A D.C. Programming Approach to the Sparse Generalized Eigenvalue Problem

Bharath K. Sriperumbudur
Dept. of ECE
UC San Diego, La Jolla
bharathsv@ucsd.edu

David A. Torres
Dept. of CSE
UC San Diego, La Jolla
datorres@cs.ucsd.edu

Gert R. G. Lanckriet
Dept. of ECE
UC San Diego, La Jolla
gert@ece.ucsd.edu

Abstract

In this paper, we consider the sparse eigenvalue problem wherein the goal is to obtain a sparse solution to the generalized eigenvalue problem. We achieve this by constraining the cardinality of the solution to the generalized eigenvalue problem and obtain sparse principal component analysis (PCA), sparse canonical correlation analysis (CCA) and sparse Fisher discriminant analysis (FDA) as special cases. Unlike the ℓ_1-norm approximation to the cardinality constraint, which previous methods have used in the context of sparse PCA, we propose a tighter approximation that is related to the negative log-likelihood of a Student's t-distribution. The problem is then framed as a d.c. (difference of convex functions) program and is solved as a sequence of convex programs by invoking the majorization-minimization method. The resulting algorithm exhibits *global convergence* behavior, i.e., for any random initialization, the sequence (subsequence) of iterates generated by the algorithm converges to a stationary point of the d.c. program. The performance of the algorithm is empirically demonstrated on a sparse PCA application.

1 Introduction

The generalized eigenvalue (GEV) problem for the matrix pair $(\boldsymbol{A}, \boldsymbol{B})$ is the problem of finding a pair $(\lambda, \boldsymbol{x})$ such that

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{B}\boldsymbol{x}, \tag{1}$$

where $\boldsymbol{A}$, $\boldsymbol{B} \in \mathbb{C}^{n\times n}$, $\mathbb{C}^n \ni \boldsymbol{x} \neq \boldsymbol{0}$ and $\lambda \in \mathbb{C}$. When $\boldsymbol{B}$ is an identity matrix, the problem in (1) is simply referred to as an eigenvalue problem.

In multivariate statistics, GEV problems are prominent and appear in problems dealing with high-dimensional data analysis, visualization and pattern recognition. In these applications, usually $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \in \mathbb{S}^n_{++}$ (see Section 2 for the notation). The variational formulation for the GEV problem in (1) is given by

$$\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = \max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \,:\, \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}, \tag{GEV-P}$$

where $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$ is the maximum generalized eigenvalue associated with the matrix pair, $(\boldsymbol{A}, \boldsymbol{B})$. The $\boldsymbol{x}$ that maximizes (GEV-P) is called the generalized eigenvector associated with $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$. Some of the well-known and widely used data analysis techniques that are specific instances of (GEV-P) are: principal component analysis (PCA), canonical correlation analysis (CCA) and Fisher discriminant analysis (FDA). Despite the simplicity and popularity of these data analysis and modeling methods, one key drawback is the lack of sparsity in their solution. They suffer from the disadvantage that their solution vector, i.e., $\boldsymbol{x}$ is a linear combination of all input variables, which often makes it difficult to interpret the results. Sparse representations are generally desirable as they aid human understanding, reduce computational and economic costs and promote better generalization.

In this paper, we consider the problem of finding sparse solutions while explaining the statistical information in the data, which can be written as

$$\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \,:\, \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1,\, \|\boldsymbol{x}\|_0 \leq k\}, \tag{SGEV-P}$$

