Sparse PCA With Multiple Components

Ryan Cory-Wright
Department of Analytics, Marketing and Operations, Imperial College Business School, London, UK
IBM Thomas J. Watson Research Center, USA
ORCID: 0000-0002-4485-0619
r.cory-wright@imperial.ac.uk

Jean Pauphilet
Management Science and Operations, London Business School, London, UK
ORCID: 0000-0001-6352-0984
jpauphilet@london.edu

Sparse Principal Component Analysis is a cardinal technique for obtaining combinations of features, or principal components (PCs), that explain the variance of high-dimensional datasets in an interpretable manner. At its heart, this involves solving a sparsity and orthogonality constrained convex maximization problem, which is extremely computationally challenging. Most existing work address sparse PCA via heuristics such as iteratively computing one sparse PC and deflating the covariance matrix, which does not guarantee the orthogonality, let alone the optimality, of the resulting solution. We challenge this status by reformulating the orthogonality conditions as rank constraints and optimizing over the sparsity and rank constraints simultaneously. We design tight semidefinite relaxations and propose tractable second-order cone versions of these relaxations which supply high-quality upper bounds. We also design valid second-order cone inequalities which hold when each PC's individual sparsity is specified, and demonstrate that these inequalities tighten our relaxations significantly. Moreover, we propose exact methods and rounding mechanisms that exploit these relaxations' tightness to obtain solutions with a bound gap on the order of 1%–5% for real-world datasets with $p = 100$s or 1000s of features and $r \in \{2, 3\}$ components. We investigate the performance of our methods in spiked covariance settings and demonstrate that simultaneously considering the orthogonality and sparsity constraints leads to improvements in the Area Under the ROC curve of 2%-8% compared to state-of-the-art deflation methods. All in all, our approach solves sparse PCA problems with multiple components to certifiable (near) optimality in a practically tractable fashion.

Key words: Sparse Principal Component Analysis; Semidefinite Optimization

1. Introduction

The ever-increasing amount of data generated and collected in the past decades has fueled a growing interest in machine learning (ML) algorithms that scale to ultra-high-dimensional settings. On the other hand, regulatory requirements such as the *right to explanation* enshrined in the European Union's General Data Protection Regulation (Goodman and Flaxman 2017) promote algorithms with outputs that are interpretable to humans. However, since most ML algorithms generate uninterpretable learners (Rudin et al. 2022), a key challenge for data scientists is to generate interpretable learners that perform well in high-dimensional settings. A standard technique

for achieving this is dimensionality reduction, wherein we reduce a p-dimensional dataset to an r-dimensional one (with $r \ll p$) with minimal information loss.

Perhaps the most popular dimensionality reduction technique is Principal Component Analysis, or PCA, initially proposed by Pearson (1901) (see also Hotelling 1933, Eckart and Young 1936). Given a normalized and centered data matrix $\boldsymbol{A} \in \mathbb{R}^{n\times p}$, and its sample covariance matrix $\boldsymbol{\Sigma} := \frac{1}{n-1}\boldsymbol{A}\boldsymbol{A}^\top$, practitioners identify the top r principal components of $\boldsymbol{\Sigma}$ by solving:

$$\max_{\boldsymbol{U}\in\mathbb{R}^{p\times r}} \quad \langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma}\rangle \text{ s.t. } \boldsymbol{U}^\top\boldsymbol{U} = \mathbb{I}, \tag{1}$$

and subsequently project the data matrix $\boldsymbol{A}$ onto the principal components $\boldsymbol{U}$. PCA is now a cardinal unsupervised learning paradigm that is practically useful across a range of fields, including pattern recognition (Naikal et al. 2011) and sequence classification (Tan et al. 2014).

Unfortunately, while modern implementations of PCA are extremely time-efficient (Udell et al. 2016, Tropp et al. 2017), PCA generates components that are dense, error-prone, and statistically meaningless in high-dimensional settings due to the curse of dimensionality (Amini and Wainwright 2008, Berthet and Rigollet 2013). Accordingly, several authors (such as Jolliffe et al. 2003, d'Aspremont et al. 2007) have proposed sparse PCA, namely augmenting Problem (1) with a sparsity constraint. When $r = 1$, sparse PCA can be formulated as the following optimization problem:

$$\max_{\boldsymbol{u}\in\mathbb{R}^{p}} \quad \langle \boldsymbol{\Sigma}, \boldsymbol{u}\boldsymbol{u}^\top\rangle \text{ s.t. } \|\boldsymbol{u}\|_2^2 = 1, \|\boldsymbol{u}\|_0 \leq k, \tag{2}$$

where $\|\boldsymbol{u}\|_0$ denotes the cardinality of $\boldsymbol{u}$ or the size of its support: $\|\boldsymbol{u}\|_0 = |\operatorname{supp}(\boldsymbol{u})| = |\{j : \boldsymbol{u}_j \neq 0\}|$. Problem (2) can be formulated as a mixed-integer semidefinite optimization problem and solved via global optimization techniques like branch-and-cut or convex optimization techniques like relax-and-round (Bertsimas et al. 2022, Li and Xie 2020). High-quality solutions can also be obtained by greedy heuristics (d'Aspremont et al. 2008) or solving its ℓ_1 relaxation (Zou et al. 2006, d'Aspremont et al. 2007, Dey et al. 2022a). Unfortunately, no consensual formulation extends Problem (2)'s formulation to $r > 1$, and there are no practically relevant algorithms with optimality guarantees that successfully address this extension. This is a significant issue for two reasons.

First, from a statistical recovery perspective, sparse PCA performs provably better than PCA in some rigorously-identified settings (see Gamarnik et al. 2021, for further details). Empirically, it is believed to generally outperform PCA as well (d'Aspremont et al. 2007, Gamarnik 2021). Furthermore, among sparse PCA methods, there exists an information-theoretic performance gap between certifiably optimal methods and heuristic approaches (Berthet and Rigollet 2013, Gamarnik 2021).

Second, certifiably optimal methods for sparse PCA are also valuable from a practical perspective. Indeed, for $r = 1$, sparse PCA methods which do not possess optimality guarantees are typically and often significantly less accurate than exact methods. For example, Berk and Bertsimas (2019) compared the performance of seven popular sparse PCA heuristics across four UCI

datasets ($k \in \{5, 10\}$) and observed that no heuristic found an optimal solution across all datasets, with some methods being routinely 20% suboptimal or more. On the other hand, recent exact methods for $r = 1$, including branch-and-bound (Berk and Bertsimas 2019) and branch-and-cut (Bertsimas et al. 2022) found the optimal solution in all cases on the same benchmark.

In this paper, we propose a generic optimization formulation that extends Problem (2) to $r > 1$, reformulate it as a sparsity and rank constrained optimization problem, and design certifiably (near) optimal techniques to solve it. As in the $r = 1$ case, we demonstrate numerically that exact methods obtain substantially higher quality solutions than existing state-of-the-art heuristics.

