The Sparse Eigenvalue Problem

Bharath K. Sriperumbudur BHARATHSV@UCSD.EDU
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

David A. Torres DATORRES@CS.UCSD.EDU
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

Gert R. G. Lanckriet GERT@ECE.UCSD.EDU
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

Editor:

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 is proved to exhibit global convergence behavior. The performance of the algorithm is empirically demonstrated on both sparse PCA (finding few relevant genes that explain as much variance as possible in a high-dimensional gene dataset) and sparse CCA (cross-language document retrieval and vocabulary selection for music retrieval) applications.

Keywords: Generalized eigenvalue problem, Principal component analysis, Canonical correlation analysis, Fisher discriminant analysis, D.c. program, Majorization-minimization, Global convergence analysis, Music annotation, Cross-language document retrieval.

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 Eq. (1) is simply referred to as an eigenvalue problem. Eigenvalue problems are so funda-

mental that they have applications in almost every area of science and engineering (Strang, 1986).

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 $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{A} \in \mathbb{S}^n$ (the set of symmetric matrices of size $n \times n$ defined over $\mathbb{R}$) and $\boldsymbol{B} \in \mathbb{S}^n_{++}$ (set of positive definite matrices of size $n \times n$ defined over $\mathbb{R}$). The variational formulation for the GEV problem in Eq. (1) is given by

$$\begin{aligned} \lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} = 1, \end{aligned} \tag{2}$$

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 Eq. (2) 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 the GEV problem in Eq. (2) are:

(a) Principal component analysis (PCA) (Hotelling, 1933; Jolliffe, 1986), a classic tool for data analysis, data compression and visualization, finds the direction of maximal variance in a given multivariate data set. This technique is used in dimensionality reduction wherein the ambient space in which the data resides is approximated by a low-dimensional subspace without significant loss of information. The variational form of PCA is obtained by choosing $\boldsymbol{A}$ to be the covariance matrix (which is a positive semidefinite matrix defined over $\mathbb{R}$) associated with the multivariate data and $\boldsymbol{B}$ to be the identity matrix in Eq. (2).

(b) Canonical correlation analysis (CCA) (Hotelling, 1936), similar to PCA, is also a data analysis and dimensionality reduction method. However, while PCA deals with only one data space $\mathcal{X}$ (from which the multivariate data is obtained), 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 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

$$\max_{\boldsymbol{w}_x \neq \boldsymbol{0},\, \boldsymbol{w}_y \neq \boldsymbol{0}} \frac{\boldsymbol{w}_x^T \boldsymbol{\Sigma}_{xy} \boldsymbol{w}_y}{\sqrt{\boldsymbol{w}_x^T \boldsymbol{\Sigma}_{xx} \boldsymbol{w}_x} \sqrt{\boldsymbol{w}_y^T \boldsymbol{\Sigma}_{yy} \boldsymbol{w}_y}}, \tag{3}$$

where $\boldsymbol{w}_x$ and $\boldsymbol{w}_y$ are the directions in $\mathcal{X}$ and $\mathcal{Y}$ along which the data is maximally correlated. $\boldsymbol{\Sigma}_{xx}$ and $\boldsymbol{\Sigma}_{yy}$ represent the covariance matrices for $\mathcal{X}$ and $\mathcal{Y}$ respectively and $\boldsymbol{\Sigma}_{xy} = \boldsymbol{\Sigma}_{yx}^T$ represents the cross-covariance matrix between $\mathcal{X}$ and $\mathcal{Y}$. Eq. (3) can be rewritten as

$$\max\{\boldsymbol{w}_x^T \boldsymbol{\Sigma}_{xy} \boldsymbol{w}_y \,:\, \boldsymbol{w}_x^T \boldsymbol{\Sigma}_{xx} \boldsymbol{w}_x = 1,\, \boldsymbol{w}_y^T \boldsymbol{\Sigma}_{yy} \boldsymbol{w}_y = 1\}, \tag{4}$$

which in turn can be written in the form of Eq. (2) with $\boldsymbol{A} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{0} \end{pmatrix}$, $\boldsymbol{B} = \begin{pmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_{yy} \end{pmatrix}$ and $\boldsymbol{x} = \begin{pmatrix} \boldsymbol{w}_x \\ \boldsymbol{w}_y \end{pmatrix}$.

(c) In the binary classification setting, Fisher discriminant analysis (FDA) finds a one dimensional subspace, $\boldsymbol{w} \in \mathbb{R}^n$, the projection of data onto which leads to maximal separation between the classes. Let $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ denote the mean vector and covariance matrix associated with class i. The variational formulation of FDA is given by

$$\max_{\boldsymbol{w} \neq \boldsymbol{0}} \quad \frac{(\boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2))^2}{\boldsymbol{w}^T(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)\boldsymbol{w}}, \tag{5}$$

which can be rewritten as

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \boldsymbol{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T\boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^T(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)\boldsymbol{w} = 1. \end{aligned} \tag{6}$$

Therefore, the FDA formulation is similar to Eq. (2) with $\boldsymbol{A} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$, called the *between-cluster variance* and $\boldsymbol{B} = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2$, called the *within-cluster variance*. For multi-class problems, similar formulations lead to multiple-discriminant analysis.

With advances in kernel methods (Schölkopf and Smola, 2002), the above mentioned techniques are even more popular as they can be kernelized (Schölkopf et al., 1998; Kuss and Graepel, 2003; Mika et al., 1999). Interestingly, the kernel version of these algorithms is also a GEV problem as in Eq. (2).[1]

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. In the following, we point to different applications where PCA/CCA/FDA is used and motivate the need for sparse solutions.

In many PCA applications, the coordinate axes have a physical interpretation; in biology, for example, each axis might correspond to a specific gene. In these cases, the interpretation of the principal components would be facilitated if they contained only few non-zero entries (or, loadings) while explaining most of the variance in the data. Moreover, in certain applications, e.g., financial asset trading strategies based on PCA techniques, the sparsity of the solution has important consequences, since fewer non-zero loadings imply fewer transaction costs. Consider a document translation application where two copies of a corpus of documents, one written in English and the other in German are given. The goal is to extract a low dimensional representation for each of the documents that explains most of the variation in the documents while providing a good translation between them. This is equivalent to representing the documents with a set of fewer words, which can be achieved by using sparse CCA. A similar motivation for sparse solutions can be provided for other CCA applications like music annotation, information retrieval, etc. In a classification setting like FDA, feature selection aids generalization performance by promoting sparse solutions. To summarize, sparse representations are generally desirable as they aid human understanding, reduce computational and economic costs and promote better generalization.

1. In addition to PCA, CCA and FDA, the popular spectral clustering algorithm also reduces to solving an eigenvalue problem. However, in this setting, one is interested in the eigenvector corresponding to the second smallest eigenvalue of the graph-Laplacian whereas in this paper, we focus on the generalized *maximum* eigenvalue problem shown in Eq. (2).

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

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} = 1, \, \|\boldsymbol{x}\|_0 \leq k, \end{aligned} \tag{7}$$

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 discrete optimization problem or as a continuous optimization problem after relaxing the cardinality constraint. In this paper, we follow the latter approach. The first step in solving Eq. (7) as a continuous optimization problem is to relax the cardinality constraint. One usual heuristic is to approximate $\|\boldsymbol{x}\|_0$ by $\|\boldsymbol{x}\|_1$ (see Section 2 for the details on notation), while another method is to approximate Eq. (7) by a semidefinite program (see Section 4 for details). Building on the earlier version of our work (Sriperumbudur et al., 2007), we approximate the cardinality constraint in Eq. (7) as the negative log-likelihood of a Student's t-distribution, which has been used earlier in many different contexts (Weston et al., 2003; Fazel et al., 2003; Candes et al., 2007). We then formulate this approximate problem as a d.c. (difference of convex functions) program and solve it using the majorization-minimization (MM) method (Hunter and Lange, 2004) resulting in a sequence of quadratically constrained quadratic programs (QCQP). Since, the proposed algorithm is an iterative procedure, using results from global convergence theory (Zangwill, 1969), we show that it is globally convergent. Since the idea behind the MM algorithm is very similar to that of the expectation-maximization (EM) algorithm (in fact, MM algorithm is a generalization of the EM algorithm), the above mentioned convergence result is similar to that of the EM algorithm (Wu, 1983). We would like to mention that the algorithm presented in this paper is more general than the one in Sriperumbudur et al. (2007) as it holds for any $\boldsymbol{A} \in \mathbb{S}^n$ unlike in Sriperumbudur et al. (2007), where $\boldsymbol{A}$ is assumed to be positive semidefinite.

As applications, we demonstrate the performance of the proposed algorithm on sparse PCA and sparse CCA problems. On the sparse PCA front, we compare our results to those of SPCA (Zou et al., 2006) and DSPCA (d'Aspremont et al., 2007) in terms of sparsity vs. explained variance on the "pit props" benchmark dataset and two high-dimensional gene datasets where the goal is to find relevant genes (as few as possible) while explaining the maximum possible variance. The proposed algorithm is used in two sparse CCA applications, one dealing with cross-language document retrieval and the other with vocabulary selection in music annotation. The cross-language document retrieval application involves a collection of documents with each document in different languages, say English and French. The goal is, given a query string in one language, retrieve the most relevant document(s) in the target language. We experimentally show that the proposed sparse CCA algorithm performs similar to the non-sparse version, however using only 10% of non-zero loadings in the canonical components. In the vocabulary selection application, we show that sparse CCA improves the performance of a statistical musical query system by selecting only those words (i.e., pruning the vocabulary) that are correlated to the underlying audio features.

The paper is organized as follows. We establish the mathematical notation in Section 2. In Section 3, the related literature, which mostly deals with the sparse PCA problem is discussed. In Section 4, we present the convex semidefinite program (SDP) (Vandenberghe

and Boyd, 1996) approximation and our proposed d.c. program formulation for the sparse GEV problem in Eq. (7). A brief introduction to MM algorithms and the solution to our d.c. program is presented in Section 5. We present the global convergence analysis of the proposed algorithm in Section 6. Finally, in Sections 7 and 8, we apply the proposed algorithm to sparse PCA and sparse CCA problems and present experimental results to demonstrate the performance of our method, while in Section 9, we discuss the applicability of our proposed algorithm to the sparse FDA problem.

2. Notation

$\mathbb{S}^n$ (*resp.* $\mathbb{S}^n_+$, $\mathbb{S}^n_{++}$) denotes the set of symmetric (*resp.* positive semidefinite, positive definite) $n \times n$ matrices defined over $\mathbb{R}$. For $\boldsymbol{X} \in \mathbb{S}^n$, $\boldsymbol{X} \succ 0$ (*resp.* $\boldsymbol{X} \succeq 0$) means that $\boldsymbol{X}$ is positive definite (*resp.* semidefinite). We denote a vector of ones and zeros by $\mathbf{1}$ and $\mathbf{0}$ respectively. Depending on the context, $\mathbf{0}$ will also be treated as a zero matrix. $|\boldsymbol{X}|$ is the matrix whose elements are the absolute values of the elements of $\boldsymbol{X}$. $[\boldsymbol{X}]_{ij}$ denotes the $(i,j)^{th}$ element of $\boldsymbol{X}$. For $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$, $\boldsymbol{x} \succeq \mathbf{0}$ denotes an element-wise inequality. $\|\boldsymbol{x}\|_0$ denotes the number of non-zero elements of vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p := (\sum_{i=1}^n |x_i|^p)^{1/p}$, $1 \leq p < \infty$ and $\|\boldsymbol{x}\|_\infty := \max_{1 \leq i \leq n} |x_i|$. $\boldsymbol{I}_n$ denotes an $n \times n$ identity matrix. $\boldsymbol{D}(\boldsymbol{x})$ represents a diagonal matrix formed with $\boldsymbol{x}$ as its principal diagonal.

3. Prior Work

The problem of sparsity in eigenvalue problems has mostly been addressed in the context of PCA, i.e., $\boldsymbol{A} \in \mathbb{S}^n_+$ and $\boldsymbol{B} = \boldsymbol{I}_n$. While PCA is numerically easy, sparse PCA is a hard combinatorial problem. The earliest attempts at "sparsifying" PCA consisted of simple axis rotations and component thresholding (Cadima and Jolliffe, 1995) for subset selection, often based on the identification of principal variables (McCabe, 1984). The first true computational technique, called SCoTLASS (Jolliffe et al., 2003), provided an optimization framework using LASSO (Tibshirani, 1996) by enforcing a sparsity constraint on the PCA solution by bounding its ℓ_1-norm, leading to a non-convex procedure. Zou et al. (2006) proposed a ℓ_1-penalized regression algorithm for PCA (called SPCA) using an *elastic net* (Zou and Hastie, 2005) and solved it very efficiently using least angle regression (Efron et al., 2004). Currently, this method seems to be the only viable option for handling very high-dimensional data sets (on the order of $n = 10,000$). Subsequently, d'Aspremont et al. (2007) proposed a convex relaxation to the non-convex cardinality constraint for PCA (called DSPCA) leading to a semidefinite program (SDP) (Vandenberghe and Boyd, 1996). Though this method shows comparable performance to SPCA on a small-scale benchmark data set, it is not scalable for high-dimensional data sets, even possibly with Nesterov's first-order method (Nesterov, 2005). Moghaddam et al. (2007a) proposed a combinatorial optimization algorithm (called GSPCA) using greedy search and branch-and-bound methods to solve the sparse PCA problem.

Moghaddam et al. (2007b) addressed the problem of sparse FDA using similar tools as in Moghaddam et al. (2007a). We show in Section 4 that the sparse FDA program can be approximated as a convex SDP, which has poor scalability as DSPCA. However, a quadratic program (QP) relaxation can be obtained for sparse FDA by using the QP formulation for

FDA (in Eq. (6)) as proposed by Mika et al. (2001) along with an ℓ_1-norm relaxation of the cardinality constraint. Therefore, it is appropriate to use the QP formulation for solving sparse FDA rather than using the SDP formulation that is presented in Section 4. See Section 9 for more details.

On sparse CCA, we are not aware of any related work. Recently, we built on our earlier work of sparse PCA (Sriperumbudur et al., 2007) and extended it to sparse CCA while applying it to a music annotation problem (Torres et al., 2007a,b).

4. Sparse Generalized Eigenvalue Formulation

Let us consider the GEV problem in Eq. (2) with $\boldsymbol{A} \succeq \boldsymbol{0}$ and $\boldsymbol{B} \succ \boldsymbol{0}$. It has to be noted that Eq. (2) is *not* a canonical convex program as it deals with *maximizing* (instead of minimizing) the convex function over a set that is not convex (because of the equality constraint). However, replacing the constraint set $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}$ by $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$ does not change the optimality conditions because by Theorem 32.1 of Rockafellar (1970), the optimum lies on the boundary of the constraint set. Though this makes the constraint set to be convex, the problem is still not a canonical convex program for the reason mentioned before. As remarked by Rockafellar (1970, p. 342), "The theory of the maximum of a convex function relative to a convex set has an entirely different character from the theory of the minimum." In fact, finding the global maximum of a convex function over a convex set is computationally hard. On the other hand, when $\boldsymbol{A}$ and $\boldsymbol{B}$ are indefinite (which means neither positive semidefinite nor negative semidefinite), then the objective and constraint functions in Eq. (2) are not convex. This means, irrespective of the positive definite behavior of $\boldsymbol{A}$ and $\boldsymbol{B}$, the program in Eq. (2) is not a canonical convex formulation and therefore can be computationally hard to solve. However, it is well known that polynomial time algorithms (e.g., QR algorithm) exist to solve Eq. (2), which makes the GEV problem *special*. The specialty of the GEV problem will be better understood when we consider the ℓ_1-norm relaxation of the sparse GEV problem. This is discussed in detail in the following paragraphs and is also shown diagrammatically in Figure 1.

Now, let us consider the variational formulation for the sparse generalized eigenvalue problem given in Eq. (7). From now on, we assume that $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \in \mathbb{S}^n_{++}$. Eq. (7) is non-convex, NP-hard and therefore intractable. The intractability of Eq. (7) is due to: (a) maximization of the non-concave objective and (b) intractability of the cardinality constraint. To get a handle on the cardinality constraint, usually the ℓ_1-norm approximation is used. To make the constraint set convex, the equality constraint is replaced by the inequality constraint.[2] Though these relaxations make the constraint set convex, they do not simplify the problem any further as the additional ℓ_1-norm constraint destroys the

2. Note that the GEV problem in Eq. (2) and $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$ are not equivalent for all $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \succ 0$, i.e., the maximizers and the optimal values are not the same. This is because if $\boldsymbol{A} \preceq 0$, then $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ is concave in $\boldsymbol{x}$ and so the program $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$ is the maximization of a concave function over a convex set and clearly the optimum value is zero which occurs at $\boldsymbol{x}_* = \boldsymbol{0}$. However, suppose $\boldsymbol{A}$ and $\boldsymbol{B}$ are such that $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) > 0$. Then $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\} = \lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$, i.e., the equality and inequality constrained programs are the same. This can be seen by solving the Lagrangian of $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$, which gives $\boldsymbol{A}\boldsymbol{x} = \mu\boldsymbol{B}\boldsymbol{x}$ where $\mu \geq 0$ is the Lagrangian multiplier. It is clear that $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \mu(\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} - 1) + \mu = \mu$, where we have invoked the complementary slackness. So, $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ can be maximized by choosing $\boldsymbol{x} \in \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}$ rather than $\boldsymbol{x} \in \{\boldsymbol{x} :$

special nature of Eq. (2) that we discussed in the last paragraph. In the following, we elaborate this behavior through Figure 1.

Figure 1 shows the geometry of the constraint set for the sparse GEV problem in Eq. (7) with $n = 2$ where the cardinality constraint is replaced with $\|\boldsymbol{x}\|_1 \leq k$. Figure 1(a) shows the level sets of $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ (black curves) and the quadratic constraint set $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1$ (red curve). Figure 1(b-d) show the constraints $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1$ and $\|\boldsymbol{x}\|_1 \leq k$ for different ranges of k. Since $\|\boldsymbol{x}\|_1 \leq k$ is relaxing $\|\boldsymbol{x}\|_0 \leq k$, k can take values between 1 and 2 (the maximum cardinality for $n = 2$). It has to be noted that $\sqrt{2} \leq k \leq 2$ is not an interesting range as the sparse GEV problem reduces to the GEV problem in Eq. (2). The interesting range for k is $1 < k < \sqrt{2}$, whose corresponding ℓ_1-norm constraint (green curve) is shown in Figure 1(c) along with the effective constraint (red curve) of the sparse GEV problem in Figure 1(d). Since the maximizer of a convex function over a convex set lies on the boundary of the constraint set, it can be seen from Figure 1(d) that the solution to the approximate problem will most likely be at one of the kinks in the red curve. However, characterizing these kinks in high dimensions is not straightforward, which therefore makes the problem hard unlike in the GEV problem where the constraint set is very easily characterized. So replacing the cardinality constraint in Eq. (7) by $\|\boldsymbol{x}\|_1 \leq k$ does not simplify the problem any further. However, using this relaxation, in the following subsection, we provide a convex relaxation to Eq. (7) which is a little different from the one proposed by d'Aspremont et al. (2007).

