A recursive divide-and-conquer approach for sparse principal component analysis

Qian Zhao, Deyu Meng*, Zongben Xu

Institute for Information and System Sciences, School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an 710049, PR China

Abstract

In this paper, a new method is proposed for sparse PCA based on the recursive divide-and-conquer methodology. The main idea is to separate the original sparse PCA problem into a series of much simpler sub-problems, each having a closed-form solution. By recursively solving these sub-problems in an analytical way, an efficient algorithm is constructed to solve the sparse PCA problem. The algorithm only involves simple computations and is thus easy to implement. The proposed method can also be very easily extended to other sparse PCA problems with certain constraints, such as the nonnegative sparse PCA problem. Furthermore, we have shown that the proposed algorithm converges to a stationary point of the problem, and its computational complexity is approximately linear in both data size and dimensionality. The effectiveness of the proposed method is substantiated by extensive experiments implemented on a series of synthetic and real data in both reconstruction-error-minimization and data-variance-maximization viewpoints.

Keywords: Face recognition, nonnegativity, principal component analysis, recursive divide-and-conquer, sparsity.

*Corresponding author. Tel.: +86 13032904180; fax: +86 2982668559.

Email addresses: zhao.qian@stu.xjtu.edu.cn (Qian Zhao), dymeng@mail.xjtu.edu.cn (Deyu Meng), zbxu@mail.xjtu.edu.cn (Zongben Xu)

1. Introduction

Principal component analysis (PCA) is one of the most classical and popular tools for data analysis and dimensionality reduction, and has a wide range of successful applications throughout science and engineering [1]. By seeking the so-called principal components (PCs), along which the data variance is maximally preserved, PCA can always capture the intrinsic latent structure underlying data. Such information greatly facilitates many further data processing tasks, such as feature extraction and pattern recognition.

Despite its many advantages, the conventional PCA suffers from the fact that each component is generally a linear combination of all data variables, and all weights in the linear combination, also called loadings, are typically non-zeros. In many applications, however, the original variables have meaningful physical interpretations. In biology, for example, each variable of gene expression data corresponds to a certain gene. In these cases, the derived PC loadings are always expected to be sparse (i.e. contain fewer non-zeros) so as to facilitate their interpretability. Moreover, in certain applications, such as financial asset trading, the sparsity of the PC loadings is especially expected since fewer nonzero loadings imply fewer transaction costs.

Accordingly, sparse PCA has attracted much attention in the recent decade, and a variety of methods for this topic have been developed [2–23]. The first attempt for this topic is to make certain post-processing transformation, e.g. rotation [2] by Jolliffe and simple thresholding [3] by Cadima and Jolliffe, on the PC loadings obtained by the conventional PCA to enforce sparsity. Jolliffe and Uddin further advanced a SCoTLASS algorithm by simultaneously calculating sparse PCs on the PCA model with additional l_1-norm penalty on loading vectors [4]. Better results have been achieved by the SPCA algorithm of Zou et al., which was developed based on iterative elastic net regression [5]. D'Aspremont et al. proposed a method, called DSPCA, for finding sparse PCs by solving a sequence of semidefinite programming (SDP) relaxations of sparse PCA [6]. Shen and Huang developed a series of methods called sPCA-rSVD (including sPCA-rSVD$_{l_0}$, sPCA-rSVD$_{l_1}$, sPCA-rSVD$_{SCAD}$), computing sparse PCs by low-rank matrix factorization under multiple sparsity-including penalties [7]. Journée et al. designed four algorithms, denoted as GPower$_{l_0}$, GPower$_{l_1}$, GPower$_{l_0,m}$, and GPower$_{l_1,m}$, respectively, for sparse PCA by formulating the issue as non-concave maximization problems with l_0- or l_1-norm sparsity-inducing penalties and extracting single unit sparse PC sequentially or block units

ones simultaneously [8]. Based on probabilistic generative model of PCA, some methods have also been attained [9–12], e.g. the EMPCA method derived by Sigg and Buhmann for sparse and/or nonnegative sparse PCA [9]. Sriperumbudur et al. provided an iterative algorithm called DCPCA, where each iteration consists of solving a quadratic programming (QP) problem [13, 14]. Recently, Lu and Zhang developed an augmented Lagrangian method (ALSPCA briefly) for sparse PCA by solving a class of non-smooth constrained optimization problems [15]. Additionally, d'Aspremont derived a PathSPCA algorithm that computes a full set of solutions for all target numbers of nonzero coefficients [16].

There are mainly two methodologies utilized by the current research on sparse PCA problem. The first is the greedy approach, including DSPCA [6], sPCA-rSVD [7], EMPCA [9], PathSPCA [16], etc. These methods mainly focus on the solving of one-sparse-PC model, and more sparse PCs can be sequentially calculated on the deflated data matrix or data covariance [24]. Under this methodology, the first several sparse PCs underlying the data can generally be properly extracted, while the computation for more sparse PCs tends to be incrementally invalidated due to the cumulation of computational error. The second is the block approach. Typical methods include SCoTLASS [4], GPower$_{l_0,m}$, GPower$_{l_1,m}$ [8], ALSPCA [15], etc. These methods aim to calculate multiple sparse PCs at once by utilizing certain block optimization techniques. The block approach for sparse PCA is expected to be more efficient than the greedy one to simultaneously attain multiple PCs, while is generally difficult to elaborately rectify each individual sparse PC based on some specific requirements in practice (e.g. the number of nonzero elements in each PC).

In this paper, a new methodology, called the recursive divide-and-conquer (ReDaC briefly), is employed for solving the sparse PCA problem. The main idea is to decompose the original large and complex problem of sparse PCA into a series of small and simple sub-problems, and then recursively solve them. Each of these sub-problems has a closed-form solution, which makes the new method simple and very easy to implement. On one hand, as compared with the greedy approach, the new method is expected to integratively achieve a collection of appropriate sparse PCs of the problem by iteratively rectifying each sparse PC in a recursive way. The group of sparse PCs attained by the proposed method is further proved being a stationary solution of the original sparse PCA problem. On the other hand, as compared with the block approach, the new method can easily handle the constraints su-

perimposed on each individual sparse PC, such as certain sparsity and/or nonnegative constraints. Besides, the computational complexity of the proposed method is approximately linear in both data size and dimensionality, which makes it well-suited to handle large-scale problems of sparse PCA.

In what follows, the main idea and the implementation details of the proposed method are first introduced in Section 2. Its convergence and computational complexity are also analyzed in this section. The effectiveness of the proposed method is comprehensively substantiated based on a series of empirical studies in Section 3. Then the paper is concluded with a summary and outlook for future research. Throughout the paper, we denote matrices, vectors and scalars by the upper-case bold-faced letters, lower-case bold-faced letters, and lower-case letters, respectively.

2. The recursive divide-and-conquer method for sparse PCA

In the following, we first introduce the fundamental models for the sparse PCA problem.

2.1. Basic models of sparse PCA

Denote the input data matrix as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T \in \mathbb{R}^{n\times d}$, where n and d are the size and the dimensionality of the given data, respectively. After a location transformation, we can assume all $\{\mathbf{x}_i\}_{i=1}^n$ to have zero mean. Let $\mathbf{\Sigma} = \frac{1}{n}\mathbf{X}^T\mathbf{X} \in \mathbb{R}^{d\times d}$ be the data covariance matrix.

The classical PCA can be solved through two types of optimization models [1]. The first is constructed by finding the $r(\leq d)$-dimensional linear subspace where the variance of the input data $\mathbf{X}$ is maximized [25]. On this data-variance-maximization viewpoint, the PCA is formulated as the following optimization model:

$$\max_{\mathbf{V}} \ \mathrm{Tr}(\mathbf{V}^T\mathbf{\Sigma}\mathbf{V}) \quad s.t. \ \ \mathbf{V}^T\mathbf{V} = \mathbf{I}, \tag{1}$$

where $\mathrm{Tr}(\mathbf{A})$ denotes the trace of the matrix $\mathbf{A}$ and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r) \in \mathbb{R}^{d\times r}$ denotes the array of PC loading vectors. The second is formulated by seeking the r-dimensional linear subspace on which the projected data and the original ones are as close as possible [26]. On this reconstruction-error-minimization viewpoint, the PCA corresponds to the following model:

$$\min_{\mathbf{U},\mathbf{V}} \ \left\|\mathbf{X} - \mathbf{U}\mathbf{V}^T\right\|_F^2 \quad s.t. \ \ \mathbf{V}^T\mathbf{V} = \mathbf{I}, \tag{2}$$

where $\|\mathbf{A}\|_F$ is the Frobenius norm of $\mathbf{A}$, $\mathbf{V} \in \mathbb{R}^{d\times r}$ is the matrix of PC loading array and $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r) \in \mathbb{R}^{n\times r}$ is the matrix of projected data. The two models are intrinsically equivalent and can attain the same PC loading vectors [1].

Corresponding to the PCA models (1) and (2), the sparse PCA problem has the following two mathematical formulations[1]:

$$\max_{\mathbf{V}} \operatorname{Tr}(\mathbf{V}^T\boldsymbol{\Sigma}\mathbf{V}) \quad s.t. \;\; \mathbf{v}_i^T\mathbf{v}_i = 1, \quad \|\mathbf{v}_i\|_p \le t_i \;(i = 1, 2, \ldots, r), \tag{3}$$

and

$$\min_{\mathbf{U},\mathbf{V}} \left\|\mathbf{X} - \mathbf{U}\mathbf{V}^T\right\|_F^2 \quad s.t. \;\; \mathbf{v}_i^T\mathbf{v}_i = 1, \quad \|\mathbf{v}_i\|_p \le t_i \;(i = 1, 2, \ldots, r), \tag{4}$$

where $p = 0$ or 1 and the corresponding $\|\mathbf{v}\|_p$ denotes the l_0- or the l_1-norm of $\mathbf{v}$, respectively. Note that the involved l_0 or l_1 penalty in the above models (3) and (4) tends to enforce sparsity of the output PCs. Methods constructed on (3) include SCoTLASS [4], DSPCA [6], DCPCA [13, 14], ALSPCA [15], etc., and those related to (4) include SPCA [5], sPCA-rSVD [7], SPC [19], GPower [8], etc. In this paper, we will construct our method on the reconstruction-error-minimization model (4), while our experiments will verify that the proposed method also performs well based on the data-variance-maximization criterion.

