Exact and Approximation Algorithms for Sparse PCA

Yongchun Li
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA 30332, ycli@gatech.edu

Weijun Xie
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Tech, Atlanta, GA 30332, wxie@gatech.edu

Sparse Principal Component Analysis (SPCA) is designed to enhance the interpretability of traditional Principal Component Analysis (PCA) by optimally selecting a subset of features that comprise the first principal component. Given the NP-hard nature of SPCA, most current approaches resort to approximate solutions, typically achieved through tractable semidefinite programs (SDPs) or heuristic methods. To solve SPCA to optimality, we propose two exact mixed-integer SDPs (MISDPs) and an arbitrarily equivalent mixed-integer linear program (MILP). The MISDPs allow us to design an effective branch-and-cut algorithm with closed-form cuts that do not need to solve dual problems. For the proposed mixed-integer formulations, we further derive the theoretical optimality gaps of their continuous relaxations. Besides, we apply the greedy and local search algorithms to solving SPCA and derive their first-known approximation ratios. Our numerical experiments reveal that the exact methods we developed can efficiently find optimal solutions for datasets containing hundreds of features. Furthermore, our approximation algorithms demonstrate both scalability and near-optimal performance when benchmarked on larger datasets, specifically those with thousands of features.

Key words: Sparse PCA, Mixed-Integer Programming, Semidefinite Programming, Greedy, Local Search

1. Introduction

This paper studies the sparse principal component analysis (SPCA) problem:

$$\text{(SPCA)} \quad w^* := \max_{\boldsymbol{x}\in\mathbb{R}^n} \left\{\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} : \|\boldsymbol{x}\|_2 = 1, \|\boldsymbol{x}\|_0 \le k\right\}, \tag{1}$$

where $\boldsymbol{A}$ is the sample covariance matrix out of a dataset with n features and thus is positive semidefinite, the zero-norm $\|\boldsymbol{x}\|_0$ denotes the number of non-zero entries in $\boldsymbol{x}$, and $k \le n$ is a positive integer. When reducing the dimensionality of a dataset, traditional PCA typically relies on all n features to calculate the first principal component. This approach, though comprehensive, may lead to a low-dimensional representation that is difficult to interpret. In contrast, the zero-norm constraint in SPCA (1) confines the first principal component to rely on at most k features, thereby offering a more interpretable and concise dimensionality reduction. Hence, the SPCA problem (1) is often capable of selecting the k most relevant and important features from a high-dimensional dataset, which significantly improves the interpretability of the dimensionality reduction in machine learning [33, 57] and is helpful for the exact recovery of sparse signals in information theory [4]. Besides, SPCA (1) tends to exhibit greater robustness to noise compared to conventional PCA [57]. These advantages of SPCA have benefited many application fields such as biology, finance, cloud computing, and healthcare, which often deal with high-dimensional datasets (see, e.g., [15, 34, 39, 44]).

1.1. Summary of Main Contributions and Organization

Given the support of variable $\boldsymbol{x}$, we observe that SPCA (1) reduces to the PCA problem, i.e., finding the largest eigenvalue of a principal submatrix of $\boldsymbol{A}$ indexed by the support. Hence, the objective of SPCA (1) can be recast as selecting a $k \times k$ principal submatrix from matrix $\boldsymbol{A}$ to maximize the largest eigenvalue. This motivates us to derive two exact mixed-integer semidefinite programs (MISDPs) and an almost equivalent mixed-integer linear program (MILP) of SPCA (1). Below, we summarize the main contributions and an outline of this paper.

(i) Sections 2 and 3 develop the equivalent MISDP (6) and MISDP (15) for SPCA (1), respectively and derive the worst-case optimality gaps of their continuous relaxations. We show that the continuous relaxation of the MISDP (15) is stronger than the one proposed by d'Aspremont et al. [21].

In Subsection 2.2, we develop a branch-and-cut algorithm for SPCA based on the MISDP (6), which can efficiently solve small- or medium-sized instances (e.g., $n \leq 100s$) to optimality.

(ii) Section 4 derives the first-known MILP (22) for SPCA (1), which can be arbitrarily close to SPCA (1). The MILP can solve small instances to optimality. We also prove the optimality gap of its continuous relaxation.

(iii) Section 5 investigates the scalable greedy and local search algorithms for approximately solving SPCA (1). We derive their first-known approximation ratios and prove the tightness when $k \leq n/2$. The numerical study demonstrates that these two approximation algorithms are superior to the existing ones in the literature.

(iv) Section 6 evaluates the computational efficiency and solution quality of our proposed methods on various real datasets, where the dimension n ranges from 13 to 2,365.

Our contributions have both theoretical and practical relevance. Theoretically, we contribute three exact mixed-integer convex programs to SPCA along with the optimality gaps of their continuous relaxations and prove the first-known approximation ratios of the greedy and local search algorithms. Table 1 displays our theoretical contributions and the comparison with existing results. Practically, our branch-and-cut algorithm and MILP (22) can efficiently yield optimal solutions for small and medium-size instances. Our approximation algorithms are scalable and successfully apply to the large-scale data analytics problem, i.e., identifying critical factors for drug abuse analysis.

Table 1 Summary of theoretical guarantees for SPCA (1)

Convex relaxation	**Optimality gap**
Continuous relaxation of MISDP (6)	$\min\{k, n/k\}$
Continuous relaxation of MISDP (15)	$\min\{k, n/k\}$
Continuous relaxation of MILP (22)	$\min\{k(\sqrt{d}/2+1/2), n/k\sqrt{d}+(n-k)(\sqrt{d}/2+1/2)\}$
SDP relaxation in [16]	$\exp\exp(\Omega(\sqrt{\log\log(n)}))$

Approximation algorithm	**Approximation ratio**
Greedy Algorithm 1	$1/k$
Local Search Algorithm 2	$1/k$
Truncation algorithm [16]	$n^{-1/3}$
Thresholding algorithm [17]	$1/2-(3/2)\operatorname{tr}(\boldsymbol{A})/w^*$
Randomized algorithm [17]	$1/2-\sqrt{\operatorname{tr}(\boldsymbol{A})/(kw^*)}$
SDP-based algorithm [17]	–

1.2. Relevant Literature

In this subsection, we survey the relevant literature on SPCA (1) from three aspects: exact mixed-integer convex programs, convex relaxations, and approximation algorithms.

Exact Mixed-Integer Programs. SPCA (1) is highly nonconvex as it maximizes a convex function subject to two nonconvex constraints (i.e., a quadratic equality constraint and a zero-norm constraint). Unlike traditional PCA, which admits closed-form solutions, SPCA (1) is notoriously known to be NP-hard and inapproximable (see, e.g., [41]). As a result, the equivalent formulations and exact algorithms for solving SPCA (1) to optimality are limited in the literature (see, e.g., [9, 27, 42]). Moghaddam et al. [42] introduced a branch-and-bound method to solve SPCA (1), and they pruned redundant nodes using the eigenvalue of principal submatrices and a greedy algorithm. Recently, Berk and Bertsimas [9] embedded various upper and lower bounds into a branch-and-bound framework, which can efficiently prune nodes and guarantee optimality for quite a few instances. It is worth mentioning that Gally and Pfetsch [27] proposed an equivalent MISDP formulation for SPCA (1). Our MISDP (15) differs from [27] by deriving additional valid inequalities. Another interesting work can be found in Dey et al. [23], where the authors developed approximate convex integer programs for SPCA (1) with an optimality gap of $(1+\sqrt{k/(k+1)})^2$.

Convex Relaxations. In addition to exact solutions of SPCA (1), researchers have actively investigated tractable convex relaxations. A common approach in the literature is developing SDP relaxations for SPCA with theoretical guarantees (e.g., [2, 20, 21, 25, 37, 56]). [2] proposed sufficient conditions for when the SDP relaxation attains the same optimal value as SPCA under the well-known spiked covariance model, in which the covariance matrix $\boldsymbol{A}$ is the identity matrix plus a sparse rank-one matrix. [16] proved a $1/\exp\exp(\Omega(\sqrt{\log\log(n)}))$ optimality gap for the SDP relaxation proposed in [21]. For another SDP relaxation in [20], [25] derived its optimality gap using randomization techniques. This paper derives the theoretical optimality gaps of the continuous relaxations of both MISDPs. Albeit convex, solvers often have difficulty in solving large-scale instances of SDP formulations (e.g., $n = 100s$). The computational challenge of these SDP problems urgently calls for more effective methods to compute the relaxation values for SPCA. From a different angle, this paper solves the continuous relaxations of the proposed MISDP formulations as the maximin saddle point problem, where the subgradient method enjoys a $O(1/T)$ convergence rate [45] based on Euclidean projections.

Approximation Algorithms. Another early thread of work on SPCA is developing high-quality heuristics for solving SPCA to near optimality, such as the greedy algorithm [20, 30], the truncation algorithm [16], the power method [35], the truncated power method [55], and the variable neighborhood search method [14]. In particular, the truncation algorithm in [16] provides the best-known approximation ratio $n^{-1/3}$, which admits an efficient implementation to return a feasible solution to SPCA. This paper investigates the greedy and local search algorithms and proves their first-known approximation ratios $1/k$.

We close this subsection by summarizing several recent works on SPCA (1) that have cited our paper since it became available online. [12] reformulated SPCA (1) as a similar MISDP around the same time as our MISDP (15); however, it is worth noting that our MISDP (15) is equipped with a novel type of valid inequalities and may yield a stronger continuous relaxation. Using the second-order cone relaxations and greedy rounding schemes, [12] focused on the computational improvement of solving SPCA in practice and achieved an optimality gap of $1-2\%$ on testing cases with $n = 1,000s$. [17] proposed three approximation algorithms based on thresholding, randomized matrix multiplication, and SDP relaxation methods, respectively. By enforcing their algorithms to satisfy the zero-norm constraint in SPCA (1), the resulting approximation ratios are presented in Table 1 which depend on the data matrix $\boldsymbol{A}$ and optimal value w^* of SPCA (1). We also compare our approximation algorithms with theirs in the numerical study. A recent study by [6] explored the underlying properties of SPCA under the spiked covariance model for statistical guarantees. Specifically, they reformulated SPCA under the spiked covariance model as a mixed-integer second-order cone program, which can be efficiently solved by their customized algorithm on large-scale instances with $n = 20,000$. Another recent work [24] studied two classical variants of the spiked covariance model: the Wigner and Wishart models. In both cases, the authors proposed a subexponential-time algorithm with a high-probability guarantee for the exact recovery of the support of $\boldsymbol{x}$ in SPCA (1). In contrast to [6, 24], our results apply to any covariance matrix $\boldsymbol{A}$ in SPCA (1). Recently, [37] introduced a novel permutation-invariant SDP relaxation for SPCA (1), providing remarkably tight upper bounds. Nevertheless, this approach may encounter computational difficulties when applied to datasets comprising hundreds of features.

Notation. The following notation is used throughout the paper. We let $\mathcal{S}^n, \mathcal{S}_+^n$ denote the set of all the $n\times n$ symmetric real matrices and the set of all the $n\times n$ symmetric positive semidefinite matrices, respectively. We use bold lower-case letters (e.g., $\boldsymbol{x}$) and bold upper-case letters (e.g., $\boldsymbol{X}$) to denote vectors and matrices, respectively and use corresponding non-bold letters (e.g., x_i, X_{ij}) to denote their components. We let $\mathbf{0}$ denote the zero vector. We use $\lceil\cdot\rceil$ to denote the ceil function. We let $\mathbb{R}^n$ denote the set of all the n dimensional vectors and let $\mathbb{R}_+^n$ denote the set of all the n-dimensional nonnegative vectors. Given a positive integer n and an integer $s\leq n$, we let $[n]:=\{1,2,\cdots,n\}$ and let $[s,n]:=\{s,s+1,\cdots,n\}$. We let $\boldsymbol{I}_n$ denote the $n\times n$ identity matrix and let $\boldsymbol{e}_i$ denote its i-th column vector. Given a set S and an integer k, we let $|S|$ denote its cardinality and let $\binom{S}{k}$ denote the collection of all the size-k subsets out of S. Given a $m\times n$ matrix $\boldsymbol{A}$ and two sets $S\subseteq[m]$, $T\subseteq[n]$, we let $\boldsymbol{A}_{S,T}$ denote a submatrix of $\boldsymbol{A}$ with rows and columns indexed by sets S and T, respectively. Given a vector $\boldsymbol{x}\in\mathbb{R}^n$, we let $\mathrm{Diag}(\boldsymbol{x})$ denote a diagonal matrix with diagonal elements $\boldsymbol{x}$, let $\|\boldsymbol{x}\|_2$ denote the two norm, let $\|\boldsymbol{x}\|_\infty$ denote the infinity norm, and let $\|\boldsymbol{x}\|_0$ denote the zero norm that counts the number of non-zero entries in $\boldsymbol{x}$. Given a symmetric matrix $\boldsymbol{A}$, let $\mathrm{tr}(\boldsymbol{A})$ denote the trace of matrix $\boldsymbol{A}$ and let $\lambda_{\max}(\boldsymbol{A})$ denote the largest eigenvalue of $\boldsymbol{A}$. The additional notation will be introduced later as needed.

2. Exact MISDP Formulation (I)

This section derives an equivalent mixed-integer semidefinite programming (MISDP) formulation for SPCA (1) based on spectral decomposition and disjunctive programming techniques. The proposed MISDP formulation facilitates the development of a branch-and-cut algorithm, which allows for exactly solving SPCA (1).

To begin with, for each $i\in[n]$, we let the binary variable $z_i=1$ if the i-th feature is selected and 0 otherwise. Thus, SPCA (1) can be written as the following mixed-integer nonconvex program:

$$\text{(SPCA)}\quad w^*:=\max_{\boldsymbol{x}\in\mathbb{R}^n,\boldsymbol{z}\in Z}\left\{\boldsymbol{x}^\top\boldsymbol{A}\boldsymbol{x}:\|\boldsymbol{x}\|_2=1,|x_i|\leq z_i,\forall i\in[n]\right\},\tag{2}$$

where we let Z denote the feasible set of binary variables $\boldsymbol{z}$ throughout the paper, i.e.,

$$Z:=\left\{\boldsymbol{z}\in\{0,1\}^n:\sum_{i\in[n]}z_i\leq k\right\}.$$

Based on the mixed-integer nonconvex formulation (2), we derive an equivalent mixed-integer convex program for SPCA in the following subsection.

2.1. An Equivalent MISDP Reformulation of SPCA

According to proposition 1 in [42], when the support of variable $\boldsymbol{x}$ is known, SPCA (1) essentially reduces to a conventional PCA problem, which is summarized in Part (i) of Lemma 1. Note that Part (ii) is a simple extension of Part (i). Furthermore, Part (iii) of Lemma 1, as proposed by [20] in their section 2, reformulates the largest eigenvalue function using the Cholesky factorization of matrix $\boldsymbol{A}$.

Lemma 1 *[20, 42] For a symmetric matrix $\boldsymbol{A}\in\mathcal{S}^n$ and a subset $S\subseteq[n]$, the following hold:*

(i) $\max_{\boldsymbol{x}\in\mathbb{R}^n}\left\{\boldsymbol{x}^\top\boldsymbol{A}\boldsymbol{x}:\|\boldsymbol{x}\|_2=1, x_i=0,\forall i\notin S\right\}=\lambda_{\max}(\boldsymbol{A}_{S,S})$,

(ii) $\max_{\boldsymbol{X}}\{\operatorname{tr}(\boldsymbol{A}_{S,S}\boldsymbol{X}):\operatorname{tr}(\boldsymbol{X})=1\}=\lambda_{\max}(\boldsymbol{A}_{S,S})$, *and*

(iii) If matrix $\boldsymbol{A}$ is positive semidefinite, then $\lambda_{\max}(\boldsymbol{A}_{S,S})=\lambda_{\max}(\sum_{i\in S}\boldsymbol{c}_i\boldsymbol{c}_i^\top)$, where $\boldsymbol{A}=\boldsymbol{C}^\top\boldsymbol{C}$, $\boldsymbol{C}\in\mathbb{R}^{d\times n}$ denotes the Cholesky factorization matrix of $\boldsymbol{A}$, d is the rank of $\boldsymbol{A}$, and $\boldsymbol{c}_i\in\mathbb{R}^d$ denotes the i-th column vector of $\boldsymbol{C}$ for each $i\in[n]$.

We make the following remarks about Lemma 1: Part (i) of Lemma 1 reduces SPCA (1) to select an at most size-$k\times k$ principal submatrix of $\boldsymbol{A}$ with the maximum largest eigenvalue, which leads to a combinatorial reformulation of SPCA (1) as shown below; Part (ii) of Lemma 1 suggests that the SDP relaxation of the largest eigenvalue problem is exact, which paves the way for the development of two exact MISDPs; and by leveraging Part (iii) in Lemma 1, we derive an exact MILP for SPCA in Section 4.

