Sparse PCA: Convex Relaxations, Algorithms and Applications

Youwei Zhang[1], Alexandre d'Aspremont[2], and Laurent El Ghaoui[3]

[1] EECS, U.C. Berkeley. Berkeley, CA 94720. zyw@eecs.berkeley.edu
[2] ORFE, Princeton University, Princeton, NJ 08544. aspremon@princeton.edu
[3] EECS, U.C. Berkeley. Berkeley, CA 94720. elghaoui@eecs.berkeley.edu

Summary. Given a sample covariance matrix, we examine the problem of maximizing the variance explained by a linear combination of the input variables while constraining the number of nonzero coefficients in this combination. This is known as sparse principal component analysis and has a wide array of applications in machine learning and engineering. Unfortunately, this problem is also combinatorially hard and we discuss convex relaxation techniques that efficiently produce good approximate solutions. We then describe several algorithms solving these relaxations as well as greedy algorithms that iteratively improve the solution quality. Finally, we illustrate sparse PCA in several applications, ranging from senate voting and finance to news data.

1 Introduction

Principal component analysis (PCA) is a classical tool for data analysis, visualization and dimensionality reduction and has a wide range of applications throughout science and engineering. Starting from a multivariate data set, PCA finds linear combinations of the variables called *principal components*, corresponding to orthogonal directions maximizing variance in the data. Numerically, a full PCA involves a singular value decomposition of the data matrix.

One of the key shortcomings of PCA is that the factors are linear combinations of *all* original variables; that is, most of factor coefficients (or loadings) are non-zero. This means that while PCA facilitates model interpretation and visualization by concentrating the information in a few factors, the factors themselves are still constructed using all variables, hence are often hard to interpret.

In many applications, the coordinate axes involved in the factors have a direct physical interpretation. In financial or biological applications, each axis might correspond to a specific asset or gene. In problems such as these, it is natural to seek a trade-off between the two goals of *statistical fidelity* (explaining most of the variance in the data) and *interpretability* (making sure that the factors involve only a few coordinate axes). Solutions that have only a few nonzero coefficients in the principal components are usually easier to interpret. Moreover, in some applications, nonzero coefficients have a direct cost (*e.g.*, transaction costs in finance) hence there may be a direct trade-off between statistical fidelity and practicality. Our aim here is to efficiently derive *sparse principal components*, i.e, a set of sparse vectors that explain a maximum amount of variance. Our motivation is that in many applications, the decrease in statistical fidelity required to obtain sparse factors is small and relatively benign.

In what follows, we will focus on the problem of finding sparse factors which explain a maximum amount of variance in the original data, which can be written

$$\max_{\|z\|\leq 1} z^T \Sigma z - \rho\,\mathbf{Card}(z) \tag{1}$$

in the variable $z \in \mathbf{R}^n$, where $\Sigma \in \mathbf{S}_n$ is the (symmetric positive semi-definite) sample covariance matrix, ρ is a parameter controlling sparsity, and $\mathbf{Card}(z)$ denotes the cardinal (or ℓ_0 norm) of z, i.e. the number of non zero coefficients of z.

While PCA is numerically easy, each factor requires computing a leading eigenvector, which can be done in $O(n^2)$ floating point operations using the Lanczos method for example (see e.g. [GVL90, §8.3, §9.1.1] or [Saa92] for details), sparse PCA is a hard combinatorial problem. In fact, [MWA06a] show that the subset selection problem for ordinary least squares, which is NP-hard [Nat95], can be reduced to a sparse generalized eigenvalue problem, of which sparse PCA is a particular instance. Sometimes factor rotation techniques are used to post-process the results from PCA and improve interpretability (see QUARTIMAX by [NW54], VARIMAX by [Kai58] or [Jol95] for a discussion). Results by e.g. [AW09] show that this naive approach has significantly worst convergence rates than the relaxations we present here. Another straightforward solution is to *threshold* to zero loadings with small magnitude [CJ95], but outside of easy cases, the methods highlighted below always perform better in situation when only a few observations are available or when significant noise is present.

A more systematic approach to the problem arose in recent years, with various researchers proposing nonconvex algorithms (e.g., SCoTLASS by [JTU03], SLRA by [ZZS02] or D.C. based methods [STL07] which find modified principal components with zero loadings. The SPCA algorithm, which is based on the representation of PCA as a regression-type optimization problem [ZHT06], allows the application of the LASSO [Tib96], a penalization technique based on the ℓ_1 norm. With the exception of simple thresholding, all the algorithms above require solving non convex problems. Recently also, [dEGJL07] derived an ℓ_1 based semidefinite relaxation for the sparse PCA problem (1) with a complexity of $O(n^4\sqrt{\log n})$ for a given ρ. [MWA06b] used greedy search and branch-and-bound methods to solve small instances of problem (1) exactly and get good solutions for larger ones. Each step of this greedy algorithm has complexity $O(n^3)$, leading to a total complexity of $O(n^4)$ for a full set of solutions. [MWA07] improve this bound in the regression/discrimination case. [JNRS08] use an extension of the power method to (locally) solve the problem defined here, as well as the "block" problem of finding several sparse principal components at once. Loss of orthogonality means that there is no natural method for deflating the matrix once a sparse principal component is found and [Mac09] discusses several options, comparing the variance vs. orthogonality/sparsity tradeoffs they imply. Finally, [AW09] derive explicit sample size thresholds for recovery of true sparse vector using either simple thresholding methods or semidefinite relaxations, in a spiked model for the covariance.

Here, we detail two semidefinite relaxations for sparse PCA, and describe algorithms to solve the relaxations efficiently. We also test these techniques on various data sets: newsgroup data, Senate voting records and stock market returns.

Notation

For a vector $z \in \mathbf{R}$, we let $\|z\|_1 = \sum_{i=1}^n |z_i|$ and $\|z\| = \left(\sum_{i=1}^n z_i^2\right)^{1/2}$, $\mathbf{Card}(z)$ is the cardinality of z, i.e. the number of nonzero coefficients of z, while the support I of z is the set $\{i : z_i \neq 0\}$ and we use I^c to denote its complement. For $\beta \in \mathbf{R}$, we write $\beta_+ = \max\{\beta, 0\}$ and for $X \in \mathbf{S}_n$ (the set of symmetric matrix of size $n \times n$) with eigenvalues λ_i, $\mathbf{Tr}(X)_+ = \sum_{i=1}^n \max\{\lambda_i, 0\}$. The vector of all ones is written $\mathbf{1}$, while the identity matrix is written $\mathbf{I}$. The diagonal matrix with the vector u on the diagonal is written $\mathbf{diag}(u)$.

2 Semidefinite relaxations

Let $\Sigma \in \mathbf{S}_n$ be a symmetric matrix. We consider the following sparse PCA problem

$$\phi(\rho) \equiv \max_{\|z\| \leq 1} z^T \Sigma z - \rho\, \mathbf{Card}(z) \tag{2}$$

in the variable $z \in \mathbf{R}^n$ where $\rho > 0$ is a parameter controlling sparsity. We assume without loss of generality that $\Sigma \in \mathbf{S}_n$ is positive semidefinite and that the n variables are ordered by decreasing marginal variances, i.e. that $\Sigma_{11} \geq \ldots \geq \Sigma_{nn}$. We also assume that we are given a square root A of the matrix Σ with $\Sigma = A^T A$, where $A \in \mathbf{R}^{n \times n}$ and we denote by $a_1, \ldots, a_n \in \mathbf{R}^n$ the columns of A. Note that the problem and our algorithms are invariant by permutations of Σ and by the choice of square root A. In practice, we are very often given the data matrix A instead of the covariance Σ.

A problem that is directly related to (2) is that of computing a cardinality constrained maximum eigenvalue, by solving

$$\begin{array}{ll} \text{maximize} & z^T \Sigma z \\ \text{subject to} & \mathbf{Card}(z) \leq k \\ & \|z\| = 1, \end{array} \tag{3}$$

in the variable $z \in \mathbf{R}^n$. Of course, this problem and (2) are related. By weak duality, an upper bound on the optimal value of (3) is given by

$$\inf_{\rho \in P} \phi(\rho) + \rho k.$$

where P is the set of penalty values for which $\phi(\rho)$ has been computed. This means in particular that if a point z is provably optimal for (2), it is also globally optimum for (3) with $k = \mathbf{Card}(z)$.