1.1. A Generic Formulation for Sparse PCA with Multiple PCs

Perhaps the most natural extension of sparse PCA to multiple principal components, and the one which we advocate in this paper, is to augment Problem (1) with a constraint on the number of non-zero entries in the matrix $\boldsymbol{U}$,

$$\|\boldsymbol{U}\|_0 = |\text{supp}(\boldsymbol{U})| = |\{(i,j) \in [p] \times [r] : U_{i,j} \neq 0\}| \leq k.$$

This gives a formulation which enforces two desirable properties on the matrix $\boldsymbol{U}$: orthogonality ($\boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}$), as present in the prototypical formulation of PCA; and sparsity, to address interpretability and accuracy concerns.

Formally, introducing a binary matrix $\boldsymbol{Z}$ to encode the support of $\boldsymbol{U}$, we consider the problem:

$$\begin{aligned} \max_{\substack{\boldsymbol{Z} \in \{0,1\}^{p \times r} \\ \langle \boldsymbol{E}, \boldsymbol{Z} \rangle \leq k}} \max_{\boldsymbol{U} \in \mathbb{R}^{p \times r}} \quad & \langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma} \rangle \\ \text{s.t.} \quad & \boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}, U_{i,t} = 0 \text{ if } Z_{i,t} = 0, \ \forall i \in [p], \ \forall t \in [r]. \end{aligned} \tag{3}$$

where $pr > k > r$ in order that the problem is well-posed and non-trivial.

Alternatively, instead of imposing an overall sparsity budget (k) on the entire matrix $\boldsymbol{U}$, we could restrict the size of the support of each column of $\boldsymbol{U}$ separately, i.e., impose $\|\boldsymbol{U}_t\|_0 \leq k_t \ \forall t \in [r]$ and adapt (3) accordingly. Indeed, optimizing this formulation over all integer combinations k_t which sum to k is equivalent to solving (3). We address both modeling options in this paper.

From a generative model perspective, Problem (3) is consistent with a spiked covariance model (see, e.g., Amini and Wainwright 2008, d'Aspremont et al. 2008), where the true covariance matrix $\boldsymbol{\Sigma}^\star$ can be decomposed as the sum of a sparse and low-rank term plus some noise:

$$\boldsymbol{\Sigma}^\star = \sum_{t \in [r]} \beta_t \boldsymbol{u}_t^\star \boldsymbol{u}_t^{\star\top} + \boldsymbol{N}. \tag{4}$$

While sparse PCA with one PC has received much attention over the past 20 years, sparse PCA with multiple PCs has received substantially less attention, arguably because of the significant technical difficulty of simultaneously optimizing over orthogonality and sparsity constraints. Indeed,

in the single PC case, various authors including Gally and Pfetsch (2016), Bertsimas et al. (2022), Li and Xie (2020), Kim et al. (2021), Dey et al. (2022a) have shown that Problem (2) can be recast as a mixed-integer semidefinite optimization (MISDO) problem and have derived both high-quality solutions and valid dual bounds. For the $r > 1$ case however, we are not aware of any equivalent results–except when very strong assumptions are made about the sparsity pattern, as discussed in Section 1.2. Moreover, as established in our numerical results (Section 4), existing heuristics routinely terminate with highly suboptimal solutions. For example, on the `pitprops` dataset, with 6 PCs, we explain 81% of the variance with an overall sparsity of 24 while previous studies could only explain less than 70% of the variance with twice times as many variables (see Section 4.2).

1.2. Literature Review

To identify the extent to which the state-of-the-art for sparse PCA could be improved, we now review methods that have been proposed to approximately solve (3) or its special cases.

Methods for generic sparse PCA: It is well known that an optimal solution to Problem (1) can be obtained via a greedy procedure where, at each iteration, the leading PC of $\mathbf{\Sigma}$, $\boldsymbol{u}$, is computed (e.g., by solving (1) with $r = 1$) and then the matrix $\mathbf{\Sigma}$ is updated (or *deflated*) to annihilate $\boldsymbol{u}$. This so-called deflation method provides an optimal solution to the PCA problem with multiple PCs (see, e.g., Mackey 2008). Consequently, Mackey (2008), building upon the work of d'Aspremont et al. (2007), propose a sparse extension of this deflation scheme where, at each iteration, a sparse PC is computed (e.g., by solving (2) or a relaxation) and $\mathbf{\Sigma}$ is updated by projecting out the eigenspace modeled by $\boldsymbol{u}$: $\mathbf{\Sigma}_{\text{new}} = (\mathbb{I} - \boldsymbol{u}\boldsymbol{u}^\top)\mathbf{\Sigma}(\mathbb{I} - \boldsymbol{u}\boldsymbol{u}^\top)$. Empirically, this method often performs reasonably well (see Berk and Bertsimas 2019, Section 5.3), particularly when Problem (2) is solved to global optimality; see also Hein and Bühler (2010), Bühler (2014) for a related deflation-based scheme. However, unlike in the traditional case, deflation need not return an optimal solution to (3). Actually, deflation need not even return a feasible solution, since the orthogonality constraint is not explicitly imposed by the method, and is therefore often violated in practice. Indeed, because of their iterative nature, deflation-based procedures can easily control the sparsity of each component but usually struggle to enforce the orthogonality constraint.

A second approach for solving Problem (3) is to apply a heuristic which (approximately) optimizes all r PCs simultaneously, rather than sequentially. Among others, Zou et al. (2006) propose an alternating minimization scheme for an ℓ_1 relaxation of Problem (3), Journée et al. (2010) propose an iterative conditional gradient method to identify a local optimum of Problem (3) without the orthogonality constraints, Lu and Zhang (2012) apply an augmented Lagrangian method which solves an ℓ_1 relaxation of (3), Vu et al. (2013) solve a semidefinite relaxation of Problem (3)'s ℓ_1 relaxation, and Benidis et al. (2016) adopt a minorization-maximization approach which also

solves Problem (3) approximately. Unfortunately, these approaches are often suboptimal if $r=1$, and only provide candidate solutions, with no indication on the optimality gap. Indeed, none of these approaches explicitly control both the sparsity and orthogonality constraints, and therefore none of the methods reviewed here are actually guaranteed to return feasible solutions to (3).

Row-Sparsity: Motivated by tractability concerns, another line of work studies a special case of (3), namely row-sparse principal component analysis or principal component analysis with global support. In this formulation, we replace the sparsity constraint on $\boldsymbol{U}$ with one requiring $\boldsymbol{U}$ has at most k non-zero rows, as advocated by Boutsidis et al. (2011), Probel and Tropp (2011), Vu and Lei (2013). This implies we can rewrite sparse PCA as performing a top-r SVD on a subset of k rows of $\boldsymbol{\Sigma}$, i.e., solving

$$\max_{\boldsymbol{z}\in\{0,1\}^p:\boldsymbol{e}^\top\boldsymbol{z}\leq k} \max_{\boldsymbol{U}\in\mathbb{R}^{p\times r}} \langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma}\rangle \text{ s.t. } \boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I}_{r\times r},\ U_{i,t}=0 \text{ if } z_i=0\ \forall i\in[p], \forall t\in[r]. \tag{5}$$

Problem (5) is a special case of Problem (3) where each PC has the same support, which corresponds to constraining $\boldsymbol{Z}$ such that $z_{i,t}=z_{j,t}\ \forall t\in[r]$ in (3). This restriction is advantageous from a computational perspective, but disadvantageous from a statistical one. Indeed, because of the global support assumption, Problem (5) can be reformulated as a mixed-integer semidefinite problem (see Bertsimas et al. 2022, Li and Xie 2021, Bertsimas and Kitane 2022, for derivations) and solved via relax-and-round (Dey et al. 2022b, Li and Xie 2021) or branch-and-bound (Del Pia 2022) strategies. However, from a generative model perspective (4), it is equivalent to making the very strong assumption that all leading eigenvectors $\boldsymbol{u}_t^\star$ have the same sparsity pattern. Indeed, if one PC $\boldsymbol{u}_t^\star$ is sparser than k ($\|\boldsymbol{u}_t\|_0\leq k_t<k$) or two PCs are partially disjoint then Problem (5) will nonetheless estimate that the support of each $\boldsymbol{u}_t^\star$ is of size k. Therefore, Problem (5) is vulnerable to consistently making false discoveries in the identification of relevant features.