where $1 \leq k \leq n$ and $\|\boldsymbol{x}\|_0$ denotes the cardinality of $\boldsymbol{x}$, i.e., the number of non-zero elements of $\boldsymbol{x}$. The above program can be solved either as a continuous optimization problem after relaxing the cardinality constraint or as a discrete optimization problem. In this paper, we follow the former approach. The first step in solving (SGEV-P) as a continuous optimization problem is to approximate the cardinality constraint. One usual heuristic is to approximate $\|\boldsymbol{x}\|_0$ by $\|\boldsymbol{x}\|_1$. In this paper, we approximate the cardinality constraint in (SGEV-P) as the negative log-likelihood of a Student's t-distribution, which has been used earlier in many different contexts [17, 5, 2]. We then formulate this approximate problem as a d.c. (difference of convex functions) program and solve it using the majorization-minimization (MM) method [7] resulting in a sequence of quadratically constrained quadratic programs (QCQPs). As a special case, when $\boldsymbol{A}$ is positive definite and $\boldsymbol{B}$ is an identity matrix (as is the case for PCA), a very simple iterative update rule (we call it as DC-PCA) can be obtained in a closed form, which has a per iteration complexity of $O(n^2)$. The proposed algorithm can be shown to be *globally convergent*, i.e., for any random initialization, the sequence (subsequence) of iterates generated by the algorithm converges to a stationary point of the d.c. program. We would like to mention that the algorithm presented in this paper is more general than the one in [13] as it holds for any $\boldsymbol{A} \in \mathbb{S}^n$ unlike in [13], where $\boldsymbol{A}$ is assumed to be positive semidefinite.

We illustrate the performance of our sparse PCA algorithm, DC-PCA on a benchmark dataset and three high-dimensional datasets, wherein we show that DC-PCA performs similar to most of the existing sparse PCA algorithms (in fact better than [18]), but at better computational speeds.

2 Notation

$\mathbb{S}^n$ (respectively $\mathbb{S}^n_+$, $\mathbb{S}^n_{++}$) denotes the set of symmetric (respectively positive semidefinite, positive definite) $n \times n$ matrices defined over $\mathbb{R}$. For $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$, $\boldsymbol{x} \succeq \boldsymbol{0}$ denotes an element-wise inequality. $\|\boldsymbol{x}\|_0$ denotes the number of non-zero elements of the vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p := (\sum_{i=1}^n |x_i|^p)^{1/p}$, $1 \leq p < \infty$. $\boldsymbol{I}_n$ denotes an $n \times n$ identity matrix. $\boldsymbol{D}(\boldsymbol{x})$ represents a diagonal matrix formed with $\boldsymbol{x}$ as its principal diagonal.

3 Sparse Generalized Eigenvalue Problem

Let us consider the variational formulation for the sparse generalized eigenvalue problem in (SGEV-P), where $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \in \mathbb{S}^n_{++}$. Suppose $\boldsymbol{A}$ is not negative definite. Then (SGEV-P) is the maximization of a non-concave objective over the non-convex constraint set $\Phi := \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\} \cap \{\boldsymbol{x} : \|\boldsymbol{x}\|_0 \leq k\}$. Although Φ can be relaxed to a convex set $\widetilde{\Phi} := \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\} \cap \{\boldsymbol{x} : \|\boldsymbol{x}\|_1 \leq k\}$, it does not simplify the problem as the maximization of a non-concave objective over a convex set is still computationally hard and intractable. So, the intractability of (SGEV-P) is due to two reasons: (a) maximization of the non-concave objective function and (b) the constraint set being non-convex. Since (SGEV-P) is intractable, instead of solving it directly, one can solve approximations to (SGEV-P) that are tractable. In the following, we present a tractable approximation to (SGEV-P).

3.1 Non-convex approximation to $\|\boldsymbol{x}\|_0$ and d.c. formulation

To this end, we consider the regularized (penalized) version of (SGEV-P) given by

$$\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\,\|\boldsymbol{x}\|_0 \,:\, \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}, \tag{SGEV-R}$$

where $\tilde{\rho} > 0$ is the regularization (penalization) parameter. Note that the quadratic equality constraint, $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1$ is relaxed to the inequality constraint, $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1$. Since

$$\|\boldsymbol{x}\|_0 = \sum_{i=1}^n \mathbb{1}_{\{|x_i| \neq 0\}} = \lim_{\varepsilon \to 0} \sum_{i=1}^n \frac{\log(1 + |x_i|/\varepsilon)}{\log(1 + 1/\varepsilon)}, \tag{2}$$

(SGEV-R) is equivalent[1] to

$$\max\Big\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}-\tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(1+|x_i|/\varepsilon)}{\log(1+1/\varepsilon)}:\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}\le 1\Big\}. \tag{3}$$

