Sparse Eigen Methods by D.C. Programming

Bharath K. Sriperumbudur BHARATHSV@UCSD.EDU
Department of Electrical and Computer Engineering, University of California, San Diego, CA 92093 USA.

David A. Torres DATORRES@CS.UCSD.EDU
Department of Computer Science and Engineering, University of California, San Diego, CA 92093 USA.

Gert R. G. Lanckriet GERT@ECE.UCSD.EDU
Department of Electrical and Computer Engineering, University of California, San Diego, CA 92093 USA.

Abstract

Eigenvalue problems are rampant in machine learning and statistics and appear in the context of classification, dimensionality reduction, etc. In this paper, we consider a cardinality constrained variational formulation of generalized eigenvalue problem with sparse principal component analysis (PCA) as a special case. Using ℓ_1-norm approximation to the cardinality constraint, previous methods have proposed both convex and non-convex solutions to the sparse PCA problem. In contrast, 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 locally convex programs. We show that the proposed method not only explains more variance with sparse loadings on the principal directions but also has better scalability compared to other methods. We demonstrate these results on a collection of datasets of varying dimensionality, two of which are high-dimensional gene datasets where the goal is to find few relevant genes that explain as much variance as possible.

1. Introduction

Principal component analysis (PCA), a popular tool for data analysis, data compression and visualization; its two-view counterpart, canonical correlation analysis (CCA) and Fisher discriminant analysis (FDA) can be seen as specific instances of a generalized maximum eigenvalue problem. Despite the simplicity and popularity of these methods, one key drawback is the lack of sparseness in their solution. Sparse representations are generally desirable as they aid human understanding, reduce computational costs and promote better generalization. For example, PCA and CCA, as dimensionality reduction tools, suffer from the disadvantage that their solution vector is a linear combination of all input variables, which often makes it difficult to interpret the results. For better interpretability (e.g., with gene expression data, financial asset trading), one would be interested to find few relevant features that explain as much variance as possible. Similarly, in a classification setting like FDA, feature selection aids generalization performance by promoting sparse solutions.

Among the generalized eigenvalue problems, sparse representations are well-studied for PCA. The earliest of attempts consisted of simple axis rotation and component thresholding for subset selection (Cadima & Jolliffe, 1995). Jolliffe et al. (2003) proposed SCoTLASS by enforcing a sparsity constraint on the principal directions by bounding their ℓ_1-norm, leading to a non-convex procedure. Zou et al. (2004) proposed SPCA, a ℓ_1-penalized regression algorithm for PCA using elastic-net, which is solved very efficiently using least angle regression. Currently, this method seems to be the only viable option for handling very high-dimensional datasets (on the order of $10,000$). Subsequently, d'Aspremont et al. (2005) proposed DSPCA, a convex relaxed solution to sparse PCA leading to a semidefinite program (SDP) (Vandenberghe & Boyd, 1996). Though this method shows comparable performance to SPCA on a small-scale benchmark dataset, it is not scalable to high-dimensional datasets, even, possibly, with Nesterov's first-order method. Re-

Appearing in *Proceedings of the 24^{th} International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

cently, Moghaddam et al. (2007) proposed GSPCA, a combinatorial optimization algorithm based on bidirectional greedy search and has outperformed these other algorithms. However, this method is also not scalable to large covariance matrices.

In this paper, we propose a sparse generalized eigenvalue algorithm by approximating the cardinality of the eigen solution as the negative log-likelihood of a Student's t-distribution. Weston et al. (2003) proposed the above approximation for feature selection in support vector machines. We show that this approximate problem can be framed as a d.c. (difference of convex functions) program and solved as a sequence of quadratically constrained quadratic programs (QCQP) using the d.c. minimization algorithm (DCA) of Tao and An (1998), which guarantees convergence to a local optimum in finite time. On the standard "pit props" benchmark dataset, known in the statistics community for its lack of sparseness and subsequent difficulty of interpretation, we show that our proposed method (DC-PCA) performs better than SPCA and DSPCA in terms of sparsity vs. explained variance. We also demonstrate the positive results of our method w.r.t. SPCA (DSPCA is not scalable) on two high dimensional datasets where the goal is to find relevant genes (as few as possible) while explaining the maximum possible variance.

In summary, two important points addressed in this paper are sparsity and scalability. Sparsity is improved by using a better approximation to the cardinality constraint. The simplicity of DCA leads to the proposed algorithm (DC-PCA) which has better scalability than SPCA and DSPCA. Therefore, we argue that our method provides better scalability and sparsity vs. explained variance than SPCA and DSPCA.

2. Notation

In this paper, $\mathbb{S}^n = \{\mathbf{X}_{n\times n} : \mathbf{X} = \mathbf{X}^T\}$, $\mathbf{1} = (1, \overset{n}{\ldots}, 1)^T$ and $\mathbf{0} = (0, \overset{n}{\ldots}, 0)^T$. For $\mathbf{X} \in \mathbb{S}^n$, $\mathbf{X} \succ 0$ (*resp.* $\mathbf{X} \succeq 0$) means that $\mathbf{X}$ is positive definite (*resp.* semidefinite). For $\mathbf{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ and $\mathbf{y} = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$, $\mathbf{x} \preceq \mathbf{y} \Rightarrow x_i \leq y_i, \forall i$. $||\mathbf{x}||_0$ denotes the number of non-zero elements of $\mathbf{x}$. $\mathbf{I}$ denotes an $n \times n$ identity matrix. $\mathbf{D}(\mathbf{x})$ represents a diagonal matrix with $\mathbf{x}$ as its principal diagonal. $\text{tr}(\mathbf{X})$ represents the trace of $\mathbf{X}$.

3. Generalized Eigenvalue Problems

The variational formulation for the generalized eigenvalue problem is given by

$$\max_{\mathbf{x}} \{\mathbf{x}^T\mathbf{A}\mathbf{x} \ : \ \mathbf{x}^T\mathbf{B}\mathbf{x} = 1\}, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{S}^n$ and $\mathbf{B} \succ 0$. Various algorithms in machine learning and statistics are specific instantiations of Eq. (1). In the binary classification setting, FDA finds a 1-D subspace onto which the projections of data lead to a maximal separation of classes. Its variational formulation is the same as Eq. (1) with $\mathbf{A} = \mathbf{S}_b$ and $\mathbf{B} = \mathbf{S}_w$ where $\mathbf{S}_b = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$ is the *between-cluster variance* and $\mathbf{S}_w = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2$ is the *within-cluster variance*. $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the mean vector and covariance matrix of class i respectively.