4.1 Semidefinite relaxation

We start with the following program obtained by relaxing the cardinality constraint with an ℓ_1-norm constraint on $\boldsymbol{x}$,

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ \|\boldsymbol{x}\|_1 \leq k. \end{aligned} \tag{8}$$

Note that the quadratic equality constraint in Eq. (7) is replaced with an inequality constraint in Eq. (8) so that the constraints forms a convex set. As mentioned before, the above program is computationally hard to solve because of the maximization of a convex function over a convex set. Since, the Lagrangian dual of Eq. (8) is always convex irrespective of the primal, we solve for the Lagrangian of Eq. (8) resulting in the following dual program:

$$\begin{aligned} \min_{\mu,\beta,\boldsymbol{r}} \quad & \frac{1}{4}\boldsymbol{r}^T(\mu\boldsymbol{B} - \boldsymbol{A})^\dagger\boldsymbol{r} + \mu + \beta k \\ \text{s.t.} \quad & -\beta\boldsymbol{1} \preceq \boldsymbol{r} \preceq \beta\boldsymbol{1},\ \mu \geq 0,\ \beta \geq 0 \\ & \mu\boldsymbol{B} - \boldsymbol{A} \succeq 0 \\ & \boldsymbol{r} \in \mathcal{R}(\mu\boldsymbol{B} - \boldsymbol{A}), \end{aligned} \tag{9}$$

where $\mathcal{R}(\boldsymbol{T})$ denotes the range space of $\boldsymbol{T}$ and $\boldsymbol{T}^\dagger$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{T}$. For a derivation of the dual in Eq. (9), we refer the reader to Appendix A. By invoking

$\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} < 1\}$. This means at the optimum $\mu_* = \boldsymbol{x}_*^T\boldsymbol{A}\boldsymbol{x}_* > 0$. Therefore, if $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) > 0$, then $\mu_* = \max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\} = \lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$.

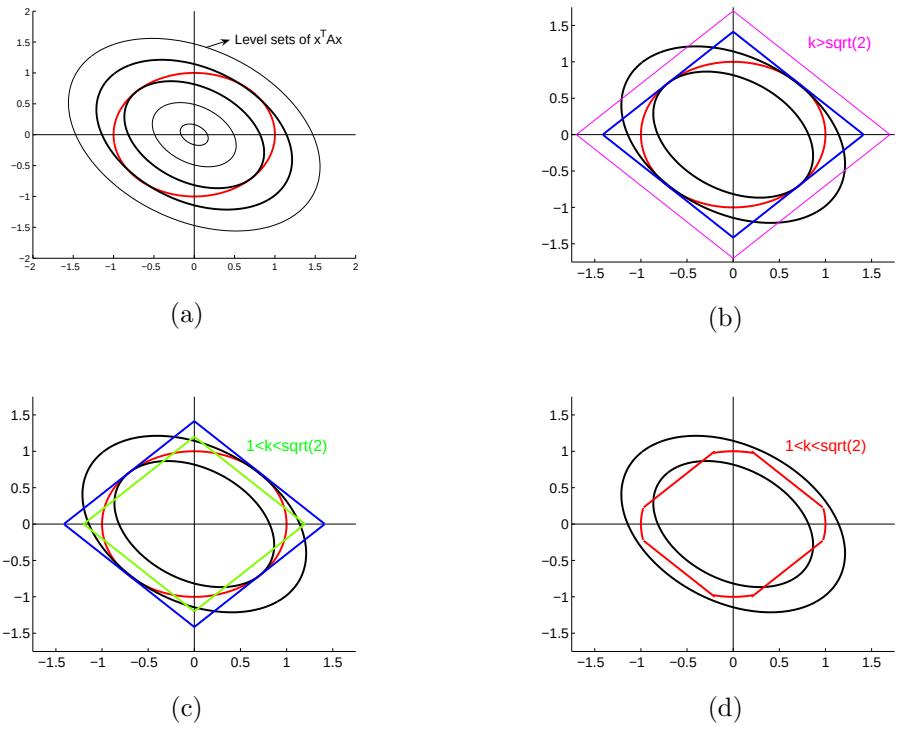

Figure 1: (a) Level sets of the objective function, $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ (black curves), and the quadratic equality constraint, $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1$ (red curve). (b) The blue curve represents the constraint $\|\boldsymbol{x}\|_1 = \sqrt{2}$. It is clear that for $k \geq \sqrt{2}$, the sparse GEV problem is same as that of the GEV problem. (c) The interesting case occurs when $1 < k < \sqrt{2}$ where the constraint $\|\boldsymbol{x}\|_1 \leq k$ is shown by the green curve. (d) The effective constraint set (intersection of $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}$ and $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 \leq k\}$) for $1 < k < \sqrt{2}$, is shown in red. Characterizing the optimum, most likely at one of the kinks in the red curve, is not straightforward in high dimensions, which makes the problem hard.

the Schur's complement lemma, Eq. (9) reduces to the following SDP,

$$\begin{aligned} \min_{\mu,\beta,\boldsymbol{r},t} \quad & t + \mu + \beta k \\ \text{s.t.} \quad & -\beta\mathbf{1} \preceq \boldsymbol{r} \preceq \beta\mathbf{1},\ \mu \geq 0,\ \beta \geq 0 \\ & \begin{pmatrix} \mu\boldsymbol{B} - \boldsymbol{A} & -\frac{1}{2}\boldsymbol{r} \\ -\frac{1}{2}\boldsymbol{r}^T & t \end{pmatrix} \succeq 0. \end{aligned} \tag{10}$$

The SDP in Eq. (10) can be related back to the primal in Eq. (8) by dualizing the dual, called the *bi-dual* (Lemaréchal and Oustry, 1999, Section 4.3), given as

$$\begin{aligned} \max_{\boldsymbol{X},\boldsymbol{x}} \quad & \mathrm{tr}(\boldsymbol{X}\boldsymbol{A}) \\ \text{s.t.} \quad & \mathrm{tr}(\boldsymbol{X}\boldsymbol{B}) \leq 1,\ \|\boldsymbol{x}\|_1 \leq k \\ & \begin{pmatrix} \boldsymbol{X} & \boldsymbol{x} \\ \boldsymbol{x}^T & 1 \end{pmatrix} \succeq 0, \end{aligned} \tag{11}$$

which is also an SDP. The bi-dual in Eq. (11) (for a derivation, see appendix A) can be directly obtained from Eq. (8) by a method called *lifting* (Lemaréchal and Oustry, 1999, Section 4.4). With $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T$ in Eq. (8)[3], we get the *lifted* version given by

$$\begin{aligned} \max_{\boldsymbol{X},\boldsymbol{x}} \quad & \operatorname{tr}(\boldsymbol{X}\boldsymbol{A}) \\ \text{s.t.} \quad & \operatorname{tr}(\boldsymbol{X}\boldsymbol{B}) \leq 1,\ \|\boldsymbol{x}\|_1 \leq k \\ & \boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T. \end{aligned} \tag{12}$$

The above program is non-convex because $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T \Leftrightarrow \boldsymbol{X} \succeq 0, \operatorname{rank}(\boldsymbol{X}) = 1$, where $\operatorname{rank}(\boldsymbol{X}) = 1$ is not a convex constraint. Relaxing $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T$ to $\boldsymbol{X} - \boldsymbol{x}\boldsymbol{x}^T \succeq 0$ results in Eq. (11) which is an SDP. Therefore, the program in Eq. (11) is the convex relaxation of the lifted version of Eq. (8). The ℓ_1-norm constraint in Eq. (11) can be replaced with $\|\boldsymbol{x}\|_1^2 \leq k^2 \Rightarrow \boldsymbol{1}^T|\boldsymbol{X}|\boldsymbol{1} \leq k^2$ so that the problem reduces to solving only for $\boldsymbol{X}$. d'Aspremont et al. (2007) applied the lifting technique to Eq. (7) where the cardinality constraint is replaced by $\|\boldsymbol{x}\|_1^2 \leq k\boldsymbol{x}^T\boldsymbol{x}$ and $\boldsymbol{B}$ by $\boldsymbol{I}_n$, resulting in the sparse PCA algorithm called DSPCA.

Though Eq. (11) is a convex approximation to the sparse GEV problem, it is computationally very intensive as the general purpose interior-point methods scale as $O(n^6 \log \epsilon^{-1})$, where ϵ is the required accuracy on the optimal value. For large-scale problems, first-order methods (Nesterov, 2005; d'Aspremont et al., 2007) can be used which scale as $O(\epsilon^{-1}n^4\sqrt{\log n})$. So, the *only*[4] convex approach possible is through SDP relaxation which is prohibitively expensive for large n. Alternatives, for large n, are either locally convergent methods (converging to some local optima of Eq. (7)) that are computationally efficient or expensive global optimization programs. In the following subsection, we consider a non-convex approximation to the cardinality constraint and propose a d.c. (difference of convex functions) formulation to the sparse GEV problem, which is then solved efficiently as a sequence of QCQPs using the majorization-minimization algorithm.

4.2 Non-convex approximation and d.c. formulation

The proposed method is motivated by the following observations.

- Because of the non-concave maximization of Eq. (7), the ℓ_1-norm relaxation of the cardinality constraint does not simplify Eq. (7). So, a better approximation to the cardinality constraint can be explored to improve sparsity.

- Since the SDP approximation to Eq. (7) scales badly in n, different approximations that yield better scalability should be explored.

3. $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \operatorname{tr}(\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}) = \operatorname{tr}(\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^T) = \operatorname{tr}(\boldsymbol{A}\boldsymbol{X})$.
4. As discussed before, the hardness of the problem in Eq. (8) is due to the maximization of a non-concave objective over a convex set. Since a linear function is both concave and convex, Eq. (8) can be turned into a convex program by making the objective linear through lifting and neglecting the rank constraint. Though this procedure gives a unique global optimum, it has to be noted that the solution obtained is only an approximation to the true solution. Ideally, the obtained solution should be projected back onto the true (unrelaxed) constraint set to achieve a feasible solution, which is usually done by random projection.

To this end, we consider the regularized[5] (penalized) version of Eq. (7) given by

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\,\|\boldsymbol{x}\|_0 \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} \leq 1, \end{aligned} \tag{13}$$

where $\tilde{\rho} > 0$ is the regularization (penalization) parameter. Different approximations to $\|\boldsymbol{x}\|_0$ have been proposed other than the ℓ_1-norm relaxation. Weston et al. (2003) replaced[6] $\|\boldsymbol{x}\|_0$ by $\sum_{i=1}^n \log(\varepsilon + |x_i|)$, where $\varepsilon > 0$, while Bradley and Mangasarian (1998) used $\sum_{i=1}^n (1 - e^{-\alpha|x_i|})$ for $\alpha > 0$.[7] These approximations were used in the context of feature selection using support vector machines. Candes et al. (2007) used an approximation similar to that of Weston et al. (2003) in the context of sparse signal recovery, which is also used by Fazel et al. (2003) in matrix rank minimization. In this paper, we use this approximation because of its interesting connection to sparse priors that are studied in Bayesian inference.[8] This approximation can be interpreted as defining a limiting Student's t-distribution prior over $\boldsymbol{x}$, an improper prior given by $p(\boldsymbol{x}) \propto \prod_{i=1}^n \frac{1}{|x_i|+\varepsilon}$ and computing its negative log-likelihood. Tipping (2001) showed that this choice of prior leads to a sparse representation and demonstrated its validity for sparse kernel expansions in the Bayesian framework.

Before proceeding further, we show that the approximation (to $\|\boldsymbol{x}\|_0$) considered in this paper, i.e., $\sum_{i=1}^n \frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})}$, $\forall\, \varepsilon > 0$, is tighter than the ℓ_1-norm approximation. To this end, let us define $a_\varepsilon := \frac{\log(1+a\varepsilon^{-1})}{\log(1+\varepsilon^{-1})}$, where $a \geq 0$ so that $\|\boldsymbol{x}\|_\varepsilon := \sum_{i=1}^n |x_i|_\varepsilon$. It is easy to check that $\|\boldsymbol{x}\|_0 = \lim_{\varepsilon\to 0} \|\boldsymbol{x}\|_\varepsilon$ and $\|\boldsymbol{x}\|_1 = \lim_{\varepsilon\to\infty} \|\boldsymbol{x}\|_\varepsilon$. In addition, we have $a > a_{\varepsilon_1} > a_{\varepsilon_2} > \ldots > 1$ for $a > 1$ and $1 > \ldots > a_{\varepsilon_2} > a_{\varepsilon_1} > a$ for $0 < a < 1$, where $\varepsilon_1 > \varepsilon_2 > \ldots$. Therefore, it is easy to see that $\|\boldsymbol{x}\|_\varepsilon$ for any $0 < \varepsilon < \infty$ is a better approximation to $\|\boldsymbol{x}\|_0$ than $\|\boldsymbol{x}\|_1$ to $\|\boldsymbol{x}\|_0$.

Starting from Eq. (13) and replacing $\|\boldsymbol{x}\|_0$ by the equality in footnote 6, the *approximate* sparse GEV problem can be written as

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} - \rho_\varepsilon \sum_{i=1}^n \log(\varepsilon + |x_i|) \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} \leq 1, \end{aligned} \tag{14}$$

where $\rho_\varepsilon = \tilde{\rho}/\log(1+\varepsilon^{-1})$. In Appendix B, we provide a derivation of the above claim.

Let us define $Q(\boldsymbol{x}) := \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\|\boldsymbol{x}\|_0$, $Q_\varepsilon(\boldsymbol{x}) := \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \rho_\varepsilon \sum_{i=1}^n \log(1+|x_i|\varepsilon^{-1})$ and $\Omega := \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$. Suppose that $\widehat{\boldsymbol{x}}$ denotes a maximizer of $Q(\boldsymbol{x})$ over Ω and $\boldsymbol{x}_\varepsilon$ denotes a maximizer of $Q_\varepsilon(\boldsymbol{x})$ over Ω. Now, one would like to know how good is the approximate solution, $\boldsymbol{x}_\varepsilon$ compared to $\widehat{\boldsymbol{x}}$. In general, it is not straightforward to either bound $\|\boldsymbol{x}_\varepsilon - \widehat{\boldsymbol{x}}\|$

5. Note the quadratic inequality constraint in Eq. (13) instead of the equality constraint. By footnote 2, we assume $\lambda_{max}(\boldsymbol{A},\boldsymbol{B}) > 0$ so that for $\tilde{\rho} = 0$, the equality and inequality constrained programs are the same. When $\boldsymbol{A} \succeq 0$ and $\boldsymbol{B} = \boldsymbol{I}_n$, El Ghaoui (2006) has derived sufficient conditions on $\tilde{\rho}$ so that the programs $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\|\boldsymbol{x}\|_0 : \boldsymbol{x}^T\boldsymbol{x} = 1\}$ and $\max\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\|\boldsymbol{x}\|_0 : \boldsymbol{x}^T\boldsymbol{x} \leq 1\}$ are equivalent. One of the open problems in this work is to derive such a condition for any $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \succ 0$.

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

7. $\|\boldsymbol{x}\|_0 = \lim_{\alpha\to\infty} \sum_{i=1}^n (1 - e^{-\alpha|x_i|})$

8. Another approximation one could use is $\sum_{i=1}^n \tan^{-1}(|x_i|/\varepsilon)$ as $\|\boldsymbol{x}\|_0 = \lim_{\varepsilon\to 0} \sum_{i=1}^n \frac{\tan^{-1}(|x_i|/\varepsilon)}{\tan^{-1}(1/\varepsilon)}$.

in terms of ε or to show that $\|\boldsymbol{x}_\varepsilon - \widehat{\boldsymbol{x}}\| \to 0$ as $\varepsilon \to 0$ because $Q(\boldsymbol{x})$ may be quite flat near its maximum over Ω. However, the following proposition shows that the difference between $Q(\widehat{\boldsymbol{x}})$ and $Q_\varepsilon(\boldsymbol{x}_\varepsilon)$ goes to 0 as $\varepsilon \to 0$. In addition, it provides a bound on the cardinality of $\widehat{\boldsymbol{x}}$ in terms of the limiting value $Q(\boldsymbol{x}_\varepsilon)$.

Proposition 1 *Let $\widehat{\boldsymbol{x}}$ and $\boldsymbol{x}_\varepsilon$ be the maximizers of Eq. (13) and Eq. (14) respectively for fixed $\tilde{\rho}$ and ε. Then, the following claims hold:*

(i) *As $\varepsilon \to 0$, $|Q_\varepsilon(\boldsymbol{x}) - Q(\boldsymbol{x})| \to 0$ uniformly on Ω.*

(ii) *$|Q_\varepsilon(\boldsymbol{x}_\varepsilon) - Q(\widehat{\boldsymbol{x}})| \to 0$ as $\varepsilon \to 0$.*

(iii) *Suppose $\lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)$ exists. Then $Q(\widehat{\boldsymbol{x}}) = \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)$. In addition, if $\lambda_{max}(\boldsymbol{A},\boldsymbol{B}) > 0$ and $\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) < 0$, then*

$$\frac{\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) - \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)}{\tilde{\rho}} \leq \|\widehat{\boldsymbol{x}}\|_0 \leq \frac{\lambda_{max}(\boldsymbol{A},\boldsymbol{B}) - \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)}{\tilde{\rho}}. \tag{15}$$

Proof
(i) Consider

$$\begin{aligned}
\max_{\boldsymbol{x}\in\Omega} |Q_\varepsilon(\boldsymbol{x}) - Q(\boldsymbol{x})| &= \tilde{\rho} \max_{\boldsymbol{x}\in\Omega} \left| \sum_{i=1}^n \frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - \|\boldsymbol{x}\|_0 \right| \\
&= \tilde{\rho} \max_{\boldsymbol{x}\in\Omega} \left| \sum_{i=1}^n \left[\frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - \mathbb{1}_{\{|x_i|\neq 0\}} \right] \right| \\
&\leq \tilde{\rho} \max \left\{ \left| \sum_{i=1}^n \left[\frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - \mathbb{1}_{\{|x_i|\neq 0\}} \right] \right| \,:\, \boldsymbol{x}^T\boldsymbol{x} \leq \lambda_{min}^{-1}(\boldsymbol{B}) \right\} \\
&\leq \tilde{\rho} \max \left\{ \sum_{i=1}^n \left| \frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - \mathbb{1}_{\{|x_i|\neq 0\}} \right| \,:\, |x_i| \leq \lambda_{min}^{-1/2}(\boldsymbol{B}),\, \forall\, i \right\} \\
&= \tilde{\rho} \sum_{i=1}^n \max \left\{ \left| \frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - \mathbb{1}_{\{|x_i|\neq 0\}} \right| \,:\, |x_i| \leq \lambda_{min}^{-1/2}(\boldsymbol{B}) \right\} \\
&= n\tilde{\rho} \left| \frac{\log(1+\lambda_{min}^{-1/2}(\boldsymbol{B})\varepsilon^{-1})}{\log(1+\varepsilon^{-1})} - 1 \right|.
\end{aligned} \tag{16}$$

(i) therefore follows by taking limits on both sides of Eq. (16).