Disjoint Sparsity: Another relevant special case of sparse PCA with multiple PCs is when the supports of all r columns of $\boldsymbol{U}$ are assumed mutually disjoint, as originally proposed by Asteris et al. (2015). This gives rise to the formulation:

$$\max_{\substack{\boldsymbol{Z}\in\{0,1\}^{p\times r}:\\ \langle \boldsymbol{E},\ \boldsymbol{Z}\rangle\leq k, \boldsymbol{Z}\boldsymbol{e}\leq e}} \max_{\boldsymbol{U}\in\mathbb{R}^{p\times r}} \langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma}\rangle \text{ s.t. } \boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I}_{r\times r},\ U_{i,t}=0 \text{ if } Z_{i,t}=0\ \forall i\in[p]. \tag{6}$$

Note that (6) is a special case of (3) where we additionally require that $\sum_{t\in[r]} Z_{i,t}\leq 1\ \forall i\in[p]$, i.e., that each PC is supported by non-overlapping features. Interestingly, this restriction allows (6) to be recast as a MISDO and solved as such (c.f. Bertsimas et al. 2022, Bertsimas and Kitane 2022).

An obvious criticism of (6)'s formulation is that it only allows each feature to be assigned to one PC, while in practice we may wish to assign a feature to multiple PCs. Therefore, (6) is best thought of as a special case of sparse PCA –and of the generative model (4). Indeed, if the true generative model involves vectors $\boldsymbol{u}_i^\star$ with partially overlapping supports, then (6) could not recover

them. Nonetheless, as we explore in our numerical experiments (Section 4), disjoint solutions often perform well when k is very small relative to p. This is because, in high-dimensional settings, there are often several disjoint submatrices which are near-optimal in the rank-one case, and selecting the leading PCs from each of them is often a reasonable approach in practice.

1.3. Contributions and Structure

To our knowledge, no existing algorithm can solve sparse PCA problems with multiple components and obtain certificates of (near) optimality, except in the very special cases of row-sparsity or disjoint support. Accordingly, in this paper, we undertake a detailed study of Problem (3).

The main contributions of the paper are twofold. First, we reformulate Problem (3) as a mixed-integer low-rank problem. Based on this reformulation, we derive tight, yet tractable, semidefinite and second-order cone relaxations of this formulation, and propose valid inequalities which strengthen the relaxation. Second, from these convex relaxations, we propose rounding techniques and an alternating minimization scheme to obtain high-quality solutions. By combining strong relaxations and good solutions, we obtain bound gaps of 5% on real-world instances of sparse PCA with multiple PCs where $p = 100$s or 1000s. All in all, we provide the first reformulation of the sparse PCA problem (3) that is amenable to certifiably optimal algorithms and admits tractable relaxations, while demonstrating its benefit compared with state-of-the-art heuristics.

The structure and detailed contributions of the paper are as follows:

- In Section 2.1, we prove that rank constraints successfully model orthogonality constraints. Leveraging this result, we reformulate Problem (3) as a sparsity and rank constrained optimization problem, and derive its semidefinite relaxation. We propose valid inequalities which strengthen the relaxation in Section 2.2, and additional valid inequalities which hold if we restrict the support of each PC separately in Section 2.3. For high-dimensional instances, we propose a tractable second-order cone relaxation in Section 2.4.
- In Section 3, we develop numerical strategies to obtain high-quality feasible solutions to Problem (3). In Section 3.1, we develop a greedy rounding strategy which rounds an optimal solution to one of Section 2's relaxations into r sparse PCs with disjoint support (hence, orthogonal). In Section 3.2, we develop an exact formulation of Problem (3) in the original space of decision variables, which can be addressed via global branch-and-bound solvers, and propose valid inequalities which improve the tightness of the formulation. Finally, in Section 3.3, we consider a Lagrangean relaxation of Problem (3) and design an alternating minimization strategy to obtain near-orthogonal solutions by iteratively solving a sequence of sparse PCA problems with $r = 1$. Compared with existing deflation-based techniques, we explicitly penalize the orthogonality violation at each iteration, with an increasing penalty parameter, thus converging towards an orthogonal and typically high-quality solution.

- In Section 4, we investigate the quality of our semidefinite relaxations and feasible methods and observe empirically that they collectively give small bound gaps in practice. First, we invoke our convex relaxations to obtain high-quality upper bounds on UCI datasets in Section 4.1. We also compare our approximate methods with existing methods from the literature, including the methods of Zou et al. (2006), Hein and Bühler (2010), Berk and Bertsimas (2019). On both synthetic and UCI datasets, we demonstrate in Sections 4.2-4.3 that our approaches, particularly the alternating minimization scheme developed in Section 3.3, explain more of the variance in the data and exhibit a lower false discovery/higher true discovery rate for a given sparsity budget. Finally, in Section 4.4, we explore the relationship between the symmetry in the size of each PCs sparsity budget and the amount of variance explained.

1.4. Preliminaries and Notation

We let nonbold face characters such as u denote scalars, lowercase bold faced characters such as $\boldsymbol{u}$ denote vectors, uppercase bold faced characters such as $\boldsymbol{U}$ denote matrices, and calligraphic uppercase characters such as $\mathcal{Z}$ denote sets. If $\boldsymbol{U}$ is a matrix then $\boldsymbol{U}_t$ denotes the tth column vector of $\boldsymbol{U}$, and $U_{i,t}$ denotes the (i,t)th entry of $\boldsymbol{U}$. We let $[p]$ denote the set of running indices $\{1,...,p\}$. We let $\mathbf{e}$ denote a vector of all 1's, $\mathbf{0}$ denote a vector of all 0's, and $\mathbb{I}$ denote the identity matrix, with dimension implied by the context.

We also use an assortment of matrix operators. We let $\langle \cdot, \cdot \rangle$ denote the Euclidean inner product between two matrices, $\| \cdot \|_F$ denote the Frobenius norm, and $\mathcal{S}^p_+$ denote the $p \times p$ positive semidefinite cone; see Horn and Johnson (1990) for a general theory of matrix operators.

