof follows by considering that the gradients of the active constraints at $(x^\star, y^\star)$ are linearly independent. □

Now, we report an analogous result for the problem with bound constraints.

Proposition 3 *Let $C = \{x \in \mathbb{R}^n : -Me \leq x \leq Me\}$. The problem*

$$\begin{array}{ll} \min & c_x' x + c_y' y \\ s.t. & -y \leq x \leq y, \\ & -Me \leq x \leq Me, \end{array} \tag{12}$$

admits $(x^\star, y^\star)$ as a solution, where

$$x_i^\star = \begin{cases} 0 & if\ |(c_x)_i| \leq (c_y)_i \\ -Msgn[(c_x)_i] & otherwise \end{cases} \qquad y_i^\star = |x_i^\star|,$$

for $i = 1, \ldots, n$.

Proof The KKT conditions for Problem (23) are the following:

$$c_x + \sigma - \rho + r - s = 0, \tag{13a}$$
$$c_y - \sigma - \rho = 0, \tag{13b}$$
$$\sigma'(x - y) = 0, \quad \sigma \geq 0, \tag{13c}$$
$$\rho'(x + y) = 0, \quad \rho \geq 0, \tag{13d}$$
$$r'(x - Me) = 0, \quad r \geq 0, \tag{13e}$$
$$s'(-x - Me) = 0, \quad s \geq 0, \tag{13f}$$
$$x - y \leq 0, \tag{13g}$$
$$-x - y \leq 0. \tag{13h}$$

We say that $(\bar{x}, \bar{y}) \in \mathbb{R}^{2n}$ is a KKT pair for problem (23) when multipliers $\bar{\sigma} \in \mathbb{R}^n$, $\bar{\rho} \in \mathbb{R}^n$, $\bar{r} \in \mathbb{R}^n$ and $\bar{s} \in \mathbb{R}^n$ exist such that $(\bar{x}, \bar{y}, \bar{\sigma}, \bar{\rho}, \bar{r}, \bar{s})$ satisfy (13a). Let us consider the two cases:

1. $|(c_x)_i| \leq (c_y)_i$ for all $i = 1, \ldots, n$. It is easy to see that the tuple $(x^\star, y^\star, \sigma^\star, \rho^\star, r^\star, s^\star)$ where

$$\begin{aligned} & x^\star = 0, \quad y^\star = 0, \quad r^\star = 0, \quad s^\star = 0, \\ & \rho_i^\star = \frac{(c_y)_i + (c_x)_i}{2}, \quad \text{for all} \quad i = 1, \ldots, n, \\ & \sigma_i^\star = \frac{(c_y)_i - (c_x)_i}{2}, \quad \text{for all} \quad i = 1, \ldots, n, \end{aligned}$$

   satisfy the KKT conditions.
2. If $|(c_x)_j| > (c_y)_j$, for at least an index $j \in \{1, \ldots, n\}$, then, for $i = 1, \ldots, n$, we consider the following cases:

– $(c_x)_i < -(c_y)_i < 0$. In this case, it can be seen that

$$\begin{aligned} &\sigma_i^\star = (c_y)_i, \quad \rho_i^\star = 0, \quad s_i^\star = 0, \\ &r_i^\star = -((c_x)_i + (c_y)_i), \quad x_i^\star = M, \quad y_i^\star = M; \end{aligned}$$

– $(c_x)_i > (c_y)_i > 0$. In this case, it can be seen that

$$\begin{aligned} &\sigma_i^\star = 0, \quad \rho_i^\star = (c_y)_i, \quad s_i^\star = (c_x)_i - (c_y)_i, \\ &r_i^\star = 0, \quad x_i^\star = -M, \quad y_i^\star = M; \end{aligned}$$

– $|(c_x)_i| \leq (c_y)_i$. In this case, it can be seen that

$$\begin{aligned} &\sigma_i^\star = \frac{(c_y)_i - (c_x)_i}{2}, \quad \rho_i^\star = \frac{(c_y)_i + (c_x)_i}{2}, \\ &s_i^\star = 0, \quad r_i^\star = 0, \quad x_i^\star = 0, \quad y_i^\star = 0. \end{aligned}$$

The proof follows by considering that the gradients of the active constraints at $(x^\star, y^\star)$ are linearly independent. □

We would like to remark that using the KKT conditions of the problems in the proof of the propositions stated above, although quite standard in this context, is not the only way to prove those results (e.g. in Proposition 3 we could use the fact that the problem is separable and calculate the solution of each one-dimensional problem to get the final solution).