2.2. Decompose original problem into small and simple sub-problems

The objective function of the sparse PCA model (4) can be equivalently formulated as follows:

$$\left\|\mathbf{X} - \mathbf{U}\mathbf{V}^T\right\|_F^2 = \left\|\mathbf{X} - \sum\nolimits_{j=1}^{r}\mathbf{u}_j\mathbf{v}_j^T\right\|_F^2 = \left\|\mathbf{E}_i - \mathbf{u}_i\mathbf{v}_i^T\right\|_F^2,$$

where $\mathbf{E}_i = \mathbf{X} - \sum_{j\neq i}\mathbf{u}_j\mathbf{v}_j^T$. It is then easy to separate the original large minimization problem, which is with respect to $\mathbf{U}$ and $\mathbf{V}$, into a series of small minimization problems, which are each with respect to a column vector $\mathbf{u}_i$ of $\mathbf{U}$ and $\mathbf{v}_i$ of $\mathbf{V}$ for $i = 1, 2, \ldots, r$, respectively, as follows:

$$\min_{\mathbf{v}_i} \left\|\mathbf{E}_i - \mathbf{u}_i\mathbf{v}_i^T\right\|_F^2 \quad s.t. \;\; \mathbf{v}_i^T\mathbf{v}_i = 1, \quad \|\mathbf{v}_i\|_p \le t_i, \tag{5}$$

[1]It should be noted that the orthogonality constraints of PC loadings in (1) and (2) are not imposed in (3) and (4). This is because simultaneously enforcing sparsity and orthogonality is generally a very difficult (and perhaps unnecessary) task. Like most of the existing sparse PCA methods [5–8], we do not enforce orthogonal PCs in the models.

and

$$\min_{\mathbf{u}_i} \left\|\mathbf{E}_i - \mathbf{u}_i\mathbf{v}_i^T\right\|_F^2. \tag{6}$$

Through recursively optimizing these small sub-problems, the recursive divide-and-conquer (ReDaC) method for solving the sparse PCA model (4) can then be naturally constructed.

It is very fortunate that both the minimization problems in (5) and (6) have closed-form solutions. This implies that the to-be-constructed ReDaC method can be fast and efficient, as presented in the following sub-sections.

2.3. The closed-form solutions of (5) and (6)

For the convenience of denotation, we first rewrite (5) and (6) as the following forms:

$$\min_{\mathbf{v}} \quad \left\|\mathbf{E} - \mathbf{u}\mathbf{v}^T\right\|_F^2 \qquad s.t. \;\; \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_p \le t, \tag{7}$$

and

$$\min_{\mathbf{u}} \quad \left\|\mathbf{E} - \mathbf{u}\mathbf{v}^T\right\|_F^2, \tag{8}$$

where $\mathbf{u}$ is n-dimensional and $\mathbf{v}$ is d-dimensional. Since the objective function $\left\|\mathbf{E} - \mathbf{u}\mathbf{v}^T\right\|_F^2$ can be equivalently transformed as:

$$\begin{aligned}\left\|\mathbf{E} - \mathbf{u}\mathbf{v}^T\right\|_F^2 &= \mathrm{Tr}((\mathbf{E} - \mathbf{u}\mathbf{v}^T)^T(\mathbf{E} - \mathbf{u}\mathbf{v}^T)) \\ &= \|\mathbf{E}\|_F^2 - 2\mathrm{Tr}(\mathbf{E}^T\mathbf{u}\mathbf{v}^T) + \mathrm{Tr}(\mathbf{v}\mathbf{u}^T\mathbf{u}\mathbf{v}^T) \\ &= \|\mathbf{E}\|_F^2 - 2\mathbf{u}^T\mathbf{E}\mathbf{v} + \mathbf{u}^T\mathbf{u}\mathbf{v}^T\mathbf{v},\end{aligned}$$

(7) and (8) are equivalent to the following optimization problems, respectively:

$$\max_{\mathbf{v}} \quad (\mathbf{E}^T\mathbf{u})^T\mathbf{v} \qquad s.t. \;\; \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_p \le t, \tag{9}$$

and

$$\min_{\mathbf{u}} \quad \mathbf{u}^T\mathbf{u} - 2(\mathbf{E}\mathbf{v})^T\mathbf{u}. \tag{10}$$

The closed-form solutions of (9) and (10), i.e. (7) and (8), can then be presented as follows.

We present the closed-form solution to (8) in the following theorem.

Theorem 1. *The optimal solution of (8) is* $\mathbf{u}^*(\mathbf{v}) = \mathbf{E}\mathbf{v}$.

The theorem is very easy to prove by calculating where the gradient of $\mathbf{u}^T\mathbf{u} - 2(\mathbf{E}\mathbf{v})^T\mathbf{u}$ is equal to zero. We thus omit the proof.

In the $p = 0$ case, the closed-form solution to (9) is presented in the following theorem. Here, we denote $\mathbf{w} = \mathbf{E}^T\mathbf{u}$, and $hard_\lambda(\mathbf{w})$ the hard thresholding function, whose i-th element corresponds to $I(|w_i| \geq \lambda)w_i$, where w_i is the i-th element of $\mathbf{w}$ and $I(x)$ (equals 1 if x is ture, and 0 otherwise) is the indicator function. The proof of the theorem is provided in Appendix A.

Theorem 2. *The optimal solution of*

$$\max_{\mathbf{v}} \ \mathbf{w}^T\mathbf{v} \quad s.t. \ \ \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_0 \leq t, \tag{11}$$

is given by:

$$\mathbf{v}_0^*(\mathbf{w}, t) = \begin{cases} \phi, & t < 1, \\ \frac{hard_{\theta_k}(\mathbf{w})}{\|hard_{\theta_k}(\mathbf{w})\|_2}, & k \leq t < k+1 \quad (k = 1, 2, \ldots, d-1), \\ \frac{\mathbf{w}}{\|\mathbf{w}\|_2} & t \geq d, \end{cases}$$

where θ_k denotes the k-th largest element of $|\mathbf{w}|$.

In the above theorem, ϕ denotes the empty set, implying that when $t < 1$, the optimum of (11) does not exist.

In the $p = 1$ case, (7) has the following closed-form solution. In the theorem, we denote $f_{\mathbf{w}}(\lambda) = \frac{soft_\lambda(\mathbf{w})}{\|soft_\lambda(\mathbf{w})\|_2}$, where $soft_\lambda(\mathbf{w})$ represents the soft thresholding function $sign(\mathbf{w})(|\mathbf{w}| - \lambda)_+$, where $(\mathbf{x})_+$ represents the vector attained by projecting $\mathbf{x}$ to its nonnegative orthant, and $(I_1, I_2, \ldots, I_d)$ denotes the permutation of $(1, 2, \ldots, d)$ based on the ascending order of $|\mathbf{w}| = (|w_1|, |w_2|, \ldots, |w_d|)^T$.

Theorem 3. *The optimal solution of*

$$\max_{\mathbf{v}} \ \mathbf{w}^T\mathbf{v} \quad s.t. \ \ \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_1 \leq t, \tag{12}$$

is given by:

$$\mathbf{v}_1^*(\mathbf{w}, t) = \begin{cases} \phi, & t < 1, \\ f_{\mathbf{w}}(\lambda_k), & \|f_{\mathbf{w}}(|w_{I_k}|)\|_1 \leq t < \|f_{\mathbf{w}}(|w_{I_{k-1}}|)\|_1 \quad (k = 2, 3, \ldots, d-1), \\ f_{\mathbf{w}}(\lambda_1), & \|f_{\mathbf{w}}(|w_{I_1}|)\|_1 \leq t < \sqrt{d}, \\ f_{\mathbf{w}}(0), & t \geq \sqrt{d}, \end{cases}$$

where for $k = 1, 2, \ldots, d-1$,

$$\lambda_k = \frac{(m-t^2)(\sum_{i=1}^{m} a_i) - \sqrt{t^2(m-t^2)(m\sum_{i=1}^{m} a_i^2 - (\sum_{i=1}^{m} a_i)^2)}}{m(m-t^2)},$$

where $(a_1, a_2, \ldots, a_m) = (|w_{I_k}|, |w_{I_{k+1}}|, \ldots, |w_{I_d}|)$, $m = d - k + 1$.

It should be noted that we have proved that $\|f_{\mathbf{w}}(|w_{I_{d-1}}|)\|_1 = 1$ and $\|f_{\mathbf{w}}(\lambda)\|_1$ is a monotonically decreasing function with respect to λ in Lemma 1 of the appendix. This means that we can conduct the optimum $\mathbf{v}^*(\mathbf{w})$ of the optimization problem (7) for any $\mathbf{w}$ based on the above theorem.

The ReDaC algorithm can then be easily constructed based on Theorems 1-3.

2.4. The recursive divide-and-conquer algorithm for sparse PCA

The main idea of the new algorithm is to recursively optimize each column, $\mathbf{u}_i$ of $\mathbf{U}$ or $\mathbf{v}_i$ of $\mathbf{V}$ for $i = 1, 2, \ldots, r$, with other $\mathbf{u}_j$s and $\mathbf{v}_j$s ($j \neq i$) fixed. The process is summarized as follows:

- Update each column $\mathbf{v}_i$ of $\mathbf{V}$ for $i = 1, 2, \ldots, r$ by the closed-form solution of (5) attained from Theorem 2 (for $p = 0$) or Theorem 3 (for $p = 1$).
- Update each column $\mathbf{u}_i$ of $\mathbf{U}$ for $i = 1, 2, \ldots, r$ by the closed-form solution of (6) calculated from Theorem 1.

Through implementing the above procedures iteratively, $\mathbf{U}$ and $\mathbf{V}$ can be recursively updated until the stopping criterion is satisfied. We summarize the aforementioned ReDaC technique as Algorithm 1.

We then briefly discuss how to specify the stopping criterion of the algorithm. The objective function of the sparse PCA model (4) is monotonically decreasing in the iterative process of Algorithm 1 since each of the step 5 and step 6 in the iterations makes an exact optimization for a column vector $\mathbf{u}_i$ of $\mathbf{U}$ or $\mathbf{v}_i$ of $\mathbf{V}$, with all of the others fixed. We can thus terminate the iterations of the algorithm when the updating rate of $\mathbf{U}$ or $\mathbf{V}$ is smaller than some preset threshold, or the maximum number of iterations is reached.

Now we briefly analyze the computational complexity of the proposed ReDaC algorithm. It is evident that the computational complexity of Algorithm 1 is essentially determined by the iterations between step 5 and step

Algorithm 1 ReDaC algorithm for sparse PCA

Input: Data matrix $\mathbf{X} \in R^{n\times d}$, number of sparse PCs r, sparsity parameters $\mathbf{t} = (t_1, \ldots, t_r)$.