We repeatedly use some basic properties of orthogonal projection matrices, often without proof. Let $\mathcal{Y}_n := \{\boldsymbol{Y} \in \mathcal{S}^n : \boldsymbol{Y}^2 = \boldsymbol{Y}\}$ denote the set of $n \times n$ orthogonal projection matrices and $\mathcal{Y}_n^k := \{\boldsymbol{Y} \in \mathcal{Y}_n : \text{tr}(\boldsymbol{Y}) \leq k\}$ denote projection matrices with rank at most k. Among others, the convex hulls of $\mathcal{Y}_n$ and $\mathcal{Y}_n^k$ are well-studied, as we now remind the reader:

LEMMA 1. *(Theorem 3 of Overton and Womersley 1992)* $\text{Conv}(\mathcal{Y}_n) = \{\boldsymbol{P} : 0 \preceq \boldsymbol{P} \preceq \mathbb{I}\}$ *and* $\text{Conv}(\mathcal{Y}_n^k) = \{\boldsymbol{P} : 0 \preceq \boldsymbol{P} \preceq \mathbb{I}, \text{tr}(\boldsymbol{P}) \leq k\}$. *Moreover, the extreme points of* $\text{Conv}(\mathcal{Y}_n)$ *are* $\mathcal{Y}_n$*, and the extreme points of* $\text{Conv}(\mathcal{Y}_n^k)$ *are* $\mathcal{Y}_n^k$.

Finally, we repeatedly reference two results on the convex hulls of convex quadratic functions under logical constraints:

LEMMA 2. *(Lemma 4 of Günlük and Linderoth 2010) The convex closure of the set*

$$\mathcal{S} = \left\{ (\boldsymbol{x}, \boldsymbol{z}, t) \in \mathbb{R}^n \times \{0,1\}^n \times \mathbb{R} : \quad t \geq \sum_{i=1}^n x_i^2, \ \boldsymbol{e}^\top \boldsymbol{z} \leq k, \ x_i = 0 \text{ if } z_i = 0, \ \forall i \in [n] \right\}$$

is given by

$$\mathcal{S}^c = \left\{ (\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}, t) \in \mathbb{R}^n \times [0,1]^n \times \mathbb{R}^n \times \mathbb{R} : \quad t \geq \sum_{i=1}^n \theta_i,\ \boldsymbol{e}^\top \boldsymbol{z} \leq k,\ \theta_i z_i \geq x_i^2\ \forall i \in [n] \right\}$$

The above result is sometimes known as a perspective reformulation, since we strengthen the quadratic constraint $t \geq \sum_i x_i^2$ by replacing x_i^2 with its perspective $z_i(x_i/z_i)^2$.

LEMMA 3. *(Theorem 1 of Atamtürk and Gomez 2019) The convex closure of the set*

$$\mathcal{T} = \left\{ (\boldsymbol{x}, \boldsymbol{z}, t) \in \mathbb{R}^n \times \{0,1\}^n \times \mathbb{R} : \quad t \geq (\boldsymbol{e}^\top \boldsymbol{x})^2,\ \boldsymbol{e}^\top \boldsymbol{z} \leq k,\ x_i = 0 \text{ if } z_i = 0,\ \forall i \in [n] \right\},$$

is given by

$$\mathcal{T}^c = \left\{ (\boldsymbol{x}, \boldsymbol{z}, t) \in \mathbb{R}^n \times [0,1]^n \times \mathbb{R} : \quad t \cdot \min(1, \boldsymbol{e}^\top \boldsymbol{z}) \geq (\boldsymbol{e}^\top \boldsymbol{x})^2,\ \boldsymbol{e}^\top \boldsymbol{z} \leq k \right\}.$$

Lemma 3 is extremely useful when a single continuous variable depends upon multiple indicator variables, as occurs in certain substructures of our reformulations of Problem (3).

2. Exact Formulations and Their Relaxations

In this section, we reformulate Problem (3) as a mixed-integer low-rank problem, study its semidefinite and second-order cone relaxations, and propose valid inequalities for strengthening these relaxations. In conjunction with the methods for generating high-quality feasible solutions we develop in the next section, this allows us to, for the first time, solve instances of Problem (3) in a manner that maintains optimality guarantees at scale.

2.1. An Extended Formulation and Its Semidefinite Relaxation

Our sparse PCA formulation (3) is a mixed-integer quadratic optimization problem that exhibits three primary sources of difficulty. First, as typical in PCA problems, Problem (3) maximizes a convex function in the decision variable $\boldsymbol{U}$. Second, there is a sparsity constraint, which is computationally challenging to model, although recent evidence suggests that sparsity constraints need not imply intractability (see, e.g., Bertsimas et al. 2020). Finally, and more consequentially, there is an orthogonality constraint. To the best of our knowledge, existing generic non-convex solvers such as `Gurobi` cannot optimize over such orthogonality constraints at scale.

To address the three aforementioned difficulties, we now derive an orthogonality-free reformulation in five steps. First, we introduce the matrix $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top \preceq \mathbb{I}$, thus linearizing the non-convex objective. Second, we introduce rank-one matrices $\boldsymbol{Y}^t$ to model the outer products of each column of $\boldsymbol{U}$, $\boldsymbol{U}_t$, with itself, $\boldsymbol{U}_t\boldsymbol{U}_t^\top$. Third, we reassign the indicator variable $Z_{i,t}$ to model whether $Y^t_{i,j}$, rather than $U_{i,t}$, is non-zero. Fourth, by letting $\boldsymbol{Y} = \sum_{t=1}^r \boldsymbol{Y}^t$, we observe that we can omit the matrix $\boldsymbol{U}$ (and the constraints involving $\boldsymbol{U}$) without altering the set of feasible $\boldsymbol{Y}$'s. Finally, we use the fact that $\boldsymbol{Y}^t_{i,j}$ is only supported on indices i,t where $Z_{i,t} > 0$ to strengthen the constraint $\boldsymbol{Y} \preceq \mathbb{I}$ to $\boldsymbol{Y} \preceq \operatorname{Diag}(\min(\boldsymbol{e}, \sum_t \boldsymbol{Z}_t))$. Formally, we have:

THEOREM 1. *Problem* (3) *attains the same optimal objective value as the problem:*

$$\max_{\substack{\boldsymbol{Z}\in\{0,1\}^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle\leq k}} \max_{\boldsymbol{Y}\in\mathcal{S}^p,\boldsymbol{Y}^t\in\mathcal{S}^p_+} \langle \boldsymbol{Y},\boldsymbol{\Sigma}\rangle \text{ s.t. } \begin{aligned} &\boldsymbol{Y}\preceq \text{Diag}\left(\min\left(\boldsymbol{e},\sum_t \boldsymbol{Z}_t\right)\right), \boldsymbol{Y}=\sum_{t=1}^r \boldsymbol{Y}^t, \\ &\text{tr}(\boldsymbol{Y}^t)=1,\ \forall t\in[r],\ Y^t_{i,j}=0 \text{ if } Z_{i,t}=0\ \forall t\in[r], i,j\in[p], \\ &\text{Rank}(\boldsymbol{Y}^t)=1,\ \forall t\in[r]. \end{aligned} \tag{7}$$

REMARK 1. We do not explicitly require that $\boldsymbol{Y}\succeq\boldsymbol{0}$, since $\boldsymbol{Y}$ is the sum of positive semidefinite matrices $\boldsymbol{Y}^t$. Indeed, omitting $\boldsymbol{Y}\succeq\boldsymbol{0}$ substantially improves the tractability of (7)'s relaxations.