4 Sparse PCA

In this section we consider the SPCA problem and its solution via the proposed FW-RD Algorithm. First we shall briefly recall PCA. PCA is a well-established tool for data processing and analysis which allows to reduce high dimensional data to a smaller dimension. Given a real matrix $A \in \mathbb{R}^{p \times n}$ which encodes p samples of n variables or features, PCA aims at finding a few linear combinations of the variables, the principal components, which are orthogonal to each other and explain as much of the variance in the data as possible. If the rows of matrix A are of zero mean, then the classical PCA problem can be formulated by using the scaled covariance matrix $Q = A'A$ as follows

$$\begin{aligned} x^* = \ &\arg\max_x \ x'Qx \\ &s.t. \ x'x \leq 1. \end{aligned} \tag{14}$$

The solution vector x^* is said the *loading vector* or the (first) Principal Component (PC) of the data, that is the component that explains the maximum amount of variance in the data. x^* is the eigenvector of Q corresponding to the maximum eigenvalue. Hence, computing all of the PC's of Q amounts to computing the Singular Value Decomposition (SVD) of Q. Usually, the PC's of Q, including x^*, will have many non-zero components.

Sparse PCA aims at finding the PC's of the covariance matrix by minimizing, at the same time, the number of their non-zero components. In [11] a term penalizing the zero-norm of x is introduced in Problem (14) thus obtaining the following formulation of the SPCA problem:

$$\begin{array}{l} x^* = \ \arg\max_x \ x'Qx - \lambda\|x\|_0 \\ \qquad s.t.\ x'x \leq 1, \end{array} \tag{15}$$

which is then approximated by using the ℓ_1-norm. While PCA is numerically easy, Sparse PCA is a hard combinatorial problem. In fact, in [23] it is shown that the subset selection problem for ordinary least squares, which is NP-hard [24], can be reduced to a sparse generalized eigenvalue problem, of which sparse PCA is a particular intance. Hence, researchers are studying ways to make Problem (15) computationally tractable.

A simple approach to Problem (15) consists in solving PCA by neglecting the zero-norm term and then to threshold the loadings with small absolute value to zero [6]. More systematic approaches to the problem appeared in recent years, with various researchers proposing the use of nonconvex algorithms (e.g., GPower in [19], SpaSM in [17], SCoTLASS in [18], SLRA in [32] or D.C. based methods [29]) which find modified principal components with zero loadings. The SPCA algorithm in [33] is based on the representation of PCA as a regression-type optimization problem thus allowing for the application of the LASSO [30]. All the mentioned approaches and algorithms require solving non convex problems. Recently in [12] an ℓ_1-based semi-definite relaxation for the sparse PCA problem has been proposed.

In this section we propose solving Problem (15) via the FW-RD method. To this purpose, following [25] and by adding some auxiliary variables, we substitute the zero-norm of vector x in the definition of Problem (15) with a concave separable function thus obtaining the problem

$$\begin{array}{l} (x^*, y^*) = \ \arg\max_{x,y} \ x'Qx - \lambda \displaystyle\sum_{i=1}^{n} \log(\epsilon + y_i) \\ \qquad\qquad s.t.\ \ x'x \leq 1, \\ \qquad\qquad\quad -y \leq x \leq y, \end{array} \tag{16}$$

where, in particular, we add variables y_i for each $i \in \{1, \ldots, n\}$. We note that the above problem (16) has the form of Problem (7).

At every iteration of the FW-RD method, we have to solve the following subproblem:

$$\begin{array}{l} (x^*, y^*) = \ \arg\min_{x,y} \ -2(Qx^k)'x + \lambda \displaystyle\sum_{i=1}^{n} \frac{y_i}{\epsilon + y_i^k} = c_x'x + c_y'y \\ \qquad\qquad s.t.\ \ x'x \leq 1, \\ \qquad\qquad\quad -y \leq x \leq y, \end{array} \tag{17}$$

where x^k is the current iterate. In order to ease the understanding of the algorithm, we do not take into account the fact that at iteration k some of the variables can be fixed to zero.

With reference to Problem (17), we know that $(c_y)_i = \frac{\lambda}{\epsilon + y_i^k} > 0$, for all $i = 1, \ldots, n$. Then, the FW-RD algorithm described in Sect. 2 can be specialized by considering that:

- solution of the Frank–Wolfe subproblem is computed as described in Proposition 2;
- since the problem (16) is a concave programming problem, the stepsize α^k is fixed to 1.