In the unsupervised setting, PCA and CCA are widely used as dimensionality reduction tools. Replacing $\mathbf{B}$ by $\mathbf{I}$ in Eq. (1) leads to the variational formulation of PCA, which identifies the direction of maximal variance in the data, with $\mathbf{A}$ being the covariance matrix. While PCA deals with only one data space $\mathcal{X}$, CCA proposes a way for dimensionality reduction by taking into account relations between samples from two spaces $\mathcal{X}$ and $\mathcal{Y}$. The assumption is that the data points coming from these two spaces contain some joint information that is reflected in correlations between them. Directions along which this correlation is high are thus assumed to be relevant directions when these relations are to be captured. The variational formulation for CCA is given by $\max_{\mathbf{w}_x, \mathbf{w}_y} \{\mathbf{w}_x^T\mathbf{S}_{xy}\mathbf{w}_y \ : \ \mathbf{w}_x^T\mathbf{S}_{xx}\mathbf{w}_x = 1, \mathbf{w}_y^T\mathbf{S}_{yy}\mathbf{w}_y = 1\}$ which can be written in the form of Eq. (1) with $\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{S}_{xy} \\ \mathbf{S}_{yx} & \mathbf{0} \end{pmatrix}$, $\mathbf{B} = \begin{pmatrix} \mathbf{S}_{xx} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{yy} \end{pmatrix}$ and $\mathbf{x} = \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_y \end{pmatrix}$ where $\mathbf{S} = \begin{pmatrix} \mathbf{S}_{xx} & \mathbf{S}_{xy} \\ \mathbf{S}_{yx} & \mathbf{S}_{yy} \end{pmatrix}$ is the covariance matrix between samples from $\mathcal{X}$ and $\mathcal{Y}$.

4. Sparse Generalized Eigenvalue Formulation

The variational formulation for the sparse generalized eigenvalue problem is given by

$$\max_{\mathbf{x}} \{\mathbf{x}^T\mathbf{A}\mathbf{x} \ : \ \mathbf{x}^T\mathbf{B}\mathbf{x} = 1, ||\mathbf{x}||_0 \leq k\}, \tag{2}$$

where $1 \leq k \leq n$. Eq. (2) is non-convex, NP-hard and therefore intractable.[1] In addition, the cardinality constraint aggravates the problem as it cannot be han-

[1]The generalized eigenvalue problem in Eq. (1) is not a convex program for $\mathbf{A} \in \mathbb{S}^n \setminus \{\mathbf{A} \colon \mathbf{A} \preceq 0\}$. In addition, the constraint set is also non-convex. However, efficient algorithms exist which can find a global optimum in polynomial time. It should be noted that any additional constraint to Eq. (1) or replacement of $\mathbf{x}^T\mathbf{B}\mathbf{x} = 1$ by any other non-quadratic convex/non-convex constraint makes the problem NP-hard. This is true even when $\mathbf{A} \succeq 0$ as it results in the *maximization of a convex objective*. In this respect, the generalized eigenvalue problem is *very special*.

dled directly (it is discontinuous and combinatorially hard). To get a handle on the cardinality constraint, usually the ℓ_1-norm approximation is used. Though it makes the constraint set convex, it does not simplify the problem any further as the additional ℓ_1 constraint destroys the special nature of Eq. (1). This is the approximation used by SCoTLASS for $\mathbf{B}=\mathbf{I}$ leading to a locally convergent algorithm. A convex method can be achieved by starting with this ℓ_1-norm approximation, performing *lifting* (Lovász & Schrijver, 1991) of Eq. (2), then relaxing the rank constraint that follows. This leads to the following SDP,

$$\max_{\mathbf{X}\succeq 0}\{\mathrm{tr}(\mathbf{AX})\,:\,\mathrm{tr}(\mathbf{BX})=1, \mathbf{1}^T|\mathbf{X}|\mathbf{1}\le k\,\mathrm{tr}(\mathbf{X})\}, \quad (3)$$

where $|\mathbf{X}|_{ij}\triangleq|\mathbf{X}_{ij}|$. With $\mathbf{B}=\mathbf{I}$, we obtain DSPCA (d'Aspremont et al., 2005). Though DSPCA is a convex method, it is computationally very intensive as the general purpose interior-point methods scale as $O(n^6\log(1/\epsilon))$, where ϵ is the required accuracy on the optimal value. For large-scale problems, first-order methods can be used which scale as $O(n^4\sqrt{\log n}/\epsilon)$. Since the *only* convex approach possible is through SDP relaxation which is prohibitively expensive for large n, one should be content with either local methods or expensive mixed-integer programs.[2] Another method to solve the sparse maximum eigenvalue program is by solving a *sequence of sparse minimum eigenvalue programs* which are convex when $\mathbf{A}\succeq 0$, $\mathbf{B}\succ 0$ and relaxing $||\mathbf{x}||_0\le k$ by $||\mathbf{x}||_1\le k$. Clearly, this brute force method is prohibited for large n.

The proposed method is motivated by the following observations. First, since the ℓ_1-norm relaxation does not simplify the original problem, we can use a better approximation to cardinality to improve sparsity. Second, since the only way to obtain a convex approximation to this problem scales terribly in size, it is wise to use a locally convergent algorithm with better scalability. To this end, we consider a related problem of Eq. (2) given by

$$\max_{\mathbf{x}}\{\mathbf{x}^T\mathbf{Ax}-\rho\,||\mathbf{x}||_0\;:\;\mathbf{x}^T\mathbf{Bx}\le 1\}, \quad (4)$$

where $\rho>0$. Note that we replaced the quadratic equality constraint with an inequality constraint. The equality of Eq. (2) (with $||\mathbf{x}||_0\le k$ absorbed into the objective function) and Eq. (4) for $\mathbf{B}=\mathbf{I}$ is discussed in El Ghaoui, (2006, Theorem 1), which says that depending on the value of ρ, both these programs either have the same optimal solution or have trivial solutions. In this paper, we consider Eq. (4) as the *sparse generalized eigenvalue problem.* Since Eq. (4) is not tractable because of $||\mathbf{x}||_0$, we approximate $||\mathbf{x}||_0$ by $\sum_{i=1}^n\log(\varepsilon+|x_i|)$ as proposed by Weston et al. (2003), leading to the following program,

$$\phi(\rho):=\max_{\mathbf{x}^T\mathbf{Bx}\le 1}\{\mathbf{x}^T\mathbf{Ax}-\rho\sum_{i=1}^n\log(\varepsilon+|x_i|)\}, \quad (5)$$

where $0\le\varepsilon\ll 1$ avoids problems when one of the x_i is zero. The following proposition shows that the global maximizers of Eq. (4) and Eq. (5) have almost the same cardinality.