(ii) $|Q_\varepsilon(\boldsymbol{x}_\varepsilon) - Q(\widehat{\boldsymbol{x}})| = |\max_{\boldsymbol{x}\in\Omega} Q_\varepsilon(\boldsymbol{x}) - \max_{\boldsymbol{x}\in\Omega} Q(\boldsymbol{x})| \leq \max_{\boldsymbol{x}\in\Omega} |Q_\varepsilon(\boldsymbol{x}) - Q(\boldsymbol{x})|$. The result therefore follows from *(i)*.

(iii) Consider

$$\begin{aligned}
Q(\widehat{\boldsymbol{x}}) - Q(\boldsymbol{x}_\varepsilon) &= Q(\widehat{\boldsymbol{x}}) - Q_\varepsilon(\widehat{\boldsymbol{x}}) + Q_\varepsilon(\widehat{\boldsymbol{x}}) - Q_\varepsilon(\boldsymbol{x}_\varepsilon) + Q_\varepsilon(\boldsymbol{x}_\varepsilon) - Q(\boldsymbol{x}_\varepsilon) \\
&\leq Q(\widehat{\boldsymbol{x}}) - Q_\varepsilon(\widehat{\boldsymbol{x}}) + Q_\varepsilon(\boldsymbol{x}_\varepsilon) - Q(\boldsymbol{x}_\varepsilon) \\
&\leq |Q(\widehat{\boldsymbol{x}}) - Q_\varepsilon(\widehat{\boldsymbol{x}})| + |Q_\varepsilon(\boldsymbol{x}_\varepsilon) - Q(\boldsymbol{x}_\varepsilon)| \\
&\leq 2 \max_{\boldsymbol{x}\in\Omega} |Q_\varepsilon(\boldsymbol{x}) - Q(\boldsymbol{x})|.
\end{aligned} \tag{17}$$

Taking limits on both sides of Eq. (17), we have $Q(\widehat{\boldsymbol{x}}) \leq \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)$, which follows from *(i)* and our assumption that $\lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)$ exists. Since $\widehat{\boldsymbol{x}}$ is a maximizer of $Q(\boldsymbol{x})$ over Ω, we also have $Q(\widehat{\boldsymbol{x}}) \geq Q(\boldsymbol{x})$, $\forall\, \boldsymbol{x} \in \Omega$ and therefore $Q(\widehat{\boldsymbol{x}}) \geq Q(\boldsymbol{x}_\varepsilon)$. Taking limits on both sides leads to the result in *(iii)*. The bound on $\|\widehat{\boldsymbol{x}}\|_0$ in Eq. (15) follows from bounding $\widehat{\boldsymbol{x}}^T \boldsymbol{A}\widehat{\boldsymbol{x}}$ as

$$\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) = \min_{\boldsymbol{x}\in\Omega} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq \widehat{\boldsymbol{x}}^T\boldsymbol{A}\widehat{\boldsymbol{x}} \leq \max_{\boldsymbol{x}\in\Omega} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \lambda_{max}(\boldsymbol{A},\boldsymbol{B})$$

in $\|\widehat{\boldsymbol{x}}\|_0 = \frac{\widehat{\boldsymbol{x}}^T\boldsymbol{A}\widehat{\boldsymbol{x}} - \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)}{\tilde{\rho}}$ where $\widehat{\boldsymbol{x}} \in \Omega$ (since $Q(\widehat{\boldsymbol{x}}) = \widehat{\boldsymbol{x}}^T\boldsymbol{A}\widehat{\boldsymbol{x}} - \tilde{\rho}\,\|\widehat{\boldsymbol{x}}\|_0 = \lim_{\varepsilon\to 0} Q(\boldsymbol{x}_\varepsilon)$). Note that we need the assumption of $\lambda_{max}(\boldsymbol{A},\boldsymbol{B}) > 0$ and $\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) < 0$. Otherwise, the programs $\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) = \min\{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}$ and $\min_{\boldsymbol{x}\in\Omega} \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ need not be equivalent. Similar is the case with $\lambda_{max}(\boldsymbol{A},\boldsymbol{B})$. See footnote 2 for details. ■

Remark 2 *(a) Suppose* $\lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}_\varepsilon)$ *exists. Then, by (ii), we have*

$$Q(\widehat{\boldsymbol{x}}) = \lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}_\varepsilon). \tag{18}$$

Eq. (71) shows that $Q(\widehat{\boldsymbol{x}}) = \max\{\lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}) : \boldsymbol{x} \in \Omega\}$*, whereas by Eq. (18) we have* $Q(\widehat{\boldsymbol{x}}) = \lim_{\varepsilon\to 0} \max\{Q_\varepsilon(\boldsymbol{x}) : \boldsymbol{x} \in \Omega\}$*. Therefore, assuming* $\lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}_\varepsilon)$ *to exist is equivalent to interchanging the limit process and maximization in Eq. (71). In this case, a bound similar to Eq. (15) can be given as*

$$\frac{\lambda_{min}(\boldsymbol{A},\boldsymbol{B}) - \lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}_\varepsilon)}{\tilde{\rho}} \leq \|\widehat{\boldsymbol{x}}\|_0 \leq \frac{\lambda_{max}(\boldsymbol{A},\boldsymbol{B}) - \lim_{\varepsilon\to 0} Q_\varepsilon(\boldsymbol{x}_\varepsilon)}{\tilde{\rho}}. \tag{19}$$

(b) Suppose $\boldsymbol{x}_*$ *is a limit point of the sequence* $\{\boldsymbol{x}_\varepsilon\}_{\varepsilon\to 0}$*. If* Q *were continuous, then by (iii),* $\boldsymbol{x}_*$ *is a maximizer of* $Q(\boldsymbol{x})$*.*

As mentioned in Remark 2, we cannot claim that any limit point of the sequence $\{\boldsymbol{x}_\varepsilon\}_{\varepsilon\to 0}$ is a maximizer of $Q(\boldsymbol{x})$. However, informally speaking, choosing a small value for ε gives a solution $\boldsymbol{x}_\varepsilon$ such that $|Q(\widehat{\boldsymbol{x}}) - Q(\boldsymbol{x}_\varepsilon)|$ is small. So, one can think of $\boldsymbol{x}_\varepsilon$ as a solution to the sparse GEV problem in Eq. (13). The task therefore reduces to solving the approximate sparse GEV problem in Eq. (14) with a small value of ε. In the discussion so far, we have fixed $\tilde{\rho}$ to some constant. If we knew a priori for what value of $\tilde{\rho}$ we would get the desired sparsity in Eq. (13), we could compute the corresponding ρ_ε to solve Eq. (14). Since we usually don't know the best $\tilde{\rho}$ a priori, the final choice of ρ_ε will mainly depend on achieving the desired sparsity in Eq. (14), for a fixed ε, making its dependence on ε less explicit. We therefore represent ρ_ε by ρ, a constant chosen to achieve the desired sparsity in Eq. (14). From now on, we consider two programs as equivalent if their optimizers are the same. So, the program in Eq. (14) is equivalent to

$$\begin{aligned} \max_{\boldsymbol{x},\boldsymbol{y}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \rho\sum_{i=1}^{n}\log(y_i+\varepsilon) \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ -\boldsymbol{y} \preceq \boldsymbol{x} \preceq \boldsymbol{y}. \end{aligned} \tag{20}$$

Since $\boldsymbol{A} \in \mathbb{S}^n$ is not assumed to be positive definite, let us choose $\tau \in \mathbb{R}$ such that $\boldsymbol{A} + \tau \boldsymbol{I}_n \succeq 0$. If $\boldsymbol{A} \succeq 0$, such a τ exists trivially (choose $\tau \geq 0$). If $\boldsymbol{A}$ is indefinite, choosing $\tau \geq -\lambda_{min}(\boldsymbol{A})$ ensures that $\boldsymbol{A} + \tau \boldsymbol{I}_n \succeq 0$.[9] So choosing $\tau \geq \max(0, -\lambda_{min}(\boldsymbol{A}))$ ensures that $\boldsymbol{A} + \tau \boldsymbol{I}_n \succeq 0$ for any $\boldsymbol{A} \in \mathbb{S}^n$. Therefore, Eq. (20) can be written as

$$\begin{aligned} -\min_{\boldsymbol{x},\boldsymbol{y}} \quad & \tau \|\boldsymbol{x}\|_2^2 - \left[\boldsymbol{x}^T (\boldsymbol{A} + \tau \boldsymbol{I}_n) \boldsymbol{x} - \rho \sum_{i=1}^{n} \log(y_i + \varepsilon) \right] \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} \leq 1, \; -\boldsymbol{y} \preceq \boldsymbol{x} \preceq \boldsymbol{y}. \end{aligned} \tag{21}$$

Since $\tau \geq 0$, the term $\tau \|\boldsymbol{x}\|_2^2$ is convex in $\boldsymbol{x}$. By construction, $\boldsymbol{x}^T (\boldsymbol{A} + \tau \boldsymbol{I}_n) \boldsymbol{x} - \rho \sum_{i=1}^{n} \log(y_i + \varepsilon)$ 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 and is usually referred to as a *d.c. program*.[10] Global optimization methods like branch and bound, cutting planes can be used to solve d.c. programs (Horst and Thoai, 1999), but are not scalable to large-scale problems. Since Eq. (21) is a constrained nonlinear optimization problem, it can be solved by methods like sequential quadratic programming, augmented Lagrangian or reduced-gradient (Bonnans et al., 2006). In Sriperumbudur et al. (2007), we solved Eq. (21) with $\tau = 0$ (assuming $\boldsymbol{A} \succeq 0$) using the d.c. programming algorithm called DCA proposed by Tao and An (1998). We recently realized that solving via DCA is equivalent to using majorization-minimization (MM) algorithms. Since MM methods are much easier to understand, in this paper, we derive our sparse GEV algorithm by solving Eq. (21) using the MM method. It has to be noted that SCoTLASS in Eq. (8) can also be formulated as a d.c. program similar to Eq. (21) by replacing $\sum_{i=1}^{n} \log(y_i + \varepsilon)$ with $\boldsymbol{y}^T \boldsymbol{1}$. In the following section, we present our sparse GEV algorithm along with its convergence analysis.

5. Solution by Majorization-Minimization

Before proceeding to solve the d.c. program in Eq. (21), we briefly discuss the idea behind MM algorithms.

5.1 MM algorithms

MM algorithms can be thought of as a generalization of the well-known EM algorithm (Dempster et al., 1977). The general principle behind MM algorithms was first enunciated by the numerical analysts Ortega and Rheinboldt (1970) in the context of line search methods. The MM principle appears in many places in statistical computation, including multidimensional scaling (deLeeuw, 1977), robust regression (Huber, 1981), correspondence analysis (Heiser, 1987), variable selection (Hunter and Li, 2005), sparse signal recovery (Candes et al., 2007), etc. We refer the interested reader to a tutorial on MM algorithms (Hunter and Lange, 2004) and the references therein.

9. One can pose this as an optimization problem, "Find the minimum value of τ for which $\boldsymbol{A} + \tau \boldsymbol{I}_n \succeq 0$?", i.e., $\tau^* = \min\{\tau : \boldsymbol{A} + \tau \boldsymbol{I}_n \succeq 0\}$. The dual program is given by $\tau^* = -\min\{\mathrm{tr}(\Lambda \boldsymbol{A}) : \mathrm{tr}(\Lambda) = 1, \Lambda \succeq 0\}$. It is easy to verify that $\tau^* = -\lambda_{min}(\boldsymbol{A})$.

10. Let Ω be a convex set of $\mathbb{R}^n$. 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 $\min_{\boldsymbol{x}} \{f_0(\boldsymbol{x}) : \boldsymbol{x} \in \Omega, f_i(\boldsymbol{x}) \leq 0, i = 1, \ldots, m\}$, where $f_i = g_i - h_i$, $i = 0, \ldots, m$, are d.c. functions are called *d.c. programs*.

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

$$f(x) \leq g(x, y), \forall\, x, y \in \Omega \qquad \text{and} \qquad f(x) = g(x, x), \forall\, x \in \Omega. \tag{22}$$

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

$$x^{(l+1)} \in \arg\min_{x \in \Omega} g(x, x^{(l)}), \tag{23}$$

unless we already have

$$x^{(l)} \in \arg\min_{x \in \Omega} g(x, x^{(l)}),$$

in which case the algorithm stops. The majorization function, g is usually constructed by using Jensen's inequality for convex functions, the first-order Taylor approximation or the quadratic upper bound principle (Böhning and Lindsay, 1988). However, any other method can also be used to construct g as long as it satisfies Eq. (22). It is easy to show that the above iterative scheme decreases the value of f monotonically in each iteration, i.e.,

$$f(x^{(l+1)}) \leq g(x^{(l+1)}, x^{(l)}) \leq g(x^{(l)}, x^{(l)}) = f(x^{(l)}), \tag{24}$$

where the first inequality and the last equality follows from Eq. (22) while the sandwiched inequality follows from Eq. (23).

Note that MM algorithms can be applied equally well to the maximization of f by simply reversing the inequality sign in Eq. (22) and changing the "min" to "max" in Eq. (23). In this case, the word MM refers to minorization-maximization, where the function g is called the *minorization function.* To put things in perspective, the EM algorithm can be obtained by constructing the minorization function g using Jensen's inequality for concave functions. The construction of such a g is referred to as the E-step, while Eq. (23) with the "min" replaced by "max" is referred to as the M-step. The algorithm in Eqs. (22-23) is used in machine learning, e.g., for non-negative matrix factorization (Lee and Seung, 2001), under the name *auxiliary function method*. Lange et al. (2000) studied this algorithm under the name *optimization transfer* while Meng (2000) referred to it as the SM algorithm, where "S" stands for the surrogate step (same as the majorization/minorization step) and "M" stands for the minimization/maximization step depending on the problem at hand. g is called the surrogate function. In the following, we consider an example that is relevant to our problem where we construct a majorization function, g, which will later be used in deriving the sparse GEV algorithm.

Example 1 (Linear Majorization) *Let us consider the optimization problem,* $\min_{\boldsymbol{x} \in \Omega} f(\boldsymbol{x})$ *where* $f = u - v$*, with* u *and* v *both convex, with* v *continuously differentiable. Since* v *is convex, we have* $v(\boldsymbol{x}) \geq v(\boldsymbol{y}) + (\boldsymbol{x} - \boldsymbol{y})^T \nabla v(\boldsymbol{y}), \forall\, \boldsymbol{x}, \boldsymbol{y} \in \Omega$*. Therefore,*

$$f(\boldsymbol{x}) \leq u(\boldsymbol{x}) - v(\boldsymbol{y}) - (\boldsymbol{x} - \boldsymbol{y})^T \nabla v(\boldsymbol{y}) =: g(\boldsymbol{x}, \boldsymbol{y}). \tag{25}$$

It is easy to verify that g *is a majorization function of* f*. Therefore, we have*

$$\boldsymbol{x}^{(l+1)} \in \arg\min_{\boldsymbol{x} \in \Omega} g(\boldsymbol{x}, \boldsymbol{x}^{(l)}) = \arg\min_{\boldsymbol{x} \in \Omega} u(\boldsymbol{x}) - \boldsymbol{x}^T \nabla v(\boldsymbol{x}^{(l)}). \tag{26}$$

If Ω is a convex set, then the above procedure solves a sequence of convex programs. Note that this is the same idea that is being used in the concave-convex procedure (CCCP) (Yuille and Rangarajan, 2003).

Suppose u and v are strictly convex, then a strict descent can be achieved in Eq. (24) unless $\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)}$, i.e., if $\boldsymbol{x}^{(l+1)} \neq \boldsymbol{x}^{(l)}$, then

$$f(\boldsymbol{x}^{(l+1)}) < g(\boldsymbol{x}^{(l+1)}, \boldsymbol{x}^{(l)}) < g(\boldsymbol{x}^{(l)}, \boldsymbol{x}^{(l)}) = f(\boldsymbol{x}^{(l)}). \tag{27}$$

The first strict inequality follows from Eq. (25). Since u is strictly convex, g is strictly convex and therefore $g(\boldsymbol{x}^{(l+1)}, \boldsymbol{x}^{(l)}) < g(\boldsymbol{x}^{(l)}, \boldsymbol{x}^{(l)})$ unless $\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)}$. This strict monotonic descent property will be helpful to analyze the convergence of the sparse GEV algorithm that is presented in the following subsection.