According to Part (i) in Lemma 1, we introduce a subset variable S to represent the support of variable $\boldsymbol{x}$ in SPCA (1) and rewrite it as

$$w^*:=\max_S\left\{\lambda_{\max}(\boldsymbol{A}_{S,S}):|S|\le k, S\subseteq[n]\right\}. \tag{3}$$

Suppose that matrix $\boldsymbol{A}$ has a rank of d. Then, by computing the Cholesky factorization $\boldsymbol{A}=\boldsymbol{C}^\top\boldsymbol{C}$ with $\boldsymbol{C}\in\mathbb{R}^{d\times n}$, Part (iii) of Lemma 1 allows us to recast the objective of SPCA (3) as below.

$$w^*:=\max_S\left\{\lambda_{\max}\left(\sum_{i\in S}\boldsymbol{c}_i\boldsymbol{c}_i^\top\right):|S|\le k, S\subseteq[n]\right\}. \tag{4}$$

Following the construction of SPCA (2), we let the binary variable z_i represent whether to select the i-th column vector $\boldsymbol{c}_i$ or not for each $i\in[n]$, which reformulates SPCA (4) as an integer program:

$$w^*:=\max_{\boldsymbol{z}\in Z}\left\{\lambda_{\max}\left(\sum_{i\in[n]}z_i\boldsymbol{c}_i\boldsymbol{c}_i^\top\right)\right\}. \tag{5}$$

In the celebrated work on SPCA by d'Aspremont et al. [20], they derived an SDP relaxation for SPCA (5) based on the Cholesky decomposition of $\boldsymbol{A}$. By leveraging their SDP relaxation and disjunctive programming techniques [5], we reduce SPCA (5) to an equivalent MISDP, as shown in the theorem below. It is worth noting that our Theorem 1 differs from [20][proposition 1] in two aspects: (i) [20] studied the modified SPCA problem (1) that moves the zero-norm constraint to the objective as a regularization term. In contrast, our Theorem 1 focuses on SPCA (1); and (ii) Theorem 1 proposes an exact MISDP that relies on the disjunction of bilinear terms, rather than the SDP relaxation in [20].

Theorem 1 *SPCA* (2) *admits an equivalent MISDP formulation:*

$$\text{(SPCA)} \quad w^* := \max_{\substack{\boldsymbol{z}\in Z,\\ \boldsymbol{X},\boldsymbol{W}_1,\cdots,\boldsymbol{W}_n\in\mathcal{S}_+^d}} \left\{ \sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{W}_i \boldsymbol{c}_i : \operatorname{tr}(\boldsymbol{X})=1, \boldsymbol{X}\succeq \boldsymbol{W}_i, \operatorname{tr}(\boldsymbol{W}_i)=z_i, \forall i\in[n] \right\}. \tag{6}$$

Proof. By leveraging Part (ii) in Lemma 1 to reformulate the objective of SPCA (5) as an SDP, we have that

$$w^* := \max_{\boldsymbol{z}\in Z, \boldsymbol{X}\in\mathcal{S}_+^d} \left\{ \sum_{i\in[n]} z_i \boldsymbol{c}_i^\top \boldsymbol{X} \boldsymbol{c}_i : \operatorname{tr}(\boldsymbol{X})=1 \right\}, \tag{7}$$

where the objective function comes from the identity $\operatorname{tr}(\boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{X}) = \boldsymbol{c}_i^\top \boldsymbol{X} \boldsymbol{c}_i$ for each $i\in[n]$.

In SPCA (7), the objective function contains bilinear terms $\{z_i\boldsymbol{X}\}_{i\in[n]}$. To convexify them, we create two copies of the matrix variable $\boldsymbol{X}$, denoting by $\boldsymbol{W}_{i1}, \boldsymbol{W}_{i2}$ for each $i\in[n]$, and one of them is equal to $\boldsymbol{X}$ depending on the value of binary variable z_i. As a result, SPCA (7) becomes

$$\begin{aligned} w^* := \max_{\boldsymbol{z}\in Z, \boldsymbol{X}, \boldsymbol{W}_{i1}, \boldsymbol{W}_{i2}\in\mathcal{S}_+^d} \Bigg\{ & \sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{W}_{i1} \boldsymbol{c}_i : \boldsymbol{X} = \boldsymbol{W}_{i1} + \boldsymbol{W}_{i2}, \forall i\in[n], \operatorname{tr}(\boldsymbol{X})=1, \\ & \operatorname{tr}(\boldsymbol{W}_{i1}) = z_i, \operatorname{tr}(\boldsymbol{W}_{i2}) = 1-z_i, \forall i\in[n] \Bigg\}. \end{aligned}$$

Above, the matrix variables $\{\boldsymbol{W}_{i2}\}_{i\in[n]}$ are redundant and can be replaced by inequality $\boldsymbol{X}\succeq\boldsymbol{W}_i$ for each $i\in[n]$. Thus, we arrive at the equivalent reformulation (6) for SPCA. □

Theorem 1 presents a novel equivalent MISDP reformulation (6) for SPCA (1). We note that the MISDP (6) has several interesting properties: (i) it can be directly solved via exact MISDP solvers such as YALMIP; (ii) matrix variables $\boldsymbol{X}$ and $\{\boldsymbol{W}_i\}_{i\in[n]}$ have a dimension of $d\times d$, where d is the rank of covariance matrix $\boldsymbol{A}$. This finding suggests that we can further reduce the problem size of the MISDP (6) when matrix $\boldsymbol{A}$ is low-rank; and (iii) in the MISDP (6), binary variables are separable from the other matrix variables. This observation motivates us to employ Benders decomposition [28] to solve the MISDP (6), as detailed in the subsequent subsection.

By relaxing binary variables, the continuous relaxation of the MISDP (6) provides an upper bound for SPCA (1), which can help evaluate the solution quality of different heuristics. In the following proposition, we provide the theoretical guarantee for the quality of this continuous relaxation. Formally, the continuous relaxation of the MISDP (6) is defined as follows:

$$\overline{w}_1 := \max_{\substack{\boldsymbol{z}\in\overline{Z},\\ \boldsymbol{X},\boldsymbol{W}_1,\cdots,\boldsymbol{W}_n\in\mathcal{S}_+^d}} \left\{ \sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{W}_i \boldsymbol{c}_i : \operatorname{tr}(\boldsymbol{X})=1, \boldsymbol{X}\succeq \boldsymbol{W}_i, \operatorname{tr}(\boldsymbol{W}_i)=z_i, \forall i\in[n] \right\}, \tag{8}$$

where we let $\overline{Z}$ denote the continuous relaxation of binary feasible set Z, i.e.,

$$\overline{Z} := \left\{ \boldsymbol{z}\in[0,1]^n : \sum_{i\in[n]} z_i \le k \right\}.$$

Proposition 1 *The continuous relaxation* (8) *of the MISDP* (6) *achieves a* $\min\{k, n/k\}$ *optimality gap of SPCA* (1)*, i.e.,*

$$w^* \le \overline{w}_1 \le \min\left\{k, \frac{n}{k}\right\} w^*.$$

Proof. First, the inequality $w^*\le \overline{w}_1$ stems from the fact that problem (8) serves as a convex relaxation of MISDP (6). Thus, it remains to show that (i) $\overline{w}_1 \le kw^*$ and (ii) $\overline{w}_1 \le n/kw^*$.

Part (i). For any feasible solution $(\boldsymbol{z}, \boldsymbol{X}, \{\boldsymbol{W}_i\}_{i\in[n]})$ to the continuous relaxation (8), we have that

$$\sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{W}_i \boldsymbol{c}_i \le \sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{c}_i \operatorname{tr}(\boldsymbol{W}_i) = \sum_{i\in[n]} z_i \boldsymbol{c}_i^\top \boldsymbol{c}_i \le \sum_{i\in[n]} z_i w^* \le k w^*,$$

where the first inequality is because the trace of the product of two symmetric positive semidefinite matrices is no larger than the product of the traces of these two matrices [18], the first equality is from $\operatorname{tr}(\boldsymbol{W}_i)=z_i$ for each $i\in[n]$, the second inequality is because

$$\boldsymbol{c}_i^\top \boldsymbol{c}_i = \lambda_{\max}\left(\boldsymbol{c}_i\boldsymbol{c}_i^\top\right) \le \max_{S\subseteq[n]:|S|=k} \lambda_{\max}\left(\sum_{j\in S} \boldsymbol{c}_j \boldsymbol{c}_j^\top\right) := w^*,$$

and the last inequality is due to $\sum_{i\in[n]} z_i \le k$.

Part (ii). In addition, given any feasible solution $(\boldsymbol{z}, \boldsymbol{X}, \{\boldsymbol{W}_i\}_{i\in[n]})$ of the continuous relaxation (8), we have

$$\sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{W}_i \boldsymbol{c}_i \le \sum_{i\in[n]} \boldsymbol{c}_i^\top \boldsymbol{X} \boldsymbol{c}_i = \frac{1}{\binom{n-1}{k-1}} \sum_{S\in\binom{[n]}{k}} \sum_{i\in S} \boldsymbol{c}_i^\top \boldsymbol{X} \boldsymbol{c}_i \le \frac{\binom{n}{k}}{\binom{n-1}{k-1}} w^* = \frac{n}{k} w^*,$$

where the first inequality is due to $\boldsymbol{W}_i \succeq \boldsymbol{X}$ for all $i\in[n]$ and the second one is from Part (ii) in Lemma 1. □

Proposition 1 shows that the continuous relaxation (8) is at most $\min\{k, n/k\}$ away from the optimal value of SPCA (6), implying that if $k=1$ or $k=n$, then the continuous relaxation value $\overline{w}_1$ exactly matches the optimal value w^*. It is important to note that our analysis of the k-factor optimality gap in Proposition 1 may not be tight. The continuous relaxation (8) nearly coincides with the optimal value of SPCA (1) in our numerical study. We leave it as an interesting future question to improve the worst-case guarantee of this upper bound.

2.2. Solving SPCA (6) by the Benders Decomposition Method

It is well-established that solving large-scale SDPs can be challenging, and the same holds for the MISDP (6). To solve the MISDP (6) efficiently, this subsection applies the Benders decomposition scheme [8, 28] to develop a branch-and-cut method. Furthermore, a side product of the Benders decomposition method is to transform the continuous relaxation (8) of the MISDP (6) into a maximin problem, enabling the use of the efficient subgradient method.

The Benders decomposition method lies in separating binary variables from continuous variables. That is, for any fixed binary variables $\boldsymbol{z}\in Z$ in the MISDP (6), the resulting subproblem is a convex program for which we can use the duality theory to develop Benders cuts. Therefore, by separating binary variables, we rewrite SPCA (6) as

$$w^* := \max_{\boldsymbol{z}\in Z} H_1(\boldsymbol{z}) := \max_{\boldsymbol{X},\boldsymbol{W}_1,\cdots,\boldsymbol{W}_d\in\mathcal{S}_+^d}\left\{\sum_{i\in[n]}\boldsymbol{c}_i^\top \boldsymbol{W}_i\boldsymbol{c}_i : \operatorname{tr}(\boldsymbol{X})=1, \boldsymbol{X}\succeq \boldsymbol{W}_i, \operatorname{tr}(\boldsymbol{W}_i)=z_i, \forall i\in[n]\right\}. \quad (9)$$

The Benders decomposition method is of particular interest when the dual of the above subproblem over $(\boldsymbol{X},\{\boldsymbol{W}_i\}_{i\in[n]})$ is easy to compute for any $\boldsymbol{z}\in Z$, ensuring that the Benders cuts are effective to generate. Using Part(ii) in Lemma 1, we show below that the strong duality of the inner maximization problem in (9) holds, and the dual problem admits a closed-form solution for any binary $\boldsymbol{z}\in Z$.

Proposition 2 *For the function $H_1(\boldsymbol{z})$ defined in SPCA* (9), *we have that*

(i) For any $\boldsymbol{z}\in\overline{Z}$, function $H_1(\boldsymbol{z})$ is equivalent to

$$H_1(\boldsymbol{z}) = \min_{\boldsymbol{\mu},\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_n\in\mathcal{S}_+^d}\left\{\lambda_{\max}\left(\sum_{i\in[n]}\boldsymbol{Q}_i\right)+\sum_{i\in[n]}\mu_i z_i : \boldsymbol{c}_i\boldsymbol{c}_i^\top \preceq \boldsymbol{Q}_i+\mu_i\boldsymbol{I}_d, 0\le\mu_i\le\|\boldsymbol{c}_i\|_2^2, \forall i\in[n]\right\}, \quad (10)$$

which is concave in $\boldsymbol{z}$.

(ii) For any binary $\boldsymbol{z}\in Z$, an optimal solution to the problem (10) *is $\mu_i^*=0$ if $z_i=1$ and $\|\boldsymbol{c}_i\|_2^2$ otherwise, and $\boldsymbol{Q}_i^* := (1-\mu_i^*/\|\boldsymbol{c}_i\|_2^2)\boldsymbol{c}_i\boldsymbol{c}_i^\top$ for each $i\in[n]$.*

Proof. See Appendix A.1. □

According to Part (i) of Proposition 2, we provide an equivalent reformulation (10) of the function $H_1(\boldsymbol{z})$ by dualizing the subproblem in (9). Plugging the closed-form solution to the dual problem (10) in Part (ii) of Proposition 2, we have that

$$H_1(\boldsymbol{z}) := \lambda_{\max}\left(\sum_{i\in S}\boldsymbol{c}_i\boldsymbol{c}_i^\top\right)+\sum_{i\in[n]\setminus S}\|\boldsymbol{c}_i\|_2^2 = \lambda_{\max}(\boldsymbol{A}_{S,S})+\sum_{i\in[n]\setminus S}\|\boldsymbol{c}_i\|_2^2,$$

for any given solution $\boldsymbol{z}\in Z$ with its support S. The result above reduces SPCA (9) to

$$w^* = \max_{\boldsymbol{z}\in Z} H_1(\boldsymbol{z}) = \max_{\boldsymbol{z}\in Z, w\in\mathbb{R}_+}\left\{w : w\le H_1(\boldsymbol{z})\le \lambda_{\max}(\boldsymbol{A}_{S,S})+\sum_{i\in[n]\setminus S}\|\boldsymbol{c}_i\|_2^2 z_i, \forall S\subseteq[n], |S|\le k\right\}. \quad (11)$$

Therefore, for any solution $(\widehat{\boldsymbol{z}}, \widehat{w}) \in Z \times \mathbb{R}$, the most violated constraint is

$$w \leq \lambda_{\max}(\boldsymbol{A}_{\widehat{S},\widehat{S}}) + \sum_{i \in [n] \setminus \widehat{S}} \|\boldsymbol{c}_i\|_2^2 z_i,$$

where set $\widehat{S}$ denotes the support of $\widehat{\boldsymbol{z}}$. Based on the closed-form valid inequalities above, we can solve MISDP (6) to optimality in a branch-and-cut framework to improve the computational performance, as shown in Section 6.

Following the framework of SPCA (9), we can rewrite the SDP relaxation (8) with the form of $\max_{\boldsymbol{z} \in \overline{Z}} H_1(\boldsymbol{z})$. However, we note that for any continuous $z \in \overline{Z}$, the dual representation of function $H_1(\boldsymbol{z})$ in problem (10) remains challenging to solve. Motivated by Part (ii) in Proposition 2, we propose an efficient function $\overline{H}_1(\boldsymbol{z})$ to approximate $H_1(\boldsymbol{z})$ by fixing $\boldsymbol{Q}_i := (1 - \mu_i / \|\boldsymbol{c}_i\|_2^2) \boldsymbol{c}_i \boldsymbol{c}_i^\top$ for each $i \in [n]$ in problem (10). As a result, we have $\overline{H}_1(\boldsymbol{z}) \geq H_1(\boldsymbol{z})$ for any $\boldsymbol{z} \in \overline{Z}$. In the following theorem, we show that the relaxed function $\overline{H}_1(\boldsymbol{z})$ becomes exact for any binary solution $\boldsymbol{z} \in Z$, and the resulting upper bound still achieves a $\min\{k, n/k\}$ optimality gap.

Theorem 2 *The following hold for the relaxed function* $\overline{H}_1(\boldsymbol{z})$*:*

(i) For any $\boldsymbol{z} \in \overline{Z}$*, function* $H_1(\boldsymbol{z})$ *is upper bounded by*

$$\overline{H}_1(\boldsymbol{z}) := \min_{\boldsymbol{\mu} \in \mathbb{R}^n} \left\{ \lambda_{\max}\left(\sum_{i \in [n]} \left(1 - \frac{\mu_i}{\|\boldsymbol{c}_i\|_2^2} \right) \boldsymbol{c}_i \boldsymbol{c}_i^\top \right) + \sum_{i \in [n]} \mu_i z_i : 0 \leq \mu_i \leq \|\boldsymbol{c}_i\|_2^2, \forall i \in [n] \right\}; \tag{12}$$

(ii) If $\boldsymbol{z} \in Z$*, then* $H_1(\boldsymbol{z}) = \overline{H}_1(\boldsymbol{z}) = \lambda_{\max}(\sum_{i \in [n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top)$*; and*

(iii) The continuous relaxation of SPCA

$$\overline{w}_2 := \max_{\boldsymbol{z} \in \overline{Z}} \overline{H}_1(\boldsymbol{z}) \tag{13}$$

achieves a $\min\{k, n/k\}$ *optimality gap, i.e.,* $w^* \leq \overline{w}_1 \leq \overline{w}_2 \leq \min\{k, n/k\} w^*$*, where* $\overline{w}_1$ *is defined in* (8)*.*

Proof. See Appendix A.2. □

We remark that (i) Compared to $H_1(\boldsymbol{z})$, the function $\overline{H}_1(\boldsymbol{z})$ is formulated by a simple convex program (12) involving an n-dimensional variable $\boldsymbol{\mu}$. Hence, the corresponding continuous relaxation (13) can be efficiently solved by the subgradient method with a convergence rate of $O(1/T)$ (see, e.g., [45]); (ii) On the other hand, the SDP relaxation $\overline{w}_1 = \max_{\boldsymbol{z} \in \overline{Z}} H_1(\boldsymbol{z})$ tends to be stronger than $\overline{w}_2 = \max_{\boldsymbol{z} \in \overline{Z}} \overline{H}_1(\boldsymbol{z})$. Thus, there is a trade-off between the computational efficiency and tight upper bounds; (iii) It is worth mentioning that both upper bounds $\overline{w}_1$ and $\overline{w}_2$ achieve the same optimality gap $\min\{k, n/k\}$, which implies that there might be room to improve the analysis of the optimality gap in Proposition 1. We leave this to interested readers; and (iv) when $\boldsymbol{z} \in Z$ is binary, both problems (10) and (12) are equivalent and admit a closed-form solution, as shown in Proposition 2, which facilitates the implementation of the branch-and-cut method.