The proof of Theorem 1 requires an intermediate result (Proposition 1). Proposition 1 shows that by imposing rank-one constraints on each $\boldsymbol{Y}^t$, the condition that the $\boldsymbol{Y}^t$'s are mutually orthogonal can be reforumlated as a linear semidefinite constraint. Independently, Proposition 1 is crucial for designing our main algorithm in Section 3.3 (proof deferred to Section EC.1).

PROPOSITION 1. *Consider r matrices, $\boldsymbol{Y}^t\in\mathcal{S}^p_+$, such that* $\text{tr}(\boldsymbol{Y}^t)=1$ *and* $\text{Rank}(\boldsymbol{Y}^t)=1$. *Then,* $\sum_{t\in[r]}\boldsymbol{Y}^t\preceq\mathbb{I}$ *if and only if* $\langle \boldsymbol{Y}^t,\boldsymbol{Y}^{t'}\rangle=0\ \forall t,t'\in[r]:\ t\neq t$.

Proof of Theorem 1 It suffices to show that for any feasible solution to (3), we can construct a feasible solution to Problem (7) with an equal or greater payoff, and vice versa.

- Let $(\boldsymbol{U},\boldsymbol{Z})$ be a solution to Problem (3). Then, since $U_{i,t}$ can only be non-zero if $Z_{i,t}=1$ and $\boldsymbol{U}^\top\boldsymbol{U}\preceq\mathbb{I}$, it follows that $\boldsymbol{U}\boldsymbol{U}^\top\preceq\text{Diag}\left(\min\left(\boldsymbol{e},\sum_t\boldsymbol{Z}_t\right)\right)$. Therefore, $(\boldsymbol{Y}:=\boldsymbol{U}\boldsymbol{U}^\top,\boldsymbol{Y}^t:=\boldsymbol{U}_t\boldsymbol{U}_t^\top,\boldsymbol{Z})$ is a feasible solution to (7) with an equal cost.
- Let $(\boldsymbol{Y},\boldsymbol{Y}^t,\boldsymbol{Z})$ denote a feasible solution to Problem (7). Then, since each $\boldsymbol{Y}^t$ is symmetric and rank-one, we can decompose $\boldsymbol{Y}^t$ as $\boldsymbol{Y}^t=\boldsymbol{U}_t\boldsymbol{U}_t^\top$ for a vector $\boldsymbol{U}_t$ such that $U_{i,t}=0$ if $Z_{i,t}=0$, and concatenate these vectors $\boldsymbol{U}_t$ into a matrix $\boldsymbol{U}$ such that $(\boldsymbol{U},\boldsymbol{Z})$ has the same cost in (3) as $(\boldsymbol{Y},\boldsymbol{Y}^t,\boldsymbol{Z})$ does in (7). Therefore, it remains to show that $\boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I}$. To see this, observe that $\boldsymbol{Y}\preceq\mathbb{I}$ implies $(\boldsymbol{U}_t^\top\boldsymbol{U}_{t'})^2=\langle\boldsymbol{Y}^t,\boldsymbol{Y}^{t'}\rangle=0$ if $t\neq t'$ by Proposition 1. □

Theorem 1 provides a formulation which is less compact than (3), but contains rank constraints rather than orthogonality constraints. Therefore, it is amenable to exact approaches for addressing sparsity (Bertsimas et al. 2021b) and rank (Bertsimas et al. 2021a) constraints.

Furthermore, this formulation already provides valid upper bounds on (3)'s objective by relaxing the rank and sparsity constraints:

$$\max_{\substack{\boldsymbol{Z}\in[0,1]^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle\leq k}} \max_{\boldsymbol{Y}\in\mathcal{S}^p_+,\boldsymbol{Y}^t\in\mathcal{S}^p_+\ \forall t\in[k]} \langle \boldsymbol{Y},\boldsymbol{\Sigma}\rangle \text{ s.t. } \begin{aligned} &\boldsymbol{Y}=\sum_{t=1}^k \boldsymbol{Y}^t, \boldsymbol{Y}\preceq \text{Diag}\left(\min\left(\boldsymbol{e},\sum_t \boldsymbol{Z}_t\right)\right), \\ &\text{tr}(\boldsymbol{Y}^t)=1,\ \forall t\in[r],\ |Y^t_{i,j}|\leq M_{i,j}Z_{i,t},\ \forall i,j\in[p], t\in[r]. \end{aligned} \tag{8}$$

where $M_{i,i}=1$ and $M_{i,j}=1/2$ if $i\neq j$ is an upper bound on $|Y^t_{i,j}|$, since $\boldsymbol{Y}^t$ was, before relaxing the rank constraint, a rank one matrix (c.f. Bertsimas et al. 2022).

2.2. Valid Inequalities for Strengthening the Extended Formulation

In this section, we propose valid inequalities which allow us to improve the quality of the convex relaxation (8) introduced in the previous section. Formally, we have the following result:

THEOREM 2. *Let $\mathcal{P}_{\text{strengthened}}$ denote the optimal value of the following problem:*

$$\max_{\substack{\boldsymbol{Z}\in[0,1]^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle\leq k}} \max_{\substack{\boldsymbol{Y}\in\mathcal{S}^p,\boldsymbol{Y}^t\in\mathcal{S}^p_+,\\ \boldsymbol{w}\in[0,1]^p}} \langle \boldsymbol{Y},\boldsymbol{\Sigma}\rangle \tag{9}$$

$$\begin{aligned}
s.t.\quad & \boldsymbol{Y}\preceq \text{Diag}(\boldsymbol{w}), \boldsymbol{Y}=\sum_{t=1}^k \boldsymbol{Y}^t, \text{tr}(\boldsymbol{Y}^t)=1, \boldsymbol{w}\leq \boldsymbol{Z}\boldsymbol{e}\\
& |Y^t_{i,j}|\leq M_{i,j}Z_{i,t} && \forall i,j\in[p], t\in[k],\\
& \sum_{j=1}^p {Y_{i,j}}^2 \leq rY_{i,i}w_i && \forall i\in[p],\\
& \sum_{j=1}^p {Y^t_{i,j}}^2 \leq Y^t_{i,i}Z_{i,t} && \forall i\in[p], t\in[r],\\
& \left(\sum_{j=1}^p |Y_{i,j}|\right)^2 \leq kY_{i,i}w_i && \forall i\in[p],\\
& \sum_{i\in[p]:i\neq j} Y^2_{i,j}\leq (k-r+1)w_j(w_j-Y_{j,j}), && \forall j\in[p],
\end{aligned}$$

where w_i models $\min(1,\sum_{t=1}^r Z_{i,t})$.