1: Initialize $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r) \in R^{n\times r}$, $\mathbf{V} = (\mathbf{v}_1, \mathbf{u}_2, \ldots, \mathbf{v}_r) \in R^{d\times r}$.
2: **repeat**
3: **for** $i = 1, \ldots r$ **do**
4: Compute $\mathbf{E}_i = \mathbf{X} - \sum_{j\neq i} \mathbf{u}_j \mathbf{v}_j^T$.
5: Update $\mathbf{v}_i$ via solving (5) based on Theorem 2 (for $p = 0$) or Theorem 3 (for $p = 1$).
6: Update $\mathbf{u}_i$ via solving (6) based on Theorem 1.
7: **end for**
8: **until** stopping criterion satisfied.

Output: The sparse PC loading vectors $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r)$.

6, i.e. the calculation of the closed-form solutions of $\mathbf{v}_i$ and $\mathbf{u}_i$ of $\mathbf{V}$ and $\mathbf{U}$, respectively. To compute $\mathbf{u}_i$, only simple operations are involved and the computation needs $O(nd)$ cost. To compute $\mathbf{v}_i$, a sorting for the elements of the d-dimensional vector $|\mathbf{w}| = |\mathbf{E}^T\mathbf{u}|$ is required, and the total computational cost is around $O(nd \log d)$ by applying the well-known heap sorting algorithm [27]. The whole process of the algorithm thus requires around $O(rnd \log d)$ computational cost in each iteration. That is, the computational complexity of the proposed algorithm is approximately linear in both the size and the dimensionality of input data.

2.5. Convergence analysis

In this section we evaluate the convergence of the proposed algorithm.

The convergence of our algorithm can actually be implied by the monotonic decrease of the cost function of (4) during the iterations of the algorithm. In specific, in each iteration of the algorithm, step 5 and step 6 optimize the column vector $\mathbf{u}_i$ of $\mathbf{U}$ or $\mathbf{v}_i$ of $\mathbf{V}$, with all of the others fixed, respectively. Since the objective function of (4) is evidently lower bounded (≥ 0), the algorithm is guaranteed to be convergent.

We want to go a further step to evaluate where the algorithm converges. Based on the formulation of the optimization problem (4), we can construct

a specific function as follows:

$$f(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r) = f_0(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r) + \sum_{i=1}^{r} f_i(\mathbf{v}_i). \tag{13}$$

where

$$f_0(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r) = \left\|\mathbf{X}-\mathbf{U}\mathbf{V}^T\right\|_F^2 = \left\|\mathbf{X}-\sum\nolimits_{i=1}^{r}\mathbf{u}_i\mathbf{v}_i^T\right\|_F^2,$$

and for each of $i = 1,\ldots,r$, $f_i(\mathbf{v}_i)$ is an indicator function defined as:

$$f_i(\mathbf{v}_i) = \begin{cases} 0, & \text{if } \|\mathbf{v}_i\|_p \le t_i \text{ and } \mathbf{v}_i^T\mathbf{v}_i = 1, \\ \infty, & \text{otherwise.} \end{cases}$$

It is then easy to show that the constrained optimization problem (4) is equivalent to the unconstrained problem

$$\min_{\{\mathbf{u}_i,\mathbf{v}_i\}_{i=1}^r} f(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r). \tag{14}$$

The proposed ReDaC algorithm can then be viewed as a block coordinate descent (BCD) method for solving (14) [28], by alteratively optimizing $\mathbf{u}_i, \mathbf{v}_i$, $i = 1, 2, \ldots, r$, respectively. Then the following theorem implies that our algorithm can converge to a stationary point of the problem.

Theorem 4 ([28])**.** *Assume that the level set $X^0 = \{x : f(x) \le f(x^0)\}$ is compact and that f is continuous on X^0. If $f(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r)$ is regular and has at most one minimum in each $\mathbf{u}_i$ and $\mathbf{v}_i$ with others fixed for $i = 1, 2, \ldots, r$, then the sequence $(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r)$ generated by Algorithm 1 converges to a stationary point of f.*

In the above theorem, the assumption that the function f, as defined in (14), is regular holds under the condition that $dom(f_0)$ is open and f_0 is Gateaux-differentiable on $dom(f_0)$ (Lemma 3.1 under Condition A1 in [28]). Based on Theorems 1-3, we can also easily see that $f(\mathbf{u}_1,\ldots,\mathbf{u}_r,\mathbf{v}_1,\ldots,\mathbf{v}_r)$ has unique minimum in each $\mathbf{u}_i$ and $\mathbf{v}_i$ with others fixed. The above theorem can then be naturally followed by Theorem 4.1(c) in [28].

Another advantage of the proposed ReDaC methodology is that it can be easily extended to other sparse PCA applications when certain constraints are needed for output sparse PCs. In the following section we give one of the extensions of our methodology — nonnegative sparse PCA problem.

2.6. The ReDaC method for nonnegative sparse PCA

The nonnegative sparse PCA [29] problem differs from the conventional sparse PCA in its nonnegativity constraint imposed on the output sparse PCs. The nonnegativity property of this problem is especially important in some applications such as microeconomics, environmental science, biology, etc. [30]. The corresponding optimization model is written as follows:

$$\min_{\mathbf{U},\mathbf{V}} \left\|\mathbf{X}-\mathbf{U}\mathbf{V}^T\right\|_F^2 \quad s.t.\ \ \mathbf{v}_i^T\mathbf{v}_i=1, \quad \|\mathbf{v}_i\|_p \le t_i, \quad \mathbf{v}_i \succeq 0\ (i=1,2,\ldots,r), \tag{15}$$

where $\mathbf{v}_i \succeq 0$ means that each element of $\mathbf{v}_i$ is greater than or equal to 0.

By utilizing the similar recursive divide-and-conquer strategy, this problem can be separated into a series of small minimization problems, each with respect to a column vector $\mathbf{u}_i$ of $\mathbf{U}$ and $\mathbf{v}_i$ of $\mathbf{V}$ for $i=1,2,\ldots,r$, respectively, as follows:

$$\min_{\mathbf{v}_i} \left\|\mathbf{E}_i-\mathbf{u}_i\mathbf{v}_i^T\right\|_F^2 \quad s.t.\ \ \mathbf{v}_i^T\mathbf{v}_i=1, \quad \|\mathbf{v}_i\|_p \le t_i, \quad \mathbf{v}_i \succeq 0 \tag{16}$$

and

$$\min_{\mathbf{u}_i} \left\|\mathbf{E}_i-\mathbf{u}_i\mathbf{v}_i^T\right\|_F^2, \tag{17}$$

where $p=0$ or 1. Since (17) is of the same formulation as (6), we only need to discuss how to solve (16). For the convenience of denotation, we first rewrite (16) as:

$$\min_{\mathbf{v}} \left\|\mathbf{E}-\mathbf{u}\mathbf{v}^T\right\|_F^2 \quad s.t.\ \ \mathbf{v}^T\mathbf{v}=1, \quad \|\mathbf{v}\|_p \le t, \quad \mathbf{v} \succeq 0. \tag{18}$$

The closed-form solution of (18) is given in the following theorem.

Theorem 5. *The closed-form solution of (18) is* $\mathbf{v}_p^*((\mathbf{w})_+,t)$ *(*$p=0,1$*), where* $\mathbf{w}=\mathbf{E}^T\mathbf{u}$*, and* $\mathbf{v}_0^*(\cdot,\cdot)$ *and* $\mathbf{v}_1^*(\cdot,\cdot)$ *are defined in Theorem 2 and Theorem 3, respectively.*

By virtue of the closed-form solution of (18) given by Theorem 5, we can now construct the ReDaC algorithm for solving nonnegative sparse PCA model (15). Since the algorithm differs from Algorithm 1 only in step 5 (i.e. updating of $\mathbf{v}_i$), we only list this step in Algorithm 2.

We then substantiate the effectiveness of the proposed ReDaC algorithms for sparse PCA and nonnegative sparse PCA through experiments in the next section.

Algorithm 2 ReDaC algorithm for nonnegative sparse PCA

5: Update $\mathbf{v}_i$ via solving (16) based on Theorem 5.

3. Experiments

To evaluate the performance of the proposed ReDaC algorithm on the sparse PCA problem, we conduct experiments on a series of synthetic and real data sets. All the experiments are implemented on Matlab 7.11(R2010b) platform in a PC with AMD Athlon(TM) 64 X2 Dual 5000+@2.60 GHz (CPU), 2GB (memory), and Windows XP (OS). In all experiments, the SVD method is utilized for initialization. The proposed algorithm under both $p = 0$ and $p = 1$ was implemented in all experiments and mostly have a similar performance. We thus only list the better one throughout.

3.1. Synthetic simulations

Two synthetic data sets are first utilized to evaluate the performance of the proposed algorithm on recovering the ground-truth sparse principal components underlying data.

3.1.1. Hastie data

Hastie data set was first proposed by Zou et al. [5] to illustrate the advantage of sparse PCA over conventional PCA on sparse PC extraction. So far this data set has become one of the most frequently utilized benchmark data for testing the effectiveness of sparse PCA methods. The data set is generated in the following way: first, three hidden factors V_1, V_2 and V_3 are created as:

$$V_1 \sim \mathcal{N}(0, 290), \quad V_2 \sim \mathcal{N}(0, 300), \quad V_3 = 0.3V_1 + 0.925V_2 + \varepsilon,$$

where $\varepsilon \sim \mathcal{N}(0, 1)$, and V_1, V_2 and ε are independent; afterwards, 10 observable variables are generated as:

$$\begin{aligned} X_i &= V_1 + \varepsilon_i^1, \quad i = 1, 2, 3, 4, \\ X_i &= V_2 + \varepsilon_i^2, \quad i = 5, 6, 7, 8, \\ X_i &= V_3 + \varepsilon_i^3, \quad i = 9, 10, \end{aligned}$$

where $\varepsilon_i^j \sim \mathcal{N}(0, 1)$ and all ε_i^js are independent. The data so generated are of intrinsic sparse PCs [5]: the first recovers the factor V_2 only using (X_5, X_6, X_7, X_8), and the second recovers V_1 only utilizing (X_1, X_2, X_3, X_4).

We generate 100 sets of data, each contains 1000 data generated in the aforementioned way, and apply Algorithm 1 to them to extract the first two sparse PCs. The results show that our algorithm can perform well in all experiments. In specific, the proposed ReDaC algorithm faithfully delivers the ground-truth sparse PCs in all experiments. The effectiveness of the proposed algorithm is thus easily substantiated in this series of benchmark data.