Proposition 1. *Let $\hat{\mathbf{x}}$ and $\check{\mathbf{x}}$ be the maximizers of Eq. (4) and Eq. (5) respectively. If $\check{\mathbf{x}}$ is independent of the choice of ε and $\delta\le|\check{x}_i|<\infty$ for some fixed $\delta>0$, where $i=\{j:|\check{x}_j|\ne 0, 1\le j\le n\}$, then $||\check{\mathbf{x}}||_0\le||\hat{\mathbf{x}}||_0+O\left(\frac{1}{\log\varepsilon}\right)$.*

Proof. The proof follows the method in Weston et al. (2003). Since $\check{\mathbf{x}}$ is the maximizer of Eq. (5), we have

$$\check{\mathbf{x}}^T\mathbf{A}\check{\mathbf{x}}-\rho\sum_{i=1}^n\log(\varepsilon+|\check{x}_i|)\ge\hat{\mathbf{x}}^T\mathbf{A}\hat{\mathbf{x}}-\rho\sum_{i=1}^n\log(\varepsilon+|\hat{x}_i|), \quad (6)$$

which is equivalent to

$$||\hat{\mathbf{x}}||_0-||\check{\mathbf{x}}||_0+\sum_{\{i\,:\,\check{x}_i\ne 0\}}\frac{\log(\varepsilon+|\check{x}_i|)}{\log\varepsilon} -\sum_{\{i\,:\,\hat{x}_i\ne 0\}}\frac{\log(\varepsilon+|\hat{x}_i|)}{\log\varepsilon}\ge\frac{\check{\mathbf{x}}^T\mathbf{A}\check{\mathbf{x}}-\hat{\mathbf{x}}^T\mathbf{A}\hat{\mathbf{x}}}{\rho\log\varepsilon}. \quad (7)$$

We see that

$$||\check{\mathbf{x}}||_0\le||\hat{\mathbf{x}}||_0+\psi(\varepsilon), \quad (8)$$

where

$$\psi(\varepsilon) = \frac{\hat{\mathbf{x}}^T\mathbf{A}\hat{\mathbf{x}}-\check{\mathbf{x}}^T\mathbf{A}\check{\mathbf{x}}}{\rho\log\varepsilon}+\sum_{\{i\,:\,\check{x}_i\ne 0\}}\frac{\log(\varepsilon+|\check{x}_i|)}{\log\varepsilon} -\sum_{\{i\,:\,\hat{x}_i\ne 0\}}\frac{\log(\varepsilon+|\hat{x}_i|)}{\log\varepsilon}\sim O\left(\frac{1}{\log\varepsilon}\right). \quad \square$$

The proposed approximation can be interpreted as defining a limiting Student's t-distribution prior over $\mathbf{x}$ (leading to an improper prior) given by $p(\mathbf{x})\propto\prod_{i=1}^n\frac{1}{|x_i|}$ and computing its negative log-likelihood. Tipping (2001) showed that this choice of prior leads to sparse representation and demonstrated its validity for sparse kernel expansions in the Bayesian framework. It can be shown that sparse FDA based on Eq. (5) is very similar to the relevance vector machine (RVM). From now on, we will assume that ε is equal to the machine precision and in fact use $\varepsilon=0$ while solving the approximate problem.

[2]DSPCA provides a unique global optima, which is only an approximation to the true solution. Ideally, the solution that is obtained should be projected back onto the true (unrelaxed) constraint set to achieve a feasible solution. This is usually done by random projection.

4.1. Solution by D.C. Programming

Consider Eq. (5) with $\varepsilon = 0$ which can be written as

$$\max_{\mathbf{x},\mathbf{y}} \{\mathbf{x}^T\mathbf{A}\mathbf{x} - \rho \sum_{i=1}^{n} \log y_i \ : \ (\mathbf{x},\mathbf{y}) \in \mathscr{F}\}, \qquad (9)$$

where $\mathscr{F} = \{(\mathbf{x},\mathbf{y}) \ : \ \mathbf{x}^T\mathbf{B}\mathbf{x} \leq 1\,, -\mathbf{y} \preceq \mathbf{x} \preceq \mathbf{y}\}$. This can be written as

$$-\phi(\rho) := \min_{\mathbf{x},\mathbf{y}} \mathcal{I}_{\mathscr{F}}(\mathbf{x},\mathbf{y}) - \left(\mathbf{x}^T\mathbf{A}\mathbf{x} - \rho \sum_{i=1}^{n} \log y_i\right), \qquad (10)$$

where the convex function $\mathcal{I}_{\mathscr{F}}$ is called the *indicator function* of the convex set, $\mathscr{F}$ given by

$$\mathcal{I}_{\mathscr{F}}(\mathbf{x},\mathbf{y}) = \begin{cases} 0 & (\mathbf{x},\mathbf{y}) \in \mathscr{F} \\ \infty & (\mathbf{x},\mathbf{y}) \notin \mathscr{F}. \end{cases} \qquad (11)$$