5 Sparse reconstruction of noisy signals

Many problems in signal/image processing and statistics can be formulated as that of finding a sparse approximate solution to a large scale underdetermined linear system. A widely-studied problem in this context is the sparse representation of signals (see, e.g., [5,10]). Various media types (i.e. imagery, video and audio) can be sparsely represented using transform-domain methods, and in fact various relevant problems dealing with these media can be easily viewed as the problem of finding sparse solutions to a linear undertermined or ill-conditioned system. In practice, given a dictionary $A \in \mathbb{R}^{m \times n}$ of elementary signals and a real noisy signal b, the goal is finding a sparse representation x of signal b in terms of the dictionary A. A quite standard approach consists in solving an ℓ_1 regularized least-squares problem having the following form:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Ax - b\|^2 + \lambda \|x\|_1 \tag{18}$$

The ℓ_1-norm term promotes sparse solutions by forcing small components of the solution vector x to be zero. Problem (18) is strictly related to the *Least Absolute Shrinkage and Selection Operator*, a widely-studied problem in statistics, described for the first time by Tibshirani in [30]:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 \\ &s.t. \quad \|x\|_1 \leq \tau, \end{aligned} \tag{19}$$

where τ is a nonnegative real parameter regulating the sparsity of the solution. The Basis Pursuit [10] problem:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \|x\|_1 \\ &s.t. \quad Ax = b, \end{aligned} \tag{20}$$

is also related to Problem (18). Another interesting application of Problem (18) is Compressed Sensing [7–9]. The idea behind Compressed Sensing is that of encoding a large sparse signal using a relatively small number of linear measurements, and minimizing the ℓ_1-norm in order to decode the signal. In the last decades, Problem (18) has become increasingly popular and various algorithms have been proposed for efficiently solving it (see e.g. [1–3,13,15,20]). An alternative way to formulate the problem of reconstructing a noisy signal by elementary signals is the following:

$$x^\star = \arg\min_x \ \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_0 \tag{21}$$

In this section we propose solving Problem (21) via the FW-RD method. To this purpose, following [25] and by adding some auxiliary variables, we substitute the zero-norm of vector x in the definition of Problem (15) with a concave separable function thus obtaining the problem

$$\begin{aligned}(x^*, y^*) = \ & \arg\min_{x,y} \ \tfrac{1}{2}\|Ax - b\|^2 + \lambda \sum_{i=1}^{n}(1 - e^{-\alpha y_i}) \\ & s.t. \ -y \le x \le y,\end{aligned} \tag{22}$$

where, in particular, we add variables y_i for each $i \in \{1, \ldots, n\}$. We note that the above Problem (22) has the form of Problem (7).

As done in Sect. 4, at every iteration of the FW-RD method, we have to solve the following linear subproblem:

$$\begin{aligned}(x^*, y^*) = \ & \arg\min_{x,y} \ (Ax^k - b)'Ax + \lambda \sum_{i=1}^{n} \alpha e^{-\alpha y_i^k} y_i = c_x' x + c_y' y \\ & s.t. \ -y \le x \le y, \\ & \quad -Me \le x \le Me,\end{aligned} \tag{23}$$

where x^k is the current iterate. The last set of constraints ($-M \le x \le M$), when $M > 0$, makes the feasible region of Problem (23) compact. With reference to Problem (23), we know that $(c_y)_i = \lambda\alpha e^{-\alpha y_i^k} > 0$, for all $i = 1, \ldots, n$.

Then, the FW-RD algorithm described in Sect. 2 can be specialized by considering that:

- solution of the Frank–Wolfe subproblem is computed as described in Proposition 3;
- Stepsize α^k is chosen by means of an Armijo rule.

6 Numerical results on sparse PCA

In this section we report the results obtained by testing our method on two different classes of sparse PCA problems. More precisely as in [19], first we experiment on random data (with an underlying sparse PCA model). Then we consider some real datasets related to the analysis of gene expressions [31]. Further, we compare our method with the methods for sparse PCA proposed in [19], namely GPower$_{\ell_0}$, and in [17], namely SpaSM.

All the numerical experiments have been conducted using Matlab 7.12 (R2011b) on an Intel core i7 with 8GB RAM and running Linux version 2.6.38.

6.1 Random data drawn from a sparse PCA model

In order to generate random data with a covariance matrix having sparse eigenvectors, we follow the procedure proposed in [28]. Let $\Sigma = VDV'$ be a covariance matrix,

where the first m columns of $V \in \mathbb{R}^{n\times n}$ are pre-specified sparse orthonormal vectors. Then, a data matrix $A \in \mathbb{R}^{p\times n}$ is generated by using a zero-mean normal distribution with covariance matrix Σ, that is, $A \sim N(0, \Sigma)$.

We consider different pairs (p, n) and for each of them we generate 100 data matrices following [19]. We then use Algorithms FW-RD, GPower$_{\ell_0}$ and SpaSM to compute two unit-norm sparse PC's of Q. This can be done by using a standard so-called deflation scheme like that used in [12]. In particular, let z_1 be the computed solution of Problem (15), then z_2 can be obtained by solving again Problem (15) with

$$Q = (A - Az_1z_1')'(A - Az_1z_1').$$

In Table 1, we report the obtained results.