3.1.2. Synthetic toy data

As [7] and [8], we adopt another interesting toy data, with intrinsic sparse PCs, to evaluate the performance of the proposed method. The data are generated from the Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with mean $\mathbf{0}$ and covariance $\mathbf{\Sigma} \in \mathbb{R}^{10\times 10}$, which is calculated by

$$\mathbf{\Sigma} = \sum_{j=1}^{10} c_j \mathbf{v}_j \mathbf{v}_j^T.$$

Here, $(c_1, c_2, ..., c_{10})$, the eigenvalues of the covariance matrix $\mathbf{\Sigma}$, are prespecified as $(250, 240, 50, 50, 6, 5, 4, 3, 2, 1)$, respectively, and $(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{10})$ are 10-dimensional orthogonal vectors, formulated by

$$\mathbf{v}_1 = (0.422, 0.422, 0.422, 0.422, 0, 0, 0, 0, 0.380, 0.380)^T,$$
$$\mathbf{v}_2 = (0, 0, 0, 0, 0.489, 0.489, 0.489, 0.489, -0.147, 0.147)^T,$$

and the rest being generated by applying Gram-Schmidt orthonormalization to 8 randomly valued 10-dimensional vectors. It is easy to see that the data generated under this distribution are of first two sparse PC vectors $\mathbf{v}_1$ and $\mathbf{v}_2$.

Four series of experiments, each involving 1000 sets of data generated from $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, are utilized, with sample sizes 500, 1000, 2000, 5000, respectively. For each experiment, the first two PCs, $\hat{\mathbf{v}}_1$ and $\hat{\mathbf{v}}_2$, are calculated by a sparse PCA method and then if both $|\hat{\mathbf{v}}_1^T \mathbf{v}_1| \geq 0.99$ and $|\hat{\mathbf{v}}_2^T \mathbf{v}_2| \geq 0.99$ are satisfied, the method is considered as a success. The proposed ReDaC method, together with the conventional PCA and 12 current sparse PCA methods, including SPCA [5], DSPCA [6], PathSPCA [16], sPCA-rSVD$_{l_0}$, sPCA-rSVD$_{l_1}$, sPCA-rSVD$_{SCAD}$ [7], EMPCA [9], GPower$_{l_0}$, GPower$_{l_1}$, GPower$_{l_{0,m}}$, GPower$_{l_{1,m}}$ [8] and ALSPCA [15], have been implemented, and the success times for four series of experiments have been recorded and summarized, respectively. The results are listed in Table 1.

Table 1: Comparison of success times of PCA and different sparse PCA methods in synthetic toy experiments with sample size varying. The best results are highlighted in bold.

	$n = 500$	$n = 1000$	$n = 2000$	$n = 5000$
PCA	0	0	0	0
SPCA	566	673	756	839
DSPCA	211	203	138	62
PathSPCA	189	187	186	171
sPCA-rSVD$_{l_0}$	646	702	797	906
sPCA-rSVD$_{l_1}$	649	715	806	909
sPCA-rSVD$_{\mathrm{SCAD}}$	649	715	806	909
EMPCA	649	715	806	909
GPower$_{l_0}$	155	154	155	139
GPower$_{l_1}$	122	127	126	126
GPower$_{l_{0,m}}$	91	76	71	16
GPower$_{l_{1,m}}$	90	92	88	82
ALSPCA	669	**749**	826	927
ReDaC	**676**	748	**827**	**928**

The advantage of the proposed ReDaC algorithm can be easily observed from Table 1. In specific, our method always attains the highest or second highest success times (in the size 1000 case, 1 less than ALSPCA) as compared with the other utilized methods in all of the four series of experiments. Considering that the ALSPCA method, which is the only comparable method in these experiments, utilizes strict constraints on the orthogonality of output PCs while the ReDaC method does not utilize any prior ground-truth information of data, the capability of the proposed method on sparse PCA calculation can be more prominently verified.

3.2. Experiments on real data

In this section, we further evaluate the performance of the proposed ReDaC method on two real data sets, including the pitprops and colon data. Two quantitative criteria are employed for performance assessment. They are designed in the viewpoints of reconstruction-error-minimization and data-variance-maximization, respectively, just corresponding to the original formulations (4) and (3) for sparse PCA problem.

- *Reconstruction-error-minimization criterion: RRE.* Once sparse PC loading matrix $\mathbf{V}$ is obtained by a method, the input data can then be reconstructed by $\hat{\mathbf{X}} = \hat{\mathbf{U}}\mathbf{V}^T$, where $\hat{\mathbf{U}} = \mathbf{X}\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}$, attained by

the least square method. Then the relative reconstruction error (RRE) can be calculated by

$$\text{RRE} = \frac{\|\mathbf{X} - \hat{\mathbf{X}}\|_F}{\|\mathbf{X}\|_F},$$

to assess the performance of the utilized method in data reconstruction point of view.

- *Data-variance-maximization criterion: PEV.* After attaining the sparse PC loading matrix $\mathbf{V}$, the input data can then be reconstructed by $\hat{\mathbf{X}} = \mathbf{X}\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T$, as aforementioned. And thus the variance of the reconstructed data can be computed by $\text{Tr}(\frac{1}{n}\hat{\mathbf{X}}^T\hat{\mathbf{X}})$. The percentage of explained variance (PEV, [7]) of the reconstructed data from the original one can then be calculated by

$$\text{PEV} = \frac{\text{Tr}(\frac{1}{n}\hat{\mathbf{X}}^T\hat{\mathbf{X}})}{\text{Tr}(\frac{1}{n}\mathbf{X}^T\mathbf{X})} \times 100\% = \frac{\text{Tr}(\hat{\mathbf{X}}^T\hat{\mathbf{X}})}{\text{Tr}(\mathbf{X}^T\mathbf{X})} \times 100\%,$$

to evaluate the performance of the utilized method in data variance point of view.

3.2.1. Pitprops data

The pitprops data set, consisting of 180 observations and 13 measured variables, was first introduced by Jeffers [31] to show the difficulty of interpreting PCs. This data set is one of the most commonly utilized examples for sparse PCA evaluation, and thus is also employed to testify the effectiveness of the proposed ReDaC method. The comparison methods include SPCA [5], DSPCA [6], PathSPCA [16], sPCA-rSVD$_{l_0}$, sPCA-rSVD$_{l_1}$, sPCA-rSVD$_{SCAD}$ [7], EMPCA [9], GPower$_{l_0}$, GPower$_{l_1}$, GPower$_{l_{0,m}}$, GPower$_{l_{1,m}}$ [8] and ALSPCA [15]. For each utilized method, 6 sparse PCs are extracted from the pitprops data, with different cardinality settings: 8-5-6-2-3-2 (altogether 26 nonzero elements), 7-4-4-1-1-1 (altogether 18 nonzero elements, as set in [5]) and 7-2-3-1-1-1 (altogether 15 nonzero elements, as set in [6]), respectively. In each experiment, both the RRE and PEV values, as defined above, are calculated, and the results are summarized in Table 2. Figure 1 further shows the the RRE and PEV curves attained by different sparse PCA methods in all experiments for more illumination. It should be noted that the GPower$_{l_0,m}$, GPower$_{l_1,m}$ and ALSPCA methods employ the block methodology, as introduced in the introduction of the paper, and calculate

Table 2: Performance comparison of different sparse PCA methods on pitprops data with different cardinality settings. The best result in each experiment is highlighted in bold.

	8-5-6-2-3-2(26)		7-4-4-1-1-1(18)		7-2-3-1-1-1(15)	
	RRE	PEV	RRE	PEV	RRE	PEV
SPCA	0.4162	82.68%	0.4448	80.22%	0.4459	80.11%
DSPCA	0.4303	81.48%	0.4563	79.18%	0.4771	77.23%
PathSPCA	0.4080	83.35%	0.4660	80.11%	0.4457	80.13%
sPCA-rSVD$_{l_0}$	0.4139	82.87%	0.4376	80.85%	0.4701	77.90%
sPCA-rSVD$_{l_1}$	0.4314	81.39%	0.4427	80.40%	0.4664	78.25%
sPCA-rSVD$_{\text{SCAD}}$	0.4306	81.45%	0.4453	80.17%	0.4762	77.32%
EMPCA	0.4070	83.44%	0.4376	80.85%	0.4451	80.18%
GPower$_{l_0}$	0.4092	83.26%	0.4400	80.64%	0.4457	80.13%
GPower$_{l_1}$	0.4080	83.35%	0.4460	80.11%	0.4457	80.13%
GPower$_{l_{0,m}}$	0.4224	82.16%	0.5089	74.10%	0.4644	78.44%
GPower$_{l_{1,m}}$	0.4187	82.46%	0.4711	77.81%	0.4589	78.94%
ALSPCA	0.4168	82.63%	0.4396	80.67%	0.4537	79.42%
ReDaC	**0.4005**	**83.50%**	**0.4343**	**81.14%**	**0.4420**	**80.46%**

all sparse PCs at once while cannot sequentially derive different numbers of sparse PCs with preset cardinality settings. Thus the results of these methods reported in Table 2 are calculated with the total sparse PC cardinalities being 26, 18 and 15, respectively, and are not included in Figure 1.

It can be seen from Table 2 that under all cardinality settings of the first 6 PCs, the proposed ReDaC method always achieves the lowest RRE and highest PEV values among all the competing methods. This means that the ReDaC method is advantageous in both reconstruction-error-minimization and data-variance-maximization viewpoints. Furthermore, from Figure 1, it is easy to see the superiority of the ReDaC method. In specific, for different number of extracted sparse PC components, the proposed ReDaC method can always get the smallest RRE values and the largest PEV values, as compared with the other utilized sparse PCA methods, in the experiments. This further substantiates the effectiveness of the proposed ReDaC method in both reconstruction-error-minimization and data-variance-maximization views.