5.2 Sparse GEV algorithm

Let us return to the sparse GEV program in Eq. (21), which is of the form $\min_{\boldsymbol{x},\boldsymbol{y}}(u(\boldsymbol{x},\boldsymbol{y}) - v(\boldsymbol{x},\boldsymbol{y}))$ where $u(\boldsymbol{x},\boldsymbol{y}) = I_\Omega(\boldsymbol{x},\boldsymbol{y}) + \tau\|\boldsymbol{x}\|_2^2$ and $v(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x} - \rho\sum_{i=1}^n \log(y_i+\varepsilon)$ with $\Omega = \{(\boldsymbol{x},\boldsymbol{y}) : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ -\boldsymbol{y} \preceq \boldsymbol{x} \preceq \boldsymbol{y}\}$. Here I_Ω represents the indicator function of the convex set Ω given by

$$I_\Omega(\boldsymbol{x},\boldsymbol{y}) = \begin{cases} 0, & (\boldsymbol{x},\boldsymbol{y}) \in \Omega \\ \infty, & \text{otherwise} \end{cases}. \tag{28}$$

It is easy to check that u and v are convex. Therefore, by Eq. (26) in Example 1, the MM algorithm gives

$$\begin{aligned} (\boldsymbol{x}^{(l+1)}, \boldsymbol{y}^{(l+1)}) = \arg\min_{\boldsymbol{x},\boldsymbol{y}} \quad & \tau\|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}^{(l)} + \rho\sum_{i=1}^n \frac{y_i}{y_i^{(l)}+\varepsilon} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ -\boldsymbol{y} \preceq \boldsymbol{x} \preceq \boldsymbol{y}, \end{aligned} \tag{29}$$

resulting in a sequence of QCQPs. It is clear that $(\boldsymbol{x}^{(l+1)}, \boldsymbol{y}^{(l+1)})$ is the unique optimal solution of Eq. (29) irrespective of whether τ is zero or not.[11] The above program can be equivalently written as

$$\begin{aligned} \boldsymbol{x}^{(l+1)} = \arg\min_{\boldsymbol{x}} \quad & \tau\|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}^{(l)} + \rho\sum_{i=1}^n \frac{|x_i|}{|x_i^{(l)}|+\varepsilon} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1. \end{aligned} \tag{30}$$

11. Suppose $\tau \neq 0$. The objective function in Eq. (29) is jointly strictly convex in $(\boldsymbol{x},\boldsymbol{y})$ and therefore $(\boldsymbol{x}^{(l+1)}, \boldsymbol{y}^{(l+1)})$ is the unique optimal solution. When $\tau = 0$, the objective function is linear in $(\boldsymbol{x},\boldsymbol{y})$ and the unique optimum lies on the boundary of the constraint set.

In Appendix C, we provide another derivation for Eq. (30). Assuming $\tau \neq 0$ [12] and defining $w_i^{(l)} := \frac{1}{|x_i^{(l)}|+\varepsilon}$, $\boldsymbol{w}^{(l)} := (w_1^{(l)}, \ldots, w_n^{(l)})$ and $\boldsymbol{W}^{(l)} := \text{diag}(\boldsymbol{w}^{(l)})$, Eq. (30) reduces to

$$\begin{aligned} \boldsymbol{x}^{(l+1)} = \arg\min_{\boldsymbol{x}} \quad & \|\boldsymbol{x} - (\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)}\|_2^2 + \frac{\rho}{\tau}\|\boldsymbol{W}^{(l)}\boldsymbol{x}\|_1 \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1. \end{aligned} \tag{32}$$

Eq. (32) is very similar to LASSO (Tibshirani, 1996) except for the *weighted* ℓ_1-penalty and the quadratic constraint. When $\boldsymbol{x}^{(0)}$ is chosen such that $\boldsymbol{x}^{(0)} = a\mathbf{1}$, then the first iteration of Eq. (32) is a LASSO minimization problem except for the quadratic constraint. Let us analyze Eq. (32) to get an intuitive interpretation.

(a) $\rho = 0$: Eq. (32) reduces to $\min\{\|\boldsymbol{x}-\boldsymbol{s}^{(l)}\|_2^2 : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$, where $\boldsymbol{s}^{(l)} = (\tau^{-1}\boldsymbol{A}+\boldsymbol{I}_n)\boldsymbol{x}^{(l)}$. So, if $\boldsymbol{s}^{(l)} \in \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$, then $\boldsymbol{x}^{(l+1)} = \boldsymbol{s}^{(l)}$, else $\boldsymbol{x}^{(l+1)} = (\boldsymbol{I}_n + \mu^{(l+1)}\boldsymbol{B})^{-1}\boldsymbol{s}^{(l)}$, where $\mu^{(l+1)}$ satisfies $[\boldsymbol{s}^{(l)}]^T(\boldsymbol{I}_n+\mu^{(l+1)}\boldsymbol{B})^{-1}\boldsymbol{B}(\boldsymbol{I}_n+\mu^{(l+1)}\boldsymbol{B})^{-1}\boldsymbol{s}^{(l)} = 1$. The first term in the objective of Eq. (32) computes the best approximation to $\boldsymbol{s}^{(l)}$ in the ℓ_2-norm so that the approximation lies in the ellipsoid $\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1$. We show in Corollary 10 that the iterative algorithm in Eq. (32) with $\rho = 0$ converges to the solution of the GEV problem in Eq. (2) and therefore, the solution $\boldsymbol{x}$ is non-sparse.

(b) $\rho = \infty$: In this case, Eq. (32) reduces to $\min\{\|\boldsymbol{W}^{(l)}\boldsymbol{x}\|_1 : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$, which is a weighted ℓ_1-norm minimization problem. Intuitively, it is clear that if $x_i^{(l)}$ is small, its weighting factor, $w_i^{(l)} = (|x_i^{(l)}| + \varepsilon)^{-1}$ in the next minimization step is large, which therefore pushes $x_i^{(l+1)}$ to be small. This way the small entries in $\boldsymbol{x}$ are generally pushed toward zero as far as the constraints on $\boldsymbol{x}$ allow, therefore yielding a sparse solution.

From the above discussion, it is clear that Eq. (32) is a trade-off between the solution to the GEV problem and the solution to the weighted ℓ_1-norm problem. Eq. (32) can also be written as follows. Define $\boldsymbol{U}^{(l)} := \left[\boldsymbol{W}^{(l)}\right]^{-1} = \text{diag}(|\boldsymbol{x}^{(l)}| + \varepsilon\mathbf{1})$. Then, we have

$$\boldsymbol{x}^{(l+1)} = \boldsymbol{U}^{(l)}\boldsymbol{z}^{(l+1)}, \tag{33}$$

where

$$\begin{aligned} \boldsymbol{z}^{(l+1)} = \arg\min_{\boldsymbol{z}} \quad & \|(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)} - \boldsymbol{U}^{(l)}\boldsymbol{z}\|_2^2 + \frac{\rho}{\tau}\|\boldsymbol{z}\|_1 \\ \text{s.t.} \quad & \boldsymbol{z}^T\boldsymbol{U}^{(l)}\boldsymbol{B}\boldsymbol{U}^{(l)}\boldsymbol{z} \leq 1. \end{aligned} \tag{34}$$

If we ignore the convex quadratic constraint in the above program, it reduces to a least squares formulation with an ℓ_1-norm regularizer, which is exactly the LASSO program.[13]

12. When $\tau = 0$ (this means $\boldsymbol{A} \succeq 0$), the program in Eq. (30) reduces to

$$\begin{aligned} \boldsymbol{x}^{(l+1)} = \arg\max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}^{(l)} - \frac{\rho}{2}\|\boldsymbol{W}^{(l)}\boldsymbol{x}\|_1 \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1. \end{aligned} \tag{31}$$

13. The objective function in Eq. (34) is special because its minimizer can be obtained very easily if it lies in the constraint set. If this minimizer lies in the constraint set, then $z_i^{(l+1)} = \frac{\text{sign}(q_i^{(l)})}{p_i^{(l)}}\left(|q_i^{(l)}| - \frac{\rho}{2\tau}\right)_+$ where $p_i^{(l)} = [\boldsymbol{U}^{(l)}\boldsymbol{U}^{(l)}]_{ii}$, $q_i^{(l)} = [\boldsymbol{q}^{(l)}]_i$ with $\boldsymbol{q}^{(l)} = \boldsymbol{U}^{(l)}(\boldsymbol{I}_n + \tau^{-1}\boldsymbol{A})\boldsymbol{x}^{(l)}$ and $(x)_+ := \max(0, x)$.

Note that the update rule in Eq. (33) has a multiplicative nature, which means, if $x_i^{(l)}$ is small, then the corresponding $z_i^{(l+1)}$ is small and therefore pushes $x_i^{(l+1)}$ towards zero for increasing l. So, at the termination, where $\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)}$, we will have $\boldsymbol{z}^{(l)} \in \{0,1\}^n$ according to Eq. (33), providing the (locally) optimal sparsity pattern. Given a sparsity pattern, $\boldsymbol{z}$, the variational re-normalization[14] (Moghaddam et al., 2007a, Proposition 2) can be applied to $\boldsymbol{x}^{(l)}$ (the solution at convergence) and almost certainly improve it, by solving the cardinality unconstrained problem,

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{D}(\boldsymbol{z}) \boldsymbol{A} \boldsymbol{D}(\boldsymbol{z}) \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} = 1, \end{aligned} \tag{35}$$

which guarantees that the optimum value of Eq. (35) is at least the optimum value of Eq. (14) and $\|\boldsymbol{z}\|_0 = \|\tilde{\boldsymbol{x}}\|_0$ where $\tilde{\boldsymbol{x}}$ is the maximizer of Eq. (35). Here $\boldsymbol{D}(\boldsymbol{z}) := \text{diag}(\boldsymbol{z})$.

One can make the algorithm in Eq. (32) or Eqs. (33-34) conservative by choosing $\varepsilon = 0$ which means $\|\boldsymbol{x}\|_0$ is approximated as $\sum_{\{i:x_i \neq 0\}} \log |x_i|$ and therefore, $w_i^{(l)} = 1/|x_i^{(l)}|$ for $x_i^{(l)} \neq 0$ and $w_i^{(l)} = \infty$ for $x_i^{(l)} = 0$. Hence, it is clear from Eq. (32) or Eq. (34) that if $x_i^{(l)} = 0$, then $x_i^{(m)} = 0$, $\forall\, m > l$, which means once an element of $\boldsymbol{x}$ is set to zero, it remains zero in all the future iterations.

We refer to either Eq. (32) or Eqs. (33-34) as the *Sparse GEV algorithm*, which is detailed in Algorithm 1. Algorithm 1 requires the knowledge of ρ, which controls 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. In addition, τ and ε have to be chosen beforehand. Since ρ is a free parameter, τ and ε can be set to any value (that satisfies the constraints in Algorithm 1) and ρ can be tuned to obtain the desired sparsity. However, it has to be noted that for a fixed value of ρ, increasing τ or ε reduces sparsity.[15] So, in practice τ is chosen to be $\max(0, -\lambda_{min}(\boldsymbol{A}))$, ε to be *close* to zero and ρ is set by searching for a value that provides the desired sparsity. In Algorithm 1, we mentioned that the iterative scheme is continued until convergence. The same holds for Eq. (58). What does convergence mean here? Does the algorithm really converge? If it converges, what does it converge to? Does it converge to an optimal solution of Eq. (14)? To address these questions, in the following section, we provide the convergence analysis of Algorithm 1 using tools from global convergence theory (Zangwill, 1969).

14. The variational re-normalization suggests that given a continuous (approximate) solution (in our case, $\boldsymbol{x}^{(l)}$ at the termination), it is almost certainly better to discard the loadings, keep only the sparsity pattern (in our case, $\boldsymbol{z}^{(l)}$) and solve the smaller unconstrained subproblem shown in Eq. (35) to obtain the final loadings, given the sparsity pattern. This procedure never decreases the variance and surely improves any continuous algorithm's performance.

15. Increasing ε increases the approximation error between $\|\boldsymbol{x}\|_0$ and $\sum_{i=1}^n \frac{\log(1+|x_i|\varepsilon^{-1})}{\log(1+\varepsilon^{-1})}$ and therefore reduces sparsity. From Eq. (32), it is clear that increasing τ reduces the weight of the term $\|\boldsymbol{W}^{(l)}\boldsymbol{x}\|_1$. So, more importance is given to reducing the approximation error, $\|\boldsymbol{x} - (\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)}\|_2^2$, leading to a less sparse solution.

Algorithm 1 Sparse Generalized Eigenvalue Algorithm

Require: $\boldsymbol{A} \in \mathbb{S}^n$, $\boldsymbol{B} \succ 0$, $\varepsilon > 0$ and $\rho > 0$

Choose $\tau \geq \max(0, -\lambda_{min}(\boldsymbol{A}))$

Choose $\boldsymbol{x}^{(0)} \in \{\boldsymbol{x} : \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} \leq 1\}$

if $\tau = 0$ **then**

  repeat

    $\boldsymbol{U}^{(l)} = \mathrm{diag}(|\boldsymbol{x}^{(l)}| + \varepsilon \boldsymbol{1})$

$$\begin{aligned} \boldsymbol{z}^{(l+1)} = \arg\max_{\boldsymbol{z}} \quad & \boldsymbol{z}^T \boldsymbol{U}^{(l)} \boldsymbol{A} \boldsymbol{x}^{(l)} - \frac{\rho}{2} \|\boldsymbol{z}\|_1 \\ \text{s.t.} \quad & \boldsymbol{z}^T \boldsymbol{U}^{(l)} \boldsymbol{B} \boldsymbol{U}^{(l)} \boldsymbol{z} \leq 1 \end{aligned} \tag{36}$$

    $\boldsymbol{x}^{(l+1)} = \boldsymbol{U}^{(l)} \boldsymbol{z}^{(l+1)}$

  until convergence

else

  repeat

    $\boldsymbol{U}^{(l)} = \mathrm{diag}(|\boldsymbol{x}^{(l)}| + \varepsilon \boldsymbol{1})$

    $\boldsymbol{t}^{(l)} = (\tau^{-1} \boldsymbol{A} + \boldsymbol{I}_n) \boldsymbol{x}^{(l)}$

$$\begin{aligned} \boldsymbol{z}^{(l+1)} = \arg\min_{\boldsymbol{z}} \quad & \|\boldsymbol{U}^{(l)} \boldsymbol{z} - \boldsymbol{t}^{(l)}\|_2^2 + \frac{\rho}{\tau} \|\boldsymbol{z}\|_1 \\ \text{s.t.} \quad & \boldsymbol{z}^T \boldsymbol{U}^{(l)} \boldsymbol{B} \boldsymbol{U}^{(l)} \boldsymbol{z} \leq 1 \end{aligned} \tag{37}$$

    $\boldsymbol{x}^{(l+1)} = \boldsymbol{U}^{(l)} \boldsymbol{z}^{(l+1)}$

  until convergence

end if

return $\boldsymbol{x}^{(l)}$, $\boldsymbol{z}^{(l)}$

6. Convergence Analysis

For an iterative procedure like Algorithm 1 to be useful, it must converge to point solutions from all or at least a significant number of initialization states and not exhibit other non-linear system behaviors, such as divergence or oscillation. *Global convergence analysis* is used to investigate this behavior. We mention up front that this *does not* deal with proving convergence to a global optimum. To summarize the result in this section, we first show in Proposition 3 that every fixed point of Algorithm 1 is a stationary point of Eq. (14). We then show in Proposition 6 that all limit points of the sequence of iterates generated by Algorithm 1 are fixed points of Algorithm 1. These results are combined in Theorem 9 that guarantees the global convergence of Algorithm 1. As a special case with $\rho = 0$, in Corollary 10, we show that Algorithm 1 is equivalent to the GEV problem in Eq. (2). In the following, we introduce some notation and terminology and proceed with the derivation of the above mentioned results.

To understand the convergence of an iterative procedure like Algorithm 1, we need to understand the notion of a *set-valued mapping*, or *point-to-set mapping*, which is central to

the theory of global convergence. A point-to-set map Ψ from a set X into a set Y is defined as $\Psi : X \to \mathscr{P}(Y)$, which assigns a subset of Y with each point of X, where $\mathscr{P}(Y)$ denotes the power set of Y. We introduce few definitions related to the properties of point-to-set maps that will be used later in proving the results. Suppose X and Y are two topological spaces. A point-to-set map Ψ is said to be *closed* at $x \in X$ if $x_k \overset{k\to\infty}{\to} x$, $x_k \in X$ and $y_k \overset{k\to\infty}{\to} y$, $y_k \in \Psi(x_k)$, imply $y \in \Psi(x)$. This concept of *closure* generalizes the concept of continuity for ordinary point-to-point mappings. A point-to-set map Ψ is said to be closed on $S \subset X$ if it is closed at every point of S. A *fixed point* of the map $\Psi : X \to \mathscr{P}(X)$ is a point x for which $\{x\} = \Psi(x)$. Ψ is said to be *uniformly compact* on X if there exists a compact set H independent of x such that $\Psi(x) \subset H$ for all $x \in X$. Note that if X is compact, then Ψ is uniformly compact on X. Let $\phi : X \to \mathbb{R}_+$ be a continuous function. Ψ is said to be *monotonic* w.r.t. ϕ whenever $y \in \Psi(x)$ implies that $\phi(y) \leq \phi(x)$. If, in addition, $y \in \Psi(x)$ and $\phi(y) = \phi(x)$ imply that $y = x$, then we say that Ψ is *strictly monotonic*.

Many iterative algorithms in mathematical programming can be described using the notion of point-to-set maps. Let X be a set and $x_0 \in X$ a given point. Then an *algorithm*, $\mathcal{A}$, with initial point x_0 is a point-to-set map $\mathcal{A} : X \to \mathscr{P}(X)$ which generates a sequence $\{x_k\}_{k=1}^{\infty}$ via the rule $x_{k+1} \in \mathcal{A}(x_k)$, $k = 0, 1, \ldots$. $\mathcal{A}$ is said to be *globally convergent* if *for any chosen initial point* x_0, the sequence $\{x_k\}_{k=0}^{\infty}$ generated by $x_{k+1} \in \mathcal{A}(x_k)$ (or a sub-sequence) converges to a point for which a necessary condition of optimality holds: the Karush-Kuhn-Tucker (KKT) conditions in the case of constrained optimization and stationarity in the case of unconstrained optimization. The property of global convergence expresses, in a sense, the certainty that the algorithm works, and it corresponds, in general, to the minimum condition (i.e., the conditions that characterize the minimality) which can be formulated for any solution methods in mathematical programming. It is very important to stress the fact that it does not imply (contrary to what the term might suggest) convergence to a global optimum for all initial points x_0. With the previously defined concepts, in the following, we present the global convergence analysis of a particular class of algorithms of which Algorithm 1 is a special case.

Let us consider the following d.c. program,

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) = u(\boldsymbol{x}) - v(\boldsymbol{x}) \\ \text{s.t.} \quad & c_i(\boldsymbol{x}) \leq 0, \, \forall \, i \in [p] \\ & d_j(\boldsymbol{x}) = 0, \, \forall \, j \in [q], \end{aligned} \tag{38}$$

where $[p] := \{0, 1, \ldots, p\}$, u, v and $\{c_i\}$ are continuously differentiable convex and moreover, u and v are also strictly convex while $\{d_j\}$ are affine. Following Example 1, we propose to solve Eq. (38) by the following iterative procedure,

$$\begin{aligned} \boldsymbol{x}^{(l+1)} = \arg\min_{\boldsymbol{x}} \quad & u(\boldsymbol{x}) - \boldsymbol{x}^T \nabla v(\boldsymbol{x}^{(l)}) \\ \text{s..t} \quad & c_i(\boldsymbol{x}) \leq 0, \, \forall \, i \in [p] \\ & d_j(\boldsymbol{x}) = 0, \, \forall \, j \in [q]. \end{aligned} \tag{39}$$

It is easy to see that the iterative procedure in Eq. (29) is exactly of the same form as in Eq. (39). We now analyze the global convergence behavior of the algorithm in Eq. (39). To

start with, in the following proposition, we show that the fixed points of Eq. (39) (assuming they exist) are the *stationary points*[16] of Eq. (38).