3. Exact MISDP Formulation (II)

In addition to the MISDP (6), this section proposes another exact MISDP reformulation (15) for SPCA (1) based on the results in Part (i) and Part (ii) of Lemma 1. For the proposed MISDP (15), we also guarantee the quality of its continuous relaxation and prove that it is stronger than the existing SDP relaxation [21].

3.1. A Naive Exact MISDP Formulation

We first establish a naive exact MISDP formulation of SPCA (2) based on Part (ii) of Lemma 1.

Proposition 3 *SPCA* (2) *admits the following MISDP formulation:*

$$\text{(SPCA)} \quad w^* := \max_{\boldsymbol{z}\in Z, \boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, X_{ii} \leq z_i, \forall i \in [n] \right\}. \tag{14}$$

and its continuous relaxation value is equal to $\lambda_{\max}(\boldsymbol{A})$.

Proof. See Appendix A.3. □

It should be noted that the work of [27] was the first to derive the equivalent MISDP formulation (14) for SPCA, which relies on characterizing sophisticated extreme points. Our proof is based on Part (ii) of Lemma 1, which is different from theirs. Although the MISDP (14) is equivalent to SPCA (2), it might be a weak formulation, provided that its continuous relaxation is equal to the trivial upper bound $\lambda_{\max}(\boldsymbol{A})$. In [27], the authors proposed valid inequalities $X_{ij} \leq z_i/2$ for all $i, j \in [n]$ to strengthen the MISDP (14). Next, we use two different types of valid inequalities that can significantly strengthen the MISDP (14).

3.2. A Stronger MISDP Reformulation with Valid Inequalities

This subsection presents valid inequalities to strengthen SPCA (14) and derives the optimality gap of the resulting continuous relaxation. To be specific, Part (i) of Lemma 2 includes the valid inequalities that are first proposed by [11] in their section 4. We derive another type of valid inequalities for the MISDP (14), as summarized in Part (ii) of Lemma 2.

Lemma 2 *The following two inequalities are valid to SPCA* (14)

(i) $\sum_{j\in[n]} X_{ij}^2 \leq X_{ii} z_i$ *for all* $i \in [n]$; *and*

(ii) $\left(\sum_{j\in[n]} |X_{ij}|\right)^2 \leq k X_{ii} z_i$ *for all* $i \in [n]$.

Proof. See Appendix A.4. □

We make the following remarks about Lemma 2.

(a) The valid inequalities for SPCA (14) in Lemma 2 exhibit significant strength, potentially dominating existing ones, such as

$$|X_{ij}| \leq z_i, X_{ij}^2 \leq X_{ii} z_j, X_{ij}^2 \leq z_i z_j, \forall i, j \in [n];$$

(b) The constraints in Lemma 2 can be expressed by second-order cones as shown in [7], which thus can be efficiently handled by SDP solvers like MOSEK and SDPT3; and

(c) Plugging the valid inequalities in Lemma 2 into the MISDP (14), we arrive at a stronger MISDP of SPCA, which is summarized below. Besides, constraints $X_{ii} \leq z_i$ for all $i \in [n]$ in the MISDP (14) are implied by those in Part (i) of Lemma 2 and are thus removed.

Theorem 3 *SPCA* (2) *is equivalent to the following MISDP formulation:*

$$\text{(SPCA)} w^* := \max_{\boldsymbol{z} \in Z, \boldsymbol{X} \in \mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{AX}) : \operatorname{tr}(\boldsymbol{X}) = 1, \sum_{j \in [n]} X_{ij}^2 \leq X_{ii} z_i, \left(\sum_{j \in [n]} |X_{ij}| \right)^2 \leq k X_{ii} z_i, \forall i \in [n] \right\}. \tag{15}$$

Suppose that $\overline{w}_3$ denotes the continuous relaxation of SPCA (15), i.e.,

$$\overline{w}_3 := \max_{\boldsymbol{z} \in \overline{Z}, \boldsymbol{X} \in \mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{AX}) : \operatorname{tr}(\boldsymbol{X}) = 1, \sum_{j \in [n]} X_{ij}^2 \leq X_{ii} z_i, \left(\sum_{j \in [n]} |X_{ij}| \right)^2 \leq k X_{ii} z_i, \forall i \in [n] \right\}. \tag{16}$$

We show that the continuous relaxation (16) is stronger than a known SDP relaxation for SPCA introduced by d'Aspremont et al. [21], denoted by $\overline{w}_4$ that admits the formulation below.

$$\overline{w}_4 := \max_{\boldsymbol{X} \in \mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{AX}) : \operatorname{tr}(\boldsymbol{X}) = 1, \sum_{i \in [n]} \sum_{j \in [n]} |X_{ij}| \leq k \right\}. \tag{17}$$

Proposition 4 *For the SDP relaxations of SPCA defined in* (16) *and* (17)*, we have that* $\overline{w}_3 \leq \overline{w}_4$*.*

Proof. See Appendix A.5. □

For the MISDP (15), analogous to Proposition 1, we also guarantee the theoretical gap of its continuous relaxation (16). The upper bound (16) is quite close to the optimal value according to our numerical study.

Proposition 5 *The continuous relaxation* (16) *of the MISDP* (15) *yields a* $\min\{k, n/k\}$ *optimality gap for SPCA, i.e.,*

$$w^* \leq \overline{w}_3 \leq \min\left\{k, \frac{n}{k}\right\} w^*.$$

Proof. See Appendix A.6. □

Albeit attaining the same theoretical optimality gaps, the proposed MISDP (6) and MISDP (15) are generally not comparable, as shown in our numerical study. Besides, we note that the continuous relaxation (8) of the MISDP (6) can be more challenging to solve due to the existence of multiple positive semidefinite matrix variables. Next, we close this section by applying Benders decomposition to solve the MISDP (15).

3.3. Solving SPCA (15) by the Benders Decomposition Method

The decomposition method developed for SPCA (15) in this subsection follows from Section 2.2. Therefore, some details are omitted for brevity. First, we decompose the proposed MISDP (15) by a master problem over binary variables $\boldsymbol{z} \in Z$ and a subproblem over the matrix variable $\boldsymbol{X} \in \mathcal{S}_+^n$, leading to the following two-stage optimization problem:

$$w^* = \max_{\boldsymbol{z} \in Z} H_2(\boldsymbol{z}) := \max_{\boldsymbol{X} \in \mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, \sum_{j \in [n]} X_{ij}^2 \le X_{ii} z_i, \left(\sum_{j \in [n]} |X_{ij}| \right)^2 \le k X_{ii} z_i, \forall i \in [n] \right\}. \tag{18}$$

It is desirable to derive an efficient dual formulation of $H_2(\boldsymbol{z})$ for any given $\boldsymbol{z} \in \overline{Z}$ such that its subgradient can be easily computed. Indeed, by leveraging Part(ii) in Lemma 1 and dualizing the second-order cone constraints, the strong duality holds for the inner maximization problem over $\boldsymbol{X}$ in (18). The proof is similar to that of Proposition 2 and is thus omitted.

Proposition 6 *For any $\boldsymbol{z} \in \overline{Z}$, function $H_2(\boldsymbol{z})$ is equivalent to*

$$\begin{aligned} H_2(\boldsymbol{z}) := \min_{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\nu}_1, \boldsymbol{\nu}_2, \boldsymbol{\Lambda}, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{\beta}} \quad & \lambda_{\max} \left(\boldsymbol{A} + \boldsymbol{\Lambda} + 1/2 \operatorname{Diag}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2 + \boldsymbol{\nu}_1 + \boldsymbol{\nu}_2) - \boldsymbol{W}_1 + \boldsymbol{W}_2 \right) \\ & + 1/2(-\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)^\top \boldsymbol{z} + k/2(-\boldsymbol{\nu}_1 + \boldsymbol{\nu}_2)^\top \boldsymbol{z}, \\ \text{s.t.} \quad & \beta_i + (\boldsymbol{W}_1)_{ij} + (\boldsymbol{W}_2)_{ij} \le 0, \forall i \in [n], j \in [n], \\ & \sum_{j \in [n]} \boldsymbol{\Lambda}_{ij}^2 + (\mu_{i1})^2 \le (\mu_{i2})^2, \forall i \in [n], \\ & \beta_i^2 + (\nu_{i1})^2 \le (\nu_{i2})^2, \forall i \in [n], \\ & (W_1)_{ij} \ge 0, (W_2)_{ij} \ge 0, \forall i \in [n], \forall j \in [n], \\ & \boldsymbol{\mu}_1, \boldsymbol{\nu}_1, \boldsymbol{\beta} \in \mathbb{R}^n, \boldsymbol{\mu}_2, \boldsymbol{\nu}_2 \in \mathbb{R}_+^n, \boldsymbol{\Lambda}, \boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathcal{S}^n, \end{aligned} \tag{19}$$

which is concave in $\boldsymbol{z}$.

For the minimization problem (19) that defines the function $H_2(\boldsymbol{z})$ in Proposition 6, we remark that: (i) Note that for any given $\boldsymbol{z} \in \overline{Z}$, function $H_2(\boldsymbol{z})$ can be solved as a second-order cone program and escape from the SDP curse. It can be solved more effectively via the first-order methods (e.g., the subgradient method) since the subgradient is easy to obtain and the projection only involves second-order cone constraints; (ii) On the other hand, the continuous relaxation (16) can be written as

$$\overline{w}_3 = \max_{\boldsymbol{z} \in \overline{Z}} H_2(\boldsymbol{z}). \tag{20}$$

Plugging the minimization problem (19) into the relaxation above, the subgradient method can be applicable to solve the entire maximin saddle problem (20) with $O(1/T)$ rate of convergence (see, e.g., [45]); and (iii) We can warm start the exact branch-and-cut algorithm by solving the continuous relaxation (20) and adding all subgradient inequalities at the root relaxed node.

4. An MILP Formulation for SPCA with Arbitrary Accuracy

In this section, inspired by the definition of eigenvalues, we derive an arbitrarily accurate mixed-integer linear program (MILP) for SPCA (5), which enables us to directly leverage the computational power of commercial solvers such as Gurobi. In addition, we establish the optimality gap of the corresponding relaxation.

4.1. An MILP Formulation for SPCA

This subsection provides a novel representation of the objective function of SPCA (5), i.e., the largest eigenvalue of matrix $\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top$, which leads to an MILP formulation. To be specific, according to the definition of eigenvalues, we observe that

$$\lambda_{\max}\left(\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top\right) = \max_{w, \boldsymbol{x}\in\mathbb{R}^d} \left\{ w : \left(\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top\right)\boldsymbol{x} = w\boldsymbol{x}, \|\boldsymbol{x}\|_\infty = 1 \right\},$$

where $\boldsymbol{x}$ represents an eigenvector, and the constraint of the infinity norm rules out the trivial solution $\boldsymbol{x}=\boldsymbol{0}$. We first replace the well-known constraint $\|\boldsymbol{x}\|_2 = 1$ with $\|\boldsymbol{x}\|_\infty = 1$ for computing eigenvalues, which lays the foundation for our MILP. Plugging the identity above into the objective, SPCA (5) becomes

$$w^* = \max_{w, \boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{z}\in Z} \left\{ w : \sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{x} = w\boldsymbol{x}, \|\boldsymbol{x}\|_\infty = 1 \right\}. \tag{21}$$

For any solution $\boldsymbol{z}\in Z$, we will show that the rest of SPCA (21) can be linearized using the disjunctive programming and binary expansion techniques. Specifically, the nonlinearity of SPCA (21) arises from three aspects:

(i) Bilinear terms in the expression $\sum_{i\in[n]} z_i \boldsymbol{x}\boldsymbol{c}_i \boldsymbol{c}_i^\top$. They can be easily linearized using the disjunctive programming techniques, provided that variable z_i is binary for each $i\in[n]$;

(ii) Constraint $\|\boldsymbol{x}\|_\infty = 1$. The nonconvex constraint $\|\boldsymbol{x}\|_\infty = 1$ can be equivalently represented as a union with $2d$ sets as follows

$$\cup_{j\in[d]}\left\{\boldsymbol{x}\in\mathbb{R}^d : x_j = 1, \|\boldsymbol{x}\|_\infty \le 1\right\} \cup_{j\in[d]} \left\{\boldsymbol{x}\in\mathbb{R}^d : x_j = -1, \|\boldsymbol{x}\|_\infty \le 1\right\}.$$

Since SPCA (21) is invariant with $\boldsymbol{x}$ and $-\boldsymbol{x}$ in, it suffices to use only d sets, i.e., $\cup_{j\in[d]}\left\{\boldsymbol{x}\in \mathbb{R}^d : x_j = 1, \|\boldsymbol{x}\|_\infty \le 1\right\}$. This can be expressed as an MILP using the techniques in [5]; and

(iii) Bilinear term $w\boldsymbol{x}$. We can start by approximating the continuous variable w using binary expansion and then linearize the resulting bilinear terms through disjunction.

Following the above analyses, we can reformulate SPCA (21) as an MILP formulation.

Theorem 4 *Given a threshold $\epsilon > 0$, the following MILP is $O(\epsilon)$-close to SPCA (2), i.e., $\epsilon \leq \widehat{w}(\epsilon) - w^* \leq \epsilon\sqrt{d}$, where $\widehat{w}(\epsilon)$ is defined by*

$$
\begin{aligned}
\widehat{w}(\epsilon) := \max_{w, \boldsymbol{z} \in Z, \boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{x}, , \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\sigma}} \quad & w \\
\text{s.t.} \quad & \boldsymbol{x} = \boldsymbol{\delta}_{i1} + \boldsymbol{\delta}_{i2}, \|\boldsymbol{\delta}_{i1}\|_\infty \leq z_i, \|\boldsymbol{\delta}_{i2}\|_\infty \leq 1 - z_i, \forall i \in [n], \\
& \boldsymbol{x} = \sum_{j \in [d]} \boldsymbol{\sigma}_j, \|\boldsymbol{\sigma}_j\|_\infty \leq y_j, \sigma_{jj} = y_j, \forall j \in [d], \sum_{j \in [d]} y_j = 1, \\
& \boldsymbol{x} = \boldsymbol{\mu}_{\ell 1} + \boldsymbol{\mu}_{\ell 2}, \|\boldsymbol{\mu}_{\ell 1}\|_\infty \leq \alpha_\ell, \|\boldsymbol{\mu}_{\ell 2}\|_\infty \leq 1 - \alpha_\ell, \forall \ell \in [m], \\
& w = w_U - (w_U - w_L)\left(\sum_{i \in [m]} 2^{-i} \alpha_i\right), \\
& \left\| \sum_{i \in [n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1} - w_U \boldsymbol{x} + (w_U - w_L) \sum_{\ell \in [m]} 2^{-\ell} \boldsymbol{\mu}_{\ell 1} \right\|_\infty \leq \epsilon, \\
& \boldsymbol{\alpha} \in \{0,1\}^m, \boldsymbol{y} \in \{0,1\}^d,
\end{aligned}
\tag{22}
$$

where w_L and w_U denote the lower and upper bounds of SPCA, respectively and $m := \lceil \log_2((w_U - w_L)\epsilon^{-1}) \rceil$.

Proof. See Appendix A.7. □

We remark about Theorem 4 that

(i) Theorem 4 provides the first-known MILP formulation (22) with arbitrary accuracy of $O(\epsilon)$ for SPCA;

(ii) It is worth noting that the MILP (22) relies on binarizing a continuous variable w, which can potentially result in poor performance (see, e.g., [49, 52]). Our numerical results also demonstrate that the branch-and-cut algorithm based on MISDP (6) exhibits higher efficiency compared to directly solving the MILP (22) on large-scale instances;

(iii) Strong lower and upper bounds of SPCA can speed up the solution process, as the number of binary variables $\boldsymbol{\alpha}$ in the MILP (22) decreases with the difference between w_L and w_U; and

(iv) One possible approach to enhance the computational efficiency is by decomposing the MILP formulation (22) into d smaller-sized subproblems and then considering each set in the union $\cup_{j \in [d]} \{\boldsymbol{x} : x_j = 1, \|\boldsymbol{x}\|_\infty \leq 1\}$, respectively, as summarized below.