From now onwards, we assume that $\mathbf{A} \succeq 0$. Based on the discussion on d.c. programming in Appendix A, it can be seen from Eq. (10) that $g(\mathbf{x},\mathbf{y}) = \mathcal{I}_{\mathscr{F}}(\mathbf{x},\mathbf{y})$ and $h(\mathbf{x},\mathbf{y}) = \mathbf{x}^T\mathbf{A}\mathbf{x} - \rho\sum_{i=1}^{n}\log y_i$, which are convex in $\mathbf{x}$ and $\mathbf{y}$. So, Eq. (10), which is an approximation to Eq. (2), is a d.c. program. Using the d.c. minimization algorithm (DCA) of Tao and An (1998) (see Appendix A),[3] Eq. (10) can be solved efficiently for a locally optimal solution. For the problem at hand, h is differentiable and combining the two DCA steps (3 and 4 of Algorithm 2 in Appendix A) for each $l \in \mathbb{N}$, we get $(\mathbf{x}_{l+1},\mathbf{y}_{l+1}) \in \partial g^*(\nabla h(\mathbf{x}_l,\mathbf{y}_l)) = \arg\max_{\mathbf{x},\mathbf{y}}\{(\mathbf{x}^T \ \ \mathbf{y}^T)\nabla h(\mathbf{x}_l,\mathbf{y}_l) - g(\mathbf{x},\mathbf{y})\}$. Therefore, $(\mathbf{x}_{l+1},\mathbf{y}_{l+1}) \in \arg\max_{(\mathbf{x},\mathbf{y})\in\mathscr{F}}\{(\mathbf{x}^T \ \ \mathbf{y}^T)\nabla h(\mathbf{x}_l,\mathbf{y}_l)\}$. With $\nabla h(\mathbf{x}_l,\mathbf{y}_l) = 2\mathbf{A}\mathbf{x}_l - \rho(\mathbf{1}\bullet\mathbf{y}_l)$, where $(\mathbf{a}\bullet\mathbf{b})_i = a_i/b_i$, we get the convex program, $(\mathbf{x}_{l+1},\mathbf{y}_{l+1}) = \arg\max_{(\mathbf{x},\mathbf{y})\in\mathscr{F}}\{\mathbf{x}_l^T\mathbf{A}\mathbf{x} - \frac{\rho}{2}(\mathbf{1}\bullet\mathbf{y}_l)^T\mathbf{y}\}$. By change of variables with $\mathbf{x} = \mathbf{x}_l \circ \bar{\mathbf{x}}$ and $\mathbf{y} = \mathbf{y}_l \circ \bar{\mathbf{y}}$ where $(\mathbf{a}\circ\mathbf{b})_i = a_ib_i$, this reduces to

$$\begin{aligned} \max_{\bar{\mathbf{x}},\bar{\mathbf{y}}} \quad & \mathbf{x}_l^T\mathbf{A}\mathbf{D}(\mathbf{x}_l)\bar{\mathbf{x}} - \frac{\rho}{2}\mathbf{1}^T\bar{\mathbf{y}} \\ \text{s.t.} \quad & \bar{\mathbf{x}}^T\mathbf{D}(\mathbf{x}_l)\mathbf{B}\mathbf{D}(\mathbf{x}_l)\bar{\mathbf{x}} \leq 1,\ -\bar{\mathbf{y}} \preceq \bar{\mathbf{x}} \preceq \bar{\mathbf{y}}, \end{aligned} \qquad (12)$$

with $\mathbf{x}_{l+1} = \mathbf{x}_l \circ \bar{\mathbf{x}}^*$ and $\mathbf{y}_{l+1} = \mathbf{y}_l \circ \bar{\mathbf{y}}^*$ where $(\bar{\mathbf{x}}^*, \bar{\mathbf{y}}^*)$ is the maximizer of Eq. (12). Removing $\bar{\mathbf{y}}$ from Eq. (12) yields the sparse generalized eigenvalue program in Algorithm 1, which is a sequence of QCQPs with a multiplicative update. The multiplicative nature of the update makes the weighting for less relevant features decay rapidly to zero and so at the termination, $\bar{\mathbf{x}}^* \in \{0,1\}^n$ provides the (locally) optimal sparsity pattern.

[3]The successive linear approximation (SLA) for concave minimization (Mangasarian, 1997) and concave-convex procedure (CCCP) (Yuille & Rangarajan., 2003) can be seen as special cases of d.c. programming.

Algorithm 1 Sparse Generalized Eigenvalue Program

```
Require: $\mathbf{A} \succeq 0$, $\mathbf{B} \succ 0$ and $\rho > 0$
1: Choose $\mathbf{x}_0 \in \mathscr{F}$ arbitrarily
2: repeat
3:
       $\bar{\mathbf{x}}^* = \arg\max_{\bar{\mathbf{x}}} \ \mathbf{x}_l^T\mathbf{A}\mathbf{D}(\mathbf{x}_l)\bar{\mathbf{x}} - \frac{\rho}{2}||\bar{\mathbf{x}}||_1$
       s.t. $\bar{\mathbf{x}}^T\mathbf{D}(\mathbf{x}_l)\mathbf{B}\mathbf{D}(\mathbf{x}_l)\bar{\mathbf{x}} \leq 1$     (13)
4:     $\mathbf{x}_{l+1} = \mathbf{x}_l \circ \bar{\mathbf{x}}^*$
5: until $\mathbf{x}_{l+1} = \mathbf{x}_l$
6: return $\mathbf{x}_l$, $\bar{\mathbf{x}}^*$
```

Using $\bar{\mathbf{x}}^*$, the variational re-normalization (Moghaddam et al., 2007, §2, Proposition 2) can be applied to $\mathbf{x}_l$ by solving the cardinality unconstrained problem, $\bar{\phi}(\rho) := \max_{\mathbf{x}}\{\mathbf{x}^T\mathbf{D}(\bar{\mathbf{x}}^*)\mathbf{A}\mathbf{D}(\bar{\mathbf{x}}^*)\mathbf{x} \ : \ \mathbf{x}^T\mathbf{B}\mathbf{x} = 1\}$, which guarantees that $\bar{\phi}(\rho) \geq \phi(\rho)$ and $||\bar{\mathbf{x}}^*||_0 = ||\tilde{\mathbf{x}}||_0$ where $\tilde{\mathbf{x}}$ is the maximizer of $\bar{\phi}(\rho)$.

The following proposition shows that when $\rho = 0$, the local optimal solution obtained by Algorithm 1 is the global optimal solution and equals that of the generalized eigenvalue problem of Eq. (1).

Proposition 2. *Let $\rho = 0$, $\mathbf{x}_l$ be the output of Algorithm 1 and $\check{\mathbf{x}} = \arg\max_{\mathbf{x}}\{\mathbf{x}^T\mathbf{A}\mathbf{x} \ : \ \mathbf{x}^T\mathbf{B}\mathbf{x} = 1\}$. Then $\mathbf{x}_l^T\mathbf{A}\mathbf{x}_l = \lambda_{max}(\mathbf{B}^{-1}\mathbf{A})$ and $\mathbf{x}_l = \check{\mathbf{x}}$, where $\lambda_{max}(\mathbf{B}^{-1}\mathbf{A})$ is the maximum eigenvalue of $\mathbf{B}^{-1}\mathbf{A}$.*