The above program is approximated by the following *approximate sparse GEV program* by neglecting the limit in (3) and choosing $\varepsilon>0$,

$$\max\Big\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}-\tilde{\rho}\sum_{i=1}^{n}\frac{\log(1+|x_i|/\varepsilon)}{\log(1+1/\varepsilon)}:\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}\le 1\Big\}, \tag{4}$$

which is equivalent to

$$\max\Big\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}-\rho_\varepsilon\sum_{i=1}^{n}\log(|x_i|+\varepsilon)\;:\;\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}\le 1\Big\}, \tag{SGEV-A}$$

where $\rho_\varepsilon:=\tilde{\rho}/\log(1+\varepsilon^{-1})$. Note that the approximate program in (SGEV-A) is a continuous optimization problem unlike the one in (SGEV-R), which has a combinatorial term. Before we present a d.c. program formulation to (SGEV-A), we briefly discuss the approximation to $\|\boldsymbol{x}\|_0$ that we considered in this paper.

Approximation to $\|\boldsymbol{x}\|_0$: The approximation (to $\|\boldsymbol{x}\|_0$) that we considered in this paper, i.e., $\|\boldsymbol{x}\|_\varepsilon:=\sum_{i=1}^{n}\frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})}$, has been used in many different contexts: feature selection using support vector machines [17], sparse signal recovery [2], matrix rank minimization [5], etc. This approximation is interesting because of its connection to sparse factorial priors that are studied in Bayesian inference, and can be interpreted as defining a Student's t-distribution prior over $\boldsymbol{x}$, an improper prior given by $\prod_{i=1}^{n}\frac{1}{|x_i|+\varepsilon}$. [16] showed that this choice of prior leads to a sparse representation and demonstrated its validity for sparse kernel expansions in the Bayesian framework. It can be shown that the approximation (to $\|\boldsymbol{x}\|_0$) considered in this paper, i.e., $\|\boldsymbol{x}\|_\varepsilon$, is tighter than the ℓ_1-norm approximation, for any $\varepsilon>0$ and therefore provides sparser solutions than the ℓ_1-norm approximation. See [14] for a detailed discussion.

D.c. formulation: Let us return to the formulation in (SGEV-A). To solve this continuous, nonconvex optimization problem and derive an algorithm for the sparse GEV problem, we explore its formulation as a d.c. program. D.c. programs are well studied and many algorithms exist to solve them [6]. They are defined as follows.

Definition 1 (D.c. program)**.** *Let Ω be a convex set in $\mathbb{R}^n$. A real valued function $f:\Omega\to\mathbb{R}$ is called a d.c. function on Ω, if there exist two* convex *functions $g,h:\Omega\to\mathbb{R}$ such that f can be expressed in the form $f(\boldsymbol{x})=g(\boldsymbol{x})-h(\boldsymbol{x})$, $\boldsymbol{x}\in\Omega$. Optimization problems of the form* $\min\{f_0(\boldsymbol{x}):\boldsymbol{x}\in\Omega, f_i(\boldsymbol{x})\le 0,\, i=1,\dots,m\}$*, where $f_i=g_i-h_i$, $i=0,\dots,m$, are d.c. functions are called* d.c. programs.

To formulate (SGEV-A) as a d.c. program, let us choose $\tau\ge\max(0,-\lambda_{min}(\boldsymbol{A}))$. (SGEV-A) is equivalently written as

$$\min\Big\{\,\big[\tau\|\boldsymbol{x}\|_2^2-\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}\big]+\rho_\varepsilon\sum_{i=1}^{n}\log(|x_i|+\varepsilon):\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}\le 1\Big\}. \tag{5}$$

Introducing the auxiliary variable, $\boldsymbol{y}$, yields the equivalent program

$$\min\Big\{\tau\|\boldsymbol{x}\|_2^2-\left[\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}-\rho_\varepsilon\sum_{i=1}^{n}\log(y_i+\varepsilon)\right]:\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}\le 1,\,-\boldsymbol{y}\preceq\boldsymbol{x}\preceq\boldsymbol{y}\Big\}, \tag{6}$$

which is a d.c. program. Indeed, the term $\tau\|\boldsymbol{x}\|_2^2$ is convex in $\boldsymbol{x}$ as $\tau\ge 0$ and, by construction, $\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}-\rho_\varepsilon\sum_{i=1}^{n}\log(y_i+\varepsilon)$ is jointly convex in $\boldsymbol{x}$ and $\boldsymbol{y}$. So, the above program is a minimization of the difference of two convex functions over a convex set. In the following section, we present an iterative algorithm to solve (6) using the majorization-minimization method.