Corollary 1 *For any $\epsilon > 0$, the optimal value of MILP* (22) *is equal to $\widehat{w}(\epsilon) = \max_{j\in[d]} \widehat{w}_j(\epsilon)$, where for each $j \in [d]$, $\widehat{w}_j(\epsilon)$ is defined as*

$$\begin{aligned}\widehat{w}_j(\epsilon) := & \max_{w,\boldsymbol{z}\in Z,\boldsymbol{y},\boldsymbol{\alpha},\boldsymbol{x},\boldsymbol{\delta},\boldsymbol{\mu}} w \\ \text{s.t.} \quad & \boldsymbol{x} = \boldsymbol{\delta}_{i1} + \boldsymbol{\delta}_{i2}, \|\boldsymbol{\delta}_{i1}\|_\infty \le z_i, \|\boldsymbol{\delta}_{i2}\|_\infty \le 1 - z_i, \forall i \in [n], \\ & \|\boldsymbol{x}\|_\infty \le 1, x_j = 1, \\ & \boldsymbol{x} = \boldsymbol{\mu}_{\ell 1} + \boldsymbol{\mu}_{\ell 2}, \|\boldsymbol{\mu}_{\ell 1}\|_\infty \le \alpha_\ell, \|\boldsymbol{\mu}_{\ell 2}\|_\infty \le 1 - \alpha_\ell, \forall \ell \in [m], \\ & w = w_U - (w_U - w_L)\left(\sum_{i\in[m]} 2^{-i}\alpha_i\right), \\ & \left\| \sum_{i\in[n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1} - w_U \boldsymbol{x} + (w_U - w_L) \sum_{i\in[m]} 2^{-i}\boldsymbol{\mu}_{i1} \right\|_\infty \le \epsilon, \\ & \boldsymbol{\alpha} \in \{0,1\}^m,\end{aligned} \tag{23}$$

where w_L and w_U denote the lower and upper bounds of SPCA, respectively and $m := \lceil \log_2((w_U - w_L)\epsilon^{-1}) \rceil$.

While the MILP (23) has a smaller size, it may be infeasible in some cases. Since the optimal value of an infeasible maximization problem is $-\infty$ by default, the result in Corollary 1 still holds.

4.2. Theoretical Optimality Gap

Analogous to the other two exact formulations, we are interested in deriving the theoretical guarantee when the binary $\boldsymbol{z} \in Z$ of MILP (22) becomes continuous. Notably, our result suggests that the upper bound obtained from the MILP (22) is generally worse than the previous ones in terms of the optimaity gap.

Proposition 7 *Given a threshold $\epsilon > 0$, by relaxing the binary variables $\boldsymbol{z}$ to be continuous, let $\overline{w}_5(\epsilon)$ denote the optimal value of the relaxed MILP formulation* (22)*. Then we have*

$$\overline{w}_5(\epsilon) \le \min\left\{k(\sqrt{d}/2 + 1/2),\ n/k\sqrt{d} + (n-k)(\sqrt{d}/2 + 1/2)\right\} w^* + \epsilon\sqrt{d}.$$

Proof. See Appendix A.8. □

5. Approximation Algorithms

In this section, motivated by the combinatorial formulation (4) of SPCA, we derive the approximation ratios of the well-known greedy and local search algorithms. We also construct worst-case examples to show the tightness of both ratios when $k \le n/2$.

5.1. Greedy Algorithm

In this subsection, we guarantee the worst-case performance of the greedy algorithm, proposed by [20] in section 3.2. Their greedy algorithm for SPCA proceeds as follows. Given a subset $\widehat{S}_G \subseteq [n]$ denoting the selected vectors, the algorithm aims to find a new vector from the unchosen set $\{\boldsymbol{c}_i\}_{i\in[n]\setminus\widehat{S}_G}$ to maximize the largest eigenvalue until the subset $\widehat{S}_G$ attaining the size k. The detailed implementation can be found in Algorithm 1.

Algorithm 1 Greedy algorithm for SPCA (4) proposed by [20]

```
Input: matrix $\boldsymbol{A} \in S_+^n$ of rank $d$, integer $k \in [n]$, and $\widehat{S}_G := \emptyset$
Compute the Cholesky factorization $\boldsymbol{A} = \boldsymbol{C}^\top \boldsymbol{C}$ of matrix $\boldsymbol{A}$ where $\boldsymbol{C} \in \mathbb{R}^{d\times n}$
Let $\boldsymbol{c}_i \in \mathbb{R}^d$ denote the $i$-th column vector of matrix $\boldsymbol{C}$ for each $i \in [n]$
for $\ell = 1, \cdots, k$ do
    Compute $j^* \in \arg\max_{j\in[n]\setminus\widehat{S}} \lambda_{\max}(\sum_{i\in\widehat{S}_G\cup\{j\}} \boldsymbol{c}_i\boldsymbol{c}_i^\top)$
    Add $j^*$ to the set $\widehat{S}_G$
end for
Output: $\widehat{S}_G$
```

We prove a $1/k$-approximation ratio for greedy Algorithm 1 in the below.

Theorem 5 *The greedy Algorithm 1 yields a $1/k$-approximation ratio for SPCA (4), i.e., the output $\widehat{S}_G$ of Algorithm 1 satisfies*

$$\lambda_{\max}\left(\sum_{i\in\widehat{S}_G} \boldsymbol{c}_i\boldsymbol{c}_i^\top\right) \geq \frac{1}{k} w^*.$$

Proof. Suppose that $S^* \subseteq [n]$ is an optimal solution of SPCA (4) is S^*. Then we have

$$\lambda_{\max}\left(\sum_{i\in S^*} \boldsymbol{c}_i\boldsymbol{c}_i^\top\right) \leq \sum_{i\in S^*} \lambda_{\max}(\boldsymbol{c}_i\boldsymbol{c}_i^\top) \leq k \max_{i\in[n]} \lambda_{\max}(\boldsymbol{c}_i\boldsymbol{c}_i^\top) \leq k\lambda_{\max}\left(\sum_{i\in\widehat{S}_G} \boldsymbol{c}_i\boldsymbol{c}_i^\top\right),$$

where the first inequality results from the convexity of the largest eigenvalue function and the last one is because the greedy Algorithm 1 chooses the largest-length vector at the first iteration. □

The approximation ratio $1/k$ of greedy Algorithm 1 is tight when $k \leq n/2$ since we construct an example whose greedy optimum exactly attain this ratio. The worst-case example is presented below.

Example 1 *For any integer k, let $d = k+1$, $n = 2k$, and we define the vectors $\{\boldsymbol{c}_i\}_{i\in[n]} \subseteq \mathbb{R}^d$ to be*

$$\boldsymbol{c}_i = \begin{cases} \boldsymbol{e}_i, & \text{if } i \in [k], \\ \boldsymbol{e}_{k+1}, & \text{if } i \in [k+1, n], \end{cases} \forall i \in [n].$$

Proposition 8 *The approximation ratio $1/k$ of greedy Algorithm 1 is tight when $k \le n/2$.*

Proof. For Example 1, the greedy Algorithm 1 sequentially selects $\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_k$, i.e., the output set is $\widehat{S}_G = [k]$. Thus, the output value of the greedy Algorithm 1 equals 1.

On the other hand, the true optimal value of Example 1 is equal to $\lambda_{\max}\left(\sum_{i\in[k+1,n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top\right) = \lambda_{\max}\left(k\boldsymbol{e}_{k+1}\boldsymbol{e}_{k+1}^\top\right) = k$. This completes the proof. □

5.2. Local Search Algorithm

The local search algorithm has been widely used to solve various machine learning and data analytics problems, including experimental design [40] and the maximum entropy sampling problem [38]. This subsection studies the local search algorithm for solving SPCA (4) and establishes its approximation ratio.

Specifically, the local search algorithm for SPCA (4) proceeds as follows: (i) initialize a feasible solution $\widehat{S}$; (ii) at each iteration, swap an element in $\widehat{S}$ with an element in $[n] \setminus \widehat{S}$ and we update the chosen set $\widehat{S}$ if the swapping strictly increases the objective value of SPCA (4); and (iii) the algorithm terminates when there is no improvement. More details are presented in Algorithm 2.

Algorithm 2 Local search algorithm for SPCA (4)

```
Input: matrix $\boldsymbol{A} \in \mathcal{S}_+^n$, integer $k \in [n]$, and initialize a size-$k$ subset $\widehat{S}_L \subseteq [n]$
Compute the Cholesky factorization $\boldsymbol{A} = \boldsymbol{C}^\top \boldsymbol{C}$ of matrix $\boldsymbol{A}$ where $\boldsymbol{C} \in \mathbb{R}^{d\times n}$
Let $\boldsymbol{c}_i \in \mathbb{R}^d$ denote the $i$-th column vector of matrix $\boldsymbol{C}$ for each $i \in [n]$
do
    for each pair $(i,j) \in \widehat{S}_L \times ([n] \setminus \widehat{S}_L)$ do
        if $\lambda_{\max}\left(\sum_{\ell\in\widehat{S}_L\cup\{j\}\setminus\{i\}} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\right) > \lambda_{\max}\left(\sum_{\ell\in\widehat{S}_L} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\right)$ then
            Update $\widehat{S}_L := \widehat{S}_L \cup \{j\} \setminus \{i\}$
        end if
    end for
while there is still an improvement
Output: $\widehat{S}_L$
```

Theorem 6 *The local search Algorithm 2 yields a $1/k$-approximation ratio of SPCA, i.e., the output $\widehat{S}_L$ of the local search Algorithm 2 satisfies*

$$\lambda_{\max}\left(\sum_{i\in\widehat{S}_L} \boldsymbol{c}_i \boldsymbol{c}_i^\top\right) \ge \frac{1}{k} w^*.$$

Proof. First, for each $j \in [n]$, we show that

$$\lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg) \geq \lambda_{\max}(\boldsymbol{c}_j \boldsymbol{c}_j^\top). \tag{24}$$

To prove it, there are two cases to be discussed depending on whether j belongs to $\widehat{S}_L$ or not.

(i) $j \in \widehat{S}_L$. The monotonicity of the largest eigenvalue of the sum of positive semidefinite matrices results in the inequality (24).

(ii) $j \in [n] \setminus \widehat{S}_L$. Then, the local optimality condition implies that there exists some $i \in \widehat{S}_L$ such that

$$\lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg) \geq \lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L \cup \{j\} \setminus \{i\}} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg) \geq \lambda_{\max}(\boldsymbol{c}_j \boldsymbol{c}_j^\top),$$

where the second inequality follows the analysis of Part (i).

Second, suppose that S^* is an optimal solution to SPCA (4). Then, by the inequality (24), we have

$$w^* = \lambda_{\max}\Bigg(\sum_{i \in S^*} \boldsymbol{c}_i \boldsymbol{c}_i^\top\Bigg) \leq \sum_{i \in S^*} \lambda_{\max}(\boldsymbol{c}_i \boldsymbol{c}_i^\top) \leq k\lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg),$$

where the first inequality is because of the convexity of the largest eigenvalue function. □

We remark that Example 1 also confirms the tightness of our analysis for the local search Algorithm 2 when $k \leq n/2$.

Proposition 9 *The approximation ratio $1/k$ of local search Algorithm 2 is tight when $k \leq n/2$.*

Proof. In Example 1, we show that the initial subset $\widehat{S}_L = [k]$ satisfies the local optimality condition. For each pair $(i, j) \in \widehat{S}_L \times ([n] \setminus \widehat{S}_L)$, we have

$$\lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L \cup \{j\} \setminus \{i\}} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg) = \lambda_{\max}(\boldsymbol{I}_d - \boldsymbol{e}_i \boldsymbol{e}_i^\top) = 1 = \lambda_{\max}(\boldsymbol{I}_d - \boldsymbol{e}_d \boldsymbol{e}_d^\top) = \lambda_{\max}\Bigg(\sum_{\ell \in \widehat{S}_L} \boldsymbol{c}_\ell \boldsymbol{c}_\ell^\top\Bigg),$$

where the identities are from the definitions of vectors $\{\boldsymbol{c}_i\}_{i \in [n]}$ in Example 1.

Therefore, the set $\widehat{S}_L$ achieves the local optimum with the largest eigenvalue of 1. Since the optimal value of SPCA is $w^* = k$, the approximation ratio of set $\widehat{S}_L$ is equal to k^{-1}. □

In practice, the local search Algorithm 2 may enhance the performance of the greedy Algorithm 1 by using its output as an initial solution, and our numerical study demonstrates that this integration works effectively. Nevertheless, the results in Theorem 6 and Proposition 9 indicate that the integrated algorithm still achieves a $1/k$-approximation ratio. In addition, we apply the power iteration method to calculate the largest eigenvalue [3] for the efficient implementation of the greedy Algorithm 1 and the local search Algorithm 2.

We close this subsection by introducing an enhancement to the local search Algorithm 2. Specifically, we consider increasing the number of swapping elements at Step 5 in Algorithm 2, which we refer to as the s-swap local search for any $s \in [k]$. This improved s-swap local search achieves a better guarantee, as shown below.

Corollary 2 *The approximation ratio of the s-swap local search is s/k for any $s \in [k]$. The ratio is tight when $k \leq n/2$.*

Proof. See Appendix A.9. □

We note that while theoretically performing well, the s-swap local search with $s \geq 2$ may not be practical due to its exponential time complexity in s. Therefore, we use the original local search Algorithm 2 in the numerical study.

6. Numerical Study

This section presents numerical experiments testing our proposed formulations and algorithms with varying-scale instances, where the dimension n spans from 13 to 2365. All the methods are implemented in Python 3.6 with calls to Gurobi 9.5.2 and MOSEK 10.0.29 on a PC equipped with a 2.3 GHz Intel Core i5 processor and 8G of memory. The codes and data used in our experiments are available at `https://github.com/yongchunli-13/Sparse-PCA`.

6.1. A comparison of exact methods for SPCA

In this subsection, we compare the computational efficiency of our exact methods, including the MISDP (6), MISDP (15), and MILP (22) on various small- and medium-sized real datasets [33, 46]. We use the custom branch-and-cut method to solve the two MISDPs. In contrast, the MILP (22) can be directly solved by the Gurobi solver. It is worth noting that the MISDP (6) admits the closed-form cuts that do not require solving dual problems, as shown in Proposition 2, while the MISDP (15) requires solving the SDP problem (19) to obtain a valid cut. This makes solving the MISDP (6) more efficient than the MISDP (15). Throughout this section, we set the target accuracy of the MILP (22) as $\epsilon := 10^{-4}$. We also test Gurobi to solve the following nonconvex SPCA formulation for comparison purposes.

$$w^* := \max_{\boldsymbol{z} \in Z, \boldsymbol{x} \in \mathbb{R}^n} \left\{ \boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x} : \|\boldsymbol{x}\|_2 = 1, \|\boldsymbol{x}\|_1 \leq \sqrt{k}, |x_i| \leq z_i, \forall i \in [n] \right\}. \tag{25}$$

First, in Table 2, we benchmark the proposed methods on the commonly-used *Pitprops* dataset with 13 features (i.e., $n = 13$) [33], testing seven cases with k chosen from $\{4, \cdots, 10\}$. We let **time(s)** denote the running time in seconds and let **SPCA** (25) denote the performance of Gurobi

Table 2 Computational results of exact methods on the Pitprops dataset

Case		MISDP (6)		MISDP (15)		MILP (22)		SPCA (25)	
n	k	w^*	time(s)	w^*	time(s)	$\widehat{w}(\epsilon)$	time(s)	w^*	time(s)
13	4	2.9375	1	2.9375	2	2.9375	1	2.9375	1
13	5	3.4062	1	3.4062	2	3.4062	1	3.4062	3
13	6	3.7710	1	3.7710	2	3.7710	2	3.7710	1
13	7	3.9962	1	3.9962	1	3.9962	1	3.9962	3
13	8	4.0686	1	4.0686	2	4.0686	2	4.0686	10
13	9	4.1386	1	4.1386	2	4.1386	1	4.1387	10
13	10	4.1726	1	4.1726	1	4.1726	1	4.1726	5

9.5.2 when solving the nonconvex SPCA (25). It is seen in Table 2 that our proposed exact formulations (6), (15), and (22) successfully solve all cases to optimality within seconds. Besides, they demonstrate a higher efficiency compared to directly using Gurobi 9.5.2.

To obtain a comprehensive understanding of the overall performance of our exact methods, we further conduct experiments on ten UCI datasets [46] with sizes ranging from 13 to 128. The information on each dataset is summarized in Table 3.