3.2.2. Colon data

The colon data set [32] consists of 62 tissue samples with the gene expression profiles of 2000 genes extracted from DNA micro-array data. This is a typical data set with high-dimension and low-sample-size property, and

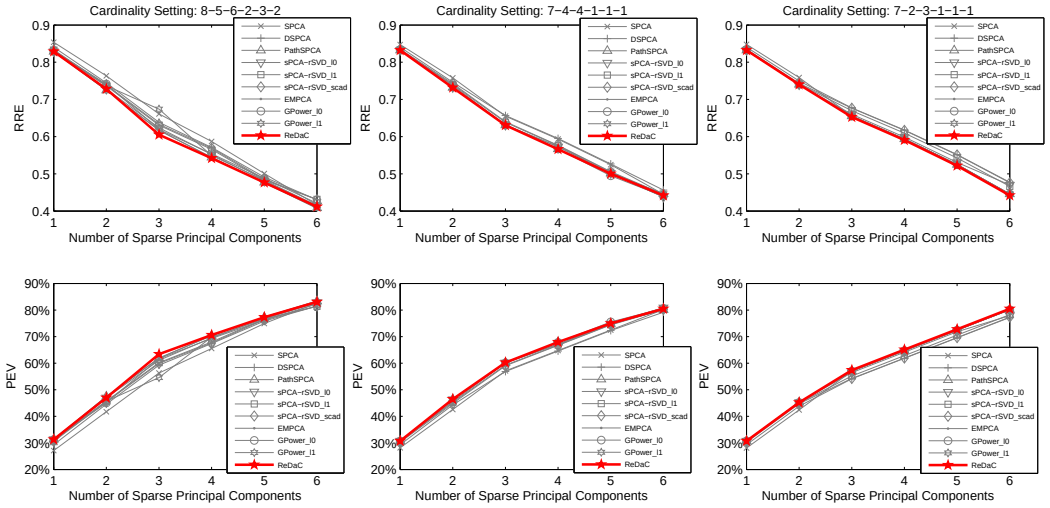

Figure 1: The tendency curves of RRE and PEV with respect to the number of extracted sparse PCs attained by different sparse PCA methods on pitprops data. Three cardinality settings for the extracted sparse PCs are utilized, including 8-5-6-2-3-2, 7-4-4-1-1-1 and 7-2-3-1-1-1.

is always employed by sparse methods for extracting interpretable information from high-dimensional genes. We thus adopt this data set for evaluation. In specific, 20 sparse PCs, each with 50 nonzero loadings, are calculated by different sparse PCA methods, including SPCA [5], PathSPCA [16], sPCA-rSVD$_{l_0}$, sPCA-rSVD$_{l_1}$, sPCA-rSVD$_{SCAD}$ [7], EMPCA [9], GPower$_{l_0}$, GPower$_{l_1}$, GPower$_{l_{0,m}}$, GPower$_{l_{1,m}}$ [8] and ALSPCA [15], respectively. Their performance is compared in Table 3 and Figure 2 in terms of RRE and PEV, respectively. It should be noted that the DSPCA method has also been tried, while cannot be terminated in a reasonable time in this experiment, and thus we omit its result in the table. Besides, we have carefully tuned the parameters of the GPower methods (including GPower$_{l_0}$, GPower$_{l_1}$, GPower$_{l_{0,m}}$ and GPower$_{l_{1,m}}$), and can get 20 sparse PCs with total cardinality around 1000, similar as the total nonzero elements number of the other utilized sparse PCA methods, while cannot get sparse PC loading sequences each with cardinality 50 as expected. The results are thus not demonstrated in Figure 2.

From Table 3, it is easy to see that the proposed ReDaC method achieves the lowest RRE and highest PEV values, as compared with the other 11 employed sparse PCA methods. Figure 2 further demonstrates that as the number of extracted sparse PCs increases, the advantage of the ReDaC method

Table 3: Performance comparison of different sparse PCA methods on colon data. The best results are highlighted in bold.

	SPCA	PathSPCA	sPCA-rSVD$_{l_0}$	sPCA-rSVD$_{l_1}$
RRE.	0.7892	0.5287	0.5236	0.5628
PEV.	37.72%	72.05%	72.58%	68.32%
	sPCA-rSVD$_{\text{SCAD}}$	EMPCA	GPower$_{l_0}$	GPower$_{l_1}$
RRE.	0.5723	0.5211	0.5042	0.5076
PEV.	67.25%	72.84%	74.56%	74.23%
	GPower$_{l_{0,m}}$	GPower$_{l_{1,m}}$	ALSPCA	ReDaC
RRE.	0.4870	0.4904	0.5917	**0.4737**
PEV.	76.29%	75.95%	64.99%	**77.56%**

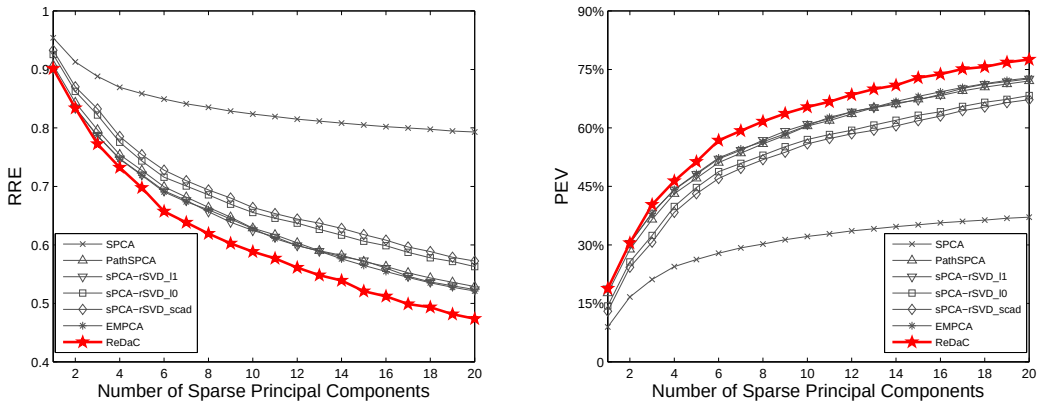

Figure 2: The tendency curves of RRE and PEV with respect to the number of extracted sparse PCs, each with cardinality 50, attained by different sparse PCA methods on colon data.

tends to be more dominant than other methods, with respect to both the RRE and PEV criteria. This further substantiates the effectiveness of the proposed method and implies its potential usefulness in applications with various interpretable components.

3.3. Nonnegative sparse PCA experiments

We further testify the performance of the proposed ReDaC method (Algorithm 2) in nonnegative sparse PC extraction. For comparison, two existing methods for nonnegative sparse PCA, NSPCA [29] and Nonnegative EMPCA (N-EMPCA, briefly) [9], are also employed.

Table 4: Performance comparison of success times attained by PCA, NSPCA, N-EMPCA and ReDaC on synthetic toy experiments with different sample sizes. The best results are highlighted in bold.

	$n = 500$	$n = 1000$	$n = 2000$	$n = 5000$
PCA	0	0	0	0
NSPCA	739	948	933	993
N-EMPCA	620	655	631	639
ReDaC	**835**	**949**	**978**	**1000**

3.3.1. Synthetic toy data

As the toy data utilized in Section 3.2, we also formulate a Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with mean $\mathbf{0}$ and covariance matrix $\mathbf{\Sigma} = \sum_{j=1}^{10} c_j \mathbf{v}_j \mathbf{v}_j^T \in \mathbb{R}^{10\times 10}$. Both the leading two eigenvectors of $\mathbf{\Sigma}$ are specified as nonnegative and sparse vectors as:

$$\mathbf{v}_1 = (0.474, 0, 0.158, 0, 0.316, 0, 0.791, 0, 0.158, 0)^T,$$
$$\mathbf{v}_2 = (0, 0.140, 0, 0.840, 0, 0.280, 0, 0.140, 0, 0.420)^T,$$

and the rest are then generated by applying Gram-Schmidt orthonormalization to 8 randomly valued 10-dimensional vectors. The 10 corresponding eigenvalues $(c_1, c_2, ..., c_{10})$ are preset as $(210, 190, 50, 50, 6, 5, 4, 3, 2, 1)$, respectively. Four series of experiments are designed, each with 1000 data sets generated from $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, with sample sizes 500, 1000, 2000 and 5000, respectively. For each experiment, the first two PCs are calculated by the conventional PCA, NSPCA, N-EMPCA and ReDaC methods, respectively. The success times, calculated in the similar way as introduced in Section 3.1.2, of each utilized method on each series of experiments are recorded, as listed in Table 4.

From Table 4, it is seen that the ReDaC method achieves the highest success rates in all experiments. The advantage of the proposed ReDaC method on nonnegative sparse PCA calculation, as compared with the other utilized methods, can thus been verified in these experiments.

3.3.2. Colon data

The colon data set is utilized again for nonnegative sparse PCA calculation. The NSPCA and N-EMPCA methods are adopted as the competing methods. Since the NSPCA method cannot directly pre-specify the cardinalities of the extracted sparse PCs, we thus first apply NSPCA on the colon

Table 5: Performance comparison of different nonnegative sparse PCA methods on colon data. The best results are highlighted in bold.

	NSPCA	N-EMPCA	ReDaC
RRE	0.3674	0.3399	**0.2706**
PEV	86.50%	88.45%	**92.68%**

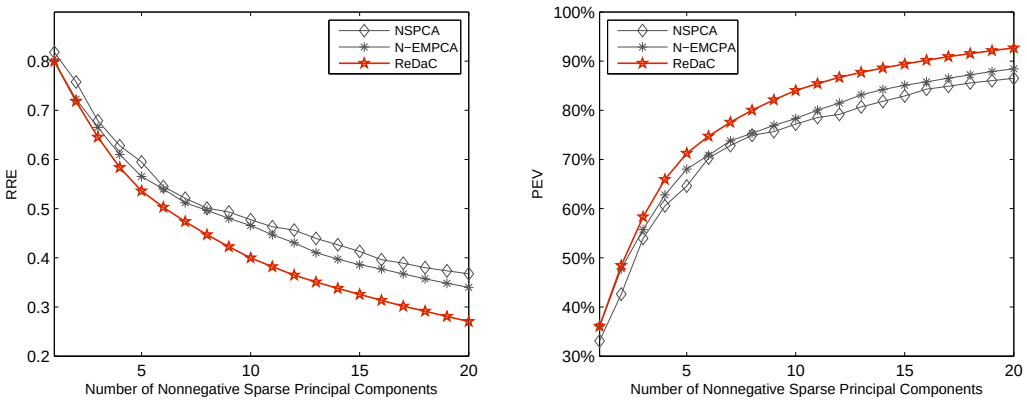

Figure 3: The tendency curves of RRE and PEV, with respect to the number of extracted nonnegative sparse PCs, attained by NSPCA, N-EMPCA and ReDaC on colon data.

data (with parameters $\alpha = 1 \times 10^6$ and $\beta = 1 \times 10^7$) and then use the cardinalities of the nonnegative sparse PCs attained by this method to preset the N-EMPCA and ReDaC methods for fair comparison. 20 sparse PCs are computed by the three methods, and the performance is compared in Table 5 and Figure 3, in terms of RRE and PEV, respectively.

Just as expected, it is evident that the proposed ReDaC method dominates in both RRE and PEV viewpoints. From Table 5, we can observe that our method achieves the lowest RRE and highest PEV on 20 extracted nonnegative sparse PCs than the other two utilized methods. Furthermore, Figure 3 shows that our method is advantageous, as compared with the other methods, for any preset number of extracted sparse PCs, and this advantage tends to be more significant as more sparse PCs are to be calculated. The effectiveness of the proposed method on nonnegative sparse PCA calculation can thus be further verified.