Proposition 3 *Suppose $\boldsymbol{x}_*$ is a fixed point of the point-to-set map in Eq. (39). Then $\boldsymbol{x}_*$ is a stationary point of the program in Eq. (38).*

Proof The convexity assumption on u, v, $\{c_i\}$ and $\{d_j\}$ makes Eq. (39) a convex program. Since $\boldsymbol{x}^{(l+1)}$ is its unique minimizer (because of strict convexity), there exist Lagrange multipliers $\{\eta_i^{(l+1)}\}_{i=0}^p \subset \mathbb{R}_+$ and $\{\mu_j^{(l+1)}\}_{j=0}^q \subset \mathbb{R}$ such that the following KKT conditions hold:

$$\begin{cases} \nabla u(\boldsymbol{x}^{(l+1)}) - \nabla v(\boldsymbol{x}^{(l)}) + \sum_{i=1}^p \eta_i^{(l+1)} \nabla c_i(\boldsymbol{x}^{(l+1)}) + \sum_{j=1}^q \mu_j^{(l+1)} \nabla d_j(\boldsymbol{x}^{(l+1)}) = 0 \\ c_i(\boldsymbol{x}^{(l+1)}) \leq 0,\ \eta_i^{(l+1)} \geq 0,\ c_i(\boldsymbol{x}^{(l+1)}) \eta_i^{(l+1)} = 0,\ \forall\, i \in [p] \\ d_j(\boldsymbol{x}^{(l+1)}) = 0,\ \mu_j^{(l+1)} \in \mathbb{R},\ \forall\, j \in [q]. \end{cases} \tag{40}$$

Since $\boldsymbol{x}_* = \mathcal{A}(\boldsymbol{x}_*)$, where $\mathcal{A}$ is the point-set-map in Eq. (39), we have that there exists unique $\{\eta_i^*\}_{i=0}^p \subset \mathbb{R}_+$ and $\{\mu_j^*\}_{j=0}^q \subset \mathbb{R}$ such that the KKT conditions in Eq. (40) hold with $\boldsymbol{x}^{(l+1)} = \boldsymbol{x}^{(l)} = \boldsymbol{x}_*$, $\eta_i^{(l+1)} = \eta_i^*$, $\forall\, i \in [p]$ and $\mu_i^{(l+1)} = \mu_j^*$, $\forall\, j \in [q]$ and therefore $\boldsymbol{x}_*$ is a stationary point of the program in Eq. (38). ■

The problem now reduces to analyzing the fixed points of Eq. (39). This is performed by invoking the following global convergence theorem due to Zangwill (1969, Convergence theorem A, page 91).

Theorem 4 (Zangwill (1969)) *Let $\mathcal{A} : X \to \mathscr{P}(X)$ be a point-to-set map (an algorithm) that given a point $x_0 \in X$ generates a sequence $\{x_k\}_{k=0}^\infty$ through the iteration $x_{k+1} \in \mathcal{A}(x_k)$. Also let a solution set[17] $\Gamma \subset X$ be given. Suppose*

(1) All points x_k are in a compact set $S \subset X$.

(2) There is a continuous function $\phi : X \to \mathbb{R}$ such that:

(a) $x \notin \Gamma \Rightarrow \phi(y) < \phi(x),\ \forall\, y \in \mathcal{A}(x)$,

(b) $x \in \Gamma \Rightarrow \phi(y) \leq \phi(x),\ \forall\, y \in \mathcal{A}(x)$.

(3) $\mathcal{A}$ is closed at x if $x \notin \Gamma$.

Then the limit of any convergent subsequence of $\{x_k\}_{k=0}^\infty$ is in Γ. Furthermore, $\lim_{k\to\infty} \phi(x_k) = \phi(x^*)$ for all limit points x^*.

We also need one more result (Gunawardana and Byrne, 2005, Proposition 7) which is useful to test the closure of $\mathcal{A}$.

16. $\boldsymbol{x}_*$ is said to be a stationary point of a constrained optimization problem if it satisfies the corresponding KKT conditions (Bonnans et al., 2006, Section 13.3). Assuming constraint qualification, KKT conditions are necessary for the local optimality of $\boldsymbol{x}_*$.

17. The general idea in showing the global convergence of an algorithm, $\mathcal{A}$ is to invoke Theorem 4 by appropriately defining ϕ and Γ. For an algorithm $\mathcal{A}$ that solves the minimization problem, $\min\{f(x) : x \in \Omega\}$, the solution set, Γ is usually chosen to be the set of corresponding stationary points and ϕ can be chosen to be the objective function itself, i.e., f.

Lemma 5 (Gunawardana and Byrne (2005)) *Given a real-valued continuous function h on $X \times Y$, define the point-to-set map $\Psi : X \to \mathscr{P}(Y)$ by*

$$\begin{aligned} \Psi(x) &= \arg\min_{y' \in Y} h(x, y') \\ &= \{y : h(x, y) \leq h(x, y'), \forall\, y' \in Y\}. \end{aligned} \tag{41}$$

Then, Ψ is closed at x if $\Psi(x)$ is nonempty.

The following proposition is a direct application of Theorem 4, which establishes the link between the limit points of $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ and the fixed points of Eq. (39).

Proposition 6 *Let $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ be any sequence generated by the point-to-set map, $\mathcal{A}$ defined by Eq. (39). Assume that $\mathcal{A}$ is uniformly compact on $\Omega := \{\boldsymbol{x} : c_i(\boldsymbol{x}) \leq 0,\, i \in [p],\, d_j(\boldsymbol{x}) = 0,\, j \in [q]\}$[18] and $\mathcal{A}(\boldsymbol{x})$ is nonempty for any $\boldsymbol{x} \in \Omega$. Then all the limit points of $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ are fixed points of $\mathcal{A}$. In addition $\lim_{l\to\infty} f(\boldsymbol{x}^{(l)}) = f(\boldsymbol{x}_*)$, where $\boldsymbol{x}_*$ is some fixed point of $\mathcal{A}$.*

Proof The assumption that $\mathcal{A}$ is uniformly compact ensures that the condition (1) in Theorem 4 is satisfied. Let Γ be the set of all fixed points of $\mathcal{A}$ and let $\phi = f$. Because of the strict descent property in Eq. (27), the condition (2) in Theorem 4 is satisfied. By Lemma 5, the assumption of non-emptiness of $\mathcal{A}(x)$ for any $\boldsymbol{x} \in \Omega$ ensures that $\mathcal{A}$ is closed on Ω and therefore satisfies the condition (3) in Theorem 4. So the result follows from Theorem 4. ■

Remark 7 *As mentioned in Example 1, the program in Eq. (39) is referred to as the concave-convex procedure (CCCP), which is used to solve Eq. (38). By combining Proposition 3 and Proposition 6, it is clear that all limit points of any sequence generated by the point-to-set map associated with Eq. (39) converges to some stationary point of Eq. (38). This shows that CCCP is globally convergent, which we believe has not been shown before.*

In the above result, the convergence of $f(\boldsymbol{x}^{(l)})$ to $f(\boldsymbol{x}_*)$ does not automatically imply the convergence of $\boldsymbol{x}^{(l)}$ to $\boldsymbol{x}_*$. Note that Proposition 6 provides sub-sequence convergence and does not guarantee the convergence of $\boldsymbol{x}^{(l)}$ to $\boldsymbol{x}_*$. However, Proposition 6 can be strengthened by using the following result due to Meyer (1976, Theorem 3.1, Corollary 3.2).

Theorem 8 (Meyer (1976)) *Let $\mathcal{A} : X \to \mathscr{P}(X)$ be a point-to-set map such that $\mathcal{A}$ is uniformly compact, closed and strictly monotone on X, where X is a closed subset of $\mathbb{R}^n$. If $\{x_k\}_{k=0}^{\infty}$ is any sequence generated by $\mathcal{A}$, then all limit points will be fixed points of $\mathcal{A}$, $\phi(x_k) \to \phi(x_*) =: \phi^*$, where x_* is a fixed point, $\|x_{k+1} - x_k\| \to 0$, and either $\{x_k\}_{k=0}^{\infty}$ converges or the set of limit points of $\{x_k\}_{k=0}^{\infty}$ is connected. Define $\mathscr{F}(a) := \{x \in \mathscr{F} : \phi(x) = a\}$ where $\mathscr{F}$ is the set of fixed points of $\mathcal{A}$. If $\mathscr{F}(\phi^*)$ is finite, then any sequence $\{x_k\}_{k=0}^{\infty}$ generated by $\mathcal{A}$ converges to some x_* in $\mathscr{F}(\phi^*)$.*

18. Instead of uniform compactness, one can assume that every for $\boldsymbol{x} \in \Omega$, the set $H(\boldsymbol{x}) := \{\boldsymbol{y} | f(\boldsymbol{y}) \leq f(\boldsymbol{x}),\, \boldsymbol{y} \in \mathcal{A}(\Omega)\}$ is bounded and the result still holds.

Note that Theorem 4 does not require $\mathcal{A}$ to be strictly monotone, while this additional property provides a strong result in Theorem 8. $\mathcal{A}$ mentioned in Proposition 6 is strongly monotonic and therefore the result can be strengthened by simply invoking Theorem 8. Now, using the results in Proposition 3, Proposition 6 and Theorem 8, the following result regarding the global convergence behavior of the sparse GEV algorithm can be obtained.

Theorem 9 (Global convergence of sparse GEV algorithm) *Let $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ be any sequence generated by the sparse GEV algorithm in Algorithm 1. Then all the limit points of $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ are stationary points of the program in Eq. (14),*

$$\rho_\varepsilon \sum_{i=1}^{n} \log(\varepsilon + |x_i^{(l)}|) - [\boldsymbol{x}^{(l)}]^T \boldsymbol{A}\boldsymbol{x}^{(l)} \rightarrow \rho_\varepsilon \sum_{i=1}^{n} \log(\varepsilon + |x_i^*|) - [\boldsymbol{x}^*]^T \boldsymbol{A}\boldsymbol{x}^* := L^*, \tag{42}$$

for some stationary point $\boldsymbol{x}^*$, $\|\boldsymbol{x}^{(l+1)} - \boldsymbol{x}^{(l)}\| \rightarrow 0$, and either $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ converges or the set of limit points of $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ is a connected and compact subset of $\mathscr{S}(L^*)$, where $\mathscr{S}(a) := \{\boldsymbol{x} \in \mathscr{S} : \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \rho_\varepsilon \sum_{i=1}^{n} \log(\varepsilon + |x_i|) = -a\}$ and $\mathscr{S}$ is the set of stationary points of Eq. (14). If $\mathscr{S}(L^*)$ is finite, then any sequence $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ generated by Algorithm 1 converges to some $\boldsymbol{x}^*$ in $\mathscr{S}(L^*)$.

Proof Since Algorithm 1 and the iterative procedure in Eq. (30) are equivalent, let $\mathcal{A}$ correspond to the point-to-set map in Eq. (30). Clearly $\Omega := \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$ is compact and therefore $\mathcal{A}$ is uniformly compact. By Weierstrass theorem[19] (Minoux, 1986), it is clear that $\mathcal{A}(\boldsymbol{x})$ is nonempty for any $\boldsymbol{x} \in \Omega$ and therefore is closed on Ω. The strict descent property in Eq. (27) ensures that $\mathcal{A}$ is strictly monotonic. So by Theorem 8, all the limit points of $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ are fixed points of $\mathcal{A}$, which either converge or form a connected compact set. From Proposition 3, the set of fixed points of $\mathcal{A}$ are already in the set of stationary points of Eq. (14) and the desired result follows from Theorem 8. ■

The following corollary shows that when $\rho = 0$, the sparse GEV algorithm (Algorithm 1) matches with the GEV problem in Eq. (2).

Corollary 10 *Let $\rho = 0$ and $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) > 0$.[20] Then, any sequence $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ generated by Algorithm 1 converges to some $\boldsymbol{x}^*$ such that $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = [\boldsymbol{x}^*]^T\boldsymbol{A}\boldsymbol{x}^*$ and $[\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* = 1$.*

Proof The stationary points of Eq. (14) with $\rho = 0$ are the generalized eigenvectors of $(\boldsymbol{A}, \boldsymbol{B})$. Therefore the set $\mathscr{S}$ as defined in Theorem 9 is finite and so any sequence $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ generated by Algorithm 1 converges to some $\boldsymbol{x}^*$ in $\mathscr{S}(L^*)$ where $L^* = -[\boldsymbol{x}^*]^T\boldsymbol{A}\boldsymbol{x}^*$. We need to show that $L^* = -\lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$. Note that $\boldsymbol{x}^*$ is a fixed point of Algorithm 1. Consider Eq. (30) which is equivalent to Algorithm 1. With $\rho = 0$, solving the Lagrangian yields $\boldsymbol{x}^{(l+1)} = (\mu^{(l+1)}\boldsymbol{B} + \tau\boldsymbol{I}_n)^{-1}(\boldsymbol{A} + \tau\boldsymbol{I}_n)\boldsymbol{x}^{(l)}$, where $\mu^{(l+1)} \geq 0$ is the Lagrangian dual variable for the constraint $[\boldsymbol{x}^{(l+1)}]^T\boldsymbol{B}\boldsymbol{x}^{(l+1)} \leq 1$. At the fixed point, $\boldsymbol{x}^*$, we have $(\mu^*\boldsymbol{B} + \tau\boldsymbol{I}_n)\boldsymbol{x}^* = (\boldsymbol{A} + \tau\boldsymbol{I}_n)\boldsymbol{x}^*$ which implies

$$\boldsymbol{A}\boldsymbol{x}^* = \mu^*\boldsymbol{B}\boldsymbol{x}^*. \tag{43}$$

19. Weierstrass theorem states: If f is a real continuous function on a compact set $K \subset \mathbb{R}^n$, then the problem $\min\{f(x) : x \in K\}$ has an optimal solution $x^* \in K$.
20. See footnote 2 for the need to impose this condition.

Multiplying both sides of Eq. (43) by $[\boldsymbol{x}^*]^T$, we have

$$\begin{aligned}[\boldsymbol{x}^*]^T\boldsymbol{A}\boldsymbol{x}^* &= \mu^*[\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* = \mu^*([\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* - 1) + \mu^* \\ &= \mu^*, \end{aligned} \tag{44}$$

where we have invoked the complementary slackness condition, $\mu^*([\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* - 1) = 0$. The optimum value of Eq. (30) at the fixed point is given by $\psi^* := -2[\boldsymbol{x}^*]^T\boldsymbol{A}\boldsymbol{x}^* - \tau\|\boldsymbol{x}^*\|_2^2$, which by Eq. (44) reduces to $\psi^* = -2\mu^* - \tau\|\boldsymbol{x}^*\|_2^2$. It is easy to see that making $\mu^* > 0$, and therefore $[\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* = 1$ minimizes ψ^* instead of choosing $\mu^* = 0$ and $[\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* < 1$. Since ψ^* is minimized by choosing the maximum μ^* that satisfies Eq. (43), $(\mu^*, \boldsymbol{x}^*)$ is indeed the eigen pair that satisfies the GEV problem in Eq. (2). ■

Based on the result in Corollary 10 and the discussion regarding $\rho = 0$ below Eq. (32), it is clear that the following algorithm converges to the solution of the GEV problem in Eq. (2),

$$\boldsymbol{x}^{(l+1)} = \begin{cases} (\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)}, & [\boldsymbol{x}^{(l)}]^T(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{B}(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)} \leq 1 \\ (\boldsymbol{I}_n + \mu^{(l+1)}\boldsymbol{B})^{-1}(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)}, & \text{otherwise} \end{cases}, \tag{45}$$

where $\mu^{(l+1)} > 0$ satisfies $[\boldsymbol{x}^{(l)}]^T(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)(\boldsymbol{I}_n + \mu^{(l+1)}\boldsymbol{B})^{-1}\boldsymbol{B}(\boldsymbol{I}_n + \mu^{(l+1)}\boldsymbol{B})^{-1}(\tau^{-1}\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{x}^{(l)} = 1$ and $\tau > 0$. To elaborate a bit, the above algorithm is obtained by solving the Lagrangian associated with Eq. (32) for $\rho = 0$, where $\mu^{(l+1)} \geq 0$ is the corresponding Lagrangian multiplier. At convergence, $\boldsymbol{x}^{(l)}$ is the generalized eigenvector associated with $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$. However, solving Eq. (32) with $\rho = 0$ through the algorithm in Eq. (45) is not straightforward as $\mu^{(l+1)}$ has to be defined for each iteration. However, when $\boldsymbol{A} \succeq 0$ and $\tau = 0$, one can obtain a simple iterative algorithm to compute the generalized eigenvector associated with $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B})$, which is shown in the following corollary.

Corollary 11 *Let $\boldsymbol{A} \succeq 0$, $\tau = 0$ and $\rho = 0$. Then, any sequence $\{\boldsymbol{x}^{(l)}\}_{l=0}^{\infty}$ generated by the following algorithm*

$$\boldsymbol{x}^{(l+1)} = \frac{\boldsymbol{B}^{-1}\boldsymbol{A}\boldsymbol{x}^{(l)}}{\sqrt{[\boldsymbol{x}^{(l)}]^T\boldsymbol{A}\boldsymbol{B}^{-1}\boldsymbol{A}\boldsymbol{x}^{(l)}}} \tag{46}$$

converges to some $\boldsymbol{x}^*$ such that $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = [\boldsymbol{x}^*]^T\boldsymbol{A}\boldsymbol{x}^*$ and $[\boldsymbol{x}^*]^T\boldsymbol{B}\boldsymbol{x}^* = 1$.

Proof Consider Eq. (30) with $\tau = 0$ and $\rho = 0$. Since the objective is linear in $\boldsymbol{x}$, the minimum occurs at the boundary of the constraint set, i.e., $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1\}$. Solving the Lagrangian, we get Eq. (46). The result therefore follows from Corollary 10 which holds for any $\tau \geq 0$. ■

For a *regular* eigenvalue problem, i.e., $\boldsymbol{B} = \boldsymbol{I}_n$, the iterative procedure in Eq. (46) reduces to the *power method* as shown by the following corollary.