Table 3 Description of UCI datasets used

Dataset	Dimension n	Number of samples	Reference
housing	13	506	[29]
keggdirected	20	48827	[43]
pol	26	15000	[26]
wdbc	30	569	[54]
dermatology	34	366	[32]
spambase	57	4601	[31]
digits	64	1797	[1]
buzz	77	583250	[36]
song	90	515345	[10]
gas	128	2565	[51]

Table 4 presents the computational results. For each UCI dataset, we consider two different values of k. It is important to note that we let "–" denote the unsolved cases within one hour throughout this section, as we set a one-hour time limit. According to the results in Table 4, it is evident that when solving the UCI datasets in Table 3, the computational efficiency of the proposed methods follows a descending pattern with the MISDP (6) being the most efficient, followed by the MILP (22), and concluding with the MISDP (15). The observed efficiency sequence of exact methods aligns with our theoretical analysis. As mentioned previously, the closed-form cuts for MISDP (6) derived in Proposition 2 can significantly enhance the branch-and-cut performance compared to the MISDP (15). Besides, the last column in Table 4 implies that Gurobi 9.5.2 can only solve a limited number of cases. Notably, there are only three testing cases in Table 4 where the MILP (22) outperforms the branch-and-cut algorithm based on the MISDP (6). Therefore, we recommend using the branch-and-cut algorithm to solve the MISDP (6) to optimality when $n = 100s$. When implementing the branch-and-cut algorithm, we first compute continuous relaxations and lower bounds returned by approximation algorithms as the warm start. Hence, having tight upper and lower bounds is desirable to expedite the branch-and-cut algorithm. We will evaluate the performance of our proposed relaxations and approximation algorithms in the following subsections.

Table 4 Computational results of exact methods on UCI datasets

Dataset	Case		MISDP (6)		MISDP (15)		MILP (22)		SPCA (25)	
	n	k	w^*	time(s)	w^*	time(s)	$\widehat{w}(\epsilon)$	time(s)	w^*	time(s)
housing	13	5	3.7239	1	3.7239	82	3.7239	2	3.7239	1
	13	10	3.7342	1	3.7342	49	3.7342	2	3.7342	10
keggdirected	20	5	451.5948	1	–	3600	451.5948	3	451.5948	1
	20	15	451.9241	7	–	3600	451.9241	3	–	3600
pol	26	5	36.5574	30	–	3600	36.5574	50	36.5574	5
	26	15	38.8281	40	–	3600	38.8281	907	38.8281	924
wdbc	30	5	5.4683	1	–	3600	5.4683	100	5.4683	7
	30	15	5.6588	1	–	3600	–	3600	–	3600
dermatology	34	5	3.3751	1	–	3600	3.3751	534	3.3751	13
	34	15	3.4161	87	–	3600	–	3600	–	3600
spambase	57	10	41.8519	727	–	3600	41.8519	21	–	3600
	57	20	–	3600	–	3600	41.8587	33	–	3600
digits	64	10	5.8801	439	–	3600	–	3600	–	3600
	64	20	–	3600	–	3600	–	3600	–	3600
buzz	77	10	2472.3111	28	–	3600	–	3600	–	3600
	77	20	3993.1748	146	–	3600	–	3600	–	3600
song	90	10	2112.4768	17	–	3600	–	3600	–	3600
	90	20	–	3600	–	3600	–	3600	–	3600
gas	128	10	18.2092	1	–	3600	18.2092	290	–	3600
	128	20	18.6831	31	–	3600	18.6831	291	–	3600

6.2. A comparison of continuous relaxations for SPCA

In this subsection, we benchmark the performance of the proposed continuous relaxations for SPCA (1). In addition to the datasets used in Tables 2 and 4, we extend our evaluation to include four larger datasets: *Eisen-1*, *Eisen-2*, *Colon*, and *Reddit*, each with dimensions n =79, 118, 500, and 2000, respectively. These datasets have been studied in the SPCA literature [23].

We obtain the continuous relaxations by relaxing binary variables $\boldsymbol{z} \in \mathcal{Z}$ in our exact formulations. These relaxations provide upper bounds for SPCA (1). Specifically, the two MISDP formulations (6) and (15) lead to the SDP relaxations (8) and (16), respectively. We use MOSEK to solve the SDP relaxations (8) and (16) directly. It is interesting to note that the SDP problem (8) inspires us an additional continuous relaxation (13) for SPCA, as detailed in Theorem 2. The relaxation (13) is free of solving SDPs and greatly improves the scalability. The continuous relaxation (13) allows for an efficient subgradient method to solve it. Although the theoretical guarantees of our proposed relaxations (8), (13), and (16) remain consistent, their practical performance varies significantly. Given the superior performance of both SDP relaxations compared to the continuous relaxation of the MILP (22), we omit reporting computational results for the latter relaxation. In addition, the SDP relaxation (17) proposed by d'Aspremont et al. [21] serves as a benchmark upper bound for SPCA.

The numerical results for the small *Pitprops* dataset can be found in Table 5, where **gap(%)** represents the optimality gap of the upper bound and is defined by $100 \times (\text{Upper Bound} - w^*)/w^*$. We see that the SDP relaxation (16) achieves the smallest gaps in the first five cases. When k is close to n, the SDP relaxation (8) tend to dominate the others. There is a trade-off between efficiency and solution quality when comparing relaxations (8) and (13), as further demonstrated in Table 6. Notably, the SDP relaxation (16) is consistently superior to the benchmark relaxation (17), which aligns with the theoretical result in Proposition 4. However, the performance of our SDP relaxation (8) is not comparable to the benchmark relaxation (17).

Table 5 Computational results of continuous relaxations on the Pitprops dataset

n=13	Relaxation (8)		Relaxation (13)		Relaxation (16)		Benchmark (17)	
k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
4	5.75	1	43.61	2	0.41	1	2.71	1
5	2.37	1	23.85	3	0.18	1	1.52	1
6	0.39	1	11.87	3	0.15	1	1.13	1
7	0.00	1	5.57	3	0.00	1	0.89	1
8	0.29	1	3.69	3	0.26	1	1.87	1
9	0.00	1	1.93	3	0.03	1	1.64	1
10	0.09	1	1.10	2	0.12	1	1.10	1

Tables 6 and 7 provide a comprehensive comparison of the proposed relaxations across various UCI datasets and large-scale datasets. When the optimal value is unavailable, we use the lower bound returned by the local search Algorithm 2 to compute **gap(%)**. For cases where $n \leq 34$, our SDP relaxation (8) can solve them to optimality within one hour. Compared to the SDP relaxation (8), the SDP relaxation (16) extends the problem-solving capacity to cases with $n \leq 100$ and achieves an optimality gap of at most 0.72%. Likewise, the benchmark SDP relaxation (17) fails to return any upper bound for cases where $n \geq 100$, as shown in Table 7. In contrast, the computational efficiency of our relaxation (13) stands out, as it can efficiently handle cases with $n \geq 100$. It is also seen in Table 7 that the benchmark relaxation (17) is not comparable with our efficient relaxation (13). In summary, we recommend using the SDP relaxation (16) to obtain an upper bound of SPCA (1) if $n \leq 100$. For the large-scale SPCA problem, we recommend using the relaxation (13).

Table 6 Computational results of continuous relaxations on UCI datasets

Dataset	Case		Relaxation (8)		Relaxation (13)		Relaxation (16)		Benchmark (17)	
	n	k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
housing	13	5	0.00	1	0.28	7	0.00	1	0.28	1
	13	10	0.00	1	0.00	7	0.00	1	0.00	1
keggdirected	20	5	0.00	3	0.07	10	0.00	1	0.07	1
	20	15	0.00	2	0.00	10	0.00	1	0.00	1
pol	26	5	0.00	26	6.36	15	0.00	1	4.96	1
	26	15	0.00	29	0.14	15	0.00	1	0.14	1
wdbc	30	5	0.00	34	0.19	15	0.00	2	0.19	1
	30	15	0.00	40	0.00	16	0.00	2	0.00	1
dermatology	34	5	0.00	201	1.50	17	0.00	1	1.50	1
	34	15	0.00	226	0.28	17	0.00	4	0.28	1
spambase	57	10	–	3600	0.02	30	0.00	74	0.02	9
	57	20	–	3600	0.01	30	0.00	85	0.01	11
digits	64	10	–	3600	88.51	36	0.38	84	2.82	26
	64	20	–	3600	31.31	36	0.72	91	4.07	24
buzz	77	10	–	3600	70.75	43	0.25	354	2.16	98
	77	20	–	3600	26.77	44	0.62	406	3.59	135
song	90	10	–	3600	5.92	60	0.00	925	1.91	249
	90	20	–	3600	2.53	61	0.00	1095	0.40	229
gas	128	10	–	3600	2.60	86	–	3600	–	3600
	128	20	–	3600	0.00	86	–	3600	–	3600

6.3. A comparison of approximation algorithms for SPCA

This subsection numerically demonstrates the scalability and high-quality outputs of our approximation algorithms using various real datasets. We compare the optimality gaps of Algorithms 1 and 2 with existing ones, including the truncation algorithm [16] and the randomized and SDP-based algorithms proposed by [17]. Notably, the thresholding algorithm in [17] reduces to the truncation

Table 7 **Computational results of continuous relaxations on four large datasets**

Dataset	Case		Relaxation (8)		Relaxation (13)		Relaxation (16)		Benchmark (17)	
	n	k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
Eisen-1	79	10	–	3600	4.38	50	0.00	461	2.10	124
	79	20	–	3600	2.27	50	0.00	376	2.27	110
Eisen-2	118	10	–	3600	48.88	74	–	3600	-	3600
	118	20	–	3600	30.19	74	–	3600	-	3600
Colon	500	10	–	3600	43.49	278	–	3600	-	3600
	500	20	–	3600	43.29	279	–	3600	-	3600
Reddit	2000	10	–	3600	–	3600	–	3600	-	3600
	2000	20	–	3600	–	3600	–	3600	-	3600

of the top eigenvector and is dominated by the truncation algorithm [16]. Hence, we exclude the thresholding algorithm from our comparison.

First, Table 8 displays the computational results for the *Pitprops* dataset, where we compute **gap(%)** as $100 \times (w^* - \text{lower bound})/w^*$ to evaluate the lower bounds. Note that we initialize the local search Algorithm 2 using the output of the greedy Algorithm 1. To enhance the computation, we employ the power iteration method to compute the largest eigenvalue [3]. We see in Table 8 that the greedy Algorithm 1 and local search Algorithm 2 successfully find the optimal solutions and outperform the others. Since both the randomized and SDP-based algorithms involve randomization, and for them, we select the best output from 50 samples for each algorithm.

Table 8 **Computational results of approximation algorithms on the Pitprops dataset**

n=13	Truncation algorithm [16]		Randomized algorithm [17]		SDP-based algorithm [17]		Greedy Algorithm 1		Local Search Algorithm 2	
k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
4	1.57	1	0.13	1	0.00	1	0.00	1	0.00	1
5	0.32	1	11.01	1	0.00	1	0.00	1	0.00	1
6	0.36	1	16.33	1	0.00	1	0.00	1	0.00	1
7	0.08	1	10.76	1	0.00	1	0.00	1	0.00	1
8	0.09	1	0.16	1	14.40	1	0.00	1	0.00	1
9	0.18	1	0.89	1	2.59	1	0.00	1	0.00	1
10	3.91	1	0.08	1	1.34	1	0.00	1	0.00	1

Then, we evaluate the performance of various approximation algorithms on larger datasets, as shown in Tables 9 and 10. For instances where the exact methods do not achieve optimality within one hour, we replace the optimal value with the best lower bound to calculate the **gap(%)**. We see that for all testing cases, the local search Algorithm 2 achieves the smallest gap, offering the best lower bound. The SDP-based algorithm relies on solving an SDP relaxation of SPCA and is less scalable. Our computational experiments show that the local search Algorithm 2 outperforms the other methods. Therefore, we recommend using this algorithm to solve practical SPCA problems.

Table 9 Computational results of approximation algorithms on UCI datasets

Dataset	Case		Truncation algorithm [16]		Randomized algorithm [17]		SDP-based algorithm [17]		Greedy Algorithm 1		Local Search Algorithm 2	
	n	k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
housing	13	5	5.67	1	0.13	1	0.00	1	0.00	1	0.00	1
	13	10	5.94	1	0.40	1	0.00	1	0.00	1	0.00	1
keggdirected	20	5	0.00	1	0.06	1	0.04	1	0.00	1	0.00	1
	20	15	0.00	1	0.11	1	0.00	1	0.00	1	0.00	1
pol	26	5	1.99	1	2.99	1	5.67	1	0.00	1	0.00	1
	26	15	15.58	1	1.78	1	0.56	1	0.00	1	0.00	1
wdbc	30	5	0.00	1	1.04	1	0.91	1	0.00	1	0.00	1
	30	15	0.00	1	1.22	1	0.00	1	0.00	1	0.00	1
dermatology	34	5	0.00	1	0.97	1	0.73	1	0.00	1	0.00	1
	34	15	0.00	1	1.54	1	0.57	1	0.00	1	0.00	1
spambase	57	10	0.00	1	0.22	1	0.13	1	0.00	1	0.00	1
	57	20	0.00	1	0.08	1	0.02	1	0.00	1	0.00	1
digits	64	10	0.01	1	23.58	1	1.85	1	0.00	1	0.00	1
	64	20	0.00	1	8.46	1	5.71	1	0.00	1	0.00	1
buzz	77	10	0.01	1	14.87	1	0.00	27	0.00	1	0.00	1
	77	20	0.00	1	6.97	1	3.99	28	0.00	1	0.00	1
song	90	10	0.00	1	2.38	1	5.65	40	0.00	1	0.00	1
	90	20	1.43	1	1.63	1	0.25	39	0.00	1	0.00	1
gas	128	10	0.01	1	2.62	1	1.13	80	0.00	1	0.00	1
	128	20	0.00	1	1.54	1	0.00	66	0.00	1	0.00	1

6.4. *Drugabuse* Dataset

In this subsection, we apply the proposed local search Algorithm 2 to the *Drugabuse* dataset with $n = 2365$ features, where the dataset comes from a questionnaire collected by the National Survey on Drug Use and Health (NSDUH) in 2018. It has been reported [48] that with the growing illicit online sale of controlled substances, deaths attributable to opioid-related drugs have quadrupled in the U.S. since 1999. Thus, it is important to select a handful of features that domain experts can further investigate. SPCA serves as an excellent tool for selecting the most representative features. In Figure 1, we present the selected $k = 10$ features, where the vertical values correspond to the selected features of the first PC scaled by 100. Among the ten selected features, there are three categories: inhalants, drug injection, and drug treatment, which play a crucial role in the analysis

Table 10 Computational results of approximation algorithms on four large datasets

Dataset	Case		Truncation algorithm [16]		Randomized algorithm [17]		SDP-based algorithm [17]		Greedy Algorithm 1		Local Search Algorithm 2	
	n	k	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
Eisen-1	79	10	10.61	1	4.28	1	14.55	5	0.00	1	0.00	1
	79	20	15.47	1	1.83	1	13.93	4	0.00	1	0.00	1
Eisen-2	118	10	0.03	1	31.23	1	2.06	41	2.62	1	0.00	1
	118	20	40.41	1	27.95	1	2.21	43	0.00	1	0.00	1
Colon	500	10	2.87	1	61.26	20	–	3600	0.00	1	0.00	1
	500	20	29.14	1	52.93	20	–	3600	0.01	1	0.00	2
Reddit	2000	10	2.56	3	18.64	283	–	3600	0.00	1	0.00	1
	2000	20	1.08	2	22.50	241	–	3600	0.83	4	0.00	1

of drug abuse. Specifically, SPCA selects six features related to drug treatment, which is consistent with the literature [19, 53] that the treatment records of drug abuse are important. The three drug injection features shed light on understanding the injection experiences of different drugs. It is well known that drug injection users face a high risk of contracting HIV and other blood-borne infections [47, 50]. Finally, the inhalants feature contributes to our understanding of the factors contributing to drug abuse [13, 22].

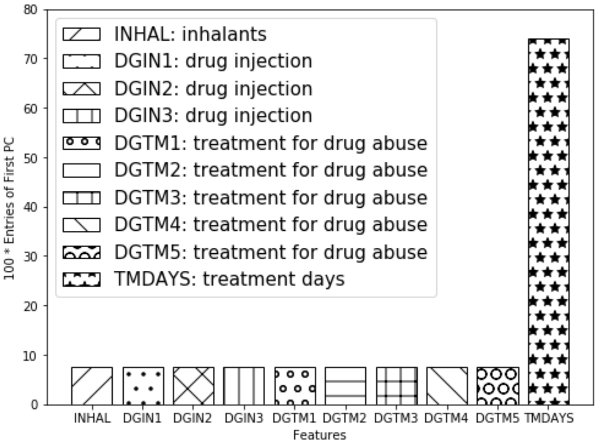

Figure 1 10 features selected by local search Algorithm 2 for Drugabuse dataset

7. Conclusion

This paper investigates the sparse PCA problem by deriving three equivalent mixed-integer exact formulations and studying two approximation algorithms: greedy and local search. We theoretically guarantee the continuous relaxations of exact formulations and the worst-case performance of approximation algorithms. We further develop a branch-and-cut algorithm for solving sparse PCA to optimality. Our numerical study demonstrates the high solution quality and computational efficiency of the proposed formulations and algorithms. The branch-and-cut algorithm manages to solve small and medium instances, and the approximation algorithms consistently yield near-optimal solutions for all the instances. The theoretical optimality gaps of the continuous relaxations may not be sufficiently tight. As a potential avenue for future research, we aim to explore and enhance these gaps.

Acknowledgment

This research has been supported in part by the National Science Foundation grants 224614 and 2246417 and the Office of Naval Research N00014-24-1-2066.

References

[1] Alpaydin E, Kaynak C (1998) Optical Recognition of Handwritten Digits. UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C50P49`.