[1] Two programs are equivalent if their optimizers are the same.

3.2 Sparse GEV algorithm

The majorization-minimization (MM) method can be thought of as a generalization of the well-known expectation-maximization (EM) algorithm [4]. The general idea of MM algorithms is as follows. Suppose we want to minimize f over $\Omega \subset \mathbb{R}^n$. The idea is to construct a *majorization function* g over $\Omega \times \Omega$ such that

$$f(x) \leq g(x,y),\, \forall\, x,y \in \Omega \qquad \text{and} \qquad f(x) = g(x,x),\, \forall\, x \in \Omega. \tag{7}$$

Thus, g as a function of x is an upper bound on f and coincides with f at y. The majorization-minimization algorithm corresponding to this majorization function g updates x at iteration l by

$$x^{(l+1)} \in \arg\min_{x\in\Omega} g(x, x^{(l)}), \tag{8}$$

unless we already have $x^{(l)} \in \arg\min_{x\in\Omega} g(x, x^{(l)})$, in which case the algorithm stops.

Let us return to the approximate sparse GEV program in (5). Let

$$f(\boldsymbol{x}) = \tau\|\boldsymbol{x}\|_2^2 + \rho_\varepsilon \sum_{i=1}^n \log(\varepsilon + |x_i|) - \boldsymbol{x}^T(\boldsymbol{A} + \tau\boldsymbol{I}_n)\boldsymbol{x}, \tag{9}$$

where $\tau \geq \max(0, -\lambda_{min}(\boldsymbol{A}))$ so that (5) can be written as $\min_{\boldsymbol{x}\in\Omega} f(\boldsymbol{x})$ and $\Omega = \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$. The main idea in deriving the sparse GEV algorithm is in obtaining a majorization function, g that satisfies (7) and then using it in (8). The following result provides such a function g for f in (9).

Proposition 2. *The following function*

$$g(\boldsymbol{x},\boldsymbol{y}) = \tau\|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y} + \boldsymbol{y}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y} + \rho_\varepsilon\sum_{i=1}^n \log(\varepsilon+|y_i|) + \rho_\varepsilon \sum_{i=1}^n \frac{|x_i| - |y_i|}{|y_i| + \varepsilon}, \tag{10}$$

majorizes f in (9).

By following the minimization step in (8) with g as in (10), the *sparse GEV algorithm* is obtained as

$$\boldsymbol{x}^{(l+1)} = \arg\min \Big\{ \tau\|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}^{(l)} + \rho_\varepsilon \sum_{i=1}^n \frac{|x_i|}{|x_i^{(l)}| + \varepsilon} \;:\; \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1 \Big\}, \tag{ALG}$$

which is a sequence of quadratically constrained quadratic programs (QCQPs) [1]. It is clear that $\boldsymbol{x}^{(l+1)}$ is the unique optimal solution of (ALG) irrespective of whether τ is zero or not. Assuming $\tau \neq 0$ and defining $w_i^{(l)} := \frac{1}{|x_i^{(l)}|+\varepsilon}$, $\boldsymbol{w}^{(l)} := (w_1^{(l)}, \ldots, w_n^{(l)})$ and $\boldsymbol{W}^{(l)} := \boldsymbol{D}(\boldsymbol{w}^{(l)})$, (ALG) reduces to

$$\boldsymbol{x}^{(l+1)} = \arg\min \Big\{ \left\|\boldsymbol{x} - (\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)}\right\|_2^2 + \frac{\rho_\varepsilon}{\tau}\left\|\boldsymbol{W}^{(l)}\boldsymbol{x}\right\|_1 : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1 \Big\}. \tag{11}$$