2.1 A semidefinite relaxation with ℓ_1 penalization

Here, we briefly recall the ℓ_1 based relaxation derived in [dEGJL07]. Following the *lifting procedure* for semidefinite relaxation described in [LS91], [Ali95], [LO99] for example, we rewrite (3) as

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(\Sigma X) \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & \mathbf{Card}(X) \leq k^2 \\ & X \succeq 0, \;\; \mathbf{Rank}(X) = 1, \end{array} \tag{4}$$

in the (matrix) variable $X \in \mathbf{S}^n$. Programs (3) and (4) are equivalent, indeed if X is a solution to the above problem, then $X \succeq 0$ and $\mathbf{Rank}(X) = 1$ mean that we have $X = xx^T$, while $\mathbf{Tr}(X) = 1$ implies that $\|x\|_2 = 1$. Finally, if $X = xx^T$ then $\mathbf{Card}(X) \leq k^2$ is equivalent to $\mathbf{Card}(x) \leq k$. We have made some progress by turning the convex maximization objective $x^T \Sigma x$ and the nonconvex constraint $\|x\|_2 = 1$ into a linear constraint and linear objective. Problem (4) is, however, still nonconvex and we need to relax both the rank and cardinality constraints.

Since for every $u \in \mathbf{R}^n$, $\mathbf{Card}(u) = q$ implies $\|u\|_1 \leq \sqrt{q}\|u\|_2$, we can replace the nonconvex constraint $\mathbf{Card}(X) \leq k^2$, by a weaker but convex constraint: $\mathbf{1}^T|X|\mathbf{1} \leq k$, where we exploit the property that $\|X\|_F = \sqrt{x^T x} = 1$ when $X = xx^T$ and $\mathbf{Tr}(X) = 1$. If we drop the rank constraint, we can form a relaxation of (4) and (3) as

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(\Sigma X) \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & \mathbf{1}^T|X|\mathbf{1} \leq k \\ & X \succeq 0, \end{array} \tag{5}$$

which is a semidefinite program in the variable $X \in \mathbf{S}^n$, where k is an integer parameter controlling the sparsity of the solution. The optimal value of this program will be an upper bound on the optimal value of the variational problem in (3). Here, the relaxation of $\mathbf{Card}(X)$ in $\mathbf{1}^T|X|\mathbf{1}$ corresponds to a classic technique which replaces the (non-convex) cardinality or l_0 norm of a vector x with its largest convex lower bound on the unit box: $|x|$, the l_1 norm of x (see [FHB01] or [DT05] for other applications).

Problem (5) can be interpreted as a robust formulation of the maximum eigenvalue problem, with additive, componentwise uncertainty in the input matrix Σ. We again assume A to be symmetric and positive semidefinite. If we consider a variation in which we penalize by the ℓ_1 norm of the matrix X instead of imposing a hard bound, to get

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(\Sigma X) - \rho \mathbf{1}^T|X|\mathbf{1} \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & X \succeq 0, \end{array} \tag{6}$$

which is a semidefinite program in the variable $X \in \mathbf{S}^n$, where $\rho > 0$ controls the magnitude of the penalty. We can rewrite this problem as

$$\max_{X \succeq 0, \mathbf{Tr}(X)=1} \ \min_{|U_{ij}| \leq \rho} \mathbf{Tr}(X(\Sigma + U)) \tag{7}$$

in the variables $X \in \mathbf{S}^n$ and $U \in \mathbf{S}^n$. This yields the following dual to (6)

$$\begin{array}{ll} \text{minimize} & \lambda^{\max}(\Sigma + U) \\ \text{subject to} & |U_{ij}| \leq \rho, \quad i, j = 1, \ldots, n, \end{array} \tag{8}$$

which is a maximum eigenvalue problem with variable $U \in \mathbf{S}^n$. This gives a natural robustness interpretation to the relaxation in (6): it corresponds to a worst-case maximum eigenvalue computation, with componentwise bounded noise of intensity ρ imposed on the matrix coefficients.

Finally, the KKT conditions (see [BV04, §5.9.2]) for problem (6) and (8) are given by

$$\begin{cases} (\Sigma + U)X = \lambda^{\max}(\Sigma + U)X \\ U \circ X = -\rho|X| \\ \mathbf{Tr}(X) = 1, \ X \succeq 0 \\ |U_{ij}| \leq \rho, \quad i, j = 1, \ldots, n. \end{cases} \tag{9}$$

If the eigenvalue $\lambda^{\max}(\Sigma+U)$ is simple (when, for example, $\lambda^{\max}(A)$ is simple and ρ is sufficiently small), the first condition means that $\mathbf{Rank}(X) = 1$ and the semidefinite relaxation is *tight*, with in particular $\mathbf{Card}(X) = \mathbf{Card}(x)^2$ if x is the dominant eigenvector of X. When the optimal solution X is not of rank one because of degeneracy (i.e. when $\lambda^{\max}(\Sigma + U)$ has multiplicity strictly larger than one), we can truncate X as in [Ali95, LO99], retaining only the dominant eigenvector x as an approximate solution to the original problem. In that degenerate scenario however, the dominant eigenvector of X is not guaranteed to be as sparse as the matrix itself.

2.2 A semidefinite relaxation with ℓ_0 penalization

We summarize here the results in [dBEG08]. We begin by reformulating (2) as a relatively simple convex maximization problem. Suppose that $\rho \geq \Sigma_{11}$. Since $z^T \Sigma z \leq \Sigma_{11}(\sum_{i=1}^n |z_i|)^2$ and $(\sum_{i=1}^n |z_i|)^2 \leq \|z\|^2 \,\mathbf{Card}(z)$ for all $z \in \mathbf{R}^n$, we have

$$\begin{aligned}\phi(\rho) &= \max\nolimits_{\|z\|\leq 1} z^T \Sigma z - \rho\,\mathbf{Card}(z)\\ &\leq (\Sigma_{11} - \rho)\,\mathbf{Card}(z)\\ &\leq 0,\end{aligned}$$

hence the optimal solution to (2) when $\rho \geq \Sigma_{11}$ is $z = 0$. From now on, we assume $\rho \leq \Sigma_{11}$ in which case the inequality $\|z\| \leq 1$ is tight. We can represent the sparsity pattern of a vector z by a vector $u \in \{0,1\}^n$ and rewrite (2) in the equivalent form

$$\begin{aligned}\phi(\rho) &= \max_{u\in\{0,1\}^n} \lambda_{\max}(\mathbf{diag}(u)\Sigma\,\mathbf{diag}(u)) - \rho\mathbf{1}^T u\\ &= \max_{u\in\{0,1\}^n} \lambda_{\max}(\mathbf{diag}(u)A^T A\,\mathbf{diag}(u)) - \rho\mathbf{1}^T u\\ &= \max_{u\in\{0,1\}^n} \lambda_{\max}(A\,\mathbf{diag}(u)A^T) - \rho\mathbf{1}^T u,\end{aligned}$$

using the fact that $\mathbf{diag}(u)^2 = \mathbf{diag}(u)$ for all variables $u \in \{0,1\}^n$ and that for any matrix B, $\lambda_{\max}(B^T B) = \lambda_{\max}(BB^T)$. We then have

$$\begin{aligned}\phi(\rho) &= \max_{u\in\{0,1\}^n} \lambda_{\max}(A\,\mathbf{diag}(u)A^T) - \rho\mathbf{1}^T u\\ &= \max_{\|x\|=1}\ \max_{u\in\{0,1\}^n} x^T A\,\mathbf{diag}(u)A^T x - \rho\mathbf{1}^T u\\ &= \max_{\|x\|=1}\ \max_{u\in\{0,1\}^n} \sum_{i=1}^n u_i((a_i^T x)^2 - \rho).\end{aligned}$$

Hence we finally get, after maximizing in u (and using $\max_{v\in\{0,1\}} \beta v = \beta_+$)

$$\phi(\rho) = \max_{\|x\|=1} \sum_{i=1}^n ((a_i^T x)^2 - \rho)_+, \tag{10}$$

which is a nonconvex problem in the variable $x \in \mathbf{R}^n$. We then select variables i such that $(a_i^T x)^2 - \rho > 0$. Note that if $\Sigma_{ii} = a_i^T a_i < \rho$, we must have $(a_i^T x)^2 \leq \|a_i\|^2 \|x\|^2 < \rho$ hence variable i will never be part of the optimal subset and we can remove it.