For every pair (p, n) we provide the average of the scalar products and computing times. Furthermore, in the column labelled "succ." we report the percentage of problems where the two pre-specified eigenvectors are successfully identified, that is when $|z_1'v_1|$ and $|z_2'v_2|$ are both greater than 0.99. As we can see from the table, the FW-RD Algorithm is competitive with GPower ℓ_0 and SpaSM in terms of CPU time and it is slightly better in terms of success rate.

6.2 Analysis of gene expressions data

DNA microarrays allow to provide the expression level of tens of thousands of genes across several hundreds of experiments thus constituting the source of a huge quantity of data. The interpretation of all these data is a challenging topic and calls for the use of advanced analytical tools. For more details and insights on microarrays and gene expression data, we refer to [27] and the references therein. Below we report results on two particular datasets [31] and precisely (a) colon cancer, (b) brown yeast and (c) lymphoma.

In the colon cancer dataset we have the expression of 2000 genes in 62 (22 normal and 40 colon cancer) tissue samples. The goal is that of determining the relevant genes to discriminate between cancerous and normal tissues. In Fig. 1, we report the proportion of adjusted variance versus the cardinality of the extracted set of discriminating genes (the so-called trade-off curve) for FW-RD, GPower$_{\ell_0}$ and SpaSM. As it can be seen, for this example our method is comparable with GPower$_{\ell_0}$ and better than SpaSM.

In the brown yeast dataset we have a total of 208 genes that have to be discriminated based on 79 gene expression data corresponding to different experimental settings. In Fig. 2 we report the trade-off curves of the three methods (SpaSM, GPower$_{\ell_0}$ and FW-RD), from which we can see that FW-RD outperforms GPower$_{\ell_0}$ (for small cardinalities) and SpaSM.

In the lymphoma problem the gene expression of 96 samples is measured with microarrays to give 4,026 features, 61 of the samples are in classes DLCL, FL or CLL (malignant) and 35 are labelled normal. As in the case of colon cancer data, the goal here is that of determining the relevant genes in discrimination. In Fig. 3, we

Table 1 Performance of SpaSM, GPower ℓ_0 and FW-RD algorithm on random data

p	n	$\lvert z_1' z_2\rvert$	$\lvert z_1' v_1\rvert$	$\lvert z_2' v_2\rvert$	Succ. (%)	Time
SpaSM						
10	100	1.28821e−03	7.50127e−01	7.35455e−01	40	5.50e−03
11	110	1.17833e−03	7.86666e−01	7.73023e−01	46	4.90e−03
12	120	1.19361e−03	7.41820e−01	7.27445e−01	42	4.40e−03
13	130	9.98098e−04	7.34747e−01	7.22297e−01	34	4.60e−03
14	140	7.67568e−04	7.43930e−01	7.32842e−01	34	4.80e−03
15	150	7.88741e−04	7.32755e−01	7.21531e−01	37	4.90e−03
16	160	6.17893e−04	7.52929e−01	7.45178e−01	40	5.30e−03
17	170	7.01328e−04	7.66798e−01	7.56287e−01	42	5.40e−03
18	180	6.05952e−04	7.84068e−01	7.75757e−01	43	6.40e−03
19	190	6.10760e−04	7.42425e−01	7.32904e−01	41	6.50e−03
20	200	5.62438e−04	7.34579e−01	7.27073e−01	42	7.80e−03
GPower ℓ_0						
10	100	2.02805e−03	7.48834e−01	7.33297e−01	40	4.00e−03
11	110	1.77439e−03	7.87859e−01	7.73650e−01	46	2.60e−03
12	120	1.87252e−03	7.40300e−01	7.24934e−01	42	2.60e−03
13	130	2.16435e−03	7.35524e−01	7.22492e−01	34	4.10e−03
14	140	2.29243e−03	7.43116e−01	7.31384e−01	35	3.10e−03
15	150	2.22725e−03	7.32765e−01	7.21021e−01	37	3.10e−03
16	160	2.69145e−03	7.52751e−01	7.44534e−01	41	3.30e−03
17	170	2.40034e−03	7.65433e−01	7.54503e−01	40	4.40e−03
18	180	2.86726e−03	7.84959e−01	7.76516e−01	43	3.40e−03
19	190	2.42358e−03	7.40279e−01	7.29977e−01	41	4.60e−03
20	200	2.60244e−03	7.32709e−01	7.24655e−01	43	3.20e−03
FW-RD						
10	100	3.52555e−03	7.49811e−01	7.38203e−01	41	4.10e−03
11	110	2.73873e−03	7.85200e−01	7.75009e−01	47	4.70e−03
12	120	2.86564e−03	7.45124e−01	7.28538e−01	42	3.40e−03
13	130	4.69557e−03	7.33128e−01	7.22478e−01	37	3.50e−03
14	140	1.98481e−03	7.40394e−01	7.31557e−01	35	4.00e−03
15	150	2.38888e−03	7.31121e−01	7.16115e−01	37	3.60e−03
16	160	9.15834e−04	7.43788e−01	7.32712e−01	38	4.90e−03
17	170	2.70908e−03	7.64909e−01	7.48966e−01	41	3.70e−03
18	180	1.70771e−03	7.73286e−01	7.61102e−01	44	4.60e−03
19	190	1.05600e−03	7.42655e−01	7.35217e−01	43	5.50e−03
20	200	3.22545e−03	7.27411e−01	7.13584e−01	39	5.00e−03