3.3.3. Application to face recognition

In this section, we introduce the performance of our method in face recognition problem [29]. The proposed ReDaC method, together with the

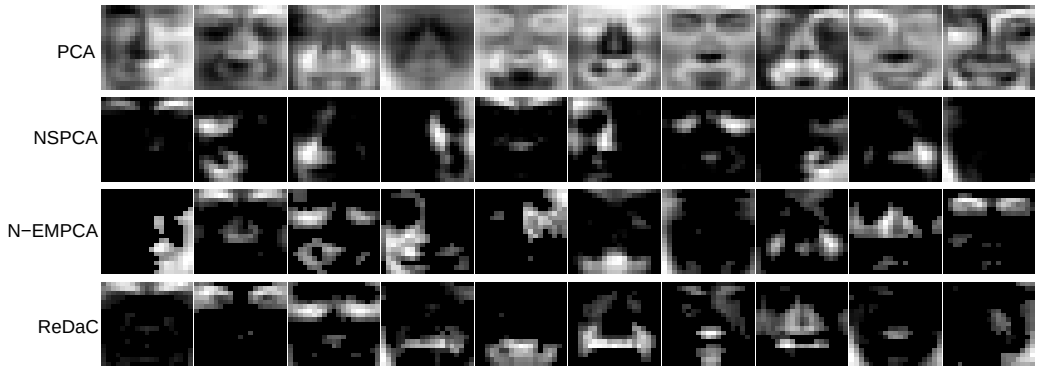

Figure 4: From top row to bottom row: 10 PCs or nonnegative sparse PCs extracted by PCA, NSPCA, N-EMPCA and ReDaC, respectively.

Table 6: Performance comparison of different nonnegative sparse PCA methods on MIT CBCL Face Dataset #1. The best results are highlighted in bold.

	NSPCA	N-EMPCA	ReDaC
RRE	0.6993	0.6912	**0.6606**
PEV	51.10%	52.22%	**56.36%**

conventional PCA, NSPCA and N-EMPCA methods, have been applied to this problem and their performance is compared in this application. The employed data set is the MIT CBCL Face Dataset #1, downloaded from "http://cbcl.mit.edu/software-datasets/FaceData2.html". This data set consists of 2429 aligned face images and 4548 non-face images, each with resolution 19×19. For each of the four utilized methods, 10 PC loading vectors are computed on face images, as shown in Figure 4, respectively. For easy comparison, we also list the RRE and PEV values of three nonnegative sparse PCA methods in Table 6.

As depicted in Figure 4, the nonnegative sparse PCs obtained by the ReDaC method more clearly exhibit the interpretable features underlying faces, as compared with the other utilized methods, e.g. the first five PCs calculated from our method clearly demonstrate the eyebrows, eyes, cheeks, mouth and chin of faces, respectively. The advantage of the proposed method can further be verified quantitatively by its smallest RRE and largest PEV values, among all employed methods, in the experiment, as shown in Table 6. The effectiveness of the ReDaC method can thus be substantiated.

To further show the usefulness of the proposed method, we apply it to face

Table 7: Performance comparison of the classification accuracy obtained by different non-negative sparse PCA methods. The best results are highlighted in bold.

	Face (%)	Non-face (%)	Total (%)
LR	96.71	93.57	94.47
PCA + LR	96.64	94.17	94.88
NSPCA + LR	94.89	93.49	93.89
N-EMPCA + LR	96.71	94.39	95.06
ReDaC + LR	**96.78**	**94.46**	**95.84**

classification under this data set as follows. First we randomly choose 1000 face images and 1000 non-face images from MIT CBCL Face Dataset #1, and take them as the training data and the rest images as testing data. We then extract 10 PCs by utilizing the PCA, NSPCA, N-EMPCA and ReDaC methods to the training set, respectively. By projecting the training data onto the corresponding 10 PCs obtained by these four methods, respectively, and then fitting the linear Logistic Regression (LR) [33] model on these dimension-reduced data (10-dimensional), we can get a classifier for testing. The classification accuracy of the classifier so obtained on the testing data is then computed, and the results are reported in Table 7. In the table, the classification accuracy attained by directly fitting the LR model on the original training data and testing on the original testing data is also listed for easy comparison.

From Table 7, it is clear that the proposed ReDaC method attains the best performance among all implemented methods, most accurately recognizing both the face images and the non-face images from the testing data. This further implies the potential usefulness of the proposed method in real applications.

4. Conclusion

In this paper we have proposed a novel recursive divide-and-conquer method (ReDaC) for sparse PCA problem. The main methodology of the proposed method is to decompose the original large sparse PCA problem into a series of small sub-problems. We have proved that each of these decomposed sub-problems has a closed-form global solution and can thus be easily solved. By recursively solving these small sub-problems, the original sparse PCA problem can always be very effectively resolved. We have also shown that the new method converges to a stationary point of the problem, and can

be easily extended to other sparse PCA problems with certain constraints, such as nonnegative sparse PCA problem. The extensive experimental results have validated that our method outperforms current sparse PCA methods in both reconstruction-error-minimization and data-variance-maximization viewpoints.

There are many interesting investigations still worthy to be further explored. For example, when we reformat the square L_2-norm error of the sparse PCA model as the L_1-norm one, the robustness of the model can always be improved for heavy noise or outlier cases, while the model is correspondingly more difficult to solve. By adopting the similar ReDaC methodology, however, the problem can be decomposed into a series of much simpler sub-problems, which are expected to be much more easily solved than the original model. Besides, although we have proved the convergence of the ReDaC method, we do not know how far the result is from the global optimum of the problem. Stochastic global optimization techniques, such as simulated annealing and evolution computation methods, may be combined with the proposed method to further improve its performance. Also, more real applications of the proposed method are under our current research.

[1] I. T. Jolliffe, Principal Component Analysis, 2nd Edition, Springer, New York, 2002.

[2] I. T. Jolliffe, Rotation of principal components - choice of normalization constraints, Journal of Applied Statistics 22 (1) (1995) 29–35.

[3] J. Cadima, I. T. Jolliffe, Loadings and correlations in the interpretation of principal components, Journal of Applied Statistics 22 (2) (1995) 203–214.

[4] I. T. Jolliffe, N. T. Trendafilov, M. Uddin, A modified principal component technique based on the lasso, Journal of Computational and Graphical Statistics 12 (3) (2003) 531–547.

[5] H. Zou, T. Hastie, R. Tibshirani, Sparse principal component analysis, Journal of Computational and Graphical Statistics 15 (2) (2006) 265–286.

[6] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, G. Lanckriet, A direct formulation for sparse pca using semidefinite programming, Siam Review 49 (3) (2007) 434–448.

[7] H. P. Shen, J. Huang, Sparse principal component analysis via regularized low rank matrix approximation, Journal of Multivariate Analysis 99 (6) (2008) 1015–1034.

[8] M. Journée, Y. Nesterov, P. Richtarik, R. Sepulchre, Generalized power method for sparse principal component analysis, Journal of Machine Learning Research 11 (2010) 517–553.

[9] C. Sigg, J. Buhmann, Expectation-maximization for sparse and non-negative pca, in: Proceedings of the 25th International Conference on Machine Learning, ACM, 2008, pp. 960–967.

[10] Y. Guan, J. Dy, Sparse probabilistic principal component analysis, in: Proceedings of 12th International Conference on Artificial Intelligence and Statistics, 2009, pp. 185–192.

[11] K. Sharp, M. Rattray, Dense message passing for sparse principal component analysis, in: Proceedings of 13th International Conference on Artificial Intelligence and Statistics, 2010, pp. 725–732.

[12] C. Archambeau, F. Bach, Sparse probabilistic projections, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems 21, MIT Press, Cambridge, MA, 2009, pp. 73–80.

[13] B. Sriperumbudur, D. Torres, G. Lanckriet, Sparse eigen methods by dc programming, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 831–838.

[14] B. K. Sriperumbudur, D. A. Torres, G. Lanckriet, A majorization-minimization approach to the sparse generalized eigenvalue problem, Machine Learning 85 (1-2) (2011) 3–39.

[15] Z. Lu, Y. Zhang, An augmented lagrangian approach for sparse principal component analysis, Mathematical Programming 135 (1-2) (2012) 149–193.

[16] A. d'Aspremont, F. Bach, L. Ghaoui, Full regularization path for sparse principal component analysis, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 177–184.

[17] B. Moghaddam, Y. Weiss, S. Avidan, Spectral bounds for sparse pca: Exact and greedy algorithms, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems 18, MIT Press, Cambridge, MA, 2006, pp. 915–922.

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

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

[20] A. Farcomeni, An exact approach to sparse principal component analysis, Computational Statistics 24 (4) (2009) 583–604.

[21] Y. Zhang, L. E. Ghaoui, Large-scale sparse principal component analysis with application to text data, in: J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, K. Weinberger (Eds.), Advances in Neural Information Processing Systems 24, MIT Press, Cambridge, MA, 2011, pp. 532–539.

[22] D. Y. Meng, Q. Zhao, Z. B. Xu, Improve robustness of sparse pca by l_1-norm maximization, Pattern Recognition 45 (1) (2012) 487–497.

[23] Y. Wang, Q. Wu, Sparse pca by iterative elimination algorithm, Advances in Computational Mathematics 36 (1) (2012) 137–151.

[24] L. Mackey, Deflation methods for sparse pca, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems 21, MIT Press, Cambridge, MA, 2009, pp. 1017–1024.

[25] H. Hotelling, Analysis of a complex of statistical variables into principal components, Journal of Educational Psychology 24 (1933) 417–441.

[26] K. Pearson, On lines and planes of closest fit to systems of points in space, Philosophical Magazine 2 (7-12) (1901) 559–572.

[27] D. Knuth, The Art of Computer Programming, Addison-Wesley, Reading, MA, 1973.

[28] P. Tseng, Convergence of a block coordinate descent method for nondifferentiable minimization, Journal of Optimization Theory and Applications 109 (3) (2001) 475–494.

[29] R. Zass, A. Shashua, Nonnegative sparse pca, in: B. Schölkopf, J. Platt, T. Hoffman (Eds.), Advances in Neural Information Processing Systems 19, MIT Press, Cambridge, MA, 2007, pp. 1561–1568.

[30] A. Cichocki, R. Zdunek, A. Phan, S. Amari, Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation, Wiley, 2009.

[31] J. Jeffers, Two case studies in the application of principal component analysis, Applied Statistics 16 (1967) 225–236.

[32] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, A. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Cell Biology 96 (12) (1999) 6745–6750.

[33] J. Friedman, T. Hastie, R. Tibshirani, The Elements of Statistical Learning, Springer, 2001.