Because the variable x appears solely through $X = xx^T$, we can reformulate the problem in terms of X only, using the fact that when $\|x\| = 1$, $X = xx^T$ is equivalent to $\mathbf{Tr}(X) = 1$, $X \succeq 0$ and $\mathbf{Rank}(X) = 1$. We thus rewrite (10) as

$$\begin{aligned}\phi(\rho) = \max.\ & \textstyle\sum_{i=1}^n (a_i^T X a_i - \rho)_+\\ \text{s.t.}\ & \mathbf{Tr}(X) = 1,\ \ \mathbf{Rank}(X) = 1\\ & X \succeq 0.\end{aligned}$$

Note that because we are maximizing a convex function $\Delta_n = \{X \in \mathbf{S}_n : \ \mathbf{Tr}(X) = 1, \ X \succeq 0\}$ which is convex, the solution must be an extreme point of Δ_n (i.e. a rank one matrix), hence we can drop the rank constraint here. Unfortunately, $X \mapsto (a_i^T X a_i - \rho)_+$, the function we are *maximizing*, is convex in X and not concave, which means that the above problem is still hard. However, we show below that on rank one elements of Δ_n, it is also equal to a concave function of X, and we use this to produce a semidefinite relaxation of problem (2).

Proposition 1. *Let $A \in \mathbf{R}^{n\times n}$, $\rho \geq 0$ and denote by $a_1, \ldots, a_n \in \mathbf{R}^n$ the columns of A, an upper bound on*

$$\begin{array}{ll} \phi(\rho) = \textit{max.} & \sum_{i=1}^n (a_i^T X a_i - \rho)_+ \\ \textit{s.t.} & \mathbf{Tr}(X) = 1, \ X \succeq 0, \ \mathbf{Rank}(X) = 1 \end{array} \tag{11}$$

can be computed by solving

$$\begin{array}{ll} \psi(\rho) = \textit{max.} & \sum_{i=1}^n \mathbf{Tr}(X^{1/2} B_i X^{1/2})_+ \\ \textit{s.t.} & \mathbf{Tr}(X) = 1, \ X \succeq 0. \end{array} \tag{12}$$

in the variables $X \in \mathbf{S}_n$, where $B_i = a_i a_i^T - \rho \mathbf{I}$, or also

$$\begin{array}{ll} \psi(\rho) = \textit{max.} & \sum_{i=1}^n \mathbf{Tr}(P_i B_i) \\ \textit{s.t.} & \mathbf{Tr}(X) = 1, \ X \succeq 0, \ X \succeq P_i \succeq 0, \end{array} \tag{13}$$

which is a semidefinite program in the variables $X \in \mathbf{S}_n, \ P_i \in \mathbf{S}_n$.

Proof. We let $X^{1/2}$ be the positive square root (i.e. with nonnegative eigenvalues) of a symmetric positive semi-definite matrix X. In particular, if $X = xx^T$ with $\|x\| = 1$, then $X^{1/2} = X = xx^T$, and for all $\beta \in \mathbf{R}$, βxx^T has one eigenvalue equal to β and $n-1$ equal to 0, which implies $\mathbf{Tr}(\beta xx^T)_+ = \beta_+$. We thus get

$$\begin{aligned} (a_i^T X a_i - \rho)_+ &= \mathbf{Tr}((a_i^T x x^T a_i - \rho) x x^T)_+ \\ &= \mathbf{Tr}(x(x^T a_i a_i^T x - \rho) x^T)_+ \\ &= \mathbf{Tr}(X^{1/2} a_i a_i^T X^{1/2} - \rho X)_+ = \mathbf{Tr}(X^{1/2}(a_i a_i^T - \rho \mathbf{I}) X^{1/2})_+. \end{aligned}$$

For any symmetric matrix B, the function $X \mapsto \mathbf{Tr}(X^{1/2} B X^{1/2})_+$ is concave on the set of symmetric positive semidefinite matrices, because we can write it as

$$\begin{aligned} \mathbf{Tr}(X^{1/2} B X^{1/2})_+ &= \max_{\{0 \preceq P \preceq X\}} \mathbf{Tr}(PB) \\ &= \min_{\{Y \succeq B, \ Y \succeq 0\}} \mathbf{Tr}(YX), \end{aligned}$$

where this last expression is a concave function of X as a pointwise minimum of affine functions. We can now relax the original problem into a convex optimization problem by simply dropping the rank constraint, to get

$$\begin{array}{ll}\psi(\rho) \equiv \max. & \sum_{i=1}^n \mathbf{Tr}(X^{1/2}a_ia_i^TX^{1/2}-\rho X)_+\\ \text{s.t.} & \mathbf{Tr}(X)=1,\ X\succeq 0,\end{array}$$

which is a convex program in $X \in \mathbf{S}_n$. Note that because B_i has at most one nonnegative eigenvalue, we can replace $\mathbf{Tr}(X^{1/2}a_ia_i^TX^{1/2}-\rho X)_+$ by $\lambda_{\max}(X^{1/2}a_ia_i^TX^{1/2}-\rho X)_+$ in the above program. Using the representation of $\mathbf{Tr}(X^{1/2}BX^{1/2})_+$ detailed above, problem (12) can be written as a semidefinite program

$$\begin{array}{ll}\psi(\rho) = \max. & \sum_{i=1}^n \mathbf{Tr}(P_iB_i)\\ \text{s.t.} & \mathbf{Tr}(X)=1,\ X\succeq 0,\ X\succeq P_i\succeq 0,\end{array}$$

in the variables $X \in \mathbf{S}_n,\ P_i \in \mathbf{S}_n$, which is the desired result.

Note that we always have $\psi(\rho) \geq \phi(\rho)$ and when the solution to the above semidefinite program has rank one, $\psi(\rho) = \phi(\rho)$ and the semidefinite relaxation (13) is *tight*. This simple fact allows us to derive sufficient global optimality conditions for the original sparse PCA problem. We recall in particular the following result from [dBEG08] which provides sufficient conditions for a particular nonzero coefficient pattern I to be globally optimal. The optimal solution x to (2) is then found by solving an eigenvalue problem on the principal submatrix of Σ with support I.

Proposition 2. *Let $A \in \mathbf{R}^{n\times n}$, $\rho \geq 0$, $\Sigma = A^TA$ with $a_1,\ldots,a_n \in \mathbf{R}^n$ the columns of A. Given a sparsity pattern I, setting x to be the largest eigenvector of $\sum_{i\in I} a_ia_i^T$, if there is a $\rho^* \geq 0$ such that the following conditions hold*

$$\max_{i\in I^c}(a_i^Tx)^2 < \rho^* < \min_{i\in I}(a_i^Tx)^2 \quad \textit{and} \quad \lambda_{\max}\left(\sum_{i=1}^n Y_i\right) \leq \sum_{i\in I}((a_i^Tx)^2-\rho^*),$$

with the dual variables Y_i defined as

$$Y_i = \max\left\{0, \rho\frac{(a_i^Ta_i-\rho)}{(\rho-(a_i^Tx)^2)}\right\}\frac{(\mathbf{I}-xx^T)a_ia_i^T(\mathbf{I}-xx^T)}{\|(\mathbf{I}-xx^T)a_i\|^2}, \quad \textit{when } i \in I^c,$$

and

$$Y_i = \frac{B_ixx^TB_i}{x^TB_ix}, \quad \textit{when } i \in I,$$

then the sparsity pattern I is globally optimal for the sparse PCA problem (2) with $\rho = \rho^*$ and we can form an optimal solution z by solving the maximum eigenvalue problem

$$z = \underset{\{z_{I^c}=0,\ \|z\|=1\}}{\operatorname{argmax}} \quad z^T\Sigma z.$$

This result also provides tractable *lower bounds* on the optimal value of (2) whenever the solution is not optimal.

3 Algorithms

In this section, we describe algorithms for solving the semidefinite relaxations detailed above. We also describe greedy methods to improve the quality of these solutions.