Proof. Solving the Lagrangian of Eq. (13) with $\rho = 0$ yields $\bar{\mathbf{x}}^* = \mathbf{D}(\mathbf{x}_l)^{-1}\mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l/\sqrt{\mathbf{x}_l^T\mathbf{A}\mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l}$ with $\mathbf{x}_{l+1} = \mathbf{D}(\mathbf{x}_l)\bar{\mathbf{x}}^*$. At the stationary point, $\mathbf{x}_{l+1} = \mathbf{x}_l$ gives $\left(\sqrt{\mathbf{x}_l^T\mathbf{A}\mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l}\right)\mathbf{x}_l = \mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l$. Defining $\mu = \sqrt{\mathbf{x}_l^T\mathbf{A}\mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l}$, we have $\mu = \sqrt{\mu\mathbf{x}_l^T\mathbf{A}\mathbf{x}_l} \Rightarrow \mu = \mathbf{x}_l^T\mathbf{A}\mathbf{x}_l$. Since $\mathbf{x}_l^T\mathbf{A}\mathbf{x}_l$ is the maximum value of Algorithm 1 and $\mathbf{B}^{-1}\mathbf{A}\mathbf{x}_l = \mu\mathbf{x}_l$, equivalently, they can be written as $\mu = \max_{\mathbf{x}_l^T\mathbf{B}\mathbf{x}_l=1}\mathbf{x}_l^T\mathbf{A}\mathbf{x}_l$, which is the generalized eigenvalue problem of Eq. (1). So, $\mu = \mathbf{x}_l^T\mathbf{A}\mathbf{x}_l = \lambda_{max}(\mathbf{B}^{-1}\mathbf{A})$ and $\mathbf{x}_l = \check{\mathbf{x}}$. □

Algorithm 1 requires the knowledge of ρ, which determines sparsity. In a supervised learning setup like FDA, ρ can be chosen by cross-validation whereas, in an unsupervised setup like PCA/CCA, Algorithm 1 has to be solved for various ρ and the solution with required cardinality is selected.

5. Sparse PCA by D.C. Programming

Based on the sparse generalized eigenvalue program in Algorithm 1, we propose our sparse PCA algorithm (DC-PCA) with $\mathbf{B} = \mathbf{I}$ and $\mathbf{A}$ being the covariance matrix. Usually, the sparse eigenvectors of $\mathbf{A}$ are obtained by applying Algorithm 1 on the sequence of deflated matrices given by $\{\mathbf{A}_0 = \mathbf{A};\ \mathbf{A}_{i+1} = \mathbf{A}_i - (\mathbf{u}_i^T\mathbf{A}_i\mathbf{u}_i)\mathbf{u}_i\mathbf{u}_i^T\}$ with the same (or different) ρ depending on the sparsity requirement, where $\mathbf{u}_i$ is the output of Algorithm 1 with $\mathbf{A} = \mathbf{A}_i$. This is appropriate only when $\mathbf{u}_i^T\mathbf{u}_j = 0,\ i \neq j$. Otherwise, there is a possibility that $\mathbf{A}_i \prec 0$, for some i. So, one should be careful in computing the cumulative variance explained by $\mathbf{u}_i$'s as $\sum_i \mathbf{u}_i^T\mathbf{A}_i\mathbf{u}_i$. Instead, the sequence of deflated matrices should be computed as $\{\mathbf{A}_0 = \mathbf{A};\ \mathbf{A}_{i+1} = \mathbf{A}_i - (\mathbf{v}_i^T\mathbf{A}_i\mathbf{v}_i)\mathbf{v}_i\mathbf{v}_i^T\}$, where $\mathbf{v}_i = \mathbf{u}_i - \mathcal{P}_{\mathscr{S}_{i-1}}\mathbf{u}_i$. $\mathcal{P}_{\mathscr{S}_{i-1}}\mathbf{u}_i$ represents the orthogonal projection of $\mathbf{u}_i$ onto the subspace, $\mathscr{S}_{i-1}$, spanned by $\{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{i-1}\}$ with $\mathbf{v}_0 = \mathbf{u}_0$. The cumulative variance is then calculated as $\sum_i \mathbf{v}_i^T\mathbf{A}_i\mathbf{v}_i$.

The following proposition shows that for $\rho = 0$, DC-PCA reduces to the *power iteration algorithm.*

Proposition 3. *Let* $\mathbf{B} = \mathbf{I}$*,* $\mathbf{A} \succeq 0$ *and* $\rho = 0$*. Then DC-PCA algorithm (Algorithm 1) is the power method for eigenvalue computation.*

Proof. From Proposition 2, $\mathbf{x}_{l+1} = \mathbf{A}\mathbf{x}_l/||\mathbf{A}\mathbf{x}_l||_2$ and at $\mathbf{x}_{l+1} = \mathbf{x}_l$, $\lambda_{max}(\mathbf{A}) = \mathbf{x}_l^T\mathbf{A}\mathbf{x}_l = ||\mathbf{A}\mathbf{x}_l||_2$. □

In the following, we compare our sparse PCA formulation (DC-PCA) to SCoTLASS and SPCA.

5.1. Comparison to SCoTLASS

The SCoTLASS program given by $\max_{\mathbf{x}}\{\mathbf{x}^T\mathbf{A}\mathbf{x} - \rho||\mathbf{x}||_1 : ||\mathbf{x}||_2^2 = 1\}$ is also non-convex because of convex maximization. Using DCA, we get an algorithm which is the same as Algorithm 1 except that Eq. (13) is replaced by $\bar{\mathbf{x}}^* = \arg\max_{\bar{\mathbf{x}}}\{\mathbf{x}_l^T\mathbf{A}\bar{\mathbf{x}} - \frac{\rho}{2}||\bar{\mathbf{x}}||_1 : ||\bar{\mathbf{x}}||_2^2 \leq 1\}$ with $\mathbf{x}_{l+1} = \bar{\mathbf{x}}^*$. Mainly, this differs from DC-PCA in the multiplicative update. Let us assume that $(\mathbf{x}_l)_i = 0$ for some l. For DC-PCA, this ensures that $(\mathbf{x}_j)_i = 0, \forall\, j > l$ which is not guaranteed for SCoTLASS. Since $(\mathbf{x}_j)_i = 0, \forall\, j > l$, $||\mathbf{x} \circ \mathbf{x}_j||_2^2 = 1$ constrains the points to the surface of a hyper-ellipsoid that is degenerate in the i^{th} dimension. The multiplicative update in DC-PCA ensures faster convergence of an irrelevant feature to zero than that in SCoTLASS, thus providing better sparsity. This is not surprising as a better approximation to $||.||_0$ is used in DC-PCA. SCoTLASS also differs from DC-PCA in the sense that its sparse eigenvectors are orthonormal, unlike in DC-PCA. When $\rho = 0$, like DC-PCA, SCoTLASS also reduces to the *power iteration algorithm.*