[2] Amini AA, Wainwright MJ (2008) High-dimensional analysis of semidefinite relaxations for sparse principal components. *2008 IEEE international symposium on information theory*, 2454–2458 (IEEE).

[3] Angelidis G, Semlyen A (1995) Efficient calculation of critical eigenvalue clusters in the small signal stability analysis of large power systems. *IEEE transactions on power systems* 10(1):427–432.

[4] Arous GB, Wein AS, Zadik I (2020) Free energy wells and overlap gap property in sparse PCA. *Conference on Learning Theory*, 479–482 (PMLR).

[5] Balas E (1975) Disjunctive programming: cutting planes from logical conditions. *Nonlinear Programming 2*, 279–312 (Elsevier).

[6] Behdin K, Mazumder R (2021) Sparse PCA: A new scalable estimator based on integer programming. *arXiv preprint arXiv:2109.11142* .

[7] Ben-Tal A, Nemirovski A (2001) *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2 (SIAM).

[8] Benders JF (1962) Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.* 4(1):238–252, ISSN 0029-599X, URL `http://dx.doi.org/10.1007/BF01386316`.

[9] Berk L, Bertsimas D (2019) Certifiably optimal sparse principal component analysis. *Mathematical Programming Computation* 11(3):381–420.

[10] Bertin-Mahieux T (2011) Year Prediction MSD. UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C50K61`.

[11] Bertsimas D, Cory-Wright R (2020) On polyhedral and second-order cone decompositions of semidefinite optimization problems. *Operations Research Letters* 48(1):78–85.

[12] Bertsimas D, Cory-Wright R, Pauphilet J (2022) Solving large-scale sparse PCA to certifiable (near) optimality. *The Journal of Machine Learning Research* 23:13–1.

[13] Breakey WR, Goodell H, Lorenz PC, McHugh PR (1974) Hallucinogenic drugs as precipitants of schizophrenia. *Psychological Medicine* 4(3):255–261.

[14] Carrizosa E, Guerrero V (2014) RS-sparse principal component analysis: A mixed integer nonlinear programming approach with VNS. *Computers & operations research* 52:349–354.

[15] Chaib S, Gu Y, Yao H (2015) An informative feature selection method based on sparse PCA for vhr scene classification. *IEEE Geoscience and Remote Sensing Letters* 13(2):147–151.

[16] Chan SO, Papailliopoulos D, Rubinstein A (2016) On the approximability of sparse PCA. *Conference on Learning Theory*, 623–646.

[17] Chowdhury A, Drineas P, Woodruff DP, Zhou S (2020) Approximation algorithms for sparse principal component analysis. *arXiv preprint arXiv:2006.12748* .

[18] Coope I (1994) On matrix trace inequalities and related topics for products of hermitian matrices. *Journal of mathematical analysis and applications* 188(3):999–1001.

[19] Coughlin LN, Tegge AN, Sheffer CE, Bickel WK (2020) A machine-learning approach to predicting smoking cessation treatment outcomes. *Nicotine and Tobacco Research* 22(3):415–422.

[20] d'Aspremont A, Bach F, El Ghaoui L (2008) Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research* 9(7).

[21] d'Aspremont A, Ghaoui LE, Jordan MI, Lanckriet GR (2005) A direct formulation for sparse PCA using semidefinite programming. *Advances in neural information processing systems*, 41–48.

[22] De Barona MS, Simpson DD (1984) Inhalant users in drug abuse prevention programs. *The American journal of drug and alcohol abuse* 10(4):503–518.

[23] Dey SS, Mazumder R, Wang G (2018) A convex integer programming approach for optimal sparse PCA. *arXiv preprint arXiv:1810.09062* .

[24] Ding Y, Kunisky D, Wein AS, Bandeira AS (2023) Subexponential-time algorithms for sparse PCA. *Foundations of Computational Mathematics* 1–50.

[25] d'Aspremont A, Bach F, Ghaoui LE (2014) Approximation bounds for sparse principal component analysis. *Mathematical Programming* 148:89–110.

[26] Evans T, Grant E (2021) Pol. UCI Machine Learning Repository, `https://github.com/treforevans/uci_datasets`.

[27] Gally T, Pfetsch ME (2016) Computing restricted isometry constants via mixed-integer semidefinite programming. *Available at `https://optimization-online.org/2016/04/5395/`* .

[28] Geoffrion AM (1972) Generalized Benders decomposition. *Journal of optimization theory and applications* 10(4):237–260.

[29] Harrison Jr D, Rubinfeld DL (1978) Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management* 5(1):81–102, `https://www.kaggle.com/datasets/heptapod/uci-ml-datasets/data`.

[30] He Y, Monteiro RD, Park H (2011) An algorithm for sparse PCA based on a new sparsity control criterion. *Proceedings of the 2011 SIAM International Conference on Data Mining*, 771–782 (SIAM).

[31] Hopkins M, Reeber E, Forman G, Suermondt J (1999) Spambase. UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C53G6X`.

[32] Ilter N, Guvenir H (1998) Dermatology. UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C5FK5P`.

[33] Jeffers J (1967) Two case studies in the application of principal component analysis. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 16(3):225–236.

[34] Jiang R, Fei H, Huan J (2012) A family of joint sparse PCA algorithms for anomaly localization in network data streams. *IEEE Transactions on Knowledge and Data Engineering* 25(11):2421–2433.

[35] Journée M, Nesterov Y, Richtárik P, Sepulchre R (2010) Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research* 11(2).

[36] Kawala F, Douzal A, Gaussier E, Diemert E (2013) Buzz in social media. UCI Machine Learning Repository, DOI: https://doi.org/10.24432/C56G6V.

[37] Kim J, Tawarmalani M, Richard JPP (2022) Convexification of permutation-invariant sets and an application to sparse principal component analysis. *Mathematics of Operations Research* 47(4):2547–2584.

[38] Li Y, Xie W (2023) Best principal submatrix selection for the maximum entropy sampling problem: scalable algorithms and performance guarantees. *Operations Research* .

[39] Luss R, d'Aspremont A (2010) Clustering and feature selection using sparse principal component analysis. *Optimization and Engineering* 11(1):145–157.

[40] Madan V, Singh M, Tantipongpipat U, Xie W (2019) Combinatorial algorithms for optimal design. *Conference on Learning Theory*, 2210–2258.

[41] Magdon-Ismail M (2017) NP-hardness and inapproximability of sparse PCA. *Information Processing Letters* 126:35–38.

[42] Moghaddam B, Weiss Y, Avidan S (2005) Spectral bounds for sparse PCA: Exact and greedy algorithms. *Advances in neural information processing systems* 18.

[43] Naeem M, Asghar S (2011) KEGG Metabolic Relation Network (Directed). UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C5CK52`.

[44] Naikal N, Yang AY, Sastry SS (2011) Informative feature selection for object recognition via sparse PCA. *2011 International Conference on Computer Vision*, 818–825 (IEEE).

[45] Nedić A, Ozdaglar A (2009) Subgradient methods for saddle-point problems. *Journal of optimization theory and applications* 142(1):205–228.

[46] Newman D, Hettich S, Blake C, Merz C (1998) UCI repository of machine learning databases. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[47] Ompad DC, Ikeda RM, Shah N, Fuller CM, Bailey S, Morse E, Kerndt P, Maslow C, Wu Y, Vlahov D, et al. (2005) Childhood sexual abuse and age at initiation of injection drug use. *American journal of public health* 95(4):703–709.

[48] Overdose O (2018) Understanding the epidemic. *Atlanta, Centers for Disease Control and Prevention* .

[49] Owen JH, Mehrotra S (2002) On the value of binary expansions for general mixed-integer linear programs. *Operations Research* 50(5):810–819.

[50] Thomas DL, Vlahov D, Solomon L, Cohn S, Taylor E, Garfein R, Nelson KE (1995) Correlates of hepatitis c virus infections among injection drug users. *Medicine* 74(4):212–220.

[51] Vergara A (2012) Gas Sensor Array Drift Dataset. UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C5RP6W`.

[52] Vielma JP, Ahmed S, Nemhauser G (2010) A note on “a superior representation method for piecewise linear functions”. *INFORMS Journal on Computing* 22(3):493–497.

[53] Volkow ND, Fowler JS, Wang GJ, Swanson JM, Telang F (2007) Dopamine in drug abuse and addiction: results of imaging studies and treatment implications. *Archives of neurology* 64(11):1575–1579.

[54] Wolberg W, Mangasarian O, Street N, Street W (1995) Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, DOI: `https://doi.org/10.24432/C5DW2B`.

[55] Yuan XT, Zhang T (2013) Truncated power method for sparse eigenvalue problems. *Journal of Machine Learning Research* 14(4).

[56] Zhang Y, d’Aspremont A, El Ghaoui L (2012) Sparse PCA: Convex relaxations, algorithms and applications. *Handbook on Semidefinite, Conic and Polynomial Optimization*, 915–940 (Springer).

[57] Zhang Y, Ghaoui LE (2011) Large-scale sparse principal component analysis with application to text data. *Advances in Neural Information Processing Systems*, 532–539.

Appendix A. Proofs

A.1 Proof of Proposition 2

Proposition 2 *For the function $H_1(\boldsymbol{z})$ defined in SPCA* (9)*, we have that*

(i) For any $\boldsymbol{z}\in\overline{Z}$, function $H_1(\boldsymbol{z})$ is equivalent to

$$H_1(\boldsymbol{z})=\min_{\boldsymbol{\mu},\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_n\in\mathcal{S}_+^d}\left\{\lambda_{\max}\left(\sum_{i\in[n]}\boldsymbol{Q}_i\right)+\sum_{i\in[n]}\mu_i z_i:\boldsymbol{c}_i\boldsymbol{c}_i^\top\preceq\boldsymbol{Q}_i+\mu_i\boldsymbol{I}_d,0\le\mu_i\le\|\boldsymbol{c}_i\|_2^2,\forall i\in[n]\right\},\tag{10}$$

which is concave in $\boldsymbol{z}$.

(ii) For any binary $\boldsymbol{z}\in Z$, an optimal solution to the problem (10) *is $\mu_i^*=0$ if $z_i=1$ and $\|\boldsymbol{c}_i\|_2^2$ otherwise, and $\boldsymbol{Q}_i^*:=(1-\mu_i^*/\|\boldsymbol{c}_i\|_2^2)\boldsymbol{c}_i\boldsymbol{c}_i^\top$ for each $i\in[n]$.*

Proof. **Part (i).** We split the proof of strong duality into two cases depending on whether $\boldsymbol{z}$ is a relative interior of set $\overline{Z}$ or not.

Case a. Suppose that $\boldsymbol{z}$ is in the relative interior of set $\overline{Z}$, i.e., $0<z_i<1$ for each $i\in[n]$. For the inner maximization problem in (9), we dualize the constraint $\boldsymbol{X}\succeq\boldsymbol{W}_i,\mathrm{tr}(\boldsymbol{W}_i)=z_i$ with Lagrangian multiplier $\boldsymbol{Q}_i\in\mathcal{S}_+^d$ and μ_i for each $i\in[n]$. Note that the constraints $\boldsymbol{X}\succeq\boldsymbol{W}_i,\mathrm{tr}(\boldsymbol{W}_i)=z_i$ for each $i\in[n]$ and $\boldsymbol{X},\boldsymbol{W}_1,\cdots,\boldsymbol{W}_n\in\mathcal{S}_+^d$ can be always strictly satisfied since $0<z_i<1$. Thus, according to the strong duality of the conic optimization problem (see, e.g., Theorem 1.4.4 in [7]), function $H_1(\boldsymbol{z})$ can be rewrite as

$$\min_{\boldsymbol{\mu},\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_n\in\mathcal{S}_+^d}\max_{\boldsymbol{X},\boldsymbol{W}_1,\cdots,\boldsymbol{W}_n\in\mathcal{S}_+^d}\left\{\sum_{i\in[n]}\boldsymbol{c}_i^\top\boldsymbol{W}_i\boldsymbol{c}_i+\sum_{i\in[n]}\mathrm{tr}\left(\boldsymbol{Q}_i(\boldsymbol{X}-\boldsymbol{W}_i)\right)+\sum_{i\in[n]}\mu_i\left(z_i-\mathrm{tr}(\boldsymbol{W}_i)\right):\right.$$
$$\left.\mathrm{tr}(\boldsymbol{X})=1\right\}.\tag{26}$$

Then the inner maximization problem (26) over $\boldsymbol{W}_i$ for each $i\in[n]$ and $\boldsymbol{X}$ yields

$$\max_{\boldsymbol{W}_i\in\mathcal{S}_+^d}\mathrm{tr}\left((\boldsymbol{c}_i\boldsymbol{c}_i^\top-\boldsymbol{Q}_i-\mu_i\boldsymbol{I}_d)\boldsymbol{W}_i\right)=\begin{cases}0, & \boldsymbol{c}_i\boldsymbol{c}_i^\top\preceq\boldsymbol{Q}_i+\mu_i\boldsymbol{I}_d,\\ \infty, & \text{otherwise.}\end{cases}$$
$$\max_{\boldsymbol{X}\in\mathcal{S}_+^d}\left\{\mathrm{tr}\left(\left(\sum_{i\in[n]}\boldsymbol{Q}_i\right)\boldsymbol{X}\right):\mathrm{tr}(\boldsymbol{X})=1\right\}=\lambda_{\max}\left(\sum_{i\in[n]}\boldsymbol{Q}_i\right),$$

where the second identity is due to Part(ii) of Lemma 1.

Thus, problem (26) can be simplified as

$$H_1(\boldsymbol{z})=\min_{\boldsymbol{\mu},\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_n\in\mathcal{S}_+^d}\left\{\lambda_{\max}\left(\sum_{i\in[n]}\boldsymbol{Q}_i\right)+\sum_{i\in[n]}\mu_i z_i:\boldsymbol{c}_i\boldsymbol{c}_i^\top\preceq\boldsymbol{Q}_i+\mu_i\boldsymbol{I}_d,\forall i\in[n]\right\}.\tag{27}$$

We show that for the minimization problem (27), any optimal solution $(\boldsymbol{\mu},\boldsymbol{Q}_1,\cdots,\boldsymbol{Q}_n)$ must satisfy $0\le\mu_i\le\|\boldsymbol{c}_i\|_2^2$ for each $i\in[n]$. We prove it by contradiction. Suppose that there exits an optimal solution $(\boldsymbol{\mu}, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n)$ to the problem (27) such that $\mu_j < 0$ for some $j \in [n]$. Then, we can construct a new feasible solution $(\overline{\boldsymbol{\mu}}, \overline{\boldsymbol{Q}}_1, \cdots, \overline{\boldsymbol{Q}}_n)$, which is exactly equal to $(\boldsymbol{\mu}, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n)$ except

$$\overline{\mu}_j = 0, \overline{\boldsymbol{Q}}_j = \boldsymbol{Q}_j + \mu_j \boldsymbol{I}_d.$$

The new solution yields the objective value

$$H_1(\boldsymbol{z}) + \mu_j - \mu_j z_j = H_1(\boldsymbol{z}) + \mu_j(1 - z_j) < H_1(\boldsymbol{z}),$$

which is a contradiction to the optimality of $(\boldsymbol{\mu}, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n)$. Similarly, suppose that there exits an optimal solution $(\boldsymbol{\mu}, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n)$ to the problem (27) such that $\mu_j > \|\boldsymbol{c}_i\|_2^2$ for some $j \in [n]$. Similarly, we can arrive at a contradiction by defining a new feasible solution $(\overline{\boldsymbol{\mu}}, \overline{\boldsymbol{Q}}_1, \cdots, \overline{\boldsymbol{Q}}_n)$, which is exactly equal to $(\boldsymbol{\mu}, \boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_n)$ except $\overline{\mu}_j = \|\boldsymbol{c}_i\|_2^2$.

Therefore, (27) can be reduced to (10).