Further, let $\mathcal{P}_{\text{Relax}}$ denote the optimal value of Problem (8), and $\mathcal{P}_{\text{Exact}}$ denote the optimal objective value of Problem (7). Then, for any covariance matrix $\boldsymbol{\Sigma}$ and any sparsity and rank (k,r):

$$\mathcal{P}_{\text{Exact}}\leq \mathcal{P}_{\text{strengthened}}\leq \mathcal{P}_{\text{Relax}}.$$

Proof of Theorem 2 The first inequality holds by verifying that any feasible solution to (7) is also a solution to (9) with equal cost.

Indeed, $\sum_{j=1}^p {Y_{i,j}}^2 \leq rY_{i,i}w_i$ follows from aggregating the 2×2 minor constraints on $\boldsymbol{Y}$ and invoking Lemma 3; $\sum_{j=1}^p {Y^t_{i,j}}^2 \leq Y^t_{i,i}Z_{i,t}$ follows from aggregating the 2×2 minor constraints on $\boldsymbol{Y}^t$, $(Y^t_{i,j})^2\leq Y^t_{i,i}Y^t_{j,j}$ and invoking Lemma 2; $\sum_{j=1}^p {Y^t_{i,j}}^2 \leq Y^t_{i,i}Z_{i,t}$ follows by invoking Cauchy-Schwarz to derive $\left(\sum_{j=1}^p |Y_{i,j}|\right)^2\leq kY_{i,i}$ and invoking Lemma 3; $\sum_{i\in[p]:i\neq j} Y^2_{i,j}\leq (k-r+1)w_j(w_j-Y_{j,j})$ follows by aggregating the 2×2 minors of $\text{Diag}(\boldsymbol{w})-\boldsymbol{Y}$ and invoking Lemma 3. We provide a full derivation of these valid inequalities in Section EC.2.

The second inequality holds by observing that (8) is a relaxation of (9). □

2.3. Strong Inequalities with a Per-Component Sparsity Budget

In this section, we consider specifying a sparsity budget k_t for each component $\boldsymbol{Y}^t$, in addition to an overall sparsity budget k (with $k=\sum_{t=1}^r k_t$), as originally proposed in Section 1.1. The primary

motivation for this approach is that it allows us to derive further valid inequalities which strengthen Problem (9), sometimes substantially. Indeed, existing works which generate sparse and sometimes orthogonal PCs (c.f. Hein and Bühler 2010, Berk and Bertsimas 2019) and we benchmark against in our numerical experiments in Section 4 require that each k_t is specified separately.

Formally, we have the following result (proof deferred to Section EC.3):

PROPOSITION 2. *Suppose that* $\sum_{i\in[p]} \boldsymbol{Z}_{i,t} \leq k_t$ *in Problem* (7)*. Then, the following inequalities hold:*

$$\left(\sum_{j=1}^{p} |Y^t_{i,j}|\right)^2 \leq k_t Y^t_{i,i} Z_{i,t} \quad \forall i \in [p], \forall t \in [r], \tag{10}$$

$$\sum_{i\in[p]:i\neq j} {Y^t_{i,j}}^2 \leq (k_t - 1) Z_{j,t}(Z_{j,t} - Y^t_{j,j}) \quad \forall j \in [p]. \tag{11}$$

Interestingly, as we observe in Section 4.1, combining Problem (9) with Constraints (10)-(11) often yields much tighter upper bounds than (9) alone, even if we take the worst-case upper bound over all feasible splits k_t's which sum to k.

To further tighten the semidefinite relaxation, we now leverage additional problem structure. For each component $t \in [r]$, the feasible set of all k_t-sparse components $\{\boldsymbol{u} \in \mathbb{R}^p : \|\boldsymbol{u}\|_0 \leq k_t\}$ is permutation and sign invariant, i.e., for any feasible vector $\boldsymbol{u}$, any vector obtained by permuting or changing the sign of the coordinates of $\boldsymbol{u}$ is also feasible. Based on this observation, Kim et al. (2021) recently propose a lifted formulation for sparse PCA with a single PC which contains additional valid inequalities. To the best of our knowledge, their formulation leads to the strongest existing relaxation for sparse PCA with $r = 1$, although it is less tractable than the relaxations developed in Bertsimas et al. (2022), Li and Xie (2020) and cannot scale beyond $p = 100$. Therefore, we now extend the valid inequalities in Kim et al. (2021)'s "T-relaxation" to sparse PCA problems with $r > 1$. To do so, for each $t \in [r]$, we introduce an additional variable $\boldsymbol{F}^t$ to capture the entry-wise absolute value of $\boldsymbol{Y}^t$, and an additional matrix $\boldsymbol{G}^t$ which contains a sorted version of $\boldsymbol{F}^t$. We obtain:

$$\max_{\substack{\boldsymbol{Z}\in[0,1]^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle \leq k,\\ \boldsymbol{w}\in[0,1]^p}} \ \max_{\substack{\boldsymbol{Y}\in\mathcal{S}^p_+, \boldsymbol{Y}^t,\boldsymbol{F}^t,\boldsymbol{G}^t\in\mathcal{S}^p_+,\\ \boldsymbol{T}^t\in\mathbb{R}^{p\times p}_+,\\ \boldsymbol{r}^{t,D}\in\mathbb{R}^{p-1}, \boldsymbol{t}^{t,D}\in\mathbb{R}^{p\times p-1}_+}} \langle \boldsymbol{Y}, \boldsymbol{\Sigma}\rangle \tag{12}$$

$$\text{s.t.} \quad \boldsymbol{Y} \preceq \text{Diag}(\boldsymbol{w}),\ \boldsymbol{Y} = \sum_{t=1}^{k} \boldsymbol{Y}^t,\ \boldsymbol{w} \leq \boldsymbol{Z}\boldsymbol{e},$$

$$\sum_{j=1}^{p} Y^2_{i,j} \leq r Y_{i,i} w_i \qquad \forall i \in [p],$$

$$\left(\sum_{j=1}^{p} |Y_{i,j}|\right)^2 \leq k Y_{i,i} w_i \qquad \forall i \in [p],$$

$$\sum_{i\in[p]:i\neq j} Y^2_{i,j} \leq (k - r + 1) w_j (w_j - Y_{j,j}) \qquad \forall j \in [p],$$

$$
\begin{aligned}
&\pm \boldsymbol{Y}^t \leq \boldsymbol{F}^t && \forall t \in [r],\\
&G^t_{i,1} \geq G^t_{i,2} \geq \ldots \geq G^t_{i,k_t} && \forall i \in [k_t],\ \forall t \in [r],\\
&G^t_{i,j} = 0 && \forall i > k_t \text{ or } j > k_t,\ \forall t \in [r],\\
&\operatorname{tr}(\boldsymbol{Y}^t) = \operatorname{tr}(\boldsymbol{G}^t) = \operatorname{tr}(\boldsymbol{F}^t) = 1 && \forall t \in [r],\\
&\langle \boldsymbol{E}, \boldsymbol{G}^t \rangle = \langle \boldsymbol{E}, \boldsymbol{F}^t \rangle && \forall t \in [r],\\
&\sum_{i=1}^{j} G^t_{i,i} \geq j r^{D,t}_j + \sum_{j=1}^{n} t^{t,D}_{i,j} && \forall j \in [p-1],\ t \in [r],\\
&Y^t_{i,i} \leq r^{t,D}_j + t^{t,D}_{i,j} && \forall i \in [p], j \in [p-1], t \in [r],\\
&\left(F^t_{i,j}\right)^2 \leq T^t_{i,j} T^t_{j,i},\ T^t_{i,i} = F^t_{i,i} && \forall i \in [p], j \in [i-1],\ t \in [r],\\
&\sum_{j \in [p]} T^t_{i,j} = Z_{i,t},\ \sum_{i \in [p]} T^t_{i,j} = k_t F^t_{j,j} && \forall i \in [p], \forall j \in [p], t \in [r],\\
&0 \leq T^t_{i,j} \leq F^t_{i,j} && \forall i, j \in [p], t \in [r].
\end{aligned}
$$