5.2. Comparison to SPCA

Zou et al., (2004, Theorem 1) derive PCA in a regression framework where sparsity is introduced by adding the ℓ_l penalty to the regression problem. We consider their "direct sparse approximation" formulation rather than the "self-contained" regression-type criterion (see §3.1 and §3.2 in Zou et al. (2004)). The corresponding optimization problem is given by $\min_{\mathbf{x}}\{||\mathbf{p}_i - \mathbf{Q}\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_2^2 + \lambda_1||\mathbf{x}||_1\}$ where $\mathbf{Q} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$ is the data matrix and $\mathbf{p}_i = (\mathbf{U}\mathbf{\Lambda})_i$ is the i^{th} principal component. This can be treated in the Bayesian setting as $\mathbf{p}_i|\mathbf{x}, \sigma^2 \sim \mathcal{G}(\mathbf{Q}\mathbf{x}, \sigma^2\mathbf{I})$ and $\mathbf{x}|\beta^2, \gamma \sim \mathcal{G}(\mathbf{0}, \beta^2\mathbf{I})\prod_i \exp(-\gamma|x_i|)$ where the prior on $\mathbf{x}$ is a product distribution of circular Gaussian and product of Laplacian densities. The parameters λ and λ_1 are related to σ^2, β^2 and γ. As aforementioned, our method can be interpreted as defining an improper prior over $\mathbf{x}$, which promotes sparsity. We use $p(\mathbf{x}) \propto \prod_i \frac{1}{|x_i|}$ (instead of $\prod_i \exp(-\gamma|x_i|)$) as the prior so that $\mathbf{x}|\beta^2 \sim \mathcal{G}(\mathbf{0},^2\mathbf{I})p(\mathbf{x})$, resulting in

$$\min_{\mathbf{x}} ||\mathbf{p}_i - \mathbf{Q}\mathbf{x}||_2^2 + \lambda||\mathbf{x}||_2^2 + \lambda_1\sum_i \log|x_i|. \qquad (14)$$

Zou et al., (2004, Theorem 1) show that with $\lambda_1 = 0$, the minimizer of Eq. (14) is a scaled version $\mathbf{V}_i$. When $\lambda_1 \neq 0$, the additional term promotes sparsity. Since this problem is equivalent to Eq. (5), we claim that DC-PCA provides sparser solutions than SPCA. It is to be noted that SPCA is not extendable to other settings like FDA/CCA whereas our formulation is more general and does extend to other settings also.

6. Experiments & Results

In this section, we illustrate the effectiveness of the proposed method (DC-PCA) in terms of sparsity and scalability on different real-life datasets. Since SPCA[4] and DSPCA[5] have demonstrated improved performance over simple thresholding and SCoTLASS, we choose these methods as our baselines to compare against the performance of our method. Also, based on the discussion in §5.1, it should be clear that DC-PCA performs better than SCoTLASS. Because of the non-availability of a code setup for GSPCA, we are not able to compare our results with it. The results show that our method has *better scalability* and achieves *more sparsity* than SPCA and DSPCA, while explaining as much variance as possible.

6.1. Pit Props Data

The pit props dataset (Jeffers, 1967) has become a standard benchmark example to test sparse PCA algorithms. The first 6 principal components (PCs)

[4]LARS-based Elastic-net SPCA MATLAB toolbox (Sjöstrand, 2005) was used to solve for SPCA.

[5]DSPCA software is available at `http://www.princeton.edu/~aspremon/DSPCA.htm`.

Table 1. Loadings for first three principal components (PCs) of the pit props data. Original SPCA and DSPCA loadings are taken from Zou et al. (2004) and d'Aspremont et al. (2005) respectively.

	PC	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}
SPCA	1	-.477	-.476	0	0	.177	0	-.250	-.344	-.416	-.400	0	0	0
	2	0	0	.785	.620	0	0	0	-.021	0	0	0	.013	0
	3	0	0	0	0	.640	.589	.492	0	0	0	0	0	-.015
DSPCA	1	-.560	-.583	0	0	0	0	-.263	-.099	-.371	-.362	0	0	0
	2	0	0	.707	.707	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	-.793	-.610	0	0	0	0	0	.012
DC-PCA	1	0.449	0.459	0	0	0	0	0.374	0.332	0.403	0.419	0	0	0
	2	0	0	0.707	0.707	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	0.816	0.578	0	0	0	0	0	0

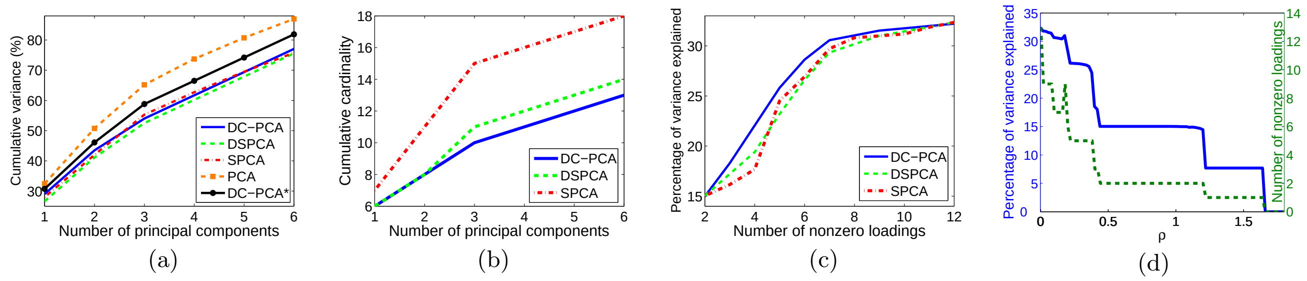

Figure 1. Pit props: (a) cumulative variance (b) cumulative cardinality for first 6 sparse principal components (PCs) (c) percentage of explained variance (PEV) vs. sparsity for the first PC (d) dependence of sparsity and PEV on ρ for the first PC. DC-PCA* in (a) represents DC-PCA evaluated at SPCA's sparsity pattern of $(7, 4, 4, 1, 1, 1)$.

capture 87% of the total variance and so all these other methods compare their explanatory power using 6 sparse PCs. Table 1 shows the first 3 PCs and their loadings for SPCA, DSPCA and DC-PCA. SPCA captures 75.8% of the variance with a cardinality pattern of $(7, 4, 4, 1, 1, 1)$ (total of 18 non-zero loadings) while DSPCA captures 75.5% with a sparsity pattern of $(6, 2, 3, 1, 1, 1)$, totaling 14 non-zero loadings. We used a sparsity pattern of $(6, 2, 2, 1, 1, 1)$ with a total of *only* 13 non-zero loadings and capture 77.1% of the total variance. In addition, when SPCA's sparsity pattern of $(7, 4, 4, 1, 1, 1)$ is used, DC-PCA (shown as DC-PCA* in Figure 1(a)) performs significantly better than SPCA and DSPCA. Comparing the cumulative variance and cumulative cardinality, Figure 1(a–b) show that DC-PCA explains more variance with fewer non-zero loadings than SPCA and DSPCA. We restate here that one should be careful in computing the cumulative variance when dealing with multiple sparse PCs (see the discussion on matrix deflation in §5). For the first PC, Figure 1(c) shows that DC-PCA consistently explains more variance with better sparsity than SPCA and DSPCA. Figure 1(d) shows the variation of sparsity and explained variance w.r.t. ρ for the first PC. This plot summarizes the method for setting ρ wherein the algorithm is run for various ρ. The value of ρ that achieves the required sparsity is chosen and its corresponding variance is calculated.

6.2. Colon Cancer Data

The colon cancer data (Alon et al., 1999) consists of 62 tissue samples (22 normal and 40 cancerous) with the gene expression profiles of $n = 2000$ genes extracted from DNA micro-array data. The high-dimensionality of the dataset makes it a suitable candidate for studying the performance of sparse PCA algorithms where feature selection is needed to get interpretable results. Its first 10 PCs explain 80% of the total variance. Due to computational reasons, we consider only the first 5 PCs in our study, which explain 70% of the total variance. By comparing the cumulative variance and cumulative cardinality for the first 5 PCs, Figure 2(a–b) show that DC-PCA explains significantly more variance with fewer non-zero loadings than SPCA. For 8% loss in the explained variance w.r.t. PCA (from 70% to 62%), DC-PCA requires $\sim$ 40% fewer genes to sufficiently reconstruct the first 5 PCs. Because of the poor scalability of DSPCA for large matrix sizes (see §6.4), the experiments on DSPCA could not be completed in reasonable time. So the results do not include the comparison to DSPCA.

6.3. Leukemia Data

Leukemia data (Golub et al., 1999) consists of a training set of 38 samples (27 ALL and 11 AML, two variants of leukemia) from bone marrow specimens and a

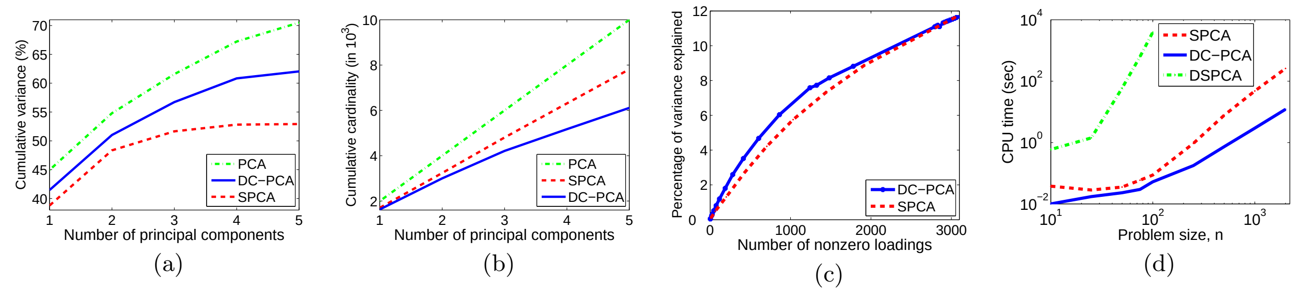

Figure 2. Colon cancer: (a) cumulative variance (b) cumulative cardinality for first 5 sparse principal components (PC). Leukemia: (c) percentage of variance explained vs. sparsity for the first PC. (d) CPU time vs. problem size for randomly chosen problems. (a–c) show that DC-PCA explains more variance with fewer non-zero loadings than SPCA.

test set of 34 samples (20 ALL and 14 AML). This dataset has been used widely in a classification setting where the goal is to distinguish between two variants of leukemia. We chose this dataset because of its large dimensionality. All samples have 7129 features, corresponding to some normalized gene expression value extracted from the micro-array image. We test the performance of DC-PCA and SPCA on this high dimensional dataset. Again, for scalability reasons, DSPCA is not considered for the performance comparison. Figure 2(c) shows the comparative performance (explained variance vs. sparsity) of DC-PCA and SPCA for the first PC. In this case, too, DC-PCA explains more variance (though marginal) with fewer variables compared to SPCA. Though this dataset is not as interesting as the colon cancer dataset because the amount of variance explained by the first PC is just 15%, we used it to show that our algorithm is scalable to high-dimensional datasets.

6.4. Computing Time vs. Problem Size

DC-PCA is a sequence of QCQPs with worst-case complexity of $O(mn^3)$, which is the same as that of SPCA as opposed to $O(n^{5.5})$ of DSPCA. Here, m is the number of iterations before convergence. To empirically compute the running time complexity of these methods, we ran[6] these algorithms on randomly chosen problems of size n ranging from 10 to 2000 for 5 different values of ρ and k (similar to the setup in d'Aspremont et al., (2005, §6.4)). Figure 2(d) shows the plot of average CPU time vs. n for these methods with the empirical complexity growing as $O(n^p)$ where $p = 1.46$ for DC-PCA, $p = 1.91$ for SPCA and $p = 3.92$ for DSPCA. This shows that DC-PCA scales much better to large-dimensional problems than SPCA and DSPCA and has to be preferred over these methods as it also has better sparsity vs. explained variance performance.

[6]The experiment was carried out on a Linux 3 GHz, 4 GB RAM workstation.

7. Conclusion & Future Work

We have proposed a sparse generalized eigenvalue algorithm using d.c. programming, the special case of which yields sparse PCA algorithm (DC-PCA). The main advantages of this algorithm are *improved sparsity* and *better scalability* than SPCA and DSPCA. Sparsity is improved by using a better approximation to the cardinality constraint. We have experimentally demonstrated on real-life data of varying dimensionality that the proposed algorithm (DC-PCA) explains more variance with sparse features than SPCA and DSPCA at much better computational speed (low CPU time).

One of the drawbacks of both DC-PCA and SPCA is the difficulty in setting the regularization parameter to attain a given sparsity, which is not the case with DSPCA wherein the sparsity requirement is explicitly mentioned. But, both these methods, though non-convex, have better scalability than the convex SDP formulation of DSPCA.

In the future, we plan to investigate this paradigm for canonical correlation analysis which has interesting applications in dictionary translation, semantic learning of multimedia content, music annotation, etc. Also, we would like to investigate path following techniques to efficiently set the regularization parameter.

Appendix A. D.C. Programming

Let $\mathscr{C}$ be a convex set of $\mathbb{R}^n$. A real valued function $f : \mathscr{C} \to \mathbb{R}$ is called a d.c. on $\mathscr{C}$, if there exist two *convex* functions $g, h : \mathscr{C} \to \mathbb{R}$ such that f can be expressed in the form $f(\mathbf{x}) = g(\mathbf{x}) - h(\mathbf{x})$, $\mathbf{x} \in \mathscr{C}$. Optimization problems of the form $\min_{\mathbf{x}} \{f_0(\mathbf{x}) : \mathbf{x} \in \mathscr{C}, f_i(\mathbf{x}) \leq 0, i = 1, \ldots, m\}$, where $f_i = g_i - h_i$, $i = 0, \ldots, m$, are d.c. functions are called *d.c. programs.* Though there exist global optimization approaches such as branch and bound, cutting planes to solve d.c. programs, they are not scalable to large-scale problems. A robust and

Algorithm 2 D.C. Minimization Algorithm (DCA)

```
Require: $g, h, \delta$
1: Choose $\mathbf{x}_0 \in \operatorname{dom} g$ arbitrarily
2: for $l \in \mathbb{N}$ do
3:     select $\mathbf{z}_l \in \partial h(\mathbf{x}_l)$ arbitrarily
4:     select $\mathbf{x}_{l+1} \in \partial g^*(\mathbf{z}_l)$ arbitrarily
5:     if $\min\left(|(\mathbf{x}_{l+1}-\mathbf{x}_l)_i|, \left|\frac{(\mathbf{x}_{l+1}-\mathbf{x}_l)_i}{(\mathbf{x}_l)_i}\right|\right) \leq \delta, \forall i$ then
6:         return $\mathbf{x}_{l+1}$
7:     end if
8: end for
```

efficient algorithm (DCA) is proposed in Tao and An (1998), which is a primal-dual subgradient method for solving general (large-scale) d.c. programs. Here, we skip details and directly present DCA. We refer the interested reader to Tao and An (1998) for more details on the theoretical guarantees of DCA and to Horst and Thoai (1999) for an overview of d.c. programming.

For a lower semi-continuous, proper convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$, we use the following standard notation (Rockafellar, 1970): domain of f, $\operatorname{dom} f = \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) < \infty\}$, conjugate function of f, $f^*(\mathbf{z}) = \sup_{\mathbf{x}\in\mathbb{R}^n}\{\mathbf{z}^T\mathbf{x} - f(\mathbf{x})\}$, subdifferential of f, $\partial f(\mathbf{z}) = \{\mathbf{w} \in \mathbb{R}^n : f(\mathbf{x}) \geq f(\mathbf{z}) + \mathbf{w}^T(\mathbf{x}-\mathbf{z}), \forall \mathbf{x} \in \mathbb{R}^n\}$ for $\mathbf{z} \in \mathbb{R}^n$. For differentiable functions, $\partial f(\mathbf{z}) = \{\nabla f(\mathbf{z})\}$. It also holds that $\partial f(\mathbf{x}) = \arg\max_{\mathbf{z}\in\mathbb{R}^n}\{\mathbf{x}^T\mathbf{z} - f^*(\mathbf{z})\}$, $\partial f^*(\mathbf{z}) = \arg\max_{\mathbf{x}\in\mathbb{R}^n}\{\mathbf{z}^T\mathbf{x} - f(\mathbf{x})\}$. The local convergence of DCA algorithm (Algorithm 2) is proven in Tao and An, (1998, Lemma 3.6, Theorem 3.7). In case of non-global solutions, one may restart DCA with a new initial point. However, Tao and An (1998) state that the DCA often converges to a global solution.

Acknowledgments

We thank the reviewers for their comments which greatly improved the paper. We wish to acknowledge support from the Fair Isaac Corporation and the University of California MICRO program.

References

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., & Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues. *Cell Biology*, *96*, 6745–6750.

Cadima, J., & Jolliffe, I. (1995). Loadings and correlations in the interpretation of principal components. *Applied Statistics*, 203–214.

d'Aspremont, A., El Ghaoui, L., Jordan, M. I., & Lanckriet, G. R. G. (2005). A direct formulation for sparse PCA using semidefinite programming. *Advances in Neural Information Processing Systems 17* (pp. 41–48). Cambridge, MA: MIT Press.

El Ghaoui, L. (2006). On the quality of a semidefinite programming bound for sparse principal component analysis. *arXive.org.*

Golub et al. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, *286*, 531–537.

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

Jeffers, J. (1967). Two case studies in the application of principal components. *Applied Statistics*, *16*, 225–236.

Jolliffe, I. T., Trendafilov, N. T., & Uddin, M. (2003). A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics*, *12*, 531–547.

Lovász, L., & Schrijver, A. (1991). Cones of matrices and set-functions and 0-1 optimization. *SIAM J. Optim.*, 166–190.

Mangasarian, O. L. (1997). Solution of general linear complementarity problems via nondifferentiable concave minimization. *Acta Mathematica Vietnamica*, *22*, 199–205.

Moghaddam, B., Weiss, Y., & Avidan, S. (2007). Spectral bounds for sparse PCA: Exact and greedy algorithms. *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press.

Rockafellar, R. T. (1970). *Convex analysis.* Princeton, NJ: Princeton University Press.

Sjöstrand, K. (2005). *Matlab implementation of LASSO, LARS, the Elastic Net and SPCA* (Technical Report). Informatics and Mathematical Modelling, Technical University of Denmark.

Tao, P. D., & An, L. T. H. (1998). D.c. optimization algorithms for solving the trust region subproblem. *SIAM J. Optim.*, 476–505.

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

Vandenberghe, L., & Boyd, S. (1996). Semidefinite programming. *SIAM Review*, 49–95.

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

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

Zou, H., Hastie, T., & Tibshirani, R. (2004). *Sparse principal component analysis* (Technical Report). Statistics Department, Stanford University.