3.1 First-order methods

Again, given a covariance matrix $\Sigma \in \mathbf{S}_n$, the DSPCA code solves a penalized formulation of problem (5), written as

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(\Sigma X) - \rho \mathbf{1}^T |X| \mathbf{1} \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & X \succeq 0, \end{array} \tag{14}$$

in the variable $X \in \mathbf{S}_n$. The dual of this program can be written as

$$\begin{array}{ll} \text{minimize} & f(U) = \lambda^{\max}(\Sigma + U) \\ \text{subject to} & |U_{ij}| \leq \rho. \end{array} \tag{15}$$

in the variable $U \in \mathbf{S}^n$. The algorithm in [dEGJL07, Nes07] regularizes the objective $f(U)$ in (15), replacing it by the smooth (i.e. with Lipschitz continuous gradient) uniform approximation

$$f_\mu(U) = \mu \log \left(\mathbf{Tr} \exp((\Sigma + U)/\mu) \right) - \mu \log n.$$

Following [Nes83], solving the smooth problem

$$\min_{U \in \mathcal{Q}} f_\mu(U)$$

where $\mathcal{Q} = \{U \in \mathbf{S}^n, |U_{ij}| \leq \rho\}$, with $\mu = \epsilon/2\log(n)$ then produces an ϵ-approximate solution to (14). The key difference between the minimization scheme developed in [Nes83] and classical gradient minimization methods is that it is not a descent method but achieves a complexity of $O(L/N^2)$ instead of $O(1/N)$ for gradient descent, where N is the number of iterations and L the Lipschitz constant of the gradient. Furthermore, this convergence rate is provably optimal for this particular class of convex minimization problems (see [Nes03, Th. 2.1.13]). Thus, by sacrificing the (local) properties of descent directions, we improve the (global) complexity estimate by an order of magnitude. For our problem here, once the regularization parameter μ is set, the algorithm is detailed as Algorithm 1.

The algorithm has four main steps. Step one computes the (smooth) function value and gradient. The second step computes the *gradient mapping*, which matches the gradient step for unconstrained problems (see [Nes03, p.86]). Step three and four update an *estimate sequence* see ([Nes03, p.72]) of f_μ whose minimum can be computed explicitly and gives an increasingly tight upper bound on the minimum of f_μ. We now present these steps in detail for our problem (we write U for U_i and X for X_i).

Algorithm 1 First-Order Algorithm.

```
Input: The covariance $\Sigma \in \mathbf{R}^{n\times n}$, and a parameter $\rho > 0$ controlling sparsity.
1: for $i = 1$ to $N$ do
2:     Compute $f_\mu(U_i)$ and $\nabla f_\mu(U_i)$
3:     Find $Y_i = \arg\min_{Y\in\mathcal{Q}} \langle \nabla f_\mu(U_i), Y\rangle + \frac{1}{2}L\|U_i - Y\|_F^2$
4:     Find $W_i = \arg\min_{W\in\mathcal{Q}} \left\{ \frac{L\|W\|_F^2}{2} + \sum_{j=0}^{N} \frac{j+1}{2}(f_\mu(U_j) + \langle \nabla f_\mu(U_j), W - U_j\rangle)\right\}$
5:     Set $U_{i+1} = \frac{2}{i+3}W_i + \frac{i+1}{i+3}Y_i$
6: end for
Output: A matrix $U \in \mathbf{S}_n$.
```

Step 1.

The most expensive step in the algorithm is the first, the computation of f_μ and its gradient. The function value can be reliably computed as

$$f_\mu(U) = d_{\max} + \mu \log\left(\sum_{i=1}^{n} \exp(\frac{d_i - d_{\max}}{\mu})\right) - \mu \log n.$$

where d_i are the eigenvalues of $\Sigma + U$. The gradient $\nabla f_\mu(U)$ can be computed explicitly as

$$\nabla f_\mu(U) := \exp\left((\Sigma + U)/\mu\right) / \mathbf{Tr}\left(\exp\left((\Sigma + U)/\mu\right)\right).$$

which means computing the same matrix exponential.

Step 2.

This step involves a problem of the form

$$\arg\min_{Y\in\mathcal{Q}} \langle \nabla f_\mu(U), Y\rangle + \frac{1}{2}L\|U - Y\|_F^2,$$

where U is given. The above problem can be reduced to a Euclidean projection

$$\arg\min_{\|Y\|_\infty \le 1} \|Y - V\|_F, \tag{16}$$

where $V = U - L^{-1}\nabla f_\mu(U)$ is given. The solution is given by

$$Y_{ij} = \mathbf{sgn}(V_{ij}) \min(|V_{ij}|, 1), \quad i, j = 1, \dots, n.$$

Step 3.

The third step involves solving a Euclidean projection problem similar to (16), with the solution V defined by

$$V = -\frac{1}{L}\sum_{i=0}^{k} \frac{i+1}{2}\nabla f_\mu(U_i).$$

Stopping criterion

We can stop the algorithm when the duality gap is smaller than ϵ:

$$\mathrm{gap}_k = \lambda_{\max}(\Sigma + U_k) - \mathbf{Tr}\, \Sigma X_i + \mathbf{1}^T |X_i| \mathbf{1} \leq \epsilon,$$

where $X_k = \nabla f_\mu(U)$ is our current estimate of the dual variable. The above gap is necessarily non-negative, since both X_i and U_i are feasible for the primal and dual problem, respectively. This is checked periodically, for example every 100 iterations.

Complexity

Overall the algorithm requires

$$O\left(\rho \frac{n\sqrt{\log n}}{\epsilon}\right) \tag{17}$$

iterations [dEGJL07, Nes07]. The main step at each iteration is computing the matrix exponential $\exp((\Sigma + U)/\mu)$ (see [MVL03] for a comprehensive survey) at a cost of $O(n^3)$ flops.

3.2 Greedy methods

We can also find good solution to problem (2), or improve existing solutions, using greedy methods. We first present very simple preprocessing solutions with complexity $O(n \log n)$ and $O(n^2)$. We then recall a simple greedy algorithm with complexity $O(n^4)$. Finally, our first contribution in this section is to derive an approximate greedy algorithm that computes a full set of (approximate) solutions for problem (2), with complexity $O(n^3)$.

Sorting and thresholding

The simplest ranking algorithm is to sort the diagonal of the matrix Σ and rank the variables by variance. This works intuitively because the diagonal is a rough proxy for the eigenvalues: the Schur-Horn theorem states that the diagonal of a matrix majorizes its eigenvalues [HJ85]; sorting costs $O(n \log n)$. Another quick solution is to compute the leading eigenvector of Σ and form a sparse vector by thresholding to zero the coefficients whose magnitude is smaller than a certain level. This can be done with cost $O(n^2)$.

Full greedy solution

Following [MWA06b], starting from an initial solution of cardinality one at $\rho = \Sigma_{11}$, we can update an increasing sequence of index sets $I_k \subseteq [1, n]$,

Algorithm 2 Greedy Search Algorithm.

```
Input: $\Sigma \in \mathbf{R}^{n \times n}$
1: Preprocessing: sort variables by decreasing diagonal elements and permute elements of $\Sigma$ accordingly.
2: Compute the Cholesky decomposition $\Sigma = A^T A$.
3: Initialization: $I_1 = \{1\}$, $x_1 = a_1/\|a_1\|$.
4: for $i = 1$ to $k^{\text{target}}$ do
5:     Compute $i_k = \text{argmax}_{i \notin I_k} \lambda_{\max}\left(\sum_{j \in I_k \cup \{i\}} a_j a_j^T\right)$.
6:     Set $I_{k+1} = I_k \cup \{i_k\}$ and compute $x_{k+1}$ as the leading eigenvector of $\sum_{j \in I_{k+1}} a_j a_j^T$.
7: end for
Output: Sparsity patterns $I_k$.
```

scanning all the remaining variables to find the index with maximum variance contribution.

At every step, I_k represents the set of nonzero elements (or sparsity pattern) of the current point and we can define z_k as the solution to problem (2) given I_k, which is:

$$z_k = \underset{\{z_{I_k^c}=0,\ \|z\|=1\}}{\text{argmax}} \; z^T \Sigma z - \rho k,$$

which means that z_k is formed by padding zeros to the leading eigenvector of the submatrix Σ_{I_k,I_k}. Note that the entire algorithm can be written in terms of a factorization $\Sigma = A^T A$ of the matrix Σ, which means significant computational savings when Σ is given as a Gram matrix. The matrices Σ_{I_k,I_k} and $\sum_{i \in I_k} a_i a_i^T$ have the same eigenvalues and if z is an eigenvector of Σ_{I_k,I_k}, then $A_{I_k} z/\|A_{I_k} z\|$ is an eigenvector of $A_{I_k} A_{I_k}^T$.

Approximate greedy solution

Computing $n - k$ eigenvalues at each iteration is costly and we can use the fact that uu^T is a subgradient of $\lambda_{\max}$ at X if u is a leading eigenvector of X [BV04], to get:

$$\lambda_{\max}\left(\sum_{j \in I_k \cup \{i\}} a_j a_j^T\right) \geq \lambda_{\max}\left(\sum_{j \in I_k} a_j a_j^T\right) + (x_k^T a_i)^2, \tag{18}$$

which means that the variance is increasing by at least $(x_k^T a_i)^2$ when variable i is added to I_k. This provides a lower bound on the objective which does not require finding $n - k$ eigenvalues at each iteration. We then derive the following algorithm.

Again, at every step, I_k represents the set of nonzero elements (or sparsity pattern) of the current point and we can define z_k as the solution to problem (2) given I_k, which is:

Algorithm 3 Approximate Greedy Search Algorithm.

```
Input: $\Sigma \in \mathbf{R}^{n \times n}$
1: Preprocessing: sort variables by decreasing diagonal elements and permute elements of $\Sigma$ accordingly.
2: Compute the Cholesky decomposition $\Sigma = A^T A$.
3: Initialization: $I_1 = \{1\}$, $x_1 = a_1/\|a_1\|$.
4: for $i = 1$ to $k^{\text{target}}$ do
5:     Compute $i_k = \operatorname{argmax}_{i \notin I_k} (x_k^T a_i)^2$.
6:     Set $I_{k+1} = I_k \cup \{i_k\}$ and compute $x_{k+1}$ as the leading eigenvector of $\sum_{j \in I_{k+1}} a_j a_j^T$.
7: end for
Output: Sparsity patterns $I_k$.
```

$$z_k = \underset{\{z_{I_k^c}=0,\ \|z\|=1\}}{\operatorname{argmax}} \; z^T \Sigma z - \rho k,$$

which means that z_k is formed by padding zeros to the leading eigenvector of the submatrix Σ_{I_k,I_k}. Better points can be found by testing the variables corresponding to the p largest values of $(x_k^T a_i)^2$ instead of picking only the best one.

Computational complexity

The complexity of computing a greedy regularization path using the classic greedy algorithm in 3.2 is $O(n^4)$: at each step k, it computes $(n-k)$ maximum eigenvalue of matrices with size k. The approximate algorithm in 3.2 computes a full path in $O(n^3)$: the first Cholesky decomposition is $O(n^3)$, while the complexity of the k-th iteration is $O(k^2)$ for the maximum eigenvalue problem and $O(n^2)$ for computing all products $(x^T a_j)$. Also, when the matrix Σ is directly given as a Gram matrix $A^T A$ with $A \in \mathbf{R}^{q \times n}$ with $q < n$, it is advantageous to use A directly as the square root of Σ and the total complexity of getting the path up to cardinality p is then reduced to $O(p^3 + p^2 n)$ (which is $O(p^3)$ for the eigenvalue problems and $O(p^2 n)$ for computing the vector products).

4 Applications

In this section, we illustrate the sparse PCA approach in applications: in news (text), finance and voting data from the US Senate.

4.1 News data

Our first dataset is a small version of the "20-newsgroups" data[4]. The data records binary occurrences of 100 specific words across 16242 postings, where the postings have been tagged by the highest level domain in Usenet.

[4] available from `http://cs.nyu.edu/~roweis/data.html`.

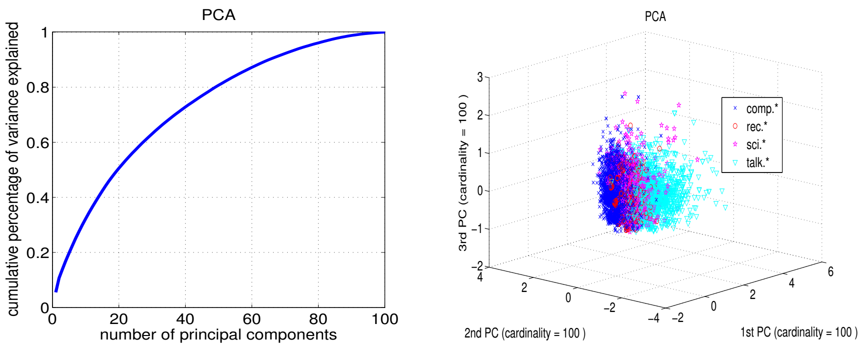

Fig. 1: PCA with 20-Newsgroups data. *Left:* Explained variance vs. number of PCs. *Right:* 3D visualization via PCA.

Each posting is viewed as one point in a 100-dimensional space. We begin with a standard PCA on the data. Fig. 1 (left) shows the cumulative percentage of variance explained as we increase the number of principal components. The slow increase means that the data does not lie within a subspace of significantly low dimension. We can anyway proceed to visualize the data: Fig. 1 (right) is the result obtained by projecting it on a subspace of dimension 3, chosen by selecting the eigenvectors corresponding to the three largest eigenvalues of the covariance matrix. Since these 3 vectors are dense, the axes in Fig. 1 (right) do not have a clear interpretation.

With sparse PCA, we hope to find a set of corresponding sparse principal components, which still help with visualization nearly as well as PCA does, and yet reveal some interesting structure. To achieve this, we have run the first-order algorithm of Section 3.1 (referred to as "DSPCA" hereafter) on the data with a range of values for the penalty parameter ρ. We obtained a plot of the variance explained by the first sparse principal component (PC), as a function of its cardinality (Fig. 2). We then selected a cardinality that can explain at least 90% of the variance explained by the the first principal component obtained from PCA. Then we have deflated the covariance matrix by taking out the part due to the first sparse PC, and then repeated the above procedure to obtain the second sparse PC. In the same way, we have solved for the third sparse PC. Fig. 2 also shows the projection of the data on the 3-dimensional subspace that is spanned by the three sparse PCs obtained above.

We first note that only a small number of words, out of the total of 100 words that can appear in each sparse PC, can explain more than 90% of variance explained by the corresponding PC. Specifically, we obtain 30 words for the first PC, 26 for the second, and 10 for the third. The lists of words associated with each sparse PCs is given in Table 1, and reveals some structure

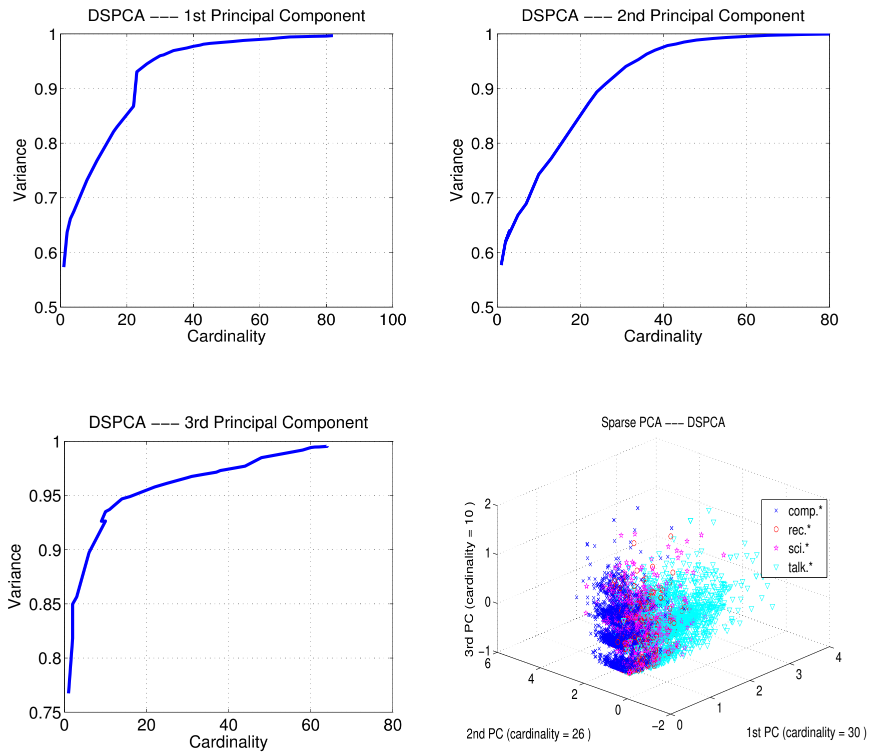

Fig. 2: Sparse PCA on the 20Newsgroups data set. First three principal components and 3D visualization. The first three principal components have cardinalities 26, 30 and 10 respectively.

about each one of the sparse PCs. That is, the 30 words associated with the first sparse PC are almost all about politics and religion, the 26 words in the second sparse PC are all computer-related, and the majority of the 10 words in the third sparse PC concerns science. Hence, applying sparse PCA to this data set allows to discover structure that is otherwise hidden in the standard PCA, for example that the first principal component is mainly related to politics and religion.

We also run the thresholded PCA and DSPCA algorithms over the collection of 1,288 news articles published by the *New York Times*'s International section mentioning the word "China." We tokenize the articles by unigrams, remove no stop words, and perform no stemming. The data encodes the binary $\{0, 1\}$ values (corresponding to appearance/non-appearance) of 86500 tokens.

Figure 3 shows the percentage of explained variance as a function of cardinality. Here we see DSPCA does outperform Thresholded PCA, though not by a big margin. Although we do not have ground truth, Table 2 and Table 3

1st PC (30 words)	2nd PC (26 words)	3rd PC (10 words)
fact	help	problem
question	problem	university
world	system	email
course	email	state
case	windows	research
problem	program	science
god	computer	phone
government	software	world
human	university	fact
state	version	question
number	files	
christian	drive	
evidence	data	
law	card	
power	dos	
religion	god	
children	disk	
jesus	pc	
system	graphics	
rights	ftp	
war	memory	
jews	christian	
help	phone	
bible	video	
earth	fact	
science	display	
research		
israel		
president		
gun		

Table 1: Words associated with the first three sparse PCs.

contains words selected by two algorithms respectively as we increase cardinality. Words selected by DSPCA appear much more meaningful than those chosen by thresholded PCA at the same cardinality.

4.2 Senate voting data

In this section, we analyze the voting records of the 109th US Senate (2000-2004) There were 101 senators (one extra Senator is due to a replacement during the term) and 48 bills involved. To simplify, the votes are divided into yes (coded as 1) or no (coded as -1), and other votes are coded as 0.

Each senator's voting record can be viewed as a point in a 48-dimensional space. By applying PCA, and projecting each senator's voting record onto a two-dimensional subspace of maximum variance, we can see that senators

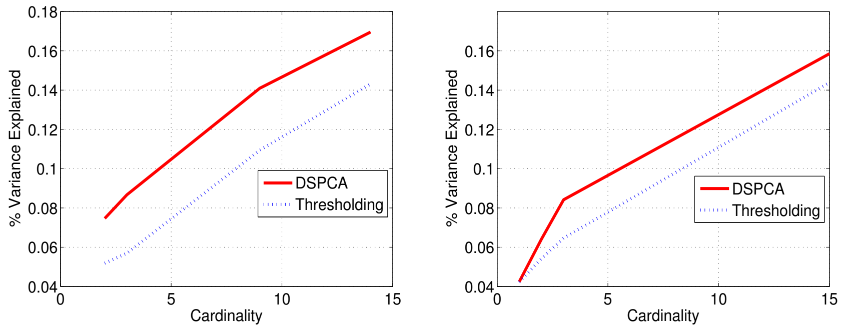

Fig. 3: Sparse PCA on 1,288 *New York Times* articles mentioning the word "China".

$k = 2$	$k = 3$	$k = 9$	$k = 14$
`united`	`american`	`washington`	`international`
`states`	`united`	`american`	`would`
	`states`	`administration`	`will`
		`united`	`washington`
		`states`	`american`
		`president`	`administration`
		`obama`	`united`
		`countries`	`states`
		`nations`	`president`
			`obama`
			`counties`
			`nations`
			`policy`
			`nuclear`

Table 2: 1st PC from DSPCA on 1,288 *New York Times* articles mentioning the word "China" for various values of the eigenvector cardinality k.

are almost perfectly separated by partisanship (Fig. 4). However, since the principal components involve all the bills, it is hard to tell which bills are most responsible for the explained variance. By applying Sparse PCA to the voting record, we aim to find a few bills that not only divide the senators according to partisanship, but also reveal which topics are most controversial within the Republican and Democratic parties. Fig. 4 (right) shows the senators' voting records, projected onto the first two sparse principal components. We note that in the two-dimensional space senators are still divided by partisanship. In fact, many republican senators perfectly coincide with each other and so are democratic senators. In contrast to Fig. 4 (left), the cardinalities associated

$k = 2$	$k = 3$	$k = 9$	$k = 14$
`even`	`even`	`even`	`would`
`like`	`like`	`we`	`new`
	`states`	`like`	`even`
		`now`	`we`
		`this`	`like`
		`will`	`now`
		`united`	`this`
		`states`	`will`
		`if`	`united`
			`states`
			`world`
			`so`
			`some`
			`if`

Table 3: 1st PC from Thresholded PCA on 1,288 *New York Times* articles mentioning the word "China" for various values of the eigenvector cardinality k.

with the first and second sparse principal components are 5 and 2 respectively, which makes it possible to interpret the coordinates.

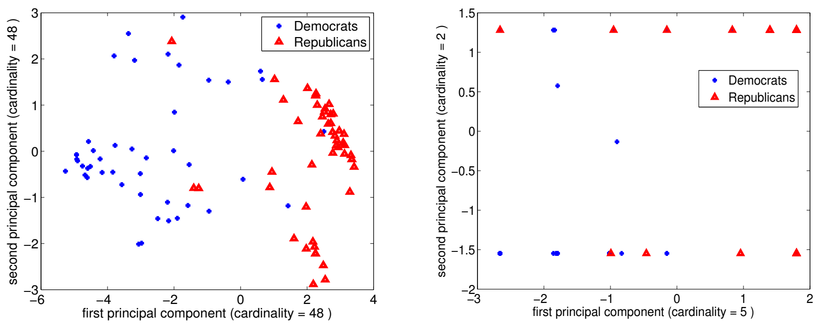

Fig. 4: 109th Senate's voting record projected onto the top 2 principal components.

Let us examine the bills appearing in the first two sparse principal components. For the first sparse PC, the corresponding bills' brief description is as follows:

- S. 1932, As Amended; Deficit Reduction Act of 2005.
- S. Con. Res. 83; An original concurrent resolution setting forth the congressional budget for the United States Government for fiscal year 2007

and including the appropriate budgetary levels for fiscal years 2006 and 2008 through 2011.
- S. 3930, As Amended; Military Commissions Act of 2006.
- S. 403, As Amended; Child Interstate Abortion Notification Act.
- Passage of S. 397, As Amended; Protection of Lawful Commerce in Arms Act.

The brief description for the two bills in the second sparse principal component are:

- H. R. 3045; Dominican Republic-Central America-United States Free Trade Agreement Implementation Act.
- S. 1307; Dominican Republic-Central America-United States Free Trade Agreement Implementation Act.

A glance at these bills tells us that the major controversial issues between Democrats and Republicans are topics such as "abortion", "military", "budget", and "free trade".

Fig 5 plots the variance explained by the first sparse principal component divided by that explained by the first PC, as a function of the cardinality of the sparse PC. Fig. 5 also shows how the cardinality of the first sparse PC varies as the penalty parameter ρ is changed in the DSPCA code.We can see that when 19 out of 48 variables (bills) are used, sparse PCA almost achieves the same statistical fidelity as standard PCA does.

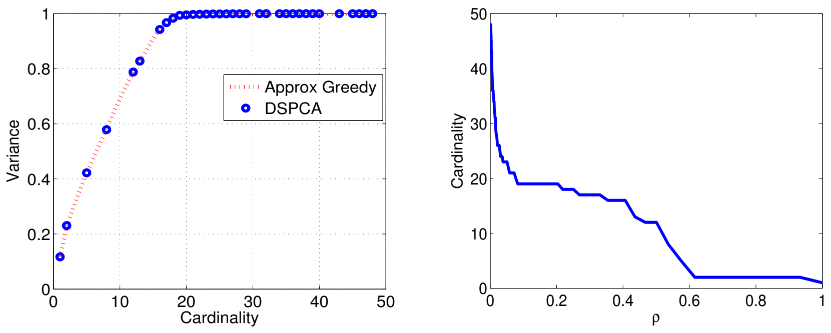

Fig. 5: *Left:* Explained variance as a function of cardinality. *Right:* Cardinality as a function of penalty parameter ρ.

4.3 Stock market data

In this section, we investigate the historical prices of S&P500 stocks over 5 years, from June 1st, 2005, through June 1st, 2010. By taking out the stocks with less than 5 years of history, we end up with 472 stocks, each having daily closing prices over 1259 trading days. The prices are first adjusted for dividends and splits and then used to calculate daily log returns. Each day's return can be represented as a point in $\mathbf{R}^{472}$.

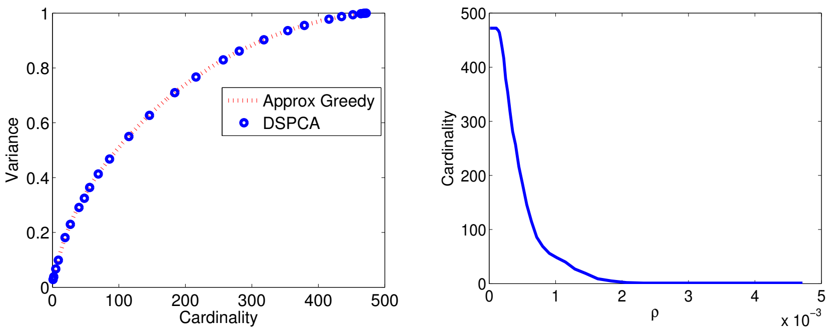

Fig. 6: *Left:* Explained variance as a function of cardinality. *Right:* Cardinality as a function of penalty parameter ρ.

Fig. 6 shows the explained variance as a function of 1st PC's cardinality. It seems hard to say that the 1st PC is sparse, since there is no natural "kink" in that curve. That is, we need almost 300 out of the total 472 stocks to explain at least 90% of the variance explained by the 1st PC from PCA. However, when we inspect the sparse PCs with increasing cardinalities, we note that initially only stocks from the "Financials" sector come to play and later until, at cardinality 32, do we see companies from other sectors appearing in the 1st sparse PC. So we take the first sparse PC with cardinality equal to 32. Then we solve for the 2nd sparse PC, and using the same guideline to arrive at a cardinality of 26.

Figure 7 show the stock returns projected onto the 2-dimensional subspaces spanned by the top 2 PCs and top 2 sparse PCs, respectively. Comparing these two plots, we observe two interesting phenomena:

- Although the top 2 principal components from PCA explain more variance (as seen from the larger range of the axes in the left over the right panel), the two sparse principal components from DSPCA involve only 58 out of 472 stocks (32 on the first PC and another distinct 26 on the second). Furthermore, 31 of the 32 stocks in the first sparse PC are all from the sector "Financials", and that almost all 26 stocks in the second sparse PC come from "Energy" and "Materials" except 2 from "Financials" and 1 from "Information Technology". Considering that there are 10 sectors in total, this is quite interesting as Sparse PCA is able to identify the right

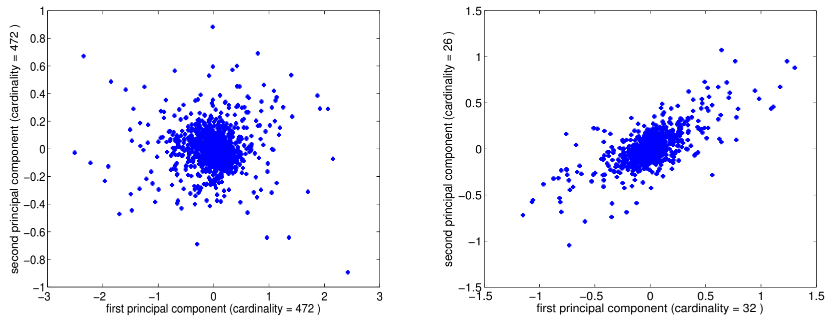

Fig. 7: S&P500 daily returns projected onto the top 2 principal components. For PCA (left) and sparse PCA (right).

groups (industry factors) that explains most of the variance. Our data covers June 2005 through June 2010 where a severe financial crisis took place, and the key role of the Financial sector is revealed purely through our sparse PCA analysis.

- In Fig. 6, the projected data appears symmetrically distributed around its center. In contrast, In Fig. 7 (right), we observe a definite orientation. Since the horizontal axis (first PC) corresponds to "Financials" and the vertical one to "Energy" and "Materials", the sparse PCA analysis tells us that these two sectors are positively correlated.

4.4 Random matrices

Sparse eigenvalues of random matrices play a central role in characterizing the performance of ℓ_1 decoders in compressed sensing applications. Testing the Restricted Isometry Property (RIP) in [CT05] amounts to bounding the maximum and minimum eigenvalues of a Gram matrix. Here, we compute the upper and lower bounds on sparse eigenvalues produced using various algorithms. We pick the data matrix to be small enough so that computing sparse eigenvalues by exhaustive search is numerically feasible. In Figure 8, we plot the maximum sparse eigenvalue versus cardinality, obtained using exhaustive search (solid line), the approximate greedy (dotted line) and fully greedy (dashed line) algorithms. We also plot the upper bounds obtained by minimizing the gap of a rank one solution (squares), by solving the semidefinite relaxation in §2.2 explicitly (stars) and by solving the DSPCA dual (diamonds). On the left, we use a matrix $\Sigma = F^T F$ with F Gaussian. On the right, $\Sigma = uu^T/\|u\|^2 + 2V^T V$, where $u_i = 1/i,\ i = 1, \ldots, n$ and V is matrix with coefficients uniformly distributed in $[0, 1]$. Almost all algorithms are provably optimal in the noisy rank one case (as well as in many example arising from "natural data"), while Gaussian random matrices are harder. Note however, that the duality gap between the semidefinite relaxations and the optimal solution is very small in both cases, while our bounds based on greedy solutions are not as good. Overall, while all algorithms seem to behave similarly on "natural" or easy data sets, only numerically expensive relaxations produce good bounds on the random matrices used in compressed sensing applications.

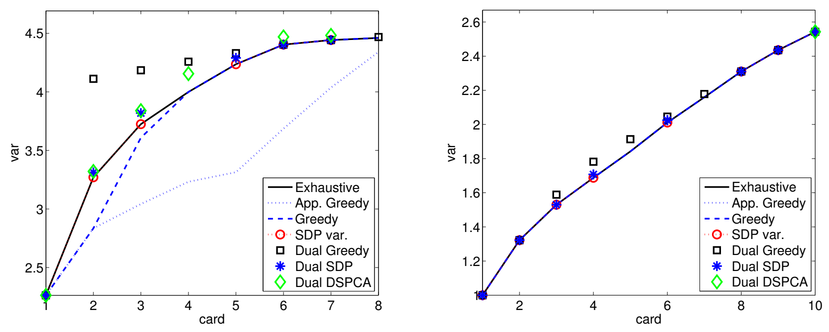

Fig. 8: Upper and lower bound on sparse maximum eigenvalues. We plot the maximum sparse eigenvalue versus cardinality, obtained using exhaustive search (solid line), the approximate greedy (dotted line) and fully greedy (dashed line) algorithms. We also plot the upper bounds obtained by minimizing the gap of a rank one solution (squares), by solving the ℓ_0 semidefinite relaxation explicitly (stars) and by solving the DSPCA dual ℓ_1 relaxation (diamonds). *Left:* On a matrix $F^T F$ with F Gaussian. *Right:* On a sparse rank one plus noise matrix.

4.5 Statistical consistency vs. computational complexity

As we hinted above, very simple methods such as thresholding or greedy algorithms often perform well enough on simple data sets, while obtaining good statistical fidelity on more complex (or random) data sets requires more complex algorithms. This is perfectly illustrated by the results in [AW09] on a spiked covariance model. To summarize these results, suppose that the sample covariance matrix $\hat{\Sigma} \in \mathbf{S}_n$ is a noisy estimate of the true population covariance $\Sigma \in \mathbf{S}_n$ with $\hat{\Sigma} = \Sigma + \Delta$ where Δ is a noise matrix, suppose also that the leading eigenvector of the true covariance is sparse with cardinality k. Under some assumptions on the noise component Δ, [AW09] show that when the ambient dimension n, the number of observations m and the number k of nonzero components in the leading eigenvector all scale to infinity, and when the ratio

$$\theta_{\text{thres}} = \frac{m}{k^2 \log(n-k)}$$

is above some critical value, then simply thresholding the diagonal of the sample covariance matrix will recover the exact support of the leading eigenvector of Σ with probability tending to one. On the other hand, simple thresholding fails with probability one when this ratio is below a certain value. Furthermore, [AW09] show that when

$$\theta_{\text{sdp}} = \frac{m}{k \log(n-k)}$$

is above some critical value, the solution of the semidefinite relaxation in Section 2.1 (if tight) will recover the exact support of the leading eigenvector of Σ with probability tending to one. On the other hand, the semidefinite relaxation fails with probability one when this ratio is below a certain value. They also show that the semidefinite programing relaxation in Section 2.1 is statistically optimal, meaning that no other method (even combinatorial ones) can recover the true support using fewer samples (up to a constant factor). This result clearly illustrates a tradeoff between statistical fidelity on one side and computational complexity on the other. In the spiked model, the semidefinite relaxation requires $O(1/k)$ fewer samples than simply thresholding the diagonal to recover the true support of the leading eigenvector of Σ, but its complexity is much higher than that of the thresholding strategy.

We can further illustrate this behavior on a simple numerical example. Suppose we are given a sample covariance $\hat{\Sigma} \in \mathbf{S}_n$ coming from a "spiked" model of covariance similar to that in [AW09], with

$$\hat{\Sigma} = uu^T + VV^T/\sqrt{m}$$

where $u \in \mathbf{R}^n$ is the true sparse leading eigenvector, with $\mathbf{Card}(u) = k$, $V \in \mathbf{R}^{n \times m}$ is a noise matrix with $V_{ij} \sim \mathcal{N}(0,1)$ and m is the number of observations. We compare the performance of the simple thresholding method (on the leading eigenvector of regular PCA here) with that of the semidefinite relaxation when recovering the support of u for various values of the number of samples. Our point here is that, while variance versus cardinality is a direct way of comparing the performance of sparse PCA algorithms, accurate recovery of the support is often a far more important objective. Many methods produce similar variance levels given a limited budget of nonzero components, but their performance in recovering the true support is often markedly different.

In Figure 9 on the left we compare ROC curves when recovering the support of u in the spiked model above using thresholded PCA, the approximate and full greedy algorithms in [dBEG08] and semidefinite relaxation (DSPCA). On the right, we plot Area Under ROC as the number of samples increase. As expected, we observe that the semidefinite relaxation performs much better when only a limited number of observations are available (m small).

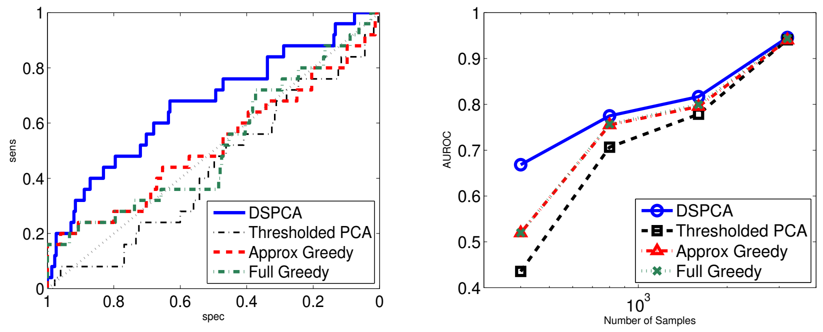

Fig. 9: *Left:* ROC curves when recovering the support of u in the spiked model using thresholding, approximate and exact greedy algorithms and the semidefinite relaxation (DSPCA) in Section 2.1 in the spiked model when $n = 250$, $m = 400$ and $k = 25$. *Right:* Area Under ROC (AUROC) versus number of samples m.

5 Conclusion

We have reviewed here several techniques for approximating the solution to the *single factor* sparse PCA problem. While the algorithms presented here perform quite well, several key questions remain open at this point.

First, outside of the (locally) convergent algorithm in [JNRS08], very few methods handle the problem of simultaneously finding several leading sparse principal components.

Also, as the examples of Section 4 illustrate, most methods (even extremely simple ones) perform well enough on easy, “natural” data sets while only the most expensive semidefinite relaxations seem to produce good bounds on the random matrices used in compressed sensing applications, or when only a few samples are available for example. Characterizing what makes “natural” data sets easier than random ones remains an open problem at this point. It is also not clear yet how to extend the statistical optimality statements of [AW09] to broader (e.g. deterministic) classes of matrices.

Finally, the question of approximation bounds à la MAXCUT for the relaxations detailed here is largely open. Basic performance bounds are discussed in [dEG08, BAd10] but they can certainly be tightened.

Acknowledgments

The authors gratefully acknowledge partial support from NSF grants SES-0835550 (CDI), CMMI-0844795 (CAREER), CMMI-0968842, a Peek junior faculty fellowship, a Howard B. Wentz Jr. award and a gift from Google.

References

[Ali95] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5:13–51, 1995.

[AW09] A.A. Amini and M. Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. *The Annals of Statistics*, 37(5B):2877–2921, 2009.

[BAd10] F. Bach, S. D. Ahipasaoglu, and A. d'Aspremont. Convex relaxations for subset selection. *working paper*, 2010.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[CJ95] J. Cadima and I. T. Jolliffe. Loadings and correlations in the interpretation of principal components. *Journal of Applied Statistics*, 22:203–214, 1995.

[CT05] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[dBEG08] A. d'Aspremont, F. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

[dEG08] A. d'Aspremont and L. El Ghaoui. Testing the nullspace property using semidefinite programming. *To appear in Mathematical Programming*, 2008.

[dEGJL07] 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.

[DT05] D. L. Donoho and J. Tanner. Sparse nonnegative solutions of underdetermined linear equations by linear programming. *Proc. of the National Academy of Sciences*, 102(27):9446–9451, 2005.

[FHB01] M. Fazel, H. Hindi, and S. Boyd. A rank minimization heuristic with application to minimum order system approximation. *Proceedings American Control Conference*, 6:4734–4739, 2001.

[GVL90] G.H. Golub and C.F. Van Loan. Matrix computation. *North Oxford Academic*, 1990.

[HJ85] R.A. Horn and C.R. Johnson. *Matrix Analysis.* Cambridge University Press, 1985.

[JNRS08] M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *arXiv:0811.4724*, 2008.

[Jol95] I. T. Jolliffe. Rotation of principal components: choice of normalization constraints. *Journal of Applied Statistics*, 22:29–35, 1995.

[JTU03] 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.

[Kai58] H.F. Kaiser. The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, 23(3):187–200, 1958.

[LO99] C. Lemaréchal and F. Oustry. Semidefinite relaxations and Lagrangian duality with application to combinatorial optimization. *INRIA, Rapport de recherche*, 3710, 1999.

[LS91] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

[Mac09] L. Mackey. Deflation methods for sparse pca. *Advances in Neural Information Processing Systems*, 21:1017–1024, 2009.

[MVL03] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[MWA06a] B. Moghaddam, Y. Weiss, and S. Avidan. Generalized spectral bounds for sparse LDA. In *International Conference on Machine Learning*, 2006.

[MWA06b] B. Moghaddam, Y. Weiss, and S. Avidan. Spectral bounds for sparse PCA: exact and greedy algorithms. *Advances in Neural Information Processing Systems*, 18, 2006.

[MWA07] B. Moghaddam, Y. Weiss, and S. Avidan. Fast Pixel/Part Selection with Sparse Eigenvectors. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[Nat95] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24(2):227–234, 1995.

[Nes83] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[Nes03] Y. Nesterov. *Introductory Lectures on Convex Optimization.* Springer, 2003.

[Nes07] Y. Nesterov. Smoothing technique and its applications in semidefinite optimization. *Mathematical Programming*, 110(2):245–259, 2007.

[NW54] JO Neuhaus and C. Wrigley. The quartimax method: an analytical approach to orthogonal simple structure. *British Journal of Statistical Psychology*, 7:81–91, 1954.

[Saa92] Y. Saad. *Numerical methods for large eigenvalue problems.* Manchester Univ Press, 1992.

[STL07] B.K. Sriperumbudur, D.A. Torres, and G.R.G. Lanckriet. Sparse eigen methods by DC programming. *Proceedings of the 24th international conference on Machine learning*, pages 831–838, 2007.

[Tib96] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal statistical society, series B*, 58(1):267–288, 1996.

[ZHT06] H. Zou, T. Hastie, and R. Tibshirani. Sparse Principal Component Analysis. *Journal of Computational & Graphical Statistics*, 15(2):265–286, 2006.

[ZZS02] Z. Zhang, H. Zha, and H. Simon. Low rank approximations with sparse factors I: basic algorithms and error analysis. *SIAM journal on matrix analysis and its applications*, 23(3):706–727, 2002.