Corollary 12 *Let $\boldsymbol{A} \succeq 0$, $\tau = 0$, $\rho = 0$ and $\boldsymbol{B} = \boldsymbol{I}_n$. Then, Algorithm 1 is the power method for computing $\lambda_{max}(\boldsymbol{A})$.*

Proof Eq. (46) in Corollary 11 is obtained by setting $\boldsymbol{A} \succeq 0$, $\rho = 0$ and $\tau = 0$ in Algorithm 1. Now, setting $\boldsymbol{B} = \boldsymbol{I}_n$ in Eq. (46) yields $\boldsymbol{x}^{(l+1)} = \boldsymbol{A}\boldsymbol{x}^{(l)}/\|\boldsymbol{A}\boldsymbol{x}^{(l)}\|_2$, which is the power iteration for the computation of $\lambda_{max}(\boldsymbol{A})$. ■

So far, we have proposed a sparse GEV algorithm and proved its global convergence behavior. In the following sections (Sections 7-9), we consider applications of the sparse GEV problem and use the proposed algorithm (Algorithm 1) to address them.

7. Sparse Principal Component Analysis

In this section, we consider sparse PCA as a special case of the sparse GEV algorithm that we presented in Section 5. Based on the sparse GEV algorithm in Algorithm 1, we propose our sparse PCA algorithm (DC-PCA) with $\boldsymbol{A}$ being the covariance matrix, $\boldsymbol{B} = \boldsymbol{I}_n$ and $\tau = 0$. This algorithm computes the sparse eigenvector of $\boldsymbol{A}$ corresponding to $\lambda_{max}(\boldsymbol{A})$. In the following, we discuss how the DC-PCA formulation relates to SCoTLASS (Jolliffe et al., 2003) and SPCA (Zou et al., 2006) and present experiments to empirically compare different approaches to sparse PCA.

7.1 Comparison to SCoTLASS

As mentioned before, the SCoTLASS program is obtained by approximating $\|\boldsymbol{x}\|_0$ with $\|\boldsymbol{x}\|_1$ in Eq. (7) given by

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \\ \text{s.t.} \quad & \|\boldsymbol{x}\|_2^2 = 1,\ \|\boldsymbol{x}\|_1 \leq k, \end{aligned} \tag{47}$$

where $\boldsymbol{A} \succeq 0$. Let us consider the regularized version of the above program given by

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} - \rho \|\boldsymbol{x}\|_1 \\ \text{s.t.} \quad & \|\boldsymbol{x}\|_2^2 \leq 1. \end{aligned} \tag{48}$$

It is clear that Eq. (48) is not a canonical convex program because of convex maximization. So applying the MM algorithm to Eq. (48), we obtain an iterative algorithm which is the same as Algorithm 1 except that Eq. (36) is replaced by

$$\begin{aligned} \boldsymbol{z}^{(l+1)} = \arg\max_{\boldsymbol{x}} \quad & \boldsymbol{z}^T \boldsymbol{A} \boldsymbol{x}^{(l)} - \frac{\rho}{2} \|\boldsymbol{z}\|_1 \\ \text{s.t.} \quad & \|\boldsymbol{z}\|_2^2 \leq 1, \end{aligned} \tag{49}$$

with $\boldsymbol{x}^{(l+1)} = \boldsymbol{z}^{(l+1)}$. Mainly, this differs from DC-PCA in the multiplicative update. Let us assume that $x_i^{(l)} = 0$ for some l. For DC-PCA, this ensures that $x_i^{(m)} = 0$, $\forall\, m > l$ which is not guaranteed for SCoTLASS. 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 the cardinality constraint is used in DC-PCA. When $\rho = 0$, like DC-PCA, SCoTLASS also reduces to the *power iteration algorithm*.

7.2 Comparison to SPCA

Let $\boldsymbol{Q}$ be a $r \times n$ matrix, where r and n are the number of observations and the number of variables respectively, with the column means being zero. Suppose $\boldsymbol{Q}$ has an SVD given by

$\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T$, where $\boldsymbol{U}$ are the principal components of unit length and the columns of $\boldsymbol{V}$ are the corresponding loadings of the principal components. Let $\boldsymbol{y}_i = [\boldsymbol{U}\boldsymbol{\Lambda}]_i$, $\forall\, i$. Zou et al. (2006, Theorem 1) posed PCA as a regression problem and showed that $[\boldsymbol{V}]_i = \boldsymbol{x}^\star/\|\boldsymbol{x}^\star\|_2$, where

$$\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y}_i - \boldsymbol{Q}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_2^2, \tag{50}$$

where $\lambda > 0$. This is equivalent to solving for an eigenvector of $\boldsymbol{Q}^T\boldsymbol{Q}$. Therefore, solving for the eigenvectors of a positive semi-definite matrix is posed as a ridge regression problem in Eq. (50). To solve for sparse eigenvectors, Zou et al. (2006) introduced an ℓ_1-penalty term in Eq. (50) resulting in the following *elastic net* called SPCA,

$$\boldsymbol{x}' = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y}_i - \boldsymbol{Q}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_2^2 + \lambda_1\|\boldsymbol{x}\|_1, \tag{51}$$

where $\lambda_1 > 0$. This problem can be treated in the Bayesian setting as: given the likelihood on $\boldsymbol{y}_i$, $\boldsymbol{y}_i|\boldsymbol{x}, \sigma^2 \sim \mathcal{G}(\boldsymbol{Q}\boldsymbol{x}, \sigma^2\boldsymbol{I})$, which is a circular normal random variable with mean $\boldsymbol{Q}\boldsymbol{x}$ (conditioned on $\boldsymbol{x}$), and a prior distribution on $\boldsymbol{x}$, $\boldsymbol{x}|\beta^2, \gamma \sim \mathcal{G}(\boldsymbol{0}, \beta^2\boldsymbol{I})\prod_i \exp(-\gamma|x_i|)$, which is the product of circular Gaussian and product of Laplacian densities, compute the maximum a posteriori (MAP) estimate of $\boldsymbol{x}$. The parameters λ and λ_1 are the hyper-parameters of the prior distribution on $\boldsymbol{x}$ and are related to σ^2, β^2 and γ. As aforementioned, our method can be interpreted as defining an improper prior over $\boldsymbol{x}$, which promotes sparsity (Tipping, 2001). We use $p(\boldsymbol{x}) \propto \prod_i \frac{1}{|x_i|+\varepsilon}$ (instead of $\prod_i \exp(-\gamma|x_i|)$) as the prior so that $\boldsymbol{x}|\varepsilon, \gamma \sim \mathcal{G}(\boldsymbol{0},^2 \boldsymbol{I})p(\boldsymbol{x})$ and therefore our formulation results in

$$\min_{\boldsymbol{x}} ||\boldsymbol{y}_i - \boldsymbol{Q}\boldsymbol{x}||_2^2 + \lambda||\boldsymbol{x}||_2^2 + \lambda_1 \sum_i \log(|x_i| + \varepsilon). \tag{52}$$

Since the problem in Eq. (52) is equivalent to Eq. (14) with $\boldsymbol{B} = \boldsymbol{I}_n$, it is clear that DC-PCA provides sparser solutions than SPCA. It is to be noted that the SPCA framework is not extendible to other settings like FDA or CCA unlike our formulation which is generic.

7.3 Experimental results

In this subsection, we illustrate the effectiveness of DC-PCA in terms of sparsity and scalability on different real-life datasets. Since SPCA[21] and DSPCA[22] have demonstrated improved performance over simple thresholding and SCoTLASS, we choose these methods as baselines to compare the performance of our method against. Also, based on the discussion in Section 7.1, it should be clear that DC-PCA performs better than SCoTLASS. The results show that our method has *better scalability* and achieves *more sparsity* than SPCA and DSPCA, while explaining at least as much variance. Based on the discussion in Section 4.2, it is clear that we would like to solve the approximate sparse GEV problem in Eq. (14) with a small value of ε. Therefore, in all our experiments, we fixed ε to 10^{-4}.

7.3.1 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) capture 87% of the total

21. LARS-based Elastic-net SPCA MATLAB toolbox (Sjöstrand, 2005) was used to solve for SPCA.
22. DSPCA software is available at `http://www.prince ton.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. (2006) and d'Aspremont et al. (2007) 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}
	1	-.48	-.48	0	0	.18	0	-.25	-.34	-.42	-.40	0	0	0
SPCA	2	0	0	.79	.62	0	0	0	-.02	0	0	0	.01	0
	3	0	0	0	0	.64	.59	.49	0	0	0	0	0	-.02
	1	-.56	-.58	0	0	0	0	-.26	-.10	-.37	-.36	0	0	0
DSPCA	2	0	0	.71	.71	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	-.79	-.61	0	0	0	0	0	.01
	1	.45	.46	0	0	0	0	.37	.33	.40	.42	0	0	0
DC-PCA	2	0	0	.71	.71	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	.82	.58	0	0	0	0	0	0

variance, so other methods compare their explanatory power using 6 sparse PCs.[23] 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 2(a)) performs significantly better than SPCA and DSPCA. Comparing the cumulative variance and cumulative cardinality, Figure 2(a–b) show that DC-PCA explains more variance with fewer non-zero loadings than SPCA and DSPCA. For the first PC, Figure 2(c) shows that DC-PCA consistently explains more variance with better sparsity than SPCA and DSPCA. Figure 2(d) shows the variation of sparsity and explained variance w.r.t. ρ for the first PC for DC-PCA. 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.

7.3.2 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

23. The discussion so far dealt with computing the sparse eigenvector corresponding to $\lambda_{max}(\boldsymbol{A})$. To compute the subsequent eigenvectors that are sparse, usually the sparse PCA algorithm is applied to a sequence of deflated matrices (for example, see d'Aspremont et al. (2007)) given by $\{\boldsymbol{A}_0 = \boldsymbol{A};\ \boldsymbol{A}_{i+1} = \boldsymbol{A}_i - (\boldsymbol{u}_i^T \boldsymbol{A}_i \boldsymbol{u}_i)\boldsymbol{u}_i \boldsymbol{u}_i^T\}$, where $\boldsymbol{u}_i$ is the output of a sparse PCA algorithm with $\boldsymbol{A} = \boldsymbol{A}_i$. This is appropriate only when $\boldsymbol{u}_i^T \boldsymbol{u}_j = 0,\ i \neq j$. Otherwise, there is a possibility that $\boldsymbol{A}_i \prec 0$, for some i. So, one should be careful in computing the cumulative variance explained by $\boldsymbol{u}_i$'s as $\sum_i \boldsymbol{u}_i^T \boldsymbol{A}_i \boldsymbol{u}_i$. Instead, the sequence of deflated matrices should be computed as $\{\boldsymbol{A}_0 = \boldsymbol{A};\ \boldsymbol{A}_{i+1} = \boldsymbol{A}_i - (\boldsymbol{v}_i^T \boldsymbol{A}_i \boldsymbol{v}_i)\boldsymbol{v}_i \boldsymbol{v}_i^T\}$, where $\boldsymbol{v}_i = \boldsymbol{u}_i - \mathcal{P}_{\mathscr{S}_{i-1}} \boldsymbol{u}_i$. $\mathcal{P}_{\mathscr{S}_{i-1}} \boldsymbol{u}_i$ represents the orthogonal projection of $\boldsymbol{u}_i$ onto the subspace, $\mathscr{S}_{i-1}$, spanned by $\{\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1}\}$ with $\boldsymbol{v}_0 = \boldsymbol{u}_0$. The cumulative variance is then calculated as $\sum_i \boldsymbol{v}_i^T \boldsymbol{A}_i \boldsymbol{v}_i$. This formulation is used in the experiments in Section 7.3 where the performance of DC-PCA is compared to other algorithms in terms of the sparsity vs. cumulative variance explained.

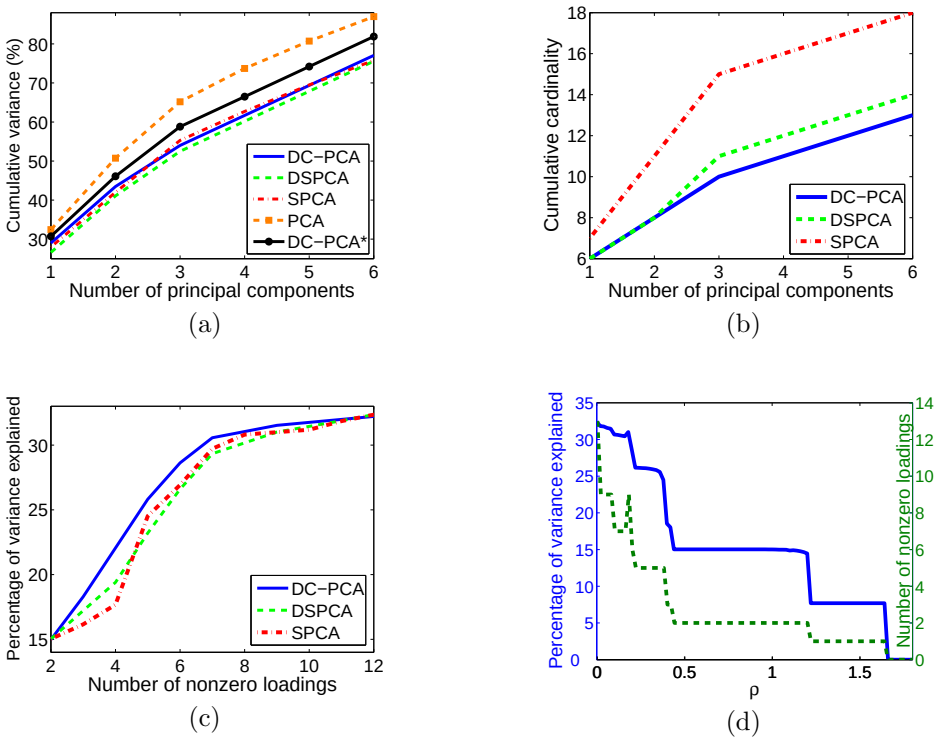

Figure 2: Pit props: (a) cumulative variance (b) cumulative cardinality for first 6 sparse principal components (PCs) (c) percentage of explained variance (PEV) vs. sparsity for the first PC (d) dependence of sparsity and PEV on ρ for the first PC for DC-PCA. DC-PCA* in (a) represents DC-PCA evaluated at SPCA's sparsity pattern of $(7, 4, 4, 1, 1, 1)$.

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 3(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 Section 7.3.4), experiments for DSPCA could not be completed in reasonable time. So, the results do not include a comparison with DSPCA.

7.3.3 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 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

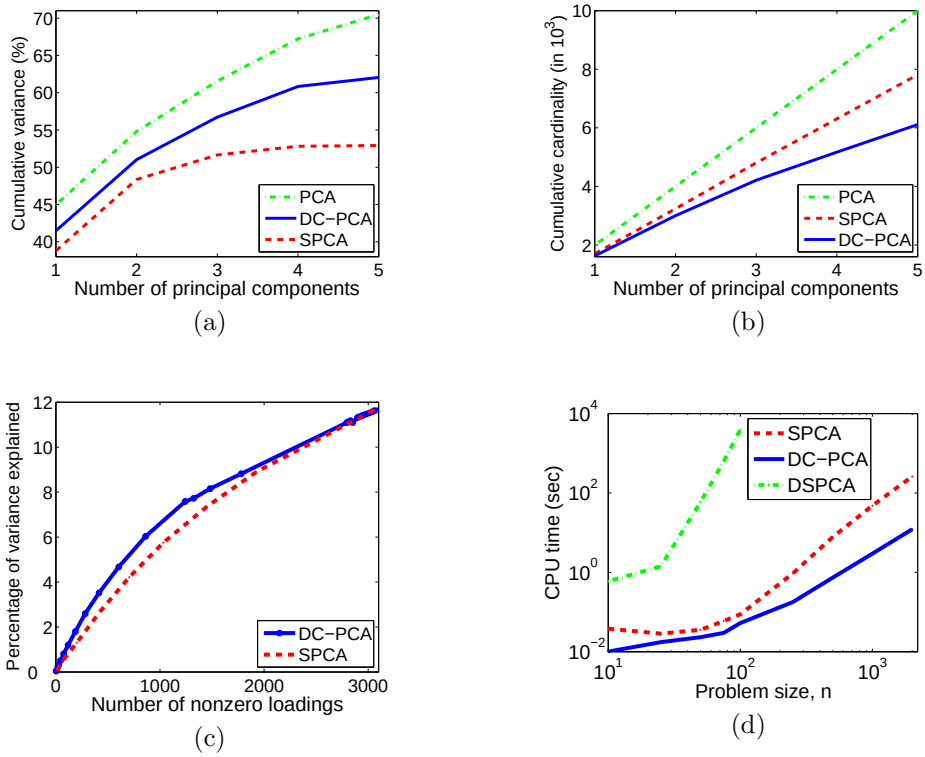

Figure 3: Colon cancer: (a) cumulative variance (b) cumulative cardinality for first 5 sparse principal components (PCs). Leukemia: (c) percentage of variance explained vs. sparsity for the first PC. (d) CPU time vs. problem size for randomly chosen problems. (a–c) show that DC-PCA explains more variance with fewer non-zero loadings than SPCA.

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 3(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, while still outperforming SPCA.

7.3.4 Computing Time vs. Problem Size

DC-PCA is a sequence of QCQPs with theoretical worst-case complexity of $O(mn^3)$, which is the same as for SPCA and better than the $O(n^{5.5})$ for DSPCA. Here, m is the number of iterations before convergence and n the dimensionality of the data. To empirically compute the running time complexity of these methods, we ran[24] 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. (2007, Section 6.4)). Figure 3(d) shows the plot of

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

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 better to large-dimensional problems than SPCA and DSPCA and is thus preferred over these methods as it also has better sparsity vs. explained variance performance.

8. Sparse Canonical Correlation Analysis

In this section, we consider sparse CCA as a special case of the sparse GEV algorithm and present two CCA applications where sparsity is helpful. We call our sparse CCA algorithm DC-CCA, where $\boldsymbol{A}$ and $\boldsymbol{B}$ are determined from the covariance and cross-covariance matrices as shown below Eq. (4). Note that $\boldsymbol{A}$ is indefinite and therefore in our experiments, we choose $\tau = -\lambda_{min}(\boldsymbol{A})$ in Algorithm 1. In the following, we present two sparse CCA applications, one related to the task of cross-language document retrieval and the other dealing with semantic annotation and retrieval of music (Torres et al., 2007a,b). We believe that this is the first time a sparse CCA algorithm has been proposed. d'Aspremont et al. (2007, Section 2.3) presents a semi-definite relaxation of a variation of CCA, where $\boldsymbol{B}$ is assumed to be an identity matrix. In Section 4, we presented a SDP relaxation for sparse CCA. However, in this section, we use DC-CCA (based on Algorithm 1) to perform sparse CCA (as it scales better for large problem sizes) and show its performance in the above mentioned applications.