Appendix A. Proof of Theorem 2

In the following, we denote $\mathbf{w} = \mathbf{E}^T\mathbf{u}$, and $hard_\lambda(\mathbf{w})$ the hard thresholding function, whose i-th element corresponds to $I(|w_i| \geq \lambda)w_i$, where w_i is the i-th element of $\mathbf{w}$ and $I(x)$ (equals 1 if x is ture, and 0 otherwise) is the indicator function

Theorem 2. The optimal solution of

$$\max_{\mathbf{v}} \ \mathbf{w}^T\mathbf{v} \quad s.t. \ \ \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_0 \leq t,$$

is given by:

$$\mathbf{v}_0^*(\mathbf{w}, t) = \begin{cases} \phi, & t < 1, \\ \frac{hard_{\theta_k}(\mathbf{w})}{\|hard_{\theta_k}(\mathbf{w})\|_2}, & k \leq t < k+1 \ (k = 1, 2, \ldots, d-1), \\ \frac{\mathbf{w}}{\|\mathbf{w}\|_2} & t \geq d. \end{cases}$$

where θ_k denotes the k-th largest element of $|\mathbf{w}|$.

Proof. In case of $t < 1$, the feasible region of the optimization problem is empty, and thus the solution of the problem does not exist.

In case of $t \geq d$, the problem is equivalent to

$$\max_{\mathbf{v}} \mathbf{w}^T\mathbf{v} \quad s.t. \ \ \mathbf{v}^T\mathbf{v} = 1.$$

It is then easy to attain the optimum of the problem $\mathbf{v}^* = \frac{\mathbf{w}}{\|\mathbf{w}\|_2}$.

In case of $k \leq t < k+1$ $(k = 1, 2, \ldots, d-1)$, the optimum $\mathbf{v}^*$ of the problem is parallel to $\mathbf{w}$ on the k-dimensional subspace where the first k largest absolute value of $\mathbf{w}$ are located. Also due to the constraint that $\mathbf{v}^T\mathbf{v} = 1$, it is then easy to deduce that the optimal solution of the optimization problem is $\frac{hard_t(\mathbf{w})}{\|hard_t(\mathbf{w})\|_2}$.

The proof is completed.

Appendix B. Proof of Theorem 3

We denote $(I_1, I_2, \ldots, I_d)$ the permutation of $(1, 2, \ldots, d)$ based on the ascending order of $|\mathbf{w}| = (|w_1|, |w_2|, \ldots, |w_d|)^T$, $soft_\lambda(\mathbf{w})$ the soft thresholding function $sign(\mathbf{w})(|\mathbf{w}| - \lambda)_+$, $f_\mathbf{w}(\lambda) = \frac{soft_\lambda(\mathbf{w})}{\|soft_\lambda(\mathbf{w})\|_2}$ and $g_\mathbf{w}(\lambda) = \mathbf{w}^T f_\mathbf{w}(\lambda)$ throughout the following.

Theorem 3. The optimal solution of

$$\max_{\mathbf{v}} \ \mathbf{w}^T\mathbf{v} \quad s.t. \ \mathbf{v}^T\mathbf{v} = 1, \quad \|\mathbf{v}\|_1 \le t,$$

is given by:

$$\mathbf{v}_1^*(\mathbf{w}) = \begin{cases} \phi, & t < 1, \\ f_\mathbf{w}(\lambda_k), & t \in [\|f_\mathbf{w}(|w_{I_k}|)\|_1, \|f_\mathbf{w}(|w_{I_{k-1}}|)\|_1) \ \ (k = 2, 3, \ldots, d-1), \\ f_\mathbf{w}(\lambda_1), & t \in [\|f_\mathbf{w}(|w_{I_1}|)\|_1, \sqrt{d}), \\ f_\mathbf{w}(0), & t \ge \sqrt{d}, \end{cases}$$

where for $k = 1, 2, \ldots, d-1$,

$$\lambda_k = \frac{(m - t^2)(\sum_{i=1}^m a_i) - \sqrt{t^2(m-t^2)(m\sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2)}}{m(m-t^2)},$$

where $(a_1, a_2, \ldots, a_m) = (|w_{I_k}|, |w_{I_{k+1}}|, \ldots, |w_{I_d}|)$, $m = d - k + 1$.

Proof. For any $\mathbf{v}$ located in the feasible region of (12), it holds that

$$\sqrt{d} = \sqrt{d\mathbf{v}^T\mathbf{v}} \ge \|\mathbf{v}\|_1 \ge \sqrt{\mathbf{v}^T\mathbf{v}} = 1.$$

We thus have that if $t < 1$, then the optimal solution $\mathbf{v}^*$ does not exist since the feasible region of the optimization problem (9) is empty.

If $t \ge \sqrt{d}$, it is easy to see that (12) is equivalent to

$$\max_{\mathbf{v}} \ \mathbf{w}^T\mathbf{v} \ \ s.t. \ \mathbf{v}^T\mathbf{v} = 1,$$

and its optimum is

$$\mathbf{v}^* = \frac{\mathbf{w}}{\|\mathbf{w}\|_2} = f_\mathbf{w}(0).$$

We then discuss the case when $t \in [1, \sqrt{d})$. Firstly we deduce the monotonic decreasing property of $h_\mathbf{w}(\lambda) = \|f_\mathbf{w}(\lambda)\|_1 = \left\| \frac{soft_\lambda(\mathbf{w})}{\|soft_\lambda(\mathbf{w})\|_2} \right\|_1$ and $g_\mathbf{w}(\lambda) = \mathbf{w}^T f_\mathbf{w}(\lambda)$ in $\lambda \in (-\infty, |w_{I_d}|)$ by the following lemmas.

Lemma 1. *$h_{\mathbf{w}}(\lambda)$ is monotonically decreasing with respect to λ in $(-\infty, |w_{I_d}|)$.*

PROOF. First, we prove that $h_{\mathbf{w}}(\lambda)$ is monototically decreasing with $\lambda \in [|w_{I_{k-1}}|, |w_{I_k}|)$, $k = 2, 3, \ldots, d$ and $(-\infty, |w_{I_1}|)$.

It is easy to see that for $\lambda \in [|w_{I_{k-1}}|, |w_{I_k}|), k = 2, 3, \ldots, d$ and $(-\infty, |w_{I_1}|)$,

$$h_{\mathbf{w}}(\lambda) = \frac{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)}{\sqrt{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2}}.$$

Then we have

$$\begin{aligned} h'_{\mathbf{w}}(\lambda) &= \frac{-(d-k+1)\sqrt{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2} + \frac{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)}{\sqrt{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2}} \sum_{i=k}^{d}(|w_{I_i}| - \lambda)}{\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2} \\ &= \left(\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2\right)^{-3/2} \left(-(d-k+1)\sum_{i=k}^{d}(|w_{I_i}| - \lambda)^2 + \left(\sum_{i=k}^{d}(|w_{I_i}| - \lambda)\right)^2\right). \end{aligned}$$

It is known that for any number sequence $s_1, s_2, \ldots, s_n$, it holds that

$$\left(\sum_{i=1}^{n} s_i\right)^2 \leq n \sum_{i=1}^{n} s_i^2.$$

Thus we have

$$h'_{\mathbf{w}}(\lambda) \leq \mathbf{0}$$

for $\lambda \in [|w_{I_{k-1}}|, |w_{I_k}|)$, $k = 2, 3, \ldots, d$ and $(-\infty, |w_{I_1}|)$. Since $h_{\mathbf{w}}(\lambda)$ is obviously a continuous function in $(-\infty, |w_{I_d}|)$, it can be easily deduced that $h_{\mathbf{w}}(\lambda)$ is monotonically decreasing in the entire set $(-\infty, |w_{I_d}|)$ with respect to λ.

The Proof is completed.

Based on Lemma 1, It is easy to deduce that the range of $h_{\mathbf{w}}(\lambda)$ for $\lambda \in (-\infty, |w_{I_d}|)$ is $[1, \sqrt{d})$, since $\lim_{\lambda \to -\infty} h_{\mathbf{w}}(\lambda) = \sqrt{d}$ and $h_{\mathbf{w}}(\lambda) = 1$ for $\lambda \in [|w_{I_{d-1}}|, |w_{I_d}|)$.

The following lemma shows the monotonic decreasing property of $g_{\mathbf{w}}(\lambda)$.

Lemma 2. *$g_{\mathbf{w}}(\lambda)$ is monotonically decreasing with respect to $\lambda \in (-\infty, |w_{I_d}|)$.*

PROOF. Please see [22] for the proof.

The next lemma proves that the optimal solution $\mathbf{v}^*$ can be expressed as $f_{\mathbf{w}}(\lambda^*)$.

Lemma 3. *The optimal solution of (12) is of the expression* $\mathbf{v}^* = f_{\mathbf{w}}(\lambda^*)$ *for* $t \in [1, \sqrt{d})$ *on some* $\lambda^* \in (-\infty, |w_{I_d}|)$.

PROOF. Please see [19, 22] for the proof.

Lemmas 1-3 imply that the optimal solution of (12) is attained at λ^* where $\|f_{\mathbf{w}}(\lambda^*)\|_1 = t$ holds. The next lamma presents the closed-form solution of this equation.

Lemma 4. *The solutuion of* $\|f_{\mathbf{w}}(\lambda)\|_1 = t$ *for* $t \in [\|f_{\mathbf{w}}(|w_{I_k}|)\|_1, \|f_{\mathbf{w}}(|w_{I_{k-1}}|)\|_1)$, $(k = 2, 3, \ldots, d-1)$, *or* $t \in [\|f_{\mathbf{w}}(|w_{I_1}|)\|_1, \sqrt{d})$ *is*

$$\lambda_k = \frac{(m-t^2)(\sum_{i=1}^m a_i) - \sqrt{t^2(m-t^2)(m\sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2)}}{m(m-t^2)},$$

where $(a_1, a_2, \ldots, a_m) = (|w_{I_k}|, |w_{I_{k+1}}|, \ldots, |w_{I_d}|)$ *and* $m = d - k + 1$.