Case b. Now we consider the case that $\boldsymbol{z}$ is not in the relative interior of $\overline{Z}$ and define two sets $T_0 := \{i \in [n] : z_i = 0\}$ and $T_1 := \{i \in [n] : z_i = 1\}$. Thus, at least one of the two sets is not empty. In this case, we first observe that $H_1(\boldsymbol{z})$ in (9) is equivalent to

$$H_1(\boldsymbol{z}) := \max_{\boldsymbol{X}, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_d \in \mathcal{S}_+^d} \left\{ \sum_{i \in [n] \setminus (T_0 \cup T_1)} \boldsymbol{c}_i^\top \boldsymbol{W}_i \boldsymbol{c}_i + \sum_{i \in T_1} \boldsymbol{c}_i^\top \boldsymbol{X} \boldsymbol{c}_i : \operatorname{tr}(\boldsymbol{X}) = 1, \right.$$
$$\left. \boldsymbol{X} \succeq \boldsymbol{W}_i, \operatorname{tr}(\boldsymbol{W}_i) = z_i, \forall i \in [n] \setminus (T_0 \cup T_1) \right\}. \tag{28}$$

Next, applying the same procedure as Case a., we have

$$H_1(\boldsymbol{z}) = \min_{\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n] \setminus (T_0 \cup T_1)} \subseteq \mathcal{S}_+^d} \left\{ \lambda_{\max}\left(\sum_{i \in [n] \setminus (T_0 \cup T_1)} \boldsymbol{Q}_i + \sum_{i \in T_1} \boldsymbol{c}_i \boldsymbol{c}_i^\top \right) + \sum_{i \in [n] \setminus (T_0 \cup T_1)} \mu_i z_i : \right.$$
$$\left. \boldsymbol{c}_i \boldsymbol{c}_i^\top \preceq \boldsymbol{Q}_i + \mu_i \boldsymbol{I}_d, 0 \le \mu_i \le \|\boldsymbol{c}_i\|_2^2, \forall i \in [n] \setminus (T_0 \cup T_1) \right\}. \tag{29}$$

To show the equivalence between (29) and (10), it remains to prove that

$$\widehat{H}_1(\boldsymbol{z}) = \min_{\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n]} \subseteq \mathcal{S}_+^d} \left\{ \lambda_{\max}\left(\sum_{i \in [n]} \boldsymbol{Q}_i \right) + \sum_{i \in [n]} \mu_i z_i : \boldsymbol{c}_i \boldsymbol{c}_i^\top \preceq \boldsymbol{Q}_i + \mu_i \boldsymbol{I}_d, 0 \le \mu_i \le \|\boldsymbol{c}_i\|_2^2, \forall i \in [n] \right\}. \tag{30}$$

First, given any feasible solution $(\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n] \setminus (T_0 \cup T_1)})$ to the problem (29), let us augment it by setting $\boldsymbol{Q}_i = \boldsymbol{0}, \mu_i = \|\boldsymbol{c}_i\|_2^2$ for each $i \in T_0$ and $\boldsymbol{Q}_i = \boldsymbol{c}_i \boldsymbol{c}_i^\top, \mu_i = 0$ for each $i \in T_1$. Then $(\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n]})$ is feasible to the problem (30) with the same objective value. Thus, we have $\widehat{H}_1(\boldsymbol{z}) \le H_1(\boldsymbol{z})$.

On the other hand, given any feasible solution $(\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n]})$ to the problem (30), then $(\boldsymbol{\mu}, \{\boldsymbol{Q}_i\}_{i \in [n] \setminus (T_0 \cup T_1)})$ is feasible to the problem (29) a smaller objective value since $\boldsymbol{c}_i \boldsymbol{c}_i^\top \preceq \boldsymbol{Q}_i + \mu_i$ for each $i \in T_1$. Thus, we have $\widehat{H}_1(\boldsymbol{z}) \ge H_1(\boldsymbol{z})$. This completes the proof.

Part (ii). For any $z \in Z$, let set S denote its support. We then construct a pair of the primal and dual solutions to the maximization problem in (9) and its dual (10) as

$$\boldsymbol{X}^* = \boldsymbol{q}_1 \boldsymbol{q}_1^\top, \boldsymbol{W}_i^* = \boldsymbol{X}^*, \forall i \in S, \boldsymbol{W}_i^* = 0, \forall i \in [n] \setminus S,$$
$$\boldsymbol{Q}_i^* = \boldsymbol{c}_i \boldsymbol{c}_i^\top, \mu_i = 0, \forall i \in S, \boldsymbol{Q}_i^* = 0, \mu_i = \|\boldsymbol{c}_i\|_2^2, \forall i \in [n] \setminus S,$$

where $\boldsymbol{q}_1$ denote the eigenvector for the largest eigenvalue of matrix $\sum_{i\in S} \boldsymbol{c}_i \boldsymbol{c}_i^\top$.

According to the results in Lemma 1, the above solutions return the same objective value for primal and dual problems, which is $\lambda_{\max}(\sum_{i\in S} \boldsymbol{c}_i \boldsymbol{c}_i^\top)$. This proves the optimality of the proposed dual solution. □

A.2 Proof of Theorem 2

Theorem 2 *The following hold for the relaxed function $\overline{H}_1(\boldsymbol{z})$:*

(i) For any $\boldsymbol{z} \in \overline{Z}$, function $H_1(\boldsymbol{z})$ is upper bounded by

$$\overline{H}_1(\boldsymbol{z}) := \min_{\boldsymbol{\mu}\in\mathbb{R}^n} \left\{ \lambda_{\max}\left(\sum_{i\in[n]} \left(1 - \frac{\mu_i}{\|\boldsymbol{c}_i\|_2^2}\right) \boldsymbol{c}_i \boldsymbol{c}_i^\top \right) + \sum_{i\in[n]} \mu_i z_i : 0 \le \mu_i \le \|\boldsymbol{c}_i\|_2^2, \forall i \in [n] \right\}; \tag{12}$$

(ii) If $\boldsymbol{z} \in Z$, then $H_1(\boldsymbol{z}) = \overline{H}_1(\boldsymbol{z}) = \lambda_{\max}(\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top)$; and

(iii) The continuous relaxation of SPCA

$$\overline{w}_2 := \max_{\boldsymbol{z}\in\overline{Z}} \overline{H}_1(\boldsymbol{z}) \tag{13}$$

achieves a $\min\{k, n/k\}$ optimality gap, i.e., $w^* \le \overline{w}_1 \le \overline{w}_2 \le \min\{k, n/k\} w^*$, where $\overline{w}_1$ is defined in (8).

Proof.

(i) The conclusion follows by choosing a feasible $\boldsymbol{Q}_i := (1 - \mu_i/\|\boldsymbol{c}_i\|_2^2)\boldsymbol{c}_i \boldsymbol{c}_i^\top$ for each $i \in [n]$ in the representation (10).

(ii) For any $\boldsymbol{z} \in Z$, we derive from Part (ii) in Proposition 2 that $\overline{H}_1(\boldsymbol{z}) \ge \lambda_{\max}(\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top)$. Thus, it is sufficient to show that $\overline{H}_1(\boldsymbol{z}) \le \lambda_{\max}(\sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top)$. Indeed, this can be done simply by letting $\mu_i = 0$ if $z_i = 0$, and $\|\boldsymbol{c}_i\|_2^2$, otherwise in (12).

(iii) By the proof of Proposition 1, to obtain the same optimality gap for $\bar{w}_2$ in (13) as SDP (8), we need to show that $\overline{H}_1(\boldsymbol{z}) \le \sum_{i\in[n]} z_i \boldsymbol{c}_i^\top \boldsymbol{c}_i$ and $\overline{H}_1(\boldsymbol{z}) \le \lambda_{\max}(\boldsymbol{A}) = \lambda_{\max}(\sum_{i\in[n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top)$ for any $\boldsymbol{z} \in \overline{Z}$.

We must have $\overline{H}_1(\boldsymbol{z}) \le \sum_{i\in[n]} z_i \boldsymbol{c}_i^\top \boldsymbol{c}_i$ by by letting $\mu_i = \boldsymbol{c}_i^\top \boldsymbol{c}_i$ for all $i \in [n]$ in (12).

We also have $\overline{H}_1(\boldsymbol{z}) \le \lambda_{\max}(\boldsymbol{A}) = \lambda_{\max}(\sum_{i\in[n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top)$ by letting $\mu_i = 0$ for all $i \in [n]$ in (12).

Then the rest of the proof follows directly from that of Proposition 1 and is thus omitted. □

A.3 Proof of Proposition 3

Proposition 3 *SPCA* (2) *admits the following MISDP formulation:*

$$\text{(SPCA)} \quad w^* := \max_{\boldsymbol{z}\in Z, \boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, X_{ii} \le z_i, \forall i \in [n] \right\}. \tag{14}$$

and its continuous relaxation value is equal to $\lambda_{\max}(\boldsymbol{A})$*.*

Proof.

(i) To show the equivalence of problem (14) and SPCA (2), we only need to show that for any feasible $\boldsymbol{z} \in Z$ with its cardinality k and support $S = \{i : z_i = 1\}$, we must have

$$\max_{\boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, X_{ii} \le z_i, \forall i \in [n] \right\} = \lambda_{\max}(\boldsymbol{A}_{S,S}). \tag{31}$$

Indeed, since $\boldsymbol{X}$ is a positive semidefinite matrix, thus $X_{ii} = 0$ for each $i \in [n] \setminus S$ implies

$$X_{ij} = 0, \forall (i,j) \notin S \times S.$$

The left-hand side of the equation (31) is equivalent to

$$\max_{\boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, X_{ii} \le z_i, \forall i \in [n] \right\} = \max_{\boldsymbol{X}\in\mathcal{S}_+^k} \{\operatorname{tr}(\boldsymbol{A}_{S,S}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1\} = \lambda_{\max}(\boldsymbol{A}_{S,S}),$$

where the second equality is due to Part (ii) in Lemma 1.

(ii) The continuous relaxation value of problem (14) is

$$\overline{w}_3 = \max_{\boldsymbol{z}\in\overline{Z}, \boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1, X_{ii} \le z_i, \forall i \in [n] \right\}.$$

Since $\operatorname{tr}(\boldsymbol{X}) = 1$, thus the linking constraint $X_{ii} \le z_i$ is redundant for each $i \in [n]$. Hence,

$$\overline{w}_3 = \max_{\boldsymbol{X}\in\mathcal{S}_+^n} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{X}) : \operatorname{tr}(\boldsymbol{X}) = 1 \right\} = \lambda_{\max}(\boldsymbol{A}),$$

where the equality is due to Part (ii) in Lemma 1. □

A.4 Proof of Lemma 2

Lemma 2 *The following two inequalities are valid to SPCA* (14)

(i) $\sum_{j\in[n]} X_{ij}^2 \le X_{ii} z_i$ *for all* $i \in [n]$*; and*

(ii) $\left(\sum_{j\in[n]} |X_{ij}|\right)^2 \le k X_{ii} z_i$ *for all* $i \in [n]$*.*

Proof. From the proof of Proposition 3, there must exists an optimal solution $(\boldsymbol{z}^*, \boldsymbol{X}^*)$ of SPCA (14) such that $\boldsymbol{X}^*$ must be rank-one. Thus, without loss of generality, for any feasible solution $(\boldsymbol{z}, \boldsymbol{X})$ of SPCA (14), we can assume that $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^\top$, where $(\boldsymbol{x}, \boldsymbol{z})$ is also feasible to SPCA (2).

Next, we split the proof into two parts.

(i) Since $\boldsymbol{X}=\boldsymbol{x}\boldsymbol{x}^\top$, thus

$$\sum_{j\in[n]} X_{ij}^2=\sum_{j\in[n]} x_i^2x_j^2=x_i^2\le z_iX_{ii}, \forall i\in[n],$$

where the last inequality follows from the facts that $X_{ii}=x_i^2\le z_i$ and z_i is binary for each $i\in[n]$.

(ii) It is known (see, e.g., [23]) that $\|\boldsymbol{x}\|_1\le\sqrt{k}$. Thus,

$$\sum_{j\in[n]}|X_{ij}|=\sum_{j\in[n]}|x_i||x_j|\le\sqrt{k}|x_i|\le\sqrt{k}\sqrt{X_{ii}z_i},$$

where the second inequality is because $X_{ii}=x_i^2\le z_i$ and z_i is binary for each $i\in[n]$. □

A.5 Proof of Proposition 4

Proposition 4 *For the SDP relaxations of SPCA defined in* (16) *and* (17)*, we have that* $\overline{w}_3\le\overline{w}_4$.

Proof. To show that $\overline{w}_4\ge\overline{w}_3$, it is sufficient to prove that any feasible solution $(\boldsymbol{z},\boldsymbol{X})$ of the continuous relaxation problem (16), must satisfy the constraints in the SDP formulation (17).

Clearly, we have $\boldsymbol{X}\in\mathcal{S}_+^n$ and $\operatorname{tr}(\boldsymbol{X})=1$. It remains that $\sum_{i\in[n]}\sum_{j\in[n]}|X_{ij}|\le k$. Indeed, we have

$$\sum_{i\in[n]}\sum_{j\in[n]}|X_{ij}|\le\sum_{i\in[n]}\sqrt{k}\sqrt{X_{ii}z_i}\le\sqrt{k}\sqrt{\sum_{i\in[n]}X_{ii}}\sqrt{\sum_{i\in[n]}z_i}\le k,$$

where the first inequality results from type (ii) inequalities in Lemma 2, the second one is due to Cauchy–Schwartz inequality, and the last one is due to $\operatorname{tr}(\boldsymbol{X})=1$ and $\sum_{i\in[n]}z_i\le k$. □

A.6 Proof of Proposition 5

Proposition 5 *The continuous relaxation* (16) *of the MISDP* (15) *yields a* $\min\{k,n/k\}$ *optimality gap for SPCA, i.e.,*

$$w^*\le\overline{w}_3\le\min\left\{k,\frac{n}{k}\right\}w^*.$$

Proof. The proof is separated into two parts: (i) $\overline{w}_3\le kw^*$ and (ii) $\overline{w}_3\le n/kw^*$.

(i) $\overline{w}_3\le kw^*$. For any feasible solution $\boldsymbol{X}$ to problem (16), we have

$$\operatorname{tr}(\boldsymbol{A}\boldsymbol{X})=\sum_{i\in[n]}\sum_{j\in[n]}A_{ij}X_{ij}\le\sum_{i\in[n]}\sum_{j\in[n]}|A_{ij}||X_{ij}|\le w^*\sum_{i\in[n]}\sum_{j\in[n]}|X_{ij}|\le kw^*,$$

where the first inequality is due to taking the absolute values, the second one is based on the fact that $\max_{i\in[n]}\{A_{i,i}\}\le w^*$ and $|A_{i,j}|\le\sqrt{A_{i,i}A_{j,j}}\le w^*$ for each pair $i,j\in[n]$, and the third one can be obtained from the proof of Proposition 4.

(ii) $\overline{w}_3\le n/kw^*$. The proof is similar to the one of Proposition 1 since $\overline{w}_3\le\lambda_{\max}(\boldsymbol{A})\le n/kw^*$. □

A.7 Proof of Theorem 4

Theorem 4 *Given a threshold $\epsilon > 0$, the following MILP is $O(\epsilon)$-close to SPCA (2), i.e., $\epsilon \leq \widehat{w}(\epsilon) - w^* \leq \epsilon\sqrt{d}$, where $\widehat{w}(\epsilon)$ is defined by*

$$\begin{aligned}
\widehat{w}(\epsilon) := & \max_{w, \boldsymbol{z}\in Z, \boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{x},, \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\sigma}} w \\
\text{s.t.} \quad & \boldsymbol{x} = \boldsymbol{\delta}_{i1} + \boldsymbol{\delta}_{i2}, \|\boldsymbol{\delta}_{i1}\|_\infty \leq z_i, \|\boldsymbol{\delta}_{i2}\|_\infty \leq 1 - z_i, \forall i \in [n], \\
& \boldsymbol{x} = \sum_{j\in[d]} \boldsymbol{\sigma}_j, \|\boldsymbol{\sigma}_j\|_\infty \leq y_j, \sigma_{jj} = y_j, \forall j \in [d], \sum_{j\in[d]} y_j = 1, \\
& \boldsymbol{x} = \boldsymbol{\mu}_{\ell 1} + \boldsymbol{\mu}_{\ell 2}, \|\boldsymbol{\mu}_{\ell 1}\|_\infty \leq \alpha_\ell, \|\boldsymbol{\mu}_{\ell 2}\|_\infty \leq 1 - \alpha_\ell, \forall \ell \in [m], \\
& w = w_U - (w_U - w_L)\left(\sum_{i\in[m]} 2^{-i}\alpha_i\right), \\
& \left\| \sum_{i\in[n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1} - w_U \boldsymbol{x} + (w_U - w_L) \sum_{\ell\in[m]} 2^{-\ell} \boldsymbol{\mu}_{\ell 1} \right\|_\infty \leq \epsilon, \\
& \boldsymbol{\alpha} \in \{0,1\}^m, \boldsymbol{y} \in \{0,1\}^d,
\end{aligned} \tag{22}$$

where w_L and w_U denote the lower and upper bounds of SPCA, respectively and $m := \lceil \log_2((w_U - w_L)\epsilon^{-1}) \rceil$.

Proof. Throughout the proof, we use indices $i \in [n]$, $j \in [d]$, and $\ell \in [m]$ to denote the elements of three different dimensional vectors, respectively. To construct the MILP by SPCA (21) and show the approximation accuracy, we split the proof into four steps.