8.1 Cross-language document retrieval

The problem of cross-language document retrieval involves a collection of documents, $\{D_i\}_{i=1}^N$ with each document being represented in different languages, say English and French. The goal of the task is, given a query string in one language, retrieve the most relevant document(s) in the target language. The first step is to obtain a semantic representation of the documents in both languages, which models the correlation between translated versions, so we can detect similarities in content between the two document spaces (one for English and the other for French). This is exactly what CCA does by finding a low-dimensional representation in both languages, with maximal correlation between them. Vinokourov et al. (2003) used CCA to address this problem and showed that the CCA approach performs better than the latent semantic indexing approach used by Littman et al. (1998). CCA provides an *efficient* basis representation (that captures the maximal correlation) for the two document spaces.

Using a bag-of-words representation for the documents, sparse CCA would allow to find a low-dimensional model based on a small subset of words in both languages. This would improve the interpretability of the model and could identify small subsets of words that are used in similar contexts in both languages and, possibly, are translations of one another. Representing documents by their similarity to all other documents (e.g., by taking inner products of bag-of-word vectors, as explained below), sparse CCA would create a low-dimensional model that only requires to measure similarity for a small subset of the training documents. This would immediately improve storage requirements and the efficiency of retrieval computations. In this study, we follow the second approach, representing documents by their similarity to all other training documents by applying a linear kernel function to

a binary bag-of-words representation of the documents, as proposed in Vinokourov et al. (2003). This will illustrate how we can achieve significant sparsity without significant loss of retrieval performance.

More specifically, each version of a document (English or French) is modeled using a bag-of-words feature vector. Within a feature vector, we associate an element in $\{0, 1\}$ with each word w_i in its language vocabulary. A value of 1 indicates that w_i is found in the document. We collect the feature vectors into the $N \times P$ matrix $\boldsymbol{E}$, where we collect the English feature vectors, and the $N \times Q$ matrix $\boldsymbol{F}$, where we collect the French feature vectors. N is the number of documents and P and Q are the vocabulary sizes of $\boldsymbol{E}$ and $\boldsymbol{F}$ respectively. Computing similarity between English documents as the inner product between their binary bag-of-words vectors (i.e., the rows of $\boldsymbol{E}$) results in computing an $N \times N$ data matrix $\boldsymbol{E}\boldsymbol{E}^T$. Similarly, we compute an $N \times N$ data matrix $\boldsymbol{F}\boldsymbol{F}^T$ and obtain two feature spaces which are both N-dimensional.

By applying sparse CCA, we effectively perform simultaneous feature selection across two vector spaces and characterize the content of and correlation between English and French documents in an efficient manner. We use the DC-CCA algorithm, using covariance and cross-variance matrices associated with the document matrices $\boldsymbol{E}\boldsymbol{E}^T$ and $\boldsymbol{F}\boldsymbol{F}^T$ and obtain successive pairs of sparse canonical components which we stack into the columns of $\boldsymbol{V}_E$ and $\boldsymbol{V}_F$. (Subsequent pairs of these sparse canonical components are obtained by deflating $\boldsymbol{E}\boldsymbol{E}^T$ and $\boldsymbol{F}\boldsymbol{F}^T$ with respect to previous canonical components. For a detailed review on deflation, we refer the reader to Shawe-Taylor and Christianini (2004).) Then, given a query in an input language, say English, we convert the query into the appropriate feature vector, $\boldsymbol{q}_E$. We project $\boldsymbol{q}_E$ into the subspace spanned by the canonical components of the English language by computing $\boldsymbol{V}_E^T \boldsymbol{q}_E$[25]. Similarly, we project all the French training documents onto the subspace spanned by the canonical components, $\boldsymbol{V}_F$ associated with the French language. Finally we perform document retrieval by selecting those French documents whose projections are closest to the projected query, where we measure similarity in a nearest neighbor sense.

8.1.1 Experimental Details

The data set used was the Aligned Hansards of the 36th Parliament of Canada (Germann, 2001), which is a collection of 1.3 million pairs of text chunks (sentences or smaller fragments) aligned into English and French translations. The text chunks are split into documents based on $***$ delimiters. Then, stop words and rare words (those that occur less than 3 times) are removed and we are left with an 1800×26328 English document-by-term matrix and a 1800×30167 French matrix. Computing $\boldsymbol{E}\boldsymbol{E}^T$ and $\boldsymbol{F}\boldsymbol{F}^T$ results in matrices of size 1800×1800.

To generate a query, we select English test documents from a test set not used for training. The appropriate retrieval result is the corresponding French language version of the query document. To perform retrieval, the query and the French test documents are projected onto the canonical components and retrieval is performed as described before.

25. Notice how this projection, onto the sparse canonical components, only requires to compute a few elements of $\boldsymbol{q}_E$, i.e., the ones corresponding to the non-zero loadings of the canonical components; differently said, we only need to compute the similarity of the query document to a small subset of all training documents.

Table 2: Average area under the ROC curve (in %) using CCA and sparse CCA (DC-CCA) in a cross-language document retrieval task. d represents the number of canonical components and *sparsity* represents the percentage of total number of zero loadings in the canonical components.

d	100	200	300	400	500
CCA	99.92	99.93	99.96	99.95	99.93
DC-CCA	95.72	97.57	98.45	98.75	99.04
Sparsity	87.15	87.56	87.95	88.21	88.44

Table 2 shows the performance of DC-CCA (sparse CCA) against CCA. We measure our results using the average area under the ROC curve (average AROC). The results in Table 2 are shown in percentages. To go into detail, for each test query we generate an ROC curve from the ranked retrieval results. Results are ranked according to their projected feature vector's Euclidean distance from the query. The area under this ROC curve is used to measure performance. For example, if the first returned document was the most relevant (i.e., the corresponding French language version of the query document) this would result in an ROC with area under the curve (AROC) of 1. If the most relevant document came in above the 75^{th} percentile of all documents this would lead to an AROC 0.75, and so on. So, we're basically measuring how highly the corresponding French language document ranks in the retrieval results. For a collection of queries we take the simple average of each query's AROC to obtain the average AROC. An average AROC with value of 1 is best, a value of 0.5 is as good as chance.

In Table 2, we compare retrieval using sparse CCA to regular CCA. For sparse CCA we used a sparsity parameter that led to loadings that were approximately 10% of the full dimensionality, which means the canonical components are approximately 90% sparse. We see that we are able to achieve good retrieval rates using sparse CCA, only slightly sacrificing performance compared to regular CCA. This is the key result of this section: we can achieve performance close to regular CCA, by using only about 12% of the number of loadings (i.e., documents) required by regular CCA. This shows that sparse CCA can narrow in on the most informative dimensions exhibited by data and can be used as an effective dimensionality reduction technique.

8.2 Vocabulary selection for music information retrieval

In this subsection we provide a short summary of the results in Torres et al. (2007a), which nicely illustrate how sparse CCA can be used to improve the performance of a statistical musical query application, by identifying problematic query words and eliminating them from the model. The application involves a computer audition system (Turnbull et al., 2008) that can annotate songs with semantically meaningful words or *tags* (such as *rock* or *mellow*), or retrieve songs from a database, based on a semantic query. This system is based on a joint probabilistic model between words and acoustic signals, learned from a training data set of songs and song tags. "Noisy" words, that are not or only weakly related to the musical content, will decrease the system's performance and waste computational

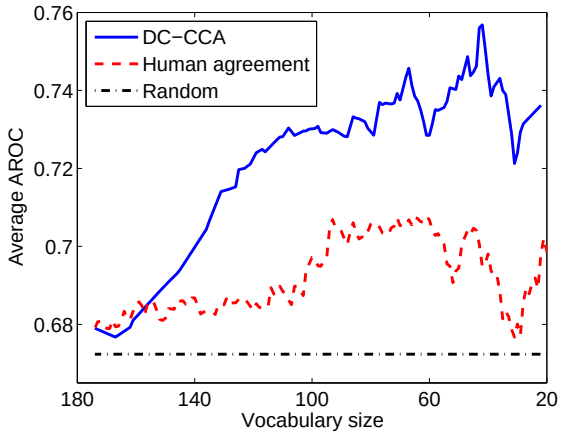

Figure 4: Comparison of vocabulary selection techniques for music retrieval.

resources. Sparse CCA is employed to prune away those noisy words and improve the system's performance.

The details of this experiment are beyond the scope of this work and can be found in Torres et al. (2007a). In short, each song from the CAL-500 data set[26] is represented in two different spaces: a semantic space based on a bag-of-words representation of a song's semantic tags and an audio space based on Mel-frequency cepstral coefficients (Mckinney, 2003) extracted from a song's audio content. This representation allows sparse CCA to identify a small subset of words spanning a semantic subspace that is highly correlated with audio content. In Figure 4, we use sparse CCA to generate a sequence of vocabularies of progressively smaller size, ranging from full size (containing about 180 words) to very sparse (containing about 20 words), depicted on the horizontal axis. For each vocabulary size, the computer audition system is trained and the average area under the receiver operating characteristic curve (AROC) is shown on the vertical axis, measuring its retrieval performance on an independent test set. The AROC (ranging between 0.5 for randomly ranked retrieval results and 1.0 for a perfect ranking) initially clearly improves, as sparse CCA (DC-CCA) generates vocabularies of smaller size: it is effectively removing noisy words that are detrimental for the system's performance. Also shown in Figure 4 are the results of training based on two alternative vocabulary selection techniques: random selection (offering no improvement) and a heuristic that eliminates words exhibiting less agreement amongst the human subjects that were surveyed to collect CAL-500 (only offering a slight improvement, initially).

In summary, Torres et al. (2007a) illustrates that vocabulary selection using sparse CCA significantly improves the retrieval performance of a computer audition system (by effectively removing noisy words), outperforming a random baseline and a human agreement heuristic.

26. The CAL-500 data set consists of a set of songs, annotated with semantic tags, by conducting human surveys. More details can be found in Turnbull et al. (2008).

9. Sparse Fisher Discriminant Analysis

In this section, we show that the FDA problem is an *interesting* special case of the GEV problem and that the special structure of $\boldsymbol{A}$ allows the sparse FDA problem to be solved more efficiently than the general sparse GEV problem.

Let us consider the GEV problem in Eq. (2) with $\boldsymbol{A} \in \mathbb{S}_+^n$, $\boldsymbol{B} \in \mathbb{S}_{++}^n$ and $\text{rank}(\boldsymbol{A}) = 1$. This is exactly the FDA problem as shown in Eq. (6) where $\boldsymbol{A}$ is of the form $\boldsymbol{A} = \boldsymbol{a}\boldsymbol{a}^T$, with $\boldsymbol{a} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \in \mathbb{R}^n$. The corresponding GEV problem is written as

$$\begin{aligned} \lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = \max_{\boldsymbol{x}} \quad & (\boldsymbol{a}^T\boldsymbol{x})^2 \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} = 1, \end{aligned} \tag{53}$$

which can also be written as $\lambda_{max}(\boldsymbol{A}, \boldsymbol{B}) = \max_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{(\boldsymbol{a}^T\boldsymbol{x})^2}{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}}$. Since we are primarily interested in the maximizer of Eq. (53), we can rewrite it as

$$\min_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}}{(\boldsymbol{a}^T\boldsymbol{x})^2} \qquad \equiv \qquad \min\{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \,:\, \boldsymbol{a}^T\boldsymbol{x} = 1\}. \tag{54}$$

The advantage of Eq. (54) will become clear when we consider its sparse version, i.e., after introducing the constraint $\{\boldsymbol{x} : \|\boldsymbol{x}\|_0 \leq k\}$ in Eq. (54). Clearly, introducing the sparsity constraint makes the problem NP-hard. However, introducing an ℓ_1-norm relaxation in this formulation gives rise to a *convex program*,

$$\min\{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \,:\, \boldsymbol{a}^T\boldsymbol{x} = 1,\, \|\boldsymbol{x}\|_1 \leq k\} \qquad \equiv \qquad \min\{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} + \nu\|\boldsymbol{x}\|_1 \,:\, \boldsymbol{a}^T\boldsymbol{x} = 1\}, \tag{55}$$

more specifically a QP. The equivalence to the regularized version with $\nu > 0$ as the regularization parameter has been shown previously.

Note that a transformation similar to the one leading to Eq. (54) can be performed for the GEV problem with any, general $\boldsymbol{A} \in \mathbb{S}^n$, i.e., writing the GEV problem as a minimization problem,

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = 1. \end{aligned} \tag{56}$$

This formulation, however, is not useful to simplify solving a GEV problem in general. Indeed, consider the sparse version of the problem in Eq. (56) with the sparsity constraint $\{\boldsymbol{x} : \|\boldsymbol{x}\|_0 \leq k\}$ relaxed to $\{\boldsymbol{x} : \|\boldsymbol{x}\|_1 \leq k\}$. Because of the quadratic equality constraint, the resulting program is non-convex for any $\boldsymbol{A}$. Suppose say that the constraint set $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = 1\}$ is relaxed to $\{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 1\}$. If $\boldsymbol{A} \notin \mathbb{S}_+^n$, the program is still non-convex as the constraint is a non-convex set. If $\boldsymbol{A} \in \mathbb{S}_+^n$, then the optimum occurs at $\boldsymbol{x} = \boldsymbol{0}$. Therefore, the minimization formulation of the GEV problem in Eq. (56) is not useful, unlike the case where $\boldsymbol{A} \in \mathbb{S}_+^n$ and $\text{rank}(\boldsymbol{A}) = 1$.

Based on the discussion so far, it is clear that the sparse FDA problem can be solved as a convex QP, unlike sparse PCA or sparse CCA, whose convex relaxation results in a SDP as discussed in Section 4.1. Suppose that one would like to use a better approximation

to $\|\boldsymbol{x}\|_0$ than $\|\boldsymbol{x}\|_1$, for sparse FDA. Using the approximation we proposed in this work, Eq. (55) reduces to

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} + \nu_\varepsilon \sum_{i=1}^{n} \log(\varepsilon + |x_i|) \\ \text{s.t.} \quad & \boldsymbol{a}^T \boldsymbol{x} = 1, \end{aligned} \tag{57}$$

where $\nu_\varepsilon := \nu / \log(1 + \varepsilon^{-1})$. Applying the MM method to the above program results in the following iterative scheme,

$$\begin{aligned} \boldsymbol{x}^{(l+1)} = \arg\min_{\boldsymbol{x}} \quad & \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} + \nu_\varepsilon \sum_{i=1}^{n} \frac{|x_i|}{|x_i^{(l)}| + \varepsilon} \\ \text{s.t.} \quad & \boldsymbol{a}^T \boldsymbol{x} = 1, \end{aligned} \tag{58}$$

which is a sequence of QPs unlike Algorithm 1, which is a sequence of QCQPs. Therefore, the nice structure of $\boldsymbol{A}$ makes the corresponding sparse GEV problem computationally efficient. So, one should solve the sparse FDA problem by using Eq. (55) or Eq. (58) instead of using the convex SDP in Eq. (11) or Algorithm 1.

Suykens et al. (2002, Chapter 3) and Mika et al. (2001, Proposition 1) have shown connections between the FDA formulation in Eq. (54) with $\boldsymbol{a} = \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$ and $\boldsymbol{B} = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2$ (see the paragraph below Eq. (6) for details) and least-squares support vector machines (classifiers that minimize the squared loss). Therefore, sparse FDA is equivalent to feature selection with a squared loss objective, i.e., the sparse FDA formulation in Eq. (55) is equivalent to LASSO, while the formulation in Eq. (57) is similar to the one considered in Weston et al. (2003). Since these are well studied problems, we do not pursue further showing the numerical performance of sparse FDA.

10. Conclusion and Discussion

We study the problem of finding sparse eigenvectors for generalized eigenvalue problems. After proposing a non-convex but tight approximation to the cardinality constraint, we formulate the resulting optimization problem as a d.c. program and derive an iterative solution algorithm, based on the majorization-minimization method. This results in solving a sequence of quadratically constrained quadratic programs, an algorithm which exhibits global convergence behavior, as we show. We also derive sparse PCA (DC-PCA) and sparse CCA (DC-CCA) algorithms as special cases of our proposed algorithm. Empirical results demonstrate the performance of the proposed algorithm for sparse PCA and sparse CCA applications. In the case of sparse PCA, we experimentally demonstrate on real-life data of varying dimensionality that the proposed algorithm (DC-PCA) explains more variance with sparser features than SPCA (Zou et al., 2006) and DSPCA (d'Aspremont et al., 2007) at better computational speed (low CPU time). Although many algorithms have been proposed for sparse PCA, we believe that this is the first time a sparse CCA algorithm has been proposed. We illustrate its practical relevance for two applications, cross-language document retrieval and vocabulary selection for music information retrieval.

The proposed algorithm does not allow to set the regularization parameter a priori, to guarantee a given sparsity level. This is similar for SPCA. Semidefinite relaxation methods,

on the other hand (e.g., DSPCA in the context of sparse PCA) are better suited to achieve a given sparsity level in one shot, by incorporating an explicit constraint on the sparsity of the solution (although, eventually, through relaxation, an approximation of the original problem is solved). Since the algorithm we propose solves a LASSO problem in each step but with a quadratic constraint, one can use a modified version of path following techniques like least angle regression (Efron et al., 2004) to learn the entire regularization path.

Acknowledgments

Bharath Sriperumbudur thanks Suvrit Sra for constructive discussions while the former was an intern at the Max Planck Institute for Biological Cybernetics, Tübingen. The authors wish to acknowledge the support from the National Science Foundation (grant DMS-MSPA 0625409), the Fair Isaac Corporation and the University of California MICRO program.

Appendix A. Derivation of the Dual and Bi-dual Programs for Eq. (8)

Consider the ℓ_1-norm relaxed sparse GEV problem in Eq. (8), which we reproduce here for convenience.