The additional variables $\boldsymbol{r}^{t,D}$ and $\boldsymbol{t}^{t,D}$ are introduced to enforce coupling constraints between the diagonal entries of $\boldsymbol{F}^t$ and $\boldsymbol{G}^t$ (Kim et al. 2021, eq. 44), while $\boldsymbol{T}^t$ allows to couple $\boldsymbol{F}^t$ with the binary variables $\boldsymbol{Z}$ (Kim et al. 2021, eq. 50). In contrast to Kim et al. (2021), we explicitly require that each $\boldsymbol{F}^t$ is positive semidefinite (rather than that its 2×2 minors are), in order to obtain a stronger relaxation; we consider the 2×2 minors when developing a more tractable relaxation in the next section of the paper.

We now offer several remarks on the strength and generality of this formulation compared to the ones proposed in the previous section:

Remark 2. Problem (12)'s relaxation requires that we have knowledge of k_t, the sparsity of each individual PC, in addition to $k = \sum_{t \in [r]} k_t$ alone. Indeed, the constraint $G^t_{i,j} = 0$ if $i > k_t$ or $j > k_t$ is only linear if k_t is known. Thus, (12)'s relaxation is less general than (9)'s.

Remark 3. Problem (12)'s relaxation dominates Problem (9)'s, even when (9) is strengthened with the inequalities proposed in (10)-(11). Indeed, Kim et al. (2021, Theorem 13) can be extended to show that (10)-(11) are redundant in (12)'s formulation. However, we did include in (12) the inequalities we introduced in Section 2.2, which capture notions of orthogonality between the $\boldsymbol{Y}^t$'s, since the lifted formulation based on Kim et al. (2021) applies to each $\boldsymbol{Y}^t$ separately.

In our numerical experiments (see Table 1 in Section 4), we observe that (12) is strictly tighter than (9), at the price of a very modest amount of tractability. Therefore, after an initial comparison, we favor (12) throughout our numerical results. Unfortunately, (12) cannot scale beyond $p = 100$, at least with current technology, due to the presence of multiple semidefinite matrices and constraints.

2.4. Second-Order Cone Relaxations for High-Dimensional Settings

We now develop a more tractable, albeit less tight, version of (12)'s relaxation which scales to $p > 100$ features. Namely, we replace all semidefinite constraints of the form $\boldsymbol{X} \in \mathcal{S}^p_+$ with the non-negativity of their 2×2 minors, $X_{i,i}X_{j,j} \geq X^2_{i,j}$ $\forall i, j \in [p]$, as proposed by Bertsimas and Cory-Wright (2020) and references therein. This gives a second-order cone relaxation of (12), (EC.5), which we lay out in Section EC.4 for brevity.

Remark 4. We can improve the second-order cone relaxation (EC.5) without compromising its tractability by iteratively solving (EC.5) and imposing linear cuts of the form

$$\langle \boldsymbol{X}, \boldsymbol{v}\boldsymbol{v}^\top \rangle \geq \boldsymbol{0},$$

for each matrix $\boldsymbol{X}$ which is positive semidefinite in (12), where $\boldsymbol{v}$ is a trailing eigenvector of $\boldsymbol{X}$ in the most recent solution to the relaxation, as presented in Bertsimas and Cory-Wright (2020). In our numerical experiments (Section 4), we consider a version of this scheme where we iteratively impose one such cut corresponding to the most negative eigenvector in the matrices $\boldsymbol{Y}^t$, $\boldsymbol{G}^t$, $\boldsymbol{F}^t$, and $\text{Diag}(\boldsymbol{w}) - \boldsymbol{Y}$, and perform up to 50 iterations of this scheme, stopping early if the most negative eigenvalue in all constraints is -10^{-4} or larger.

3. Algorithmic Strategies

In this section, we propose three numerical strategies which provide high-quality solutions to Problem (3). First, in Section 3.1, we propose a rounding mechanism which converts a solution to one of Section 2's convex relaxations into a certifiably near-optimal solution to Problem (3). Second, in Section 3.2, we propose a formulation of Problem (3) which is amenable to existing spatial branch-and-bound codes, and can be warm-started using the rounding mechanism proposed in Section 3.1. Finally, in Section 3.3, we propose an iterative deflation heuristic that exploits the relative maturity of sparse PCA technology in the rank-one case to obtain high-quality PCs in the rank-r case for $r > 1$. All in all, we provide a suite of algorithms that allow practitioners to address instances of Problem (3) in an accurate yet practically efficient fashion. Moreover, when combined with the upper bounds developed in Section 2, these strategies allow us to, for the first time, solve large-scale instances of Problem (3) to provable near-optimality (see Section 4).

3.1. Feasible Solutions from the Relaxation via Greedy Disjoint Rounding

In this section, we develop a rounding mechanism that converts an optimal solutions to a convex relaxation (see Section 2) into a high-quality feasible solution. Historically, a numerically useful strategy for similar integer optimization problems has been to (a) solve a convex relaxation in $(\boldsymbol{Z}, \boldsymbol{Y})$, ($b$) greedily round $\boldsymbol{Z}^\star$, the solution to the relaxation, to obtain a feasible binary matrix $\hat{\boldsymbol{Z}}$

that is close to $\boldsymbol{Z}^\star$, and (c) resolve for $\boldsymbol{U}$ under the constraints $U_{i,t}=0$ if $\hat{Z}_{i,t}=0$. For example, in our prior work Bertsimas et al. (2022), we established that a similar rounding strategy performs well for the single component case. In addition, such strategies are typically an important building block for exact approaches, such as spatial branch-and-bound, to obtain good upper and lower bounds for each subproblem, especially at the root node. Therefore, a natural question is whether rounding remains numerically useful in the presence of the greater technical difficulties induced by optimizing over multiple components simultaneously.