report the trade-off curves of the three methods (SpaSM, GPower$_{\ell_0}$ and FW-RD). We can see once again that FW-RD outperforms GPower$_{\ell_0}$ (for small cardinalities) and SpaSM.

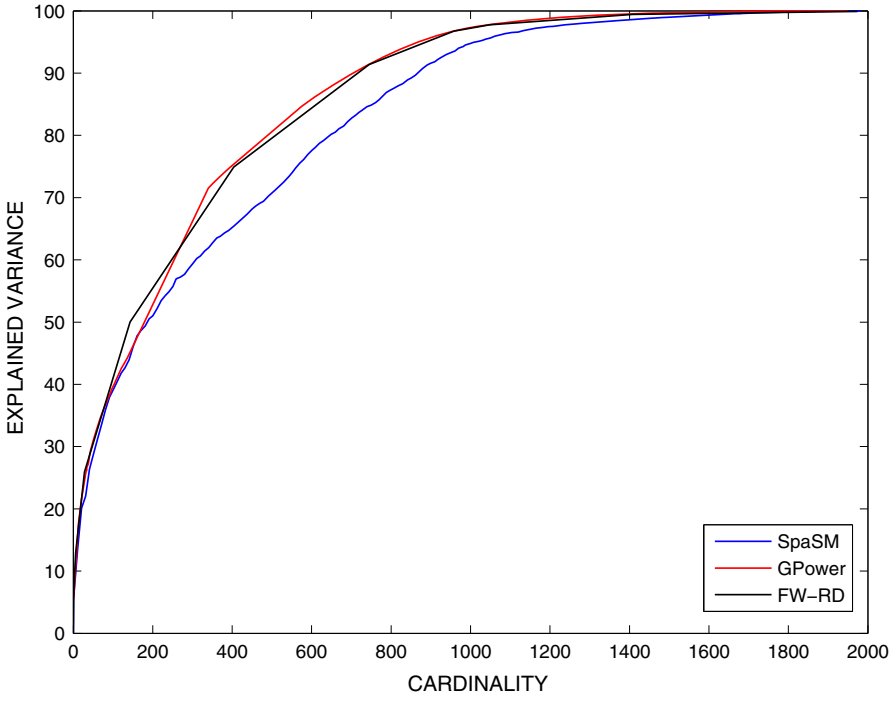

Fig. 1 Trade-off curves for gene expressions in colon cancer dataset

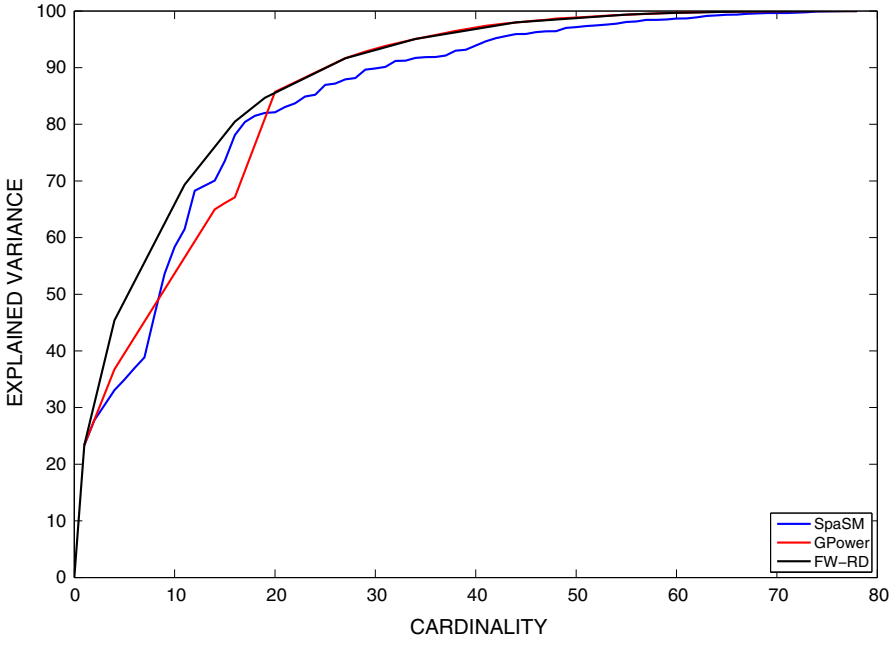

Fig. 2 Trade-off curves for gene expressions in yeast dataset

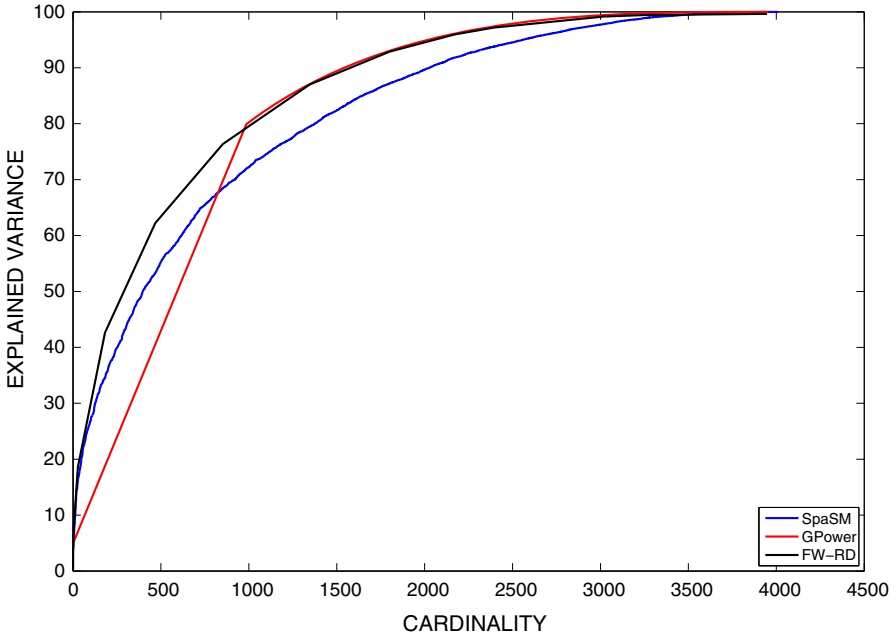

Fig. 3 Trade-off curves for gene expressions in lymphoma dataset

7 Numerical results on sparse representation of noisy signals

We compare FW-RD algorithm on a set of sparse signal reconstruction problems with the codes SALSA [1,2], FPC-AS [15] and SpaRSA [13]. We generated matrix A and vector b according to five different basic compressed sensing scenarios, like those described in [15,16]. In practice, we first randomly generate matrix A (according to one of the given scenarios), then we choose $b = Ax^\star + \nu$, where ν is a Gaussian white vector with variance 10^{-4} and $x^\star$ is a vector with T randomly placed ± 1 spikes and zeroes in the other components.

In Table 2, we report the experimentation and comparison of the proposed algorithm FW-RD with SALSA, FPC-AS and SpaRSA, where parameter λ is chosen as proposed in [13]. n, m and T denote, respectively, the number of columns, rows of the matrix A and the number of spikes of the sparse solution $x^\star$. $Prob$ denotes the compressed sensing scenario used, namely, partial discrete cosine transform ($Prob = 0$), random ± 1 Bernoulli ($Prob = 1$), partial Hadamard ($Prob = 2$), normally distributed random ($Prob = 3$) and scaled normally distributed random ($Prob = 4$) matrix. In the columns labelled time and MSE we report, respectively, the CPU computing time and the mean squared error of the reconstructions with respect to $x^\star$. Every table row reports the avarage results over 10 runs of the algorithms. All the numerical experiments have been conducted using Matlab 7.12 (R2011b) on an Intel core i7 with 8GB RAM and running Linux version 2.6.38.

As we can easily see by taking a look at the table, FW-RD is very competitive with SALSA, FPC-AS and SpaRSA both in terms of CPU time and MSE in all scenarios.