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ \|\boldsymbol{x}\|_1 \leq k. \end{aligned} \tag{59}$$

The above problem can be re-written as

$$\begin{aligned} \max_{\boldsymbol{x},\,\boldsymbol{y}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1,\ -\boldsymbol{y} \preceq \boldsymbol{x} \preceq \boldsymbol{y} \\ & \boldsymbol{y}^T\mathbf{1} \leq k. \end{aligned} \tag{60}$$

The corresponding Lagrangian dual problem is given by

$$\min_{\substack{\beta\geq 0,\mu\geq 0\\ \boldsymbol{u}\succeq\mathbf{0},\boldsymbol{s}\succeq\mathbf{0}}} \ \max_{\boldsymbol{x},\,\boldsymbol{y}} \ L(\boldsymbol{x},\boldsymbol{y},\beta,\mu,\boldsymbol{u},\boldsymbol{s}),$$

where

$$L(\boldsymbol{x},\boldsymbol{y},\beta,\mu,\boldsymbol{u},\boldsymbol{s}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \mu(\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}-1) - \beta(\boldsymbol{y}^T\mathbf{1}-k) - \boldsymbol{u}^T(\boldsymbol{x}-\boldsymbol{y}) + \boldsymbol{s}^T(\boldsymbol{x}+\boldsymbol{y}). \tag{61}$$

Let us first maximize L over $\boldsymbol{x}$. By Lemma 3.6 of Lemaréchal and Oustry (1999), the necessary and sufficient condition for $Q(\boldsymbol{x}) = \boldsymbol{x}^T(\boldsymbol{A}-\mu\boldsymbol{B})\boldsymbol{x} + \boldsymbol{x}^T(\boldsymbol{s}-\boldsymbol{u})$ to have a finite upper bound over $\mathbb{R}^n$ is $\mu\boldsymbol{B}-\boldsymbol{A} \succeq 0$ and $\boldsymbol{s}-\boldsymbol{u} \in \mathcal{R}(\mu\boldsymbol{B}-\boldsymbol{A})$. Differentiating L w.r.t. $\boldsymbol{x}$ yields $\boldsymbol{x} = \frac{1}{2}(\mu\boldsymbol{B}-\boldsymbol{A})^\dagger(\boldsymbol{s}-\boldsymbol{u})$. Similarly, while maximizing L w.r.t. $\boldsymbol{y}$, the necessary and sufficient condition for $R(\boldsymbol{y}) = \boldsymbol{y}^T(\boldsymbol{s}+\boldsymbol{u}-\beta\mathbf{1})$ to have a finite upper bound over $\mathbb{R}^n$ is $\boldsymbol{s}+\boldsymbol{u} = \beta\mathbf{1}$. Therefore, the dual program can be written as

$$\begin{aligned} \min_{\boldsymbol{u},\boldsymbol{s},\beta,\mu} \quad & \frac{1}{4}(\boldsymbol{u}-\boldsymbol{s})^T(\mu\boldsymbol{B}-\boldsymbol{A})^\dagger(\boldsymbol{u}-\boldsymbol{s}) + \beta k + \mu \\ \text{s.t.} \quad & \mu\boldsymbol{B}-\boldsymbol{A} \succeq 0,\ \boldsymbol{u}-\boldsymbol{s} \in \mathcal{R}(\mu\boldsymbol{B}-\boldsymbol{A}) \\ & \boldsymbol{s}+\boldsymbol{u} = \beta\mathbf{1},\ \beta \geq 0,\ \mu \geq 0,\ \boldsymbol{u} \succeq \mathbf{0},\ \boldsymbol{s} \succeq \mathbf{0}, \end{aligned} \tag{62}$$

which is equivalent to

$$\begin{aligned} \min_{\boldsymbol{r},\beta,\mu} \quad & \frac{1}{4}\boldsymbol{r}^T(\mu\boldsymbol{B}-\boldsymbol{A})^\dagger\boldsymbol{r}+\beta k+\mu \\ \text{s.t.} \quad & \mu\boldsymbol{B}-\boldsymbol{A}\succeq 0,\ \boldsymbol{r}\in\mathcal{R}(\mu\boldsymbol{B}-\boldsymbol{A}) \\ & -\beta\mathbf{1}\preceq\boldsymbol{r}\preceq\beta\mathbf{1},\ \beta\geq 0,\ \mu\geq 0, \end{aligned} \tag{63}$$

resulting in Eq. (9). By invoking the Schur's complement lemma, the dual can be written as

$$\begin{aligned} \min_{\boldsymbol{r},t,\beta,\mu} \quad & t+\beta k+\mu \\ \text{s.t.} \quad & -\beta\mathbf{1}\preceq\boldsymbol{r}\preceq\beta\mathbf{1},\ \beta\geq 0,\ \mu\geq 0 \\ & \begin{pmatrix} \mu\boldsymbol{B}-\boldsymbol{A} & -\frac{1}{2}\boldsymbol{r} \\ -\frac{1}{2}\boldsymbol{r}^T & t \end{pmatrix}\succeq 0. \end{aligned} \tag{64}$$

The bi-dual associated with Eq. (59) is given by

$$\max_{\substack{\phi\in\mathbb{R},\alpha\geq 0,\theta\geq 0\\ \boldsymbol{\tau}\succeq\mathbf{0},\boldsymbol{\eta}\succeq\mathbf{0},\boldsymbol{x}\succeq\mathbf{0}\\ \boldsymbol{X}\succeq 0}} \ \min_{\substack{\boldsymbol{r}\succeq\mathbf{0},t\in\mathbb{R}\\ \beta\geq 0,\mu\geq 0}} \tilde{L}(\boldsymbol{r},t,\beta,\mu,\phi,\alpha,\theta,\boldsymbol{\tau},\boldsymbol{X},\boldsymbol{x},\boldsymbol{\eta}), \tag{65}$$

where $\tilde{L}$ is the Lagrangian associated with Eq. (64) given by

$$\begin{aligned} \tilde{L}(\boldsymbol{r},t,\beta,\mu,\phi,\alpha,\theta,\boldsymbol{\tau},\boldsymbol{X},\boldsymbol{x},\boldsymbol{\eta}) &= t+\beta k+\mu+\boldsymbol{\eta}^T(\boldsymbol{r}-\beta\mathbf{1})-\boldsymbol{\tau}^T(\boldsymbol{r}+\beta\mathbf{1})-\alpha\mu-\theta\beta \\ &\quad -\mathrm{tr}\left[\begin{pmatrix} \boldsymbol{X} & \boldsymbol{x} \\ \boldsymbol{x}^T & \phi \end{pmatrix}\begin{pmatrix} \mu\boldsymbol{B}-\boldsymbol{A} & -\frac{1}{2}\boldsymbol{r} \\ -\frac{1}{2}\boldsymbol{r}^T & t \end{pmatrix}\right] \\ &= \mathrm{tr}(\boldsymbol{X}\boldsymbol{A})+\mu(1-\alpha-\mathrm{tr}(\boldsymbol{X}\boldsymbol{B}))+t(1-\phi) \\ &\quad +\beta(k-\boldsymbol{\eta}^T\mathbf{1}-\boldsymbol{\tau}^T\mathbf{1}-\theta)+\boldsymbol{r}^T(\boldsymbol{\eta}-\boldsymbol{\tau}+\boldsymbol{x}). \end{aligned} \tag{66}$$

Minimizing the above Lagrangian results in

$$\begin{aligned} \max_{\alpha,\theta,\boldsymbol{\tau},\boldsymbol{\eta},\boldsymbol{x},\boldsymbol{X}} \quad & \mathrm{tr}(\boldsymbol{X}\boldsymbol{A}) \\ \text{s.t.} \quad & \alpha+\mathrm{tr}(\boldsymbol{X}\boldsymbol{B})=1,\ \boldsymbol{x}+\boldsymbol{\eta}=\boldsymbol{\tau},\ (\boldsymbol{\eta}+\boldsymbol{\tau})^T\mathbf{1}+\theta=k \\ & \begin{pmatrix} \boldsymbol{X} & \boldsymbol{x} \\ \boldsymbol{x}^T & 1 \end{pmatrix}\succeq 0, \end{aligned} \tag{67}$$

which is equivalent to

$$\begin{aligned} \max_{\boldsymbol{x},\boldsymbol{X}} \quad & \mathrm{tr}(\boldsymbol{X}\boldsymbol{A}) \\ \text{s.t.} \quad & \mathrm{tr}(\boldsymbol{X}\boldsymbol{B})\leq 1, \|\boldsymbol{x}\|_1\leq k \\ & \begin{pmatrix} \boldsymbol{X} & \boldsymbol{x} \\ \boldsymbol{x}^T & 1 \end{pmatrix}\succeq 0, \end{aligned} \tag{68}$$

as shown in Eq. (11).

Appendix B. Derivation of the approximate sparse GEV program in Eq. (14)

Starting from the sparse GEV program in Eq. (13), the approximate sparse GEV program in Eq. (14) can be obtained as follows. Replacing $\|\boldsymbol{x}\|_0$ by $\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(1+|x_i|/\varepsilon)}{\log(1+1/\varepsilon)}$ in Eq. (13), we have

$$\begin{aligned}\max_{\boldsymbol{x}} \quad & \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(1+|x_i|/\varepsilon)}{\log(1+1/\varepsilon)} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1. \end{aligned} \tag{69}$$

Consider the objective function in the above program, given by

$$\begin{aligned}\phi(\boldsymbol{x},\tilde{\rho}) &:= \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(1+|x_i|/\varepsilon)}{\log(1+1/\varepsilon)} \\ &= \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(\varepsilon+|x_i|)}{\log(1+1/\varepsilon)} + \tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log\varepsilon}{\log(1+1/\varepsilon)} \\ &= \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}\lim_{\varepsilon\to 0}\sum_{i=1}^{n}\frac{\log(\varepsilon+|x_i|)}{\log(1+1/\varepsilon)} - n\tilde{\rho} \\ &= \lim_{\varepsilon\to 0}\left[\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}_\varepsilon\sum_{i=1}^{n}\log(\varepsilon+|x_i|) - n\tilde{\rho}\right], \end{aligned} \tag{70}$$

where $\tilde{\rho}_\varepsilon := \tilde{\rho}/\log(1+\varepsilon^{-1})$. Therefore, Eq. (69) reduces to

$$\begin{aligned}\max_{\boldsymbol{x}} \quad & \lim_{\varepsilon\to 0}\left[\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \tilde{\rho}_\varepsilon\sum_{i=1}^{n}\log(\varepsilon+|x_i|)\right] - n\tilde{\rho} \\ \text{s.t.} \quad & \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1, \end{aligned} \tag{71}$$

and is equivalent to the sparse GEV program in Eq. (13). The approximate sparse GEV program in Eq. (14) is obtained by neglecting the limit in Eq. (71) and choosing $\varepsilon > 0$.

Appendix C. Derivation of Eq. (30)

Eq. (30) can be derived differently as follows. Let

$$\begin{aligned} f(\boldsymbol{x}) &= \rho\sum_{i=1}^{n}\log(\varepsilon+|x_i|) - \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \\ &= \tau\|\boldsymbol{x}\|_2^2 + \rho\sum_{i=1}^{n}\log(\varepsilon+|x_i|) - \boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x}, \end{aligned} \tag{72}$$

where $\tau \geq \max(0, -\lambda_{min}(\boldsymbol{A}))$. Using the inequality $\log(z) \leq z-1$, $\forall\, z \in \mathbb{R}_+$ with $z = \frac{|x_i|+\varepsilon}{|y_i|+\varepsilon}$, we have

$$\log(\varepsilon+|x_i|) \leq \log(\varepsilon+|y_i|) + \frac{|x_i|-|y_i|}{|y_i|+\varepsilon},\ \forall\, \boldsymbol{x},\boldsymbol{y}.$$

Since $\boldsymbol{A}+\tau\boldsymbol{I}_n \succeq 0$, we have $\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{x} \geq \boldsymbol{y}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y}+2(\boldsymbol{x}-\boldsymbol{y})^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y}$, $\forall\, \boldsymbol{x},\boldsymbol{y}$. It is easy to check that

$$g(\boldsymbol{x},\boldsymbol{y}) = \tau\|\boldsymbol{x}\|_2^2 - 2\boldsymbol{x}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y}+\boldsymbol{y}^T(\boldsymbol{A}+\tau\boldsymbol{I}_n)\boldsymbol{y}+\rho\sum_{i=1}^{n}\log(\varepsilon+|y_i|)+\rho\sum_{i=1}^{n}\frac{|x_i|-|y_i|}{|y_i|+\varepsilon} \quad (73)$$

majorizes f over $\mathbb{R}^n \times \mathbb{R}^n$ and therefore over $\Omega \times \Omega$ where $\Omega = \{\boldsymbol{x} : \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} \leq 1\}$. In addition, the inequalities are strict unless $\boldsymbol{x} = \boldsymbol{y}$. Eq. (14) is equivalent to minimizing f over Ω and therefore the minimization step in Eq. (23) with g in Eq. (73) results in Eq. (30). Note that g is strictly convex in $\boldsymbol{x}$ and so $g(\boldsymbol{x}',\boldsymbol{y}) < g(\boldsymbol{x},\boldsymbol{y})$, $\forall\, \boldsymbol{x} \in \Omega$ unless $\boldsymbol{x} = \boldsymbol{x}'$ where $\boldsymbol{x}' = \arg\min\{g(\boldsymbol{x},\boldsymbol{y}) : \boldsymbol{x} \in \Omega\}$. Therefore, $f(\boldsymbol{x}') < g(\boldsymbol{x}',\boldsymbol{y}) < g(\boldsymbol{y},\boldsymbol{y}) = f(\boldsymbol{y})$ unless $\boldsymbol{x}' = \boldsymbol{y}$ where the equality holds.

References

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

D. Böhning and B. G. Lindsay. Monotonicity of quadratic-approximation algorithms. *Annals of the Institute of Statistical Mathematics*, 40(4):641–663, 1988.

J. F. Bonnans, J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal. *Numerical Optimization: Theoretical and Practical Aspects*. Springer-Verlag, 2006.

P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *Proc. 15th International Conf. on Machine Learning*, pages 82–90. Morgan Kaufmann, San Francisco, CA, 1998.

J. Cadima and I. Jolliffe. Loadings and correlations in the interpretation of principal components. *Applied Statistics*, 22:203–214, 1995.

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

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.

J. deLeeuw. Applications of convex analysis to multidimensional scaling. In J. R. Barra, F. Brodeau, G. Romier, and B. Van Cutsem, editors, *Recent advantages in Statistics*, pages 133–146, Amsterdam, The Netherlands, 1977. North Holland Publishing Company.

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.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.

L. El Ghaoui. On the quality of a semidefinite programming bound for sparse principal component analysis, 2006. http://arxiv.org/PS_cache/math/pdf/0601/0601448.pdf.

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.

U. Germann. Aligned Hansards of the 36^{th} parliament of Canada, 2001. http://www.isi.edu/natural-language/download/hansard/.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M. K. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, October 1999.

A. Gunawardana and W. Byrne. Convergence theorems for generalized alternating minimization procedures. *Journal of Machine Learning Research*, 6:2049–2073, 2005.

W. J. Heiser. Correspondence analysis with least absolute residuals. *Comput. Stat. Data Analysis*, 5:337–356, 1987.

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

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

P. J. Huber. *Robust Statistics*. John Wiley, New York, 1981.

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

D. R. Hunter and R. Li. Variable selection using MM algorithms. *The Annals of Statistics*, 33:1617–1642, 2005.

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

I. Jolliffe. *Principal component analysis*. Springer-Verlag, New York, USA, 1986.

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

M. Kuss and T. Graepel. The geometry of kernel canonical correlation analysis. Technical report 108, Max Planck Institute for Biological Cybernetics, Tübingen, Germany, 2003.

K. Lange, D. R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions with discussion. *Journal of Computational and Graphical Statistics*, 9(1):1–59, 2000.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, Cambridge, 2001.

C. Lemaréchal and F. Oustry. Semidefinite relaxations and Lagrangian duality with application to combinatorial optimization. Technical Report RR3710, INRIA, 1999.

M. L. Littman, S. T. Dumais, and T. K. Landauer. Automatic cross-language information retrieval using latent semantic indexing. In G. Grefenstette, editor, *Cross-Language Information Retrieval*, pages 51–62. Kluwer Academic Publishers, 1998.

G. McCabe. Principal variables. *Technometrics*, 26:137–144, 1984.

M. F. Mckinney. Features for audio and music classification. In *Proc. of the International Symposium on Music Information Retrieval*, pages 151–158, 2003.

X.-L. Meng. Discussion on "optimization transfer using surrogate objective functions". *Journal of Computational and Graphical Statistics*, 9(1):35–43, 2000.

R. R. Meyer. Sufficient conditions for the convergence of monotonic mathematical programming algorithms. *Journal of Computer and System Sciences*, 12:108–121, 1976.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX*, pages 41–48. IEEE, 1999.

S. Mika, G. Rätsch, and K.-R. Müller. A mathematical programming approach to the kernel Fisher algorithm. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

M. Minoux. *Mathematical Programming: Theory and Algorithms*. John Wiley & Sons Ltd., 1986.

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, 2007a. MIT Press.

B. Moghaddam, Y. Weiss, and S. Avidan. Generalized spectral bounds for sparse LDA. In *Proc. of International Conference on Machine Learning*, 2007b.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming, Series A*, 103:127–152, 2005.

J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, 1970.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

J. Shawe-Taylor and N. Christianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

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

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

G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, 1986.

J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific Publishing, Singapore, 2002.

Pham Dinh Tao and Le Thi Hoai An. D.c. optimization algorithms for solving the trust region subproblem. *SIAM Journal of Optimization*, 8:476–505, 1998.

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

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

D. Torres, D. Turnbull, L. Barrington, and G. R. G. Lanckriet. Identifying words that are musically meaningful. In *Proc. of International Symposium on Music Information and Retrieval*, 2007a.

D. A. Torres, D. Turnbull, B. K. Sriperumbudur, L. Barrington, and G. R. G. Lanckriet. Finding musically meaningful words using sparse CCA. In *Music, Brain & Cognition Workshop, NIPS*, 2007b.

D. Turnbull, L. Barrington, D. Torres, and G. R. G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Trans. on Audio, Speech and Language Processing*, 16:467–476, 2008.

L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

A. Vinokourov, J. Shawe-Taylor, and N. Cristianini. Inferring a semantic representation of text via cross-language correlation analysis. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1473–1480, Cambridge, MA, 2003. MIT Press.

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.

C. F. J. Wu. On the convergence properties of the EM algorithm. *Annals of Statistics*, 11 (1):95–103, 1983.

A. L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15: 915–936, 2003.

W. I. Zangwill. *Nonlinear Programming: A Unified Approach*. Prentice-Hall, Englewood Cliffs, N.J., 1969.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J. R. Statist. Soc. B*, 67:301–320, 2005.

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