To answer this, observe that the "relax" and "round" steps are tractable for any r, since they respectively involve solving a convex relaxation and sorting. Therefore, the numerical tractability of greedy rounding hinges on the tractability of solving for $\boldsymbol{U}$ with $\hat{\boldsymbol{Z}}$ fixed. This is equivalent to:

$$\max_{\boldsymbol{U}\in\mathbb{R}^{p\times r}} \quad \langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma}\rangle \text{ s.t. } \boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I},\ U_{i,t}=0 \text{ if } \hat{Z}_{i,t}=0,\ \forall i\in[p], t\in[r]. \tag{13}$$

Unfortunately, (13) is very expensive to solve in high-dimensional settings, due to the orthogonality constraint $\boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I}$. Indeed, in preliminary experiments on the UCI pitprops dataset ($p=13$), spatial branch-and-bound routinely required more than 7200s to solve (13), and the solution found within the first 7200 seconds was usually significantly worse than solutions identified by other methods developed in this paper (even when the same support was identified). Indeed, the orthogonality constraints currently constitute the main bottleneck for spatial branch-and-bound codes like `Gurobi`, to the extent that, on the same pitprops instances, we observed that solving (13) was not significantly faster than jointly optimizing for $\boldsymbol{Z}$ and $\boldsymbol{U}$. In order for our rounding scheme to be scalable and potentially useful for accelerating exact approaches, we avoid solving (13) directly.

To do so, we modify the rounding step (b) of the relax-round-and-resolve strategy described above. In particular, we greedily round $\boldsymbol{Z}^\star$ to the closest binary solution $\hat{\boldsymbol{Z}}$ *encoding for disjoint supports*, i.e., where $\sum_{t\in[r]}\hat{Z}_{i,t}\leq 1,\ \forall i\in[p]$. As a result, any matrix $\boldsymbol{U}$ with support $\hat{\boldsymbol{Z}}$ automatically satisfies the orthogonality constraint. We can then obtain a solution of (13) by solving for each PC independently. For each $t\in[r]$, we consider the submatrix of $\boldsymbol{\Sigma}$ over the indices $\{i:\hat{Z}_{i,t}=1\}$, extract its leading eigenvector via SVD, and pad it with zeros to construct $\boldsymbol{U}_t$. Although the restriction to disjoint support is not without loss of optimality, we will observe numerically in Section 4 that disjoint solutions are not *particularly* suboptimal for Problem (3) when k and r are small relative to p—an observation already made by Asteris et al. (2015).

We formalize our approach in Algorithm 1. In the rounding step, we require that each PC is supported on at least one feature, since otherwise the constraint $\boldsymbol{U}^\top\boldsymbol{U}=\mathbb{I}$ cannot be satisfied. Interestingly, Algorithm 2 of Bertsimas et al. (2022) can be seen as a special case of Algorithm 1, where there is only one PC and thus the solution is trivially disjoint.

Algorithm 1 A disjoint greedy rounding method for Problem (7)

Require: Covariance matrix $\mathbf{\Sigma}$, rank parameter r, sparsity parameter k

Compute $\boldsymbol{Z}^\star$ solution of (12) with (10)-(11) or (EC.5) with (10)-(11)

Construct $\boldsymbol{Z} \in \{0,1\}^{p\times r}$ solution of

$$\max_{\boldsymbol{Z}\in\{0,1\}^{p\times r}} \langle \boldsymbol{Z}, \boldsymbol{Z}^\star\rangle \text{ s.t. } \sum_{t=1}^{r} Z_{i,t} \leq 1,\ \forall i\in[p],\ \sum_{i=1}^{p} Z_{i,t} \geq 1,\ \forall t\in[r], \langle \boldsymbol{e}, \boldsymbol{Z}\rangle \leq k.$$

Compute $\boldsymbol{U}$ solution of (13) via SVD

return $\boldsymbol{Z}, \boldsymbol{U}$.

3.2. Exact Non-Convex Formulation with Warmstart and Presolving

Problem (3) is a non-convex mixed-integer quadratically constrained problem which is addressable via non-convex MIQCP solvers. Since non-convex quadratic constraints cannot be satisfied exactly, we relax the orthogonality constraint $\boldsymbol{U}^\top\boldsymbol{U} = \mathbb{I}$ to require that it is satisfied to within an elementwise tolerance of $\epsilon_{t_1,t_2} : \sum_{t_1,t_2\in[r]} \epsilon_{t_1,t_2} \leq \rho$ for some $\rho > 0$. This gives:

$$\begin{aligned}
\max_{\substack{\boldsymbol{Z}\in\{0,1\}^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle\leq k}} \max_{\boldsymbol{U}\in\mathbb{R}^{p\times r}, \boldsymbol{\epsilon}\in\mathbb{R}^{r\times r}} \quad & \langle \boldsymbol{U}\boldsymbol{U}^\top, \mathbf{\Sigma}\rangle & (14)\\
\text{s.t.} \quad & \left|\sum_{j\in[p]} U_{j,t_1}U_{j,t_2} - \delta_{t_1,t_2}\right| \leq \epsilon_{t_1,t_2}, & \forall t_1,t_2\in[r],\\
& U_{i,t} = 0 \text{ if } Z_{i,t} = 0, & \forall i\in[p],\ t\in[r],\\
& \sum_{t_1,t_2\in[r]} \epsilon_{t_1,t_2} \leq \rho,
\end{aligned}$$

where $\delta_{i,j} = 1$ if $i = j$, and 0 otherwise. Problem (14) is a non-convex quadratically constrained mixed-integer problem with pr continuous variables, pr binaries, and r^2 quadratic constraints.

However, as previously discussed, spatial branch-and-bound technology can only solve problems with $pr < 50$. Therefore, we now propose techniques to accelerate the convergence of spatial branch-and-bound by leveraging information from the convex relaxations derived in Section 2.

First, from the solution of a convex relaxation, Algorithm 1 generates a feasible solution $(\boldsymbol{Z}, \boldsymbol{U})$ to Problem (14). We propose to use this solution as a high-quality warm-start for branch-and-bound. Indeed, it is well documented in the mixed-integer optimization literature that warm-starts can dramatically accelerate the convergence of branch-and-bound (see, e.g., Bertsimas et al. 2016).

Second, when searching for a high-quality solution, we use information from the convex relaxation to pre-solve Problem (14) and restrict the search space. Namely, for any index i for which $\sum_{t\in[r]} Z^\star_{i,t} < 10^{-4}$ in the solution of the convex relaxation $\boldsymbol{Z}^\star$, we set $\sum_{t\in[r]} Z_{i,t} = 0$ in (14). Recent

work (c.f. Fischetti and Monaci 2013, Bertsimas and Digalakis 2022) similarly and successfully used information from convex relaxations to reduce the number of binary variables in other contexts. As we observe in our numerical results in Section 4, this numerical strategy accelerates branch-and-bound's convergence at the price of almost no loss in solution quality.

In addition, note that we strengthen Problem (14)'s formulation with valid inequalities derived from sparse PCA's ℓ_1 relaxation, as explored by Dey et al. (2022b,a) and in Section EC.2. Indeed, if each PC's sparsity k_t is specified a priori, we have the valid inequalities

$$\|\boldsymbol{U}_t\|_1 \leq \sqrt{k_t}, \ \forall t \in [r]. \tag{15}$$

Moreover, if k is specified but k_t is not, we instead impose the second-order cone inequalities

$$\|\boldsymbol{U}_t\|_1^2 \leq \sum_{i\in[p]} Z_{i,t}, \ \forall t \in [r], \tag{16}$$

which allows us to model $k_t = \sum_{j=1}^p Z_{j,t}$ in a more expensive yet tractable fashion. