LARGE-SCALE PCA WITH SPARSITY CONSTRAINTS

CLÉMENT J. PROBEL AND JOEL A. TROPP

Technical Report No. 2011-02
August 2011

APPLIED & COMPUTATIONAL MATHEMATICS
CALIFORNIA INSTITUTE OF TECHNOLOGY
mail code 9-94 · pasadena, ca 91125

Large-Scale PCA with Sparsity Constraints

Clément J. Probel
École des Mines
Paris, France
clement.probel@mines-paristech.fr

Joel A. Tropp
California Institute of Technology
Pasadena, CA 91125-5000
jtropp@cms.caltech.edu

Abstract

This paper describes a new thresholding technique for constructing sparse principal components. Large-scale implementation issues are addressed, and a mathematical analysis describes situations where the algorithm is effective. In experiments, this method compares favorably with more sophisticated algorithms.

1 Sparsity constraints in PCA

Principal component analysis (PCA) is a basic statistical method for decomposing data matrices. The idea is to extract a small number of meta-variables, called *principal components* or *factors*, that can be combined linearly to explain the responses of the subjects who have been surveyed. PCA is usually accomplished by means of a truncated singular value decomposition (SVD).

Researchers often criticize PCA because it yields factors that include *all* the measured variables [1]. In practice, it is desirable to produce *sparse factors*, which involve a small subset of variables. The plainest reason for preferring sparse factors is interpretability; it is difficult for the human mind to penetrate a vast conspiracy of interacting variables. In other problems, using a small number of variables may reduce monetary or time costs. For example, when modeling a financial market, we might wish to explain most of the volatility using a small collection of assets because trading charges scale with the diversity of a portfolio. The paper [2] describes clustering and feature selection applications; see [3, Sec. 1] for a variety of others.

Statistical theory provides a sound reason for enforcing sparsity. PCA is sensitive to measurement noise, so the estimated factors are not reliable when many measured variables participate. For modern data sets, in which the number of variables vastly exceeds the number of subjects, factors computed using standard PCA may be so noisy that they explain nothing [4, Thm. 1].

Sparsity-constrained principal component analysis (SC.PCA) attempts to address these concerns. Suppose that $\boldsymbol{X}$ is an $n \times p$ data matrix whose rows index subjects and whose columns index measured variables. The goal is to identify a subset of s variables that we can combine linearly into k factors to explain the energy in the data matrix. We insist that these *sparse factors* form an orthonormal family so each factor captures a different aspect of the data. Thus, we require $s \geq k$.

Core to most approaches to SC.PCA is a combinatorial optimization that looks for a single sparse factor [5, Sec. 2.1]. This formulation alloys the variational formula [6, Cor. III.1.2] of a top right singular vector with a sparsity constraint. Formally, we seek a p-dimensional vector $\boldsymbol{w}$ that solves

$$\max \|\boldsymbol{X}\boldsymbol{w}\|_2 \quad \text{subject to} \quad \|\boldsymbol{w}\|_2 = 1 \quad \text{and} \quad \|\boldsymbol{w}\|_0 \leq s. \tag{1.1}$$

The notation $\|\cdot\|_q$ refers to the usual ℓ_q norm, and we introduce the convention that $\|\cdot\|_0$ returns the number of nonzero *rows* in a vector or matrix.

The problem (1.1) is computationally hard for worst-case instances [5, Sec. 2.1]. Therefore, it is generally impossible to complete the optimization (1.1). Although the paper [7] offers techniques for obtaining certificates of optimality for (1.1), we typically must relax our expectations and find special cases where we can obtain reasonable sparse factors.

To construct a set of k sparse factors, most (but not all) earlier proposals rely on iterative deflation. This technique constructs a sparse factor by attempting to solve (1.1), removes the contribution of the computed factor from the data matrix, and then repeats the process. This approach tends to be costly because it requires repeated optimization of (1.1).

Remarks. Our description of SC.PCA assumes that the desired outcome is a set of sparse factors that involve as few total variables as possible. In contrast, many algorithms for SC.PCA allow each sparse factor to involve different variables. When the columns of $\boldsymbol{X}$ are centered, SC.PCA can be treated as a search for sparse factors that explain the maximum *variance* in the data. Thus, many papers perform SC.PCA on the (empirical) covariance matrix; others study the correlation matrix.

1.1 Proposal

Let us express the problem of computing multiple sparse factors in terms of a single mathematical program. This formulation combines the variational characterization of the top k right singular vectors [6, Ex. II.1.13] with a sparsity bound. Formally, we want a $p \times k$ matrix $\boldsymbol{W}$ that optimizes

$$\max \|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}} \quad \text{subject to} \quad \boldsymbol{W}^*\boldsymbol{W} = \mathbf{I}_k \quad \text{and} \quad \|\boldsymbol{W}\|_0 \leq s. \tag{1.2}$$

We have written $\|\cdot\|_{\mathrm{F}}$ for the Frobenius norm and $\mathbf{I}_k$ for the $k \times k$ identity matrix. The equality constraint requires the columns of $\boldsymbol{W}$ to form an orthonormal set. When $s = k$, this problem can be solved efficiently. Owing to the sparsity constraint, the optimization problem (1.2) is computationally hard when $s \gg k$, but we have found theoretical evidence (beyond the scope of this paper) that (1.2) with $k \gg 1$ should be *easier* to solve approximately than (1.1).

Section 2 proposes an algorithm, called SC.PCA by Joint Thresholding (SC.PCA.JT), for attempting the optimization (1.2). The formulation (1.2) and the algorithm rely on several ideas.

1. We can obtain better sparse factors if we construct them simultaneously, rather than attempting to produce them sequentially. This approach also requires less computation.
2. In practice, the right singular vectors of the matrix $\boldsymbol{X}$ contain a substantial amount of information about which variables are significant. In particular, we argue that for many types of data, the singular vectors of $\boldsymbol{X}$ exhibit joint decay.
3. On account of this decay, the identities of the key variables can be extracted from the right singular vectors of the data matrix by a joint thresholding operation.
4. To compute the singular vectors of a very large data matrix efficiently, we propose to use a new class of SVD algorithms based on randomized dimension reduction [8].

We argue that the SC.PCA.JT algorithm produces sparse factors that are comparable with or superior to the output of earlier methods. Even so, the computational costs of our method are usually lower, which makes the algorithm valuable for large-scale applications. We offer a mathematical analysis that identifies situations where the method is effective, and we present evidence that data matrices often meet these criteria. Finally, we summarize some preliminary numerical experiments with real data that provide an empirical demonstration of the efficacy of SC.PCA.JT.

2 Large-scale SC.PCA

Figure 1 presents a high-level description of the SC.PCA.JT algorithm for approaching (1.2). The algorithm selects variables by finding the largest rows of the matrix of k dominant singular vectors—which is a joint thresholding operation. The final collection of k sparse factors is obtained by computing the right singular vectors of the reduced data matrix. Although this procedure is simple and effective, it does not appear in the literature. We remark that SC.PCA.JT extends the standard simple thresholding method ($k = 1$), and it is inspired by the diagonal thresholding method [4]. Moghaddam et al. have emphasized the importance of Step 3 in their work [9]. Let us continue with a discussion of implementation issues and a mathematical analysis of the algorithm.

2.1 Implementation Issues

The most expensive step in the SC.PCA.JT algorithm is easily Step 1, which requires us to obtain k right singular vectors of the full data matrix. The approaches we consider for large data are

Input. An $n \times p$ matrix $\boldsymbol{X}$; sparsity level s; number k of sparse factors.
Output. A $p \times k$ matrix $\boldsymbol{W}$ whose columns are jointly s-sparse factors.

---

1. Construct a $p \times k$ matrix $\boldsymbol{V}$ that solves

$$\max_{\boldsymbol{V}} \|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}} \quad \text{subject to} \quad \boldsymbol{V}^*\boldsymbol{V} = \mathbf{I}_k.$$

2. Identify a set S that indexes the s largest rows of $\boldsymbol{V}$.

$$r_i = \|\boldsymbol{v}_{i,:}\|_2 \quad \text{and} \quad S \in \arg\max_{|I| \le s} \sum\nolimits_{i \in I} r_i.$$

3. Construct a $p \times k$ matrix $\boldsymbol{W}$ that solves

$$\max_{\boldsymbol{W}} \|\boldsymbol{X}\boldsymbol{P}_S\boldsymbol{W}\|_{\mathrm{F}} \quad \text{subject to} \quad \boldsymbol{W}^*\boldsymbol{W} = \mathbf{I}_k,$$

where $\boldsymbol{P}_S$ is the orthogonal projector onto the coordinates listed in S.

Figure 1: SC.PCA BY JOINT THRESHOLDING (SC.PCA.JT)

practical because they access the data only through matrix–vector or matrix–matrix multiplication. We mention several techniques and the situation where each is preferred.

Small data sets. For moderately sized data, classical dense methods [10, Sec. 8.6] are adequate (`svd` in Matlab). The cost is on the order of $\min\{n^2p, np^2\}$ arithmetic operations.

Sparse data. When the data matrix is sparse, Krylov subspace methods (`svds` in Matlab) are often effective [10, Sec. 9.4]. The nominal cost is on the order of $k \cdot \mathrm{nnz}(\boldsymbol{X}) + k^2p$ operations.

General data, rapid spectral decay. For a (dense) matrix with a rapidly decaying singular spectrum, we recommend randomized algorithms that incorporate a fast transform; see [8, Sec. 4.6] or [11]. The total cost of this approach is on the order of $np\log(k) + k^2p$ operations.

General data, slow spectral decay. For a matrix whose singular spectrum decays slowly or not at all, the most effective method is probably the randomized PCA algorithm [8, p. 9]; see also [12]. The cost is about $k \cdot \mathrm{nnz}(\boldsymbol{X}) + k^2p$ operations.

In Step 3, we need to calculate singular vectors of the reduced data matrix $\boldsymbol{X}\boldsymbol{P}_S$. Typically, the reduced matrix is relatively small, so it is appropriate to apply dense methods at a cost of s^2n. For very large problems, it may be beneficial to use one of the other methods described above. The remaining part of the computation, Step 2, is negligible in comparison with Steps 1 and 3. The easiest implementation just computes and sorts the row norms with about $np + p\log(p)$ operations.

2.2 Analysis of algorithm

The analysis of the SC.PCA.JT algorithm describes how its performance depends on properties of the data matrix $\boldsymbol{X}$. The proof is based on a simple numerical inequality [13, Lem. 7].

Lemma 2.1 *Consider a weakly decreasing sequence $\{a_i : i = 1, 2, 3, \dots\}$ of nonnegative numbers. For each positive integer s, we have the bound*

$$\Big[\sum\nolimits_{i>s} a_i^2\Big]^{1/2} \le \frac{1}{2\sqrt{s}} \sum\nolimits_{i \ge 1} a_i.$$

Proof. Since $\{a_i\}$ is weakly decreasing, we have the chain of inequalities

$$\Big[\sum\nolimits_{i>s} a_i^2\Big]^{1/2} \le \sqrt{a_s} \cdot \Big[\sum\nolimits_{i>s} a_i\Big]^{1/2} \le \frac{1}{\sqrt{s}} \cdot \Big[\sum\nolimits_{i \le s} a_i\Big]^{1/2} \Big[\sum\nolimits_{i>s} a_i\Big]^{1/2}.$$

Invoke the geometric mean–arithmetic mean inequality to combine the two sums. □

By applying Lemma 2.1 to (the decreasing rearrangement of) the row norms of the matrix $\boldsymbol{V}$ computed in Step 2 of the algorithm, we can bound the total energy that falls outside the largest rows.

Corollary 2.2 *Consider a $p \times k$ matrix $\boldsymbol{V}$. As in Step 2 of the SC.PCA.JT algorithm, compute the sequence $\{r_i : i = 1, 2, \dots, p\}$ of row norms of $\boldsymbol{V}$ and the set S of the largest s rows. Then*

$$\|\boldsymbol{P}_{S^c}\boldsymbol{V}\|_{\mathrm{F}} = \Big[\sum\nolimits_{i \notin S} r_i^2\Big]^{1/2} \leq \frac{1}{2\sqrt{s}} \sum\nolimits_{i \geq 1} r_i.$$

The symbol $\boldsymbol{P}_{S^c}$ denotes the orthogonal projector onto the coordinates not *listed in S.*

The following result contains *qualitative* information about the issues that influence the performance of the algorithm. We discuss the meaning below.

Theorem 2.3 (SC.PCA.JT Algorithm) *Consider an $n \times p$ data matrix $\boldsymbol{X}$. Define the quantity*

$$R = \sum\nolimits_{i \geq 1} r_i$$

where r_i are the row norms of $\boldsymbol{V}$ obtained in Step 2. Then the output $\boldsymbol{W}$ of the algorithm verifies

$$\|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}} \geq \left[1 - \frac{R}{2\sqrt{s}} \cdot \frac{\|\boldsymbol{X}\|}{\|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}}}\right] \cdot \|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}} \tag{2.1}$$

Proof. The nonzero rows of $\boldsymbol{W}$ are listed in the set S, so we have

$$\|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}} = \|\boldsymbol{X}\boldsymbol{P}_S\boldsymbol{W}\|_{\mathrm{F}} \geq \|\boldsymbol{X}\boldsymbol{P}_S\boldsymbol{V}\|_{\mathrm{F}},$$

where the inequality depends on the variational definition of $\boldsymbol{W}$ in Step 3 of the algorithm. To continue, introduce the decomposition $\boldsymbol{P}_S = \mathbf{I} - \boldsymbol{P}_{S^c}$, and apply the lower triangle inequality.

$$\|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}} \geq \|\boldsymbol{X}(\mathbf{I} - \boldsymbol{P}_{S^c})\boldsymbol{V}\|_{\mathrm{F}} \geq \|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}} - \|\boldsymbol{X}\boldsymbol{P}_{S^c}\boldsymbol{V}\|_{\mathrm{F}}.$$

Invoke the standard operator norm bound, and then refer to Corollary 2.2 to discover that

$$\|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}} \geq \|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}} - \|\boldsymbol{X}\| \cdot \|\boldsymbol{P}_{S^c}\boldsymbol{V}\|_{\mathrm{F}} \geq \|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}} - \frac{R}{2\sqrt{s}} \cdot \|\boldsymbol{X}\|.$$

Factor this expression to complete the proof. □

The left-hand side $\|\boldsymbol{X}\boldsymbol{W}\|_{\mathrm{F}}$ is the energy in the data that we can explain with the computed sparse factors. The theorem compares this quantity with $\|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}}$, the energy captured by the *best* set of k orthonormal factors. In particular, $\|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}}$ exceeds the optimal value of (1.2). The SC.PCA.JT algorithm misses some energy because of several phenomena.

1) Decay of singular vectors. The quantity R reflects how much the sorted row norms of $\boldsymbol{V}$ decay.

$$k \leq R \leq \sqrt{kp}.$$

The left-hand inequality is in force when exactly k rows of $\boldsymbol{V}$ are nonzero; the right-hand inequality applies if all p row norms are equal. Therefore, the SC.PCA.JT algorithm prefers when the energy in $\boldsymbol{V}$ is concentrated in relatively few rows.

2) Decay of singular values. The norm ratio reflects the decay of the first k singular values of $\boldsymbol{X}$.

$$\frac{1}{\sqrt{k}} \leq \frac{\|\boldsymbol{X}\|}{\|\boldsymbol{X}\boldsymbol{V}\|_{\mathrm{F}}} \leq 1.$$

The left-hand inequality holds if the largest k singular values of $\boldsymbol{X}$ are the same; the right-hand inequality holds if $\boldsymbol{X}$ has rank one. As a result, the SC.PCA.JT algorithm is most successful at computing a set of k sparse factors when each of the k dominant right singular vectors of $\boldsymbol{X}$ explains a lot of energy in the data.

3) Sparsity parameter. As the sparsity parameter s grows, the energy explained by the sparse factors increases rapidly at first but the marginal improvement declines. For sufficiently large s, the sparse factors explain essentially as much energy as the best *nonsparse* factors.

In summary, the ideal situation occurs when (i) the row norms of the matrix $\boldsymbol{V}$ decay; and (ii) the first k singular values of $\boldsymbol{X}$ are comparable. In the next section, we argue that, in practice, the first condition is often fulfilled. The second condition suggests that we should choose the number k of factors by finding a knee in the sequence of singular values.

Remark. Related ideas yield a nonparametric analysis of the diagonal thresholding algorithm [4].

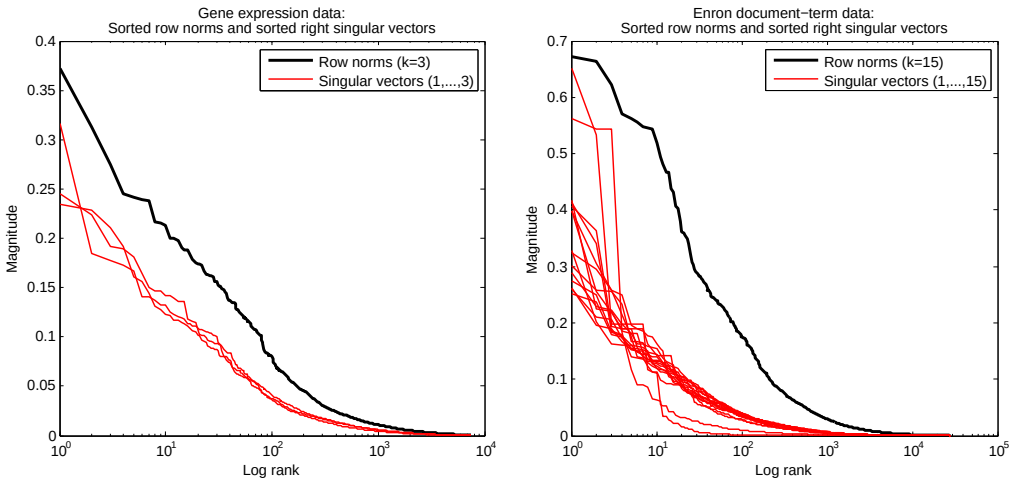

Figure 2: **Individual and joint decay of singular vectors.** The red lines show the individual decay of the dominant singular vectors of $\boldsymbol{X}$; the black lines mark the row norms of $\boldsymbol{V}$, which capture the joint decay of the singular vectors. [*Left*] Colon cancer gene expression data matrix (36 subjects $\times$ 7457 genes). [*Right*] Enron document–term matrix (39861 documents $\times$ 28102 terms).

3 When singular vectors decay

We offer some empirical and theoretical evidence that many types of data matrices have decaying singular vectors. Figure 2 displays the sorted components of the singular vectors of two data matrices, along with the sorted row norms of the matrix $\boldsymbol{V}$ of k dominant right singular vectors for an appropriate choice of k. In both cases, we see clear evidence of individual and joint decay. See Section 5 for details about the data. To continue, we describe two distinct settings where some theoretical analysis has been performed: (i) functional data and (ii) graph and network data.

3.1 Functional data

Functional data usually consists of point samples from a function defined on a Euclidean space. For example, the ith row of the data matrix might tabulate noisy values from a function f_i of a real variable sampled on a uniform grid:

$$\boldsymbol{X}_{ij} = f_i(j) + z_{ij} \quad i = 1, \dots, n \text{ and } j = 1, \dots, p.$$

Here, z_{ij} denote statistical errors. In many situations, we possess a priori information about the underlying functions f_i. For example, a sensor network measuring the evolution of a one-dimensional temperature field would report snapshots of a slowly varying smooth function.

We can use our prior information to select an orthonormal basis that represents the functions efficiently. More precisely, we would like the transform coefficients to decay quickly when we sort them in order of decreasing magnitude because a decaying vector is well approximated by a sparse vector (the basic principle behind Lemma 2.1). Therefore, we transform the rows of the data matrix *before* we apply the SC.PCA.JT algorithm. The field of computational harmonic analysis [14] has developed detailed theory and algorithms that justify this approach. The work of Johnstone et al. [4, 3] on SC.PCA relies on the same idea. Let us sketch two settings where the approach works.

1) Smooth functions. The Fourier coefficients of a smooth function decay at a rate connected with the level of smoothness. Indeed, for a function $f : [0, 1] \to \mathbb{R}$ with m derivatives, we have

$$\int_0^1 \left| f^{(m)}(t) \right|^2 \mathrm{d}t = \sum\nolimits_{j=-\infty}^{\infty} |c_j|^2 \, |j|^{2m} \quad \text{where} \quad c_j = \int_0^1 f(t) \, \mathrm{e}^{-2\pi \mathrm{i} j t} \, \mathrm{d}t.$$

Since the high-frequency coefficients are weighted heavily, they must be small. Thus, the low frequencies are usually best for representing a collection of smooth functions. Of course, an adaptive choice of frequencies will typically be superior to a fixed choice.

2) Piecewise smooth functions. A function that is smooth between discontinuities has decaying wavelet coefficients. Indeed, this type of function is well approximated by coarse-scale wavelets; the fine-scale wavelets concentrate near the discontinuities and are comparatively less important. Thus, a collection of piecewise smooth functions can be jointly represented by coarse wavelet components with a few fine-scale coefficients for detail.

3.2 Networks and graphs

Extensive empirical research on networks and graphs has demonstrated that the sequence of node degrees decays. In addition, the sequence of singular values of the adjacency matrix and the components of the singular vectors also exhibit a decay pattern. See [15, Sec. 2] for an overview.

There is one setting where there is a clear theoretical link between combinatorial graph properties and the eigenvectors of a related matrix. Consider a random walk on a connected, undirected graph. The dominant left eigenvector of the transition matrix provides the stationary distribution of the random walk, which is proportional to the degrees of the nodes. Therefore, when the node distribution of the graph decays, so does the dominant left eigenvector of the transition matrix.

The Kronecker graph model [15, Sec. 3] provides a potential explanation for the observed decay in the singular values and singular vectors of the adjacency matrix of a network. This approach forms a network by assuming that the associations among large communities follow the same pattern as the associations within a small "seed" community. The global adjacency matrix takes the form $\boldsymbol{A} \otimes \boldsymbol{A} \otimes \cdots \otimes \boldsymbol{A}$, where $\otimes$ denotes the Kronecker product. The singular vectors of the repeated Kronecker product are simply repeated Kronecker products of the singular vectors of $\boldsymbol{A}$. Thus, when the singular vectors of $\boldsymbol{A}$ exhibit even a small amount of decay, this decay is amplified in the singular vectors of the full network. A similar result holds for the singular values [15, Sec. 3.2.2]

4 Previous algorithms and theory

The literature contains a substantial number of algorithmic approaches to SC.PCA. These techniques fall into several clear categories, which we arrange in increasing order of computational cost.

1) Thresholding. Simple thresholding selects variables associated with the largest entries of the dominant singular vector of the data matrix. Diagonal thresholding [4] selects variables by identifying the largest-norm columns of the data matrix. The latter approach is very effective, but we emphasize that it is inappropriate when the columns are standardized or have comparable norms!

2) Penalty methods. These approaches marry the variational characterization of a dominant eigenvector with a penalty constraint that promotes sparsity [1, 16, 17, 18]. Researchers have proposed many algorithms that attempt the resulting nonconvex optimization. We have found that alternating maximization [18] offers the best computational cost and quality of output.

3) Greedy approaches. The forward method successively identifies a new variable that offers the greatest improvement and adds it to the active set [19, 9]. A faster approximate greedy algorithm appears in [7]. There are also reverse and bi-directional greedy methods [9].

4) Semidefinite relaxation. The sparse singular vector problem (1.1) is relaxed to a semidefinite program (SDP), and the cardinality constraint is replaced by an ℓ_1 bound on the semidefinite matrix variable. The SDP is solved using Nesterov's smoothing approach [5, 2] or the Burer–Monteiro factorization approach [20].

5) Brute force. These algorithms perform an exhaustive evaluation of all subsets of variables, perhaps using branch-and-bound techniques to reduce the cost of the search [19, 9].

Most theoretical work on SC.PCA starts from the premise that the data matrix is drawn from the *spiked covariance model*. In its simplest incarnation, this model assumes that each row of the data matrix is an independent realization of the random vector $\boldsymbol{v}^* + \sigma^2 \boldsymbol{z}^*$, where $\boldsymbol{v} \in \mathbb{R}^p$ is a fixed s-sparse vector, σ^2 is a variance parameter, and $\boldsymbol{z} \in \mathbb{R}^p$ is a standard normal vector. In principle, the goal of SC.PCA is to correctly identify the nonzero components of $\boldsymbol{v}$ from the noisy observations.

Researchers have developed asymptotic results when various estimators are applied to data from the spiked covariance model. Johnstone and Lu [4] demonstrate that PCA is asymptotically *inconsistent*

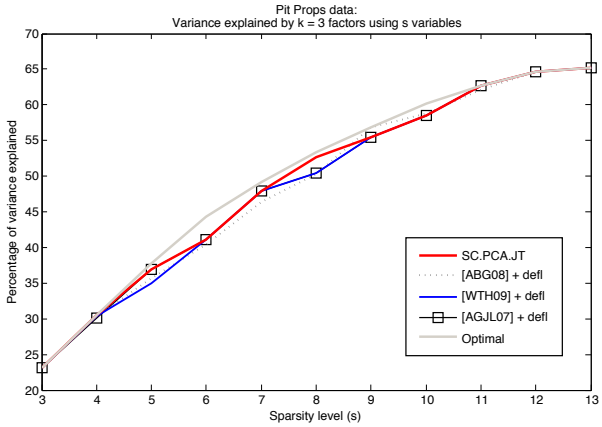

Figure 3: **Pit props.** Fraction of energy captured by 3 sparse factors using a total of s variables.

when $p/n \to \mathrm{c} > 0$. They also develop consistency results for diagonal thresholding. Amini and Wainwright [21] describe how the performance of algorithms depends on the quantities

$$\theta_{\mathrm{diag}} = \frac{n}{s^2 \log(p-s)} \quad \text{and} \quad \theta_{\mathrm{sdp}} = \frac{n}{s \log(p-s)}.$$

Diagonal thresholding (resp. semidefinite relaxation) asymptotically almost surely (a.a.s.) identifies the support of $\boldsymbol{v}$ if and only if θ_{diag} (resp. θ_{sdp}) is above a certain threshold. Furthermore, when θ_{sdp} is below a fixed level, *no algorithm* can a.a.s. identify the support of $\boldsymbol{v}$.

Since SC.PCA.JT is a thresholding technique, its performance for the spiked covariance model resembles that of the diagonal thresholding method rather than the SDP method. Nevertheless, our empirical studies indicate that we accrue little advantage by invoking sophisticated algorithms. This observation suggests that the spiked covariance model does not capture key aspects of our data.

5 Preliminary numerical evidence

We have performed basic comparisons between the SC.PCA.JT algorithm and related methods for some small and moderate data sets. We leave larger experiments for future work.

Pit props. The standard test case for SC.PCA is the 13×13 pit props correlation matrix [22]. We compute 3 sparse eigenvectors of total sparsity s using several methods. Figure 3 charts the proportion of variance explained. SC.PCA.JT outperforms the other methods, including semidefinite relaxation, almost uniformly. Note that diagonal thresholding does not apply to this example.

Gene expression data. We study the Notterman colon cancer dataset [23], which consists of expression levels of $p = 7457$ genes in each of $n = 36$ subjects, half cancerous. We center the expression levels for each gene, and look for a small set of genes that explains the remaining variability. See Figure 4 for the variance explained by each algorithm using $k = 3$ sparse factors, along with the running times. The two thresholding algorithms are almost uniformly the most effective; diagonal thresholding is slightly better for small s and SC.PCA.JT is slightly better for large s.

Enron document–term matrix. Finally, we applied SC.PCA.JT to the document–term matrix harvested from the Enron email database [24]. Figure 5 displays the energy explained by k dominant sparse factors for various k, along with the singular values of the data matrix with the applicable values of k marked; the red series corresponds with $k = 15$. Table 1 shows the most significant terms in five dominant sparse factors computed with $k = 15$ and $s = 500$.

Summary. The outcome of this project is somewhat depressing because it suggests that simple algorithms based on thresholding are not only faster but also more effective than fancier methods. Note that the $1 - s^{-1/2}$ behavior predicted by Theorem 2.3 appears in the curves for all of the algorithms, which indicates that this phenomenon may be intrinsic to the SC.PCA problem.

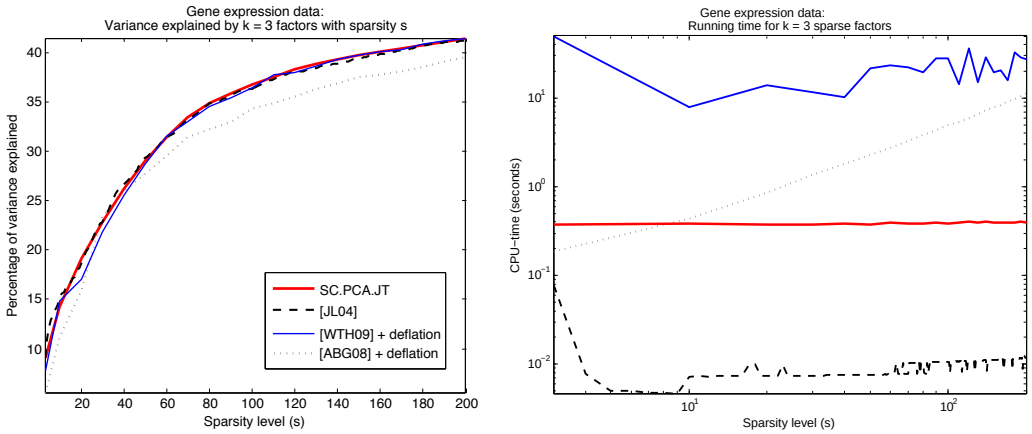

Figure 4: **Gene expression data.** [*Left*] Fraction of variance captured by $k = 3$ factors with joint sparsity s computed with several algorithms. [*Right*] Running times for algorithms in left panel.

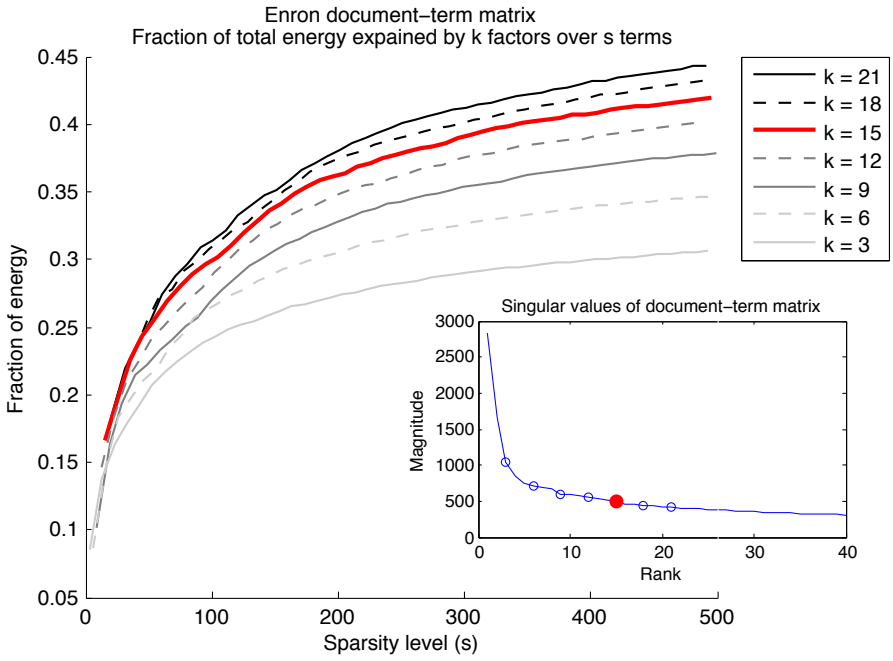

Figure 5: **Enron document–term matrix.** Fraction of energy captured by k sparse factors for various k. [*Inset*] Singular values of matrix with applicable values of k marked.

Factor 1: `company power energy california electricity`
Factor 2: `company -power -energy -california firm fund round investor ...`
Factor 3: `company stock -davis -california -firm round ventures ...`
Factor 4: `texas game allowed yard defense team rank passing fantasy ...`
Factor 5: `pst columbia mid avista aquila`

Table 1: **Enron document–term matrix.** Terms with the highest loadings in the five dominant sparse factors obtained by SC.PCA.JT with $k = 15$ total factors and $s = 500$ total terms. Signs indicate the signs of the loadings. Terms are ordered by global importance.

References

[1] I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the LASSO. *J. Compu. Graph. Statist.*, 12(3):531–547, Sep. 2003.

[2] R. Luss and A. d'Asprémont. Clustering and feature selection using sparse principal component analysis. *Optim. Engr.*, 11(1):145–157, Feb. 2010.

[3] D. Paul and I. M. Johnstone. Augmented sparse principal component analysis for high-dimensional data. Technical report, Univ. California at Davis, 2007.

[4] I. M. Johnstone and A. Lu. Sparse principal components. Technical report, Stanford Univ., 2004.

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

[6] R. Bhatia. *Matrix Analysis*. Number 169 in GTM. Springer, Berlin, 1997.

[7] A. d'Asprémont, F. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *J. Machine Learning Res.*, 9:1269–1294, Jul. 2008.

[8] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. ACM TR 2009-05, California Inst. Tech., Sep. 2009.

[9] B. Moghaddam, Y. Weis, and S. Avidan. Spectral bounds for sparse PCA: Exact and greedy algorithms. In *Neural Information Processing Systems*, Vancouver, Dec. 2005.

[10] G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins Univ. Press, Baltimore, MD, 3rd edition, 1996.

[11] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A fast randomized algorithm for the approximation of matrices. *Appl. Comp. Harmon. Anal.*, 25(3):335–366, 2008.

[12] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM J. Matrix Anal. Appl.*, 31(3):1100–1124, 2009.

[13] A. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: Fast algorithms for compressed sensing. In *STOC '07: Proc. 39th Ann. ACM Symp. Theory of Computing*, San Diego, 2007. Available from `http://www.acm.caltech.edu/~jtropp/conf/GSTV06-One-Sketch-complete.pdf`.

[14] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, London, 2nd edition, 1999.

[15] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. Available from `arXiv:0812.4905`, Aug. 2009.

[16] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *J. Comput Graph. Statist.*, 15:262–286, 2006.

[17] H. Shen and J. Z. Huang. Sparse principal component analysis via regularized low rank matrix approximation. *J. Multivar. Anal.*, 99:1015–1034, 2008.

[18] D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatist.*, 10(3):515–534, 2009.

[19] G. P. McCabe. Principal variables. *Technometrics*, 26(2):137–144, May 1984.

[20] R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. Technical report, INRIA, Sep. 2009.

[21] A. A. Amini and M. J. Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. *Ann. Statist.*, 37(5B):2877–2921, 2009.

[22] J. Jeffers. Two case studies in the application of principal components. *Appl. Statist.*, 16:225–236, 1967.

[23] D. A. Notterman, U. Alon, A. J. Sierk, and A. J. Levine. Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays. *Cancer Res.*, 61:3124–3130, 2001. Data available from `http://microarray.princeton.edu`.

[24] A. Frank and A. Asuncion. UCI machine learning repository, Bag of Words data set, 2010.