Table 2 Performance of SALSA, FPC-AS, SparSA and FW-RD Algorithm on randomly generated SRNS problems

Prob	FPC-AS		SpaRSA		SALSA		FW-RD	
	Time	MSE	Time	MSE	Time	MSE	Time	MSE
$n = 2{,}048$; $T = 25$; $m = 512$;								
0	5.324e+00	5.167e+02	8.660e−01	9.342e−02	3.958e+00	1.164e−02	3.580e−01	4.347e−02
1	4.738e+00	2.347e−04	1.568e+00	2.347e−02	6.755e+00	7.507e−01	5.220e−01	3.311e−05
2	4.917e+00	2.149e−04	4.350e−01	2.149e−02	6.929e+00	7.585e−01	6.620e−01	3.022e−05
3	4.925e+00	2.188e−04	1.658e+00	2.188e−02	6.704e+00	7.521e−01	5.820e−01	3.079e−05
4	4.805e+00	2.500e−04	9.300e−02	2.501e−02	6.517e+00	2.632e−04	1.390e−01	4.297e−03
$n = 4{,}096$; $T = 50$; $m = 1,024$;								
0	1.848e+01	2.319e+02	5.139e+00	7.817e−02	1.545e+01	5.243e−02	1.213e+00	6.246e−02
1	1.877e+01	2.657e−04	1.271e+01	2.657e−02	2.348e+01	7.492e−01	1.971e+00	3.760e−05
2	1.861e+01	2.445e−04	1.532e+00	2.445e−02	2.384e+01	7.478e−01	1.733e+00	3.452e−05
3	1.878e+01	2.548e−04	1.234e+01	2.548e−02	2.475e+01	7.481e−01	2.008e+00	3.604e−05
4	1.868e+01	2.705e−04	2.990e−01	2.705e−02	2.406e+01	2.850e−04	5.710e−01	2.176e−03
$n = 8{,}192$; $T = 100$; $m = 2,048$								
0	6.002e+01	1.516e+02	4.281e+01	1.606e−01	6.120e+01	7.075e−02	4.791e+00	1.191e−01
1	5.890e+01	3.005e−04	5.351e+01	7.784e−02	9.158e+01	7.486e−01	6.404e+00	1.095e−03
2	6.018e+01	2.299e−04	7.532e+00	2.299e−02	9.204e+01	7.504e−01	6.557e+00	3.238e−05
3	6.111e+01	2.936e−04	5.418e+01	7.191e−02	9.264e+01	7.499e−01	6.190e+00	2.145e−03
4	6.084e+01	2.797e−04	1.054e+00	2.796e−02	9.250e+01	2.950e−04	2.556e+00	4.850e−05

We finally want to notice that the cost in terms of CPU time for the FW-RD algorithm does not strongly depend on the scenario chosen and it does not grow that much even when the size of matrix A is quite large.

8 Conclusions

In this paper we considered a class of ℓ_0-penalized problems with simple constraints and adapted the first-order method proposed in [25] for their solution. Despite the assumption made on the feasible set C, the considered class of problems is still sufficiently general to encompass many significant applicative problems. We proposed a Frank–Wolfe method for the solution of the problem and showed that the FW subproblem can be solved analytically which is beneficial to the overall algorithm efficiency.

To show the effectiveness and efficiency of the proposed approach, we presented numerical results and comparison with other well-know software packages. In particular, we consider two numerical problems: (a) sparse PCA problems, for which we report comparison with the GPower method and SpaSM (b) sparse reconstruction of noisy signals problems, for which we report comparison with SALSA, FPC-AS and SpaRSA. For the former class of problem we tested our code on both random generated problems and biological problems. Our method performs well and compare quite favorably with GPower on random generated problems and is slightly superior on biological data. For the latter class of problems (reconstruction of noisy signals), we run our experimentation on five different signal scenarios. The results show that FW-RD is quite efficient and robust and often gives better results than SALSA, FPC-AS and SpaRSA both in terms of CPU time and reconstruction error.

To conclude, the results confirm viability of the proposed method and its efficiency when compared with state-of-the-art software packages for the solution of special classes of sparse problems. In this regard, we point out that our method is more general than the other algorithms, namely SpaSM, GPower, SALSA, FPC-AS and SpaRSA, since it allows to solve a more general class of sparse problems than those addressed by the competing solvers. Furthermore, we would like to remark that the method is very easy to implement (just a few lines of Matlab code).

References

1. Afonso, M., Bioucas-Dias, J., Figueiredo, M.: Fast image recovery using variable splitting and constrained optimization. IEEE Trans. Image Process. **19**(9), 2345–2356 (2010)
2. Afonso, M., Bioucas-Dias, J., Figueiredo, M.: An augmented lagrangian based method for the constrained formulation of imaging inverse problems. IEEE Trans. Image Process. **20**(3), 681–695 (2011)
3. Beck, A., Teboulle, M.: A fast iterative shrinkage-threshold algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(2), 183–202 (2009)
4. Bienstock, D.: Computational study of a family of mixed-integer quadratic programming problems. Math. Progr. **74**, 121–140 (1996)
5. Bruckstein, A.M., Donoho, D.L., Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev. **51**(1), 34–81 (2009)
6. Cadima, J., Jolliffe, I.: Loadings and correlations in the interpretation of principal components. J. Appl. Stat. **22**, 203–214 (1995)

7. Candès, E., Romberg, J.: Quantitative robust uncertainty principles and optimally sparse decompositions. Found. Comput. Math. **6**(2), 227–254 (2006)
8. Candès, E., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**(2), 489–509 (2006)
9. Candès, E., Romberg, J., Tao, T.: Stable signal recovery from incomplete and inaccurate measurements. Commun. Pure Appl. Math. **59**(8), 1207–1223 (2006)
10. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition basis pursuit. SIAM Rev. **43**, 129–159 (2001)
11. D'Aspremont, A., Bach, F.R., El Ghaoui, L.: Optimal solutions for sparse principal component analysis. J. Mach. Learn. Res. **9**, 1269–1294 (2008)
12. D'Aspremont, A., El Ghaoui, L., Jordan, N.I., Lanckriet, G.R.G.: A direct formulation for sparce pca using semidefinite programming. SIAM Rev. **49**(3), 434–448 (2007)
13. Figueiredo, M.A.T., Nowak, R.D., Wright, S.J.: Sparse reconstruction by separable approximation. IEEE Trans. Signal Process. **57**(7), 2479–2493 (2009)
14. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Nav. Res. Logist. Q. **3**, 95–110 (1956)
15. Hale, E.T., Yin, W., Zhang, Y.: Fixed-point continuation for l1-minimization: methodology and convergence. SIAM J. Optim. **19**(3), 1107–1130 (2008)
16. Hale, E.T., Yin, W., Zhang, Y.: http://www.caam.rice.edu/~optimization/l1/fpc/, (2012)
17. Hein, M., Bühler, T.: An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse pca. In: Advances in Neural Information Processing Systems, pp. 847–855 (2010)
18. Jolliffe, I.T., Trendafilov, N.T., Uddin, M.: A modified principal component technique based on the lasso. J. Comput. Graph. Stat. **12**, 531–547 (2003)
19. Journeé, M., Nesterov, Y., Richtárik, P., Sepulchre, R.: Generalized power method for sparse principal component analysis. J. Mach. Learn. Res. **11**, 517–553 (2010)
20. Kim, S.-J., Koh, K., Lustig, M., Boyd, S., Gorinevsky, D.: An interior-point method for large-scale l1-regularized least squares. IEEE J. Select. Topics Signal Process. **1**(4), 606–617 (2007)
21. Luss, R., Teboulle, M.: Conditional gradient algorithms for rank-one matrix approximations with a sparsity constraint. SIAM Rev. **55**(1), 65–98 (2013)
22. Mangasarian, O.L.: Machine learning via polyhedral concave minimization. In: Fischer, H., Riedmueller, B., Schaeffler, S. (eds.) Applied Mathematics and Parallel Computing Festschrift for Klaus Ritter, pp. 175–188. Physica-Verlag, Germany (1996)
23. Moghaddam, B., Weiss, Y., Avidan, S.: Generalized spectral bounds for sparse lda. In: ICML '06 Proceedings of the 23rd International Conference on Machine Learning, pp. 641–648 (2006)
24. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM J. Comput. **24**(2), 227–234 (1995)
25. Rinaldi, F.: Concave programming for finding sparse solutions to problems with convex constraints. Optim. Methods Softw. **26**(6), 971–992 (2011)
26. Rinaldi, F., Schoen, F., Sciandrone, M.: Concave programming for minimizing the zero-norm over polyhedral sets. Comput. Optim. Appl. **46**(3), 467–486 (2010)
27. Riva, A., Carpentier, A.-S., Torrésani, B., Hénaut, A.: Comments on selected fundamental aspects of microarray analysis. Comput. Biol. Chem. **29**(5), 319–336 (2005)
28. Shen, H., Huang, J.Z.: Sparse principal component analysis via regularized low rank matrix approximation. J. Multivar. Anal. **99**(6), 1015–1034 (2008)
29. Sriperumbudur, B.K., Torres, D.A., Lanckriet, G.R.G.: Sparse eigen methods by dc programming. In: Proceedings of the 24th International Conference on Machine Learning, pp. 831–838 (2007)
30. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B **58**(1), 267–288 (1996)
31. Weston, J., Elisseef, A., Schölkopf, B.: Use of the zero norm with linear models and kernel methods. J. Mach. Learn. Res. **3**, 1439–1461 (2003)
32. Zhang, Z., Zha, H., Simon, H.: Low rank approximations with sparse factors i: basic algorithms and error analysis. SIAM J. Matrix Anal. Appl. **23**(3), 706–727 (2002)
33. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. J. Comput. Graph. Stat. **15**(2), 265–286 (2006)