PROOF. Let's transform the equation

$$\|f_{\mathbf{w}}(\lambda)\|_1 = \frac{\sum_{i=k}^d (|w_{I_i}| - \lambda)}{\sqrt{\sum_{i=k}^d (|w_{I_i}| - \lambda)^2}} = \frac{\sum_{i=1}^m (a_i - \lambda)}{\sqrt{\sum_{i=1}^m (a_i - \lambda)^2}} = t \tag{19}$$

as the following expression

$$(\sum_{i=1}^m a_i - m\lambda)^2 = t^2 \sum_{i=1}^m (a_i - \lambda)^2.$$

Then we can get the quadratic equation with respect to λ as:

$$m(m-t^2)\lambda^2 - 2(m-t^2)(\sum_{i=1}^m a_i)\lambda + (\sum_{i=1}^m a_i)^2 - t^2 \sum_{i=1}^m a_i^2 = 0. \tag{20}$$

We first claim that $t^2 < m$ for $t \in [\|f_{\mathbf{w}}(|w_{I_k}|)\|_1, \|f_{\mathbf{w}}(|w_{I_{k-1}}|)\|_1)$, $k = 2, 3, \ldots, d-1$, or $t \in [\|f_{\mathbf{w}}(|w_{I_1}|)\|_1, \sqrt{d})$. In fact, by the definition of $f_{\mathbf{w}}(\lambda)$, we have that

$$\begin{aligned} t < \|f_{\mathbf{w}}(|w_{I_{k-1}}|)\|_1 &= \frac{\sum_{i=1}^m (a_i - |w_{I_{k-1}}|)}{\sqrt{\sum_{i=1}^m (a_i - |w_{I_{k-1}}|)^2}} \\ &\leq \left(m \sum_{i=1}^m \left(\frac{(a_i - |w_{I_{k-1}}|)}{\sqrt{\sum_{i=1}^m (a_i - |w_{I_{k-1}}|)^2}} \right)^2 \right)^{\frac{1}{2}} \\ &= \sqrt{m}, \end{aligned}$$

for $t \in [\|f_{\mathbf{w}}(|w_{I_k}|)\|_1, \|f_{\mathbf{w}}(|w_{I_{k-1}}|)\|_1)$, $k = 2, \ldots, d-1$, and

$$\begin{aligned} t < \|f_{\mathbf{w}}(|\sqrt{d}|)\|_1 &= \frac{\sum_{i=1}^d (a_i - |\sqrt{d}|)}{\sqrt{\sum_{i=1}^d (a_i - |\sqrt{d}|)^2}} \\ &\leq \left(d \sum_{i=1}^d \left(\frac{(a_i - |\sqrt{d}|)}{\sqrt{\sum_{i=1}^d (a_i - |\sqrt{d}|)^2}} \right)^2 \right)^{\frac{1}{2}} \\ &= \sqrt{d} = \sqrt{m}, \end{aligned}$$

for $t \in [\|f_{\mathbf{w}}(|w_{I_1}|)\|_1, \sqrt{d})$. Then it can be seen that the discriminant of equation (20)

$$\Delta = t^2(m - t^2)(m \sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2) \geq 0,$$

using the fact that $(\sum_{i=1}^m a_i)^2 \leq m \sum_{i=1}^m a_i^2$. Therefore, the solutions of equation (20) can be expressed as

$$\lambda = \frac{(m - t^2)(\sum_{i=1}^m a_i) \pm \sqrt{t^2(m - t^2)(m \sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2)}}{m(m - t^2)}.$$

It holds that

$$\begin{aligned}
\lambda^+ &= \frac{(m-t^2)(\sum_{i=1}^m a_i) + \sqrt{t^2(m-t^2)(m\sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2)}}{m(m-t^2)} \\
&\geq \frac{(m-t^2)(\sum_{i=1}^m a_i)}{m(m-t^2)} \\
&= \frac{\sum_{i=1}^m a_i}{m} (= \frac{\sum_{i=k}^d |w_{I_i}|}{d-k+1}) \\
&\geq |w_{I_k}|.
\end{aligned}$$

If $\lambda^+ > |w_{I_k}|$, since $\lambda \leq |w_{I_k}|$ required by equation (19), then

$$\lambda_k = \lambda^- = \frac{(m-t^2)(\sum_{i=1}^m a_i) - \sqrt{t^2(m-t^2)(m\sum_{i=1}^m a_i^2 - (\sum_{i=1}^m a_i)^2)}}{m(m-t^2)}.$$

Otherwise, if $\lambda^+ = |w_{I_k}|$, then it holds that $(\sum_{i=1}^m a_i)^2 = m\sum_{i=1}^m a_i^2$, which naturally leads to $\lambda_k = \lambda^+ = \lambda^-$.

The proof is then completed.

Based on the above Lemmas 1-4, the conclusion of Theorem 3 can then be obtained.

Appendix C. Proof of Theorem 5

Theorem 5. The global optimal solution to (18) is $\mathbf{v}_p^*((\mathbf{w})_+, t)$ $(p = 0, 1)$, where $\mathbf{w} = \mathbf{E}^T\mathbf{u}$, and $\mathbf{v}_0^*(\cdot,\cdot)$ and $\mathbf{v}_1^*(\cdot,\cdot)$ are defined in Theorem 2 and Theorem 3, respectively.

It is easy to prove this theorem based on the following lemma.

Lemma 5. *Assume that there is at least one element of* $\mathbf{w}$ *is positive, then the optimization problem*

$$(P1) \quad \max_{\mathbf{v}} \mathbf{w}^T\mathbf{v} \quad s.t. \;\; \mathbf{v}^T\mathbf{v} = 1, \;\; \|\mathbf{v}\|_p \leq t, \;\; \mathbf{v} \succeq 0,$$

can be equivalently soved by

$$(P2) \quad \max_{\mathbf{v}} (\mathbf{w})_+^T\mathbf{v} \quad s.t. \;\; \mathbf{v}^T\mathbf{v} = 1, \;\; \|\mathbf{v}\|_p \leq t,$$

where p *is* 0 *or* 1.

Proof. Denote the optimal solutions of $(P1)$ and $(P2)$ as $\mathbf{v1}$ and $\mathbf{v2}$, respectively.

First, we prove that $\mathbf{w}^T\mathbf{v1} \geq \mathbf{w}^T\mathbf{v2}$. Based on Theorem 2 and 3, the elements of $\mathbf{v2}$ are of the same signs (or zeros) with the corresponding ones of $(\mathbf{w})_+$. This means that $\mathbf{v2} \succeq 0$ natrually holds. That is, $\mathbf{v2}$ belongs to the feasible region of $(P1)$. Since $\mathbf{v1}$ is the optimum of $(P1)$, we have $\mathbf{w}^T\mathbf{v1} \geq \mathbf{w}^T\mathbf{v2}$.

Then we prove that $\mathbf{w}^T\mathbf{v1} \leq \mathbf{w}^T\mathbf{v2}$ through the following three steps.

(C1): The nonzero elements of $\mathbf{v1} = (v_1^{(1)}, v_2^{(1)}, ..., v_d^{(1)})$ lie on the positions where the nonnegative entries of $\mathbf{w}$ are located.

If all elements of $\mathbf{w}$ are nonnegative, then (C1) is evidently satisfied.

Otherwise, there is an element, denoted as the i-th element w_i of $\mathbf{w}$, is negative and the corresponding element, $v_i^{(1)}$, of $\mathbf{v1}$ is nonzero (i.e. positive). We further pick up a nonnegative element, denoted as w_j, from $\mathbf{w}$. Then we can construct a new d-dimensional vector $\widetilde{\mathbf{v}} = (\tilde{v}_1, \tilde{v}_2, ..., \tilde{v}_i)$ as

$$\tilde{v}_k = \begin{cases} 0, & k = i, \\ \sqrt{(v_i^{(1)})^2 + (v_j^{(1)})^2}, & k = j, \\ v_k^{(1)}, & k \neq i, j. \end{cases}$$

Then we have

$$\begin{aligned}\mathbf{w}^T\tilde{\mathbf{v}} &= \sum_k w_k v_k = w_j\sqrt{(v_i^{(1)})^2+(v_j^{(1)})^2} + \sum_{k\neq i,j} w_k v_k^{(1)} \\ &> w_i v_i^{(1)} + w_j v_j^{(1)} + \sum_{k\neq i,j} w_k v_k^{(1)} \\ &= \mathbf{w}^T\mathbf{v1}.\end{aligned}$$

We get the inequality by the fact that $w_j\sqrt{(v_i^{(1)})^2+(v_j^{(1)})^2} \geq w_j v_j^{(1)}$ and $0 > w_i v_i^{(1)}$. This is contradict to the fact that $\mathbf{v1}$ is the optimal solution of $(P1)$, noting that $\|\tilde{\mathbf{v}}\|_p \leq \|\mathbf{v}\|_p \leq t$.

The conclusion (C1) is then proved.

(C2): The nonzero elements of $\mathbf{v2}$ $=(v_1^{(2)}, v_2^{(2)}, ..., v_d^{(2)})$ lie on the positions where the nonzero entries of $(\mathbf{w})_+$ are located.

Denote $(\mathbf{w})_+ = (w_1^+, w_2^+, ..., w_d^+)$. If all elements of $(\mathbf{w})_+$ are positive, then (C2) is evidently satisfied.

Otherwise, let w_i^+ be a zero element of $\mathbf{w}$ and the corresponding element, $v_i^{(2)}$, of $\mathbf{v2}$ is nonzero, and let w_j^+ be a positive element of $\mathbf{w}$. Then we can construct a new d-dimensional vector $\overline{\mathbf{v}} = (\overline{v}_1, \overline{v}_2, ..., \overline{v}_i)$ as

$$\overline{v}_k = \begin{cases} 0, & k = i, \\ \sqrt{(v_i^{(2)})^2+(v_j^{(2)})^2}, & k = j, \\ v_k^{(2)}, & k \neq i, j. \end{cases}$$

Then we have

$$\begin{aligned}(\mathbf{w})_+^T\overline{\mathbf{v}} &= \sum_k w_k^+\overline{v}_k = w_j^+\sqrt{(v_i^{(2)})^2+(v_j^{(2)})^2} + \sum_{k\neq i,j} w_k^+ v_k^{(2)} \\ &> w_i^+ v_i^{(2)} + w_j^+ v_j^{(2)} + \sum_{k\neq i,j} w_k^+ v_k^{(1)} \\ &= (\mathbf{w})_+^T\mathbf{v}2.\end{aligned}$$

We get the first inequality by the fact that $w_j^+\sqrt{(v_i^{(2)})^2+(v_j^{(2)})^2} > w_j^+ v_j^{(2)}$ and $0 = w_i^+ v_i^{(2)}$. This is contradict to the fact that $\mathbf{v2}$ is the optimal solution of $(P2)$, noting that $\|\tilde{\mathbf{v}}\|_p \leq \|\mathbf{v}\|_p \leq t$.

The conclusion (C2) is then proved.

(C3): We can then prove that $\mathbf{w}^T\mathbf{v1} \leq \mathbf{w}^T\mathbf{v2}$ based on the conclusions (C1) and (C2) as follows:

$$\mathbf{w}^T\mathbf{v1} = (\mathbf{w})_+^T\mathbf{v1} \leq (\mathbf{w})_+^T\mathbf{v2} = \mathbf{w}^T\mathbf{v2}.$$

In the above equation, the first equality is conducted by (C1), the second inequality is based on the fact that $\mathbf{v2}$ is the optimal solution of $(P2)$, and the third equality is followed by (C2).

Thus it holds that $\mathbf{w}^T\mathbf{v1} = \mathbf{w}^T\mathbf{v2}$. This implies that the optimization problem $(P1)$ can be equivalently solved by $(P2)$.