Step 1. Linearize the bilinear terms $\{z_i \boldsymbol{x}\}_{i\in[n]}$ in (21). This can be done by introducing two copies $\boldsymbol{\delta}_{i1}, \boldsymbol{\delta}_{i2}$ of vector $\boldsymbol{x}$ for each $i \in [n]$ such that

$$\boldsymbol{x} = \boldsymbol{\delta}_{i1} + \boldsymbol{\delta}_{i2}, \|\boldsymbol{\delta}_{i1}\|_\infty \leq z_i, \|\boldsymbol{\delta}_{i2}\|_\infty \leq 1 - z_i, \forall i \in [n], \sum_{i\in[n]} z_i \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{x} = \sum_{i\in[n]} \boldsymbol{c}_i \boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1}.$$

Step 2. Linearize the nonconvex constraint $\|\boldsymbol{x}\|_\infty = 1$. We first observe that due to symmetry, $\|\boldsymbol{x}\|_\infty = 1$ can be equivalently written as a disjunction with d sets as below

$$\cup_{j\in[d]} \left\{ \boldsymbol{x} \in \mathbb{R}^d : x_j = 1, \|\boldsymbol{x}\|_\infty \leq 1 \right\}.$$

Next, for each $j \in d$, we introduce a binary variable $y_j = 1$ indicating the j-th set is active and 0, otherwise, and then create a copy $\boldsymbol{\sigma}_j \in \mathbb{R}^d$ of variable $\boldsymbol{x}$ such that

$$\boldsymbol{x} = \sum_{j\in[d]} \boldsymbol{\sigma}_j, \|\boldsymbol{\sigma}_j\|_\infty \leq y_j, \sigma_{jj} = y_j, \forall j \in [d], \sum_{j\in[d]} y_j = 1, \boldsymbol{y} \in \{0,1\}^d.$$

Step 3. Approximate and linearize bilinear term $w\boldsymbol{x}$. We first approximate variable w using m binary variables $\boldsymbol{\alpha} \in \mathbb{R}^m$ with $m := \lceil \log_2((w_U - w_L)/\epsilon) \rceil$. Thus, we have

$$w \approx w_U - (w_U - w_L)\left(\sum_{\ell\in[m]} 2^{-\ell}\alpha_\ell\right)$$

with approximation accuracy at most $(w_U - w_L)/2^m \leq \epsilon$. The bilinear term $w\boldsymbol{x}$ is now approximated by

$$w\boldsymbol{x} \approx w_U\boldsymbol{x} - (w_U - w_L)\Bigg(\sum_{\ell\in[m]} 2^{-\ell}\alpha_\ell \boldsymbol{x}\Bigg). \tag{32}$$

With binary variables $\boldsymbol{\alpha}$, the resulting bilinear terms $\{\alpha_\ell \boldsymbol{x}\}_{\ell\in[m]}$ can be further linearized following the same arguments as Step 2, i.e.,

$$\begin{aligned}
&\boldsymbol{x} = \boldsymbol{\mu}_{\ell 1} + \boldsymbol{\mu}_{\ell 2}, \|\boldsymbol{\mu}_{\ell 1}\|_\infty \leq \alpha_\ell, \|\boldsymbol{\mu}_{\ell 2}\|_\infty \leq 1 - \alpha_\ell, \forall \ell \in [m],\\
&w_U\boldsymbol{x} - (w_U - w_L)\Bigg(\sum_{\ell\in[m]} 2^{-\ell}\alpha_\ell \boldsymbol{x}\Bigg) = w_U\boldsymbol{x} - (w_U - w_L)\sum_{\ell\in[m]} 2^{-\ell}\boldsymbol{\mu}_{\ell 1}.
\end{aligned}$$

Step 4. Finally, following the approximation and linearization results in Step 3, the equality constraint $\sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{\sigma}_{i1} = w\boldsymbol{x}$ in (21) might not hold exactly. Thus, we replace the equality by the following inequality

$$\begin{aligned}
&\Bigg\|\sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1} - w_U\boldsymbol{x} + (w_U - w_L)\sum_{i\in[m]} 2^{-i}\boldsymbol{\mu}_{i1}\Bigg\|_\infty\\
&= \Bigg\|\sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top z_i\boldsymbol{x} - w_U\boldsymbol{x} + (w_U - w_L)\sum_{i\in[m]} 2^{-i}\alpha_i\boldsymbol{x}\Bigg\|_\infty\\
&= \Bigg\|w\boldsymbol{x} - w_U\boldsymbol{x} + (w_U - w_L)\sum_{i\in[m]} 2^{-i}\alpha_i\boldsymbol{x}\Bigg\|_\infty \leq (w_U - w_L)/2^m \leq \epsilon,
\end{aligned}$$

which holds for any feasible solution of formulation (21).

First, we have $\widehat{w}(\epsilon) \geq w^* - \epsilon$ since $w := w^* - \epsilon$ is feasible to the MILP (22).

Moreover, given an optimal solution $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{z}}, \widehat{w}(\epsilon))$ to the MILP (22), we must have

$$\begin{aligned}
&\Bigg\|\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top \widehat{\boldsymbol{x}} - \widehat{w}(\epsilon)\widehat{\boldsymbol{x}}\Bigg\|_\infty \leq \epsilon\\
(\Rightarrow)\quad & \min_{\boldsymbol{x}:\|\boldsymbol{x}\|_\infty = 1} \Bigg\|\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{x} - \widehat{w}(\epsilon)\boldsymbol{x}\Bigg\|_\infty \leq \epsilon\\
(\Rightarrow)\quad & d^{-1/2}\min_{\boldsymbol{x}:\|\boldsymbol{x}\|_\infty = 1} \Bigg\|\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{x} - \widehat{w}(\epsilon)\boldsymbol{x}\Bigg\|_2 \leq \epsilon\\
(\Rightarrow)\quad & d^{-1/2}\min_{\boldsymbol{x}:\|\boldsymbol{x}\|_2 \geq 1} \Bigg\|\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{x} - \widehat{w}(\epsilon)\boldsymbol{x}\Bigg\|_2 \leq \epsilon\\
(\Leftrightarrow)\quad & d^{-1/2}\min_{\boldsymbol{x}:\|\boldsymbol{x}\|_2 = 1} \Bigg\|\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{x} - \widehat{w}(\epsilon)\boldsymbol{x}\Bigg\|_2 \leq \epsilon
\end{aligned}$$

where the first implication is due to $\|\widehat{\boldsymbol{x}}\|_\infty = 1$, the second one is due to $\|\boldsymbol{x}\|_\infty \geq d^{-1/2}\|\boldsymbol{x}\|_2$ since $\boldsymbol{x} \in \mathbb{R}^d$, the third one is because $\|\boldsymbol{x}\|_\infty = 1$ implies $\|\boldsymbol{x}\|_2 \geq 1$, and the equivalence is because of monotonicity and positive homogeneity of the objective function. According to the last inequality, there exists an eigenvalue w of matrix $\sum_{i\in[n]} \widehat{z}_i\boldsymbol{c}_i\boldsymbol{c}_i^\top$ such that $|\widehat{w}(\epsilon) - w| \leq \epsilon\sqrt{d}$, which further implies that $\widehat{w}(\epsilon) - w^* \leq \epsilon\sqrt{d}$ since $w \leq w^*$. □

A.8 Proof of Proposition 7

Proposition 7 *Given a threshold $\epsilon > 0$, by relaxing the binary variables $\boldsymbol{z}$ to be continuous, let $\overline{w}_5(\epsilon)$ denote the optimal value of the relaxed MILP formulation* (22). *Then we have*

$$\overline{w}_5(\epsilon) \le \min\left\{k(\sqrt{d}/2+1/2),\ n/k\sqrt{d}+(n-k)(\sqrt{d}/2+1/2)\right\}w^* + \epsilon\sqrt{d}.$$

Proof. From the proof of Theorem 4, we know that $\overline{w}_5(\epsilon) \le \overline{w}_5(0) + \epsilon\sqrt{d}$. Thus, it is sufficient to show that

$$\overline{w}_5(0) \le k(\sqrt{d}/2+1/2)w^*.$$

We observe that when $\epsilon = 0$, the resulting formulation by relaxing binary variables $\boldsymbol{z}$ to be continuous becomes:

$$\begin{aligned}\overline{w}_5(0) = \max_{\substack{w,\boldsymbol{z}\in\overline{Z},\boldsymbol{x},\\ \{\boldsymbol{\delta}_{i1}\}_{i\in[n]},\{\boldsymbol{\delta}_{i2}\}_{i\in[n]}}} & \Bigg\{ w : \sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top \boldsymbol{\delta}_{i1} = w\boldsymbol{x}, \|\boldsymbol{x}\|_\infty = 1, \\ & \boldsymbol{x} = \boldsymbol{\delta}_{i1} + \boldsymbol{\delta}_{i2}, \|\boldsymbol{\delta}_{i1}\|_\infty \le z_i, \|\boldsymbol{\delta}_{i2}\|_\infty \le 1 - z_i, \forall i\in[n] \Bigg\}, \end{aligned} \tag{33}$$

Next, we split the proof into three steps.

Step 1. For any feasible solution to problem (33), we have

$$\begin{aligned} w = \frac{\|\sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{\delta}_{i1}\|_\infty}{\|\boldsymbol{x}\|_\infty} &= \|\sum_{i\in[n]} \boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{\delta}_{i1}\|_\infty \le \sum_{i\in[n]} \|\boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{\delta}_{i1}\|_\infty = \sum_{i\in[n]} \|\boldsymbol{c}_i\|_\infty |\boldsymbol{c}_i^\top\boldsymbol{\delta}_{i1}| \\ &\le \sum_{i\in[n]} \|\boldsymbol{c}_i\|_\infty \|\boldsymbol{c}_i\|_1 \|\boldsymbol{\delta}_{i1}\|_\infty \le \sum_{i\in[n]} \|\boldsymbol{c}_i\|_\infty\|\boldsymbol{c}_i\|_1 z_i \le k \max_{i\in[n]} \|\boldsymbol{c}_i\|_\infty \|\boldsymbol{c}_i\|_1, \end{aligned}$$

where the first inequality is due to triangle inequality, the second one is because of Holder's inequality, the third one is because $\|\boldsymbol{\delta}_{i1}\|_\infty \le z_i$, and the last one is due to $\|\boldsymbol{c}_i\|_\infty\|\boldsymbol{c}_i\|_1 \le \max_{j\in[n]} \|\boldsymbol{c}_j\|_\infty\|\boldsymbol{c}_j\|_1$ for each $i\in[n]$ and $\sum_{i\in[n]} z_i \le k$.

Step 2. Now it remains to show that for each $i\in[n]$

$$\|\boldsymbol{c}_i\|_\infty\|\boldsymbol{c}_i\|_1 \le \frac{\sqrt{d}+1}{2}w^*.$$

Let ς be a permutation of index set $[d]$ such that $c_{i,\varsigma(1)},\cdots,c_{i,\varsigma(d)}$ are sorted in an ascending order. Then we have

$$c_{i,\varsigma(1)}^2 + \frac{1}{d-1}\left(\sum_{j\in[2,d]} |c_{i,\varsigma(j)}|\right)^2 \le c_{i,\varsigma(1)}^2 + \cdots + c_{i,\varsigma(d)}^2 = \|\boldsymbol{c}_i\|_2^2 \le w^*,$$

where the first inequality is from the arithmetic and quadratic mean inequality and the second inequality follows from $\|\boldsymbol{c}_i\|_2^2 = \lambda_{\max}(\boldsymbol{c}_i\boldsymbol{c}_i^\top) \le w^*$.

For ease of exposition, let us introduce $v_1 = |c_{i,\varsigma(1)}|$ and $v_2 = \sum_{j\in[2,d]} |c_{i,\varsigma(j)}|$. Next, let us consider an optimization problem

$$\nu = \max_{\boldsymbol{v}\in\mathbb{R}^2_+} \left\{ v_1(v_1+v_2) : v_1^2 + 1/(d-1)v_2^2 \le w^* \right\}, \tag{34}$$

whose optimal value clearly provides an upper bound of $\|\boldsymbol{c}_i\|_\infty \|\boldsymbol{c}_i\|_1$.

To solve (34), we first rewrite v_1, v_2 as

$$v_1 = r\sin(\theta)r, v_2 = r\sqrt{d-1}\cos(\theta), \theta\in[0,\pi/2], r \le \sqrt{w^*}.$$

In this way, the objective function (34) is equal to

$$\begin{aligned} v_1(v_1+v_2) = v_1^2 + v_1v_2 = r^2\sin^2(\theta) + r^2\sqrt{d-1}\sin(\theta)\cos(\theta) = r^2\frac{1-\cos(2\theta)}{2} + r^2\sqrt{d-1}\frac{\sin(2\theta)}{2} \\ = \frac{r^2}{2} - \frac{r^2}{2}\cos(2\theta) + \frac{1}{2}r^2\sqrt{d-1}\sin(2\theta) \le \frac{1}{2}r^2 + \frac{\sqrt{d}}{2}r^2 \le \frac{\sqrt{d}+1}{2}w^*, \end{aligned}$$

where the first inequality is due to Cauchy-Schwartz inequality and the second one is because $r^2 \le w^*$. Thus, we must have $\|\boldsymbol{c}_i\|_\infty\|\boldsymbol{c}_i\|_1 \le \frac{\sqrt{d}+1}{2}w^*$. This proves the first bound $k(\sqrt{d}/2 + 1/2)$ together with Step 1.

Step 3. We now prove the second bound. Plugging the equations $\boldsymbol{\delta}_{i1} = \boldsymbol{x} - \boldsymbol{\delta}_{i2}$ for all $i\in[n]$, we rewrite the continuous relaxation value as

$$\begin{aligned} w = \frac{\|\sum_{i\in[n]}\boldsymbol{c}_i\boldsymbol{c}_i^\top(\boldsymbol{x}-\boldsymbol{\delta}_{i2})\|_\infty}{\|\boldsymbol{x}\|_\infty} \le \frac{\|\sum_{i\in[n]}\boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} + \frac{\|\sum_{i\in[n]}\boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{\delta}_{i2}\|_\infty}{\|\boldsymbol{x}\|_\infty} \\ \le \frac{\|\sum_{i\in[n]}\boldsymbol{c}_i\boldsymbol{c}_i^\top\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} + (n-k)\frac{\sqrt{d}+1}{2}w^* \le \max_{i\in[d]}\sum_{j\in[d]}|\overline{C}_{ij}| + (n-k)\frac{\sqrt{d}+1}{2}w^*, \end{aligned}$$

where $\overline{\boldsymbol{C}} := \boldsymbol{C}\boldsymbol{C}^\top = \sum_{i\in[n]}\boldsymbol{c}_i\boldsymbol{c}_i^\top$ and the first inequality is from the triangle inequality, the second one follows from the derivations in Steps 1 and 2, and the third one is due to $x_i \le 1$ for each $i\in[d]$.

Next, the first term of the right-hand side above can be upper bounded by

$$\max_{i\in[d]}\sum_{j\in[d]}|\overline{C}_{ij}| = \|\overline{\boldsymbol{C}}\|_1 \le \sqrt{d}\|\overline{\boldsymbol{C}}\|_2 = \sqrt{d}\lambda_{\max}(\overline{\boldsymbol{C}}) \le \frac{n}{k}\sqrt{d}w^*,$$

where the equations are from the definition of ℓ_1-norm and ℓ_2-norm of a matrix and the second inequality is due to $\lambda_{\max}(\overline{\boldsymbol{C}}) = \lambda_{\max}(\boldsymbol{A}) \le n/kw^*$. □

A.9 Proof of Corollary 2

Corollary 2 *The approximation ratio of the s-swap local search is s/k for any $s\in[k]$. The ratio is tight when $k\le n/2$.*

Proof. First, let set $\widehat{S}_L$ denote the indices of selected vectors by s-swap local search algorithm. Then following the same proof as that in Theorem 6, for any size-s set $T \subseteq [n]$, we have

$$\lambda_{\max}\bigg(\sum_{i\in \widehat{S}_L} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg) \geq \lambda_{\max}\bigg(\sum_{i\in T} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg). \tag{35}$$

Let S^* denote an optimal solution to SPCA (4). Using the result in (35), the optimal value of SPCA w^* is upper bounded by

$$w^* = \lambda_{\max}\bigg(\sum_{i\in S^*} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg) = \lambda_{\max}\bigg(\frac{1}{\binom{k-1}{s-1}}\sum_{T\subseteq S^*, |T|=s}\sum_{i\in T} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg) \leq \frac{\binom{k}{s}}{\binom{k-1}{s-1}}\lambda_{\max}\bigg(\sum_{i\in \widehat{S}_L} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg) = \frac{k}{s}\bigg(\sum_{i\in \widehat{S}_L} \boldsymbol{c}_i\boldsymbol{c}_i^\top\bigg).$$

Second, to show the tightness, let us consider the following example.

Example 2 *For any integer $k \in [d]$, let $d = k+1$, $n = (s+1)k$, and the vectors $\{\boldsymbol{c}_i\}_{i\in[n]} \subseteq \mathbb{R}^d$ be*

$$\boldsymbol{c}_i = \begin{cases} \boldsymbol{e}_i, & \text{if } i \in [k], \\ \vdots & \\ \boldsymbol{e}_{i-(s-1)k}, & \text{if } i \in [(s-1)k+1, sk], \\ \boldsymbol{e}_{k+1}, & \text{if } i \in [sk+1, n], \end{cases} \quad \forall i \in [n].$$

In Example 2, we show that the subset $\widehat{S}_L = [k-s+1] \cup \{\ell k+1\}_{\ell\in[s-1]}$ satisfies the s-swap local optimality condition.

Indeed, for each pair (T_1, T_2) such that $T_1 \subseteq \widehat{S}_L, T_2 \subseteq ([n] \setminus \widehat{S}_L)$ with $|T_1| = |T_2| = s$, we have

$$\lambda_{\max}\bigg(\sum_{\ell\in \widehat{S}_L\cup T_2\setminus T_1} \boldsymbol{c}_\ell\boldsymbol{c}_\ell^\top\bigg) \leq s.$$

Therefore, the set $\widehat{S}_L$ achieves s-swap local optimum with largest eigenvalue of s. Since the optimal value of SPCA is $w^* = k$, the approximation ratio of set $\widehat{S}_L$ is equal to sk^{-1} for SPCA. □