(11) is very similar to LASSO [15] except for the *weighted* ℓ_1-penalty and the quadratic constraint. When $\boldsymbol{A} \succeq 0$, $\boldsymbol{B} = \boldsymbol{I}_n$ and $\tau = 0$, (ALG) reduces to a very simple iterative rule:

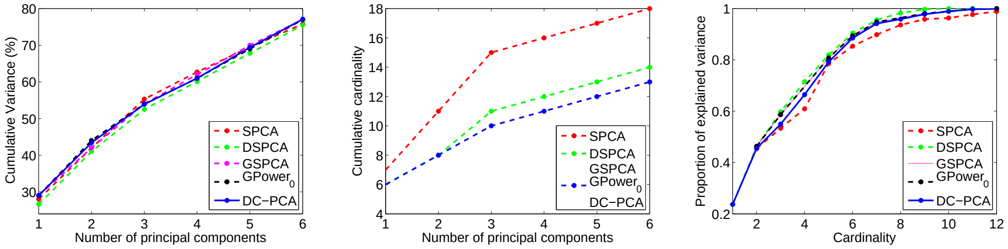
$$x_i^{(l+1)} = \frac{\left[|(\boldsymbol{A}\boldsymbol{x}^{(l)})_i| - \frac{\rho_\varepsilon}{2}w_i^{(l)}\right]_+ \operatorname{sign}((\boldsymbol{A}\boldsymbol{x}^{(l)})_i)}{\sqrt{\sum_{i=1}^n \left[|(\boldsymbol{A}\boldsymbol{x}^{(l)})_i| - \frac{\rho_\varepsilon}{2}w_i^{(l)}\right]_+^2}},\, \forall i, \tag{ALG-S}$$

where $[a]_+ := \max(0,a)$, which we call as DC-PCA. Note that (ALG-S) can be used to solve sparse PCA while (ALG) is to be used to solve for sparse CCA as the corresponding $\boldsymbol{A}$ is indefinite.

As mentioned before, it can be shown that (ALG) is globally convergent. For details, see [12, 14].

4 Experiments

In this section, we illustrate the effectiveness of DC-PCA in terms of sparsity and scalability on various datasets. On small datasets, the performance of DC-PCA is compared against SPCA [18], DSPCA [3], GSPCA [11] and GPower$_{\ell_0}$ [9], while on large datasets, DC-PCA is compared to all these algorithms except DSPCA and GSPCA due to scalability issues. The results show that the performance of DC-PCA is comparable to the performance of many of these algorithms, but with *better scalability*. For more detailed experiments, see [14]. On the implementation side, we fix ε to be the machine precision in all our experiments.



(a) (b) (c)

Figure 1: Pit props: (a) cumulative variance and (b) cumulative cardinality for the first 6 sparse principal components (PCs); (c) proportion of explained variance (PEV) vs. cardinality for the first sparse PC (obtained by varying the sparsity parameter and computing the cardinality and explained variance for the solution vector).

Table 1: Gene expression datasets

Dataset	Samples (p)	Genes (n)
Colon cancer	62	2000
Leukemia	38	7129
Ramaswamy	127	16063

4.1 Pit props data

The pit props dataset [8] has become a standard benchmark example to test sparse PCA algorithms. The first 6 principal components (PCs) capture 87% of the total variance (see [10] for the deflation technique to obtain sparse eigenvectors other than the dominant one). Therefore, the explanatory power of sparse PCA methods is often compared on the first 6 sparse PCs. Comparing the cumulative variance and cumulative cardinality, Figures 1(a–b) show that DC-PCA explains more variance with fewer non-zero loadings than SPCA and DSPCA. In addition, its performance is similar to that of GSPCA and GPower$_{\ell_0}$. For the first sparse PC, Figure 1(c) shows that DC-PCA consistently explains more variance with better sparsity than SPCA, while performing similar to other algorithms.

4.2 Gene expression data

Usually, gene expression data is specified by a $p \times n$ matrix (say $\boldsymbol{C}$) of p samples and n genes. The covariance matrix, $\boldsymbol{A}$ is therefore computed as $\boldsymbol{C}^T\boldsymbol{C}$. In our experiments, we consider three gene expression datasets which are tabulated in Table 1. Figures 2(a-c) show the proportion of explained variance versus the cardinality for the first sparse PC for the datasets shown in Table 1. It can be seen that DC-PCA performs similar to GPower$_{\ell_0}$ and performs better than SPCA. The average computation time required by the sparse PCA algorithms on each dataset is shown in Table 2. The indicated times are averages over n computations, one for each cardinality ranging from n down to 1. The results show that DC-PCA and GPower$_{\ell_0}$ are significantly faster than SPCA, which, for a long time, was widely accepted as the algorithm that can handle large datasets.

Overall, the results in this section demonstrate that DC-PCA performs similar to GPower$_{\ell_0}$, the state-of-the-art, and better than SPCA, both in terms of scalability and proportion of variance explained vs. cardinality. We would like to mention that our sparse PCA algorithm (DC-PCA) is derived from a more general framework, that can be used to address other generalized eigenvalue problems as well, e.g., sparse CCA, sparse FDA, etc.

References

[1] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] E. J. Candes, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *J. Fourier Anal. Appl.*, 2007. To appear.

[3] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

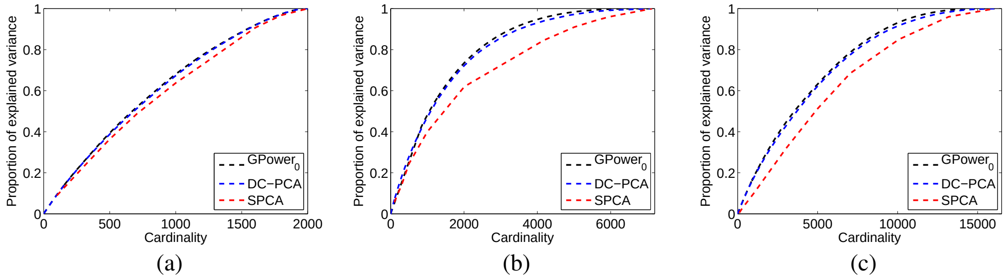


Figure 2: Trade-off curves between explained variance and cardinality for (a) Colon cancer, (b) Leukemia and (c) Ramaswamy datasets. The proportion of variance explained is computed on the first sparse principal component. (a–c) show that DC-PCA performs similar to GPower$_{\ell_0}$, while explaining more variance (for a fixed cardinality) than SPCA.

Table 2: Computation time (in seconds) to obtain the first sparse PC, averaged over cardinalities ranging from 1 to n, for the Colon cancer, Leukemia and Ramaswamy datasets.

	Colon cancer	Leukemia	Ramaswamy
n	2000	7129	16063
SPCA	2.057	3.548	38.731
GPower$_{\ell_0}$	0.182	0.223	2.337
DC-PCA	0.034	0.156	0.547

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc. B*, 39:1–38, 1977.

[5] M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. In *Proc. American Control Conference*, Denver, Colorado, 2003.

[6] R. Horst and N. V. Thoai. D.c. programming: Overview. *Journal of Optimization Theory and Applications*, 103:1–43, 1999.

[7] D. R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58:30–37, 2004.

[8] J. Jeffers. Two case studies in the application of principal components. *Applied Statistics*, 16:225–236, 1967.

[9] M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *http://arxiv.org/abs/0811.4724v1*, November 2008.

[10] L. Mackey. Deflation methods for sparse pca. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1017–1024. MIT Press, 2009.

[11] B. Moghaddam, Y. Weiss, and S. Avidan. Spectral bounds for sparse PCA: Exact and greedy algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

[12] B. K. Sriperumbudur and G. R. G. Lanckriet. On the convergence of the concave-convex procedure. In *NIPS*, 2009. To appear.

[13] B. K. Sriperumbudur, D. A. Torres, and G. R. G. Lanckriet. Sparse eigen methods by d.c. programming. In *Proc. of the 24th Annual International Conference on Machine Learning*, 2007.

[14] B. K. Sriperumbudur, D. A. Torres, and G. R. G. Lanckriet. A d.c. programming approach to the sparse generalized eigenvalue problem. *http://arxiv.org/abs/0901.1504v2*, 2009.

[15] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[16] M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

[17] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, March 2003.

[18] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15:265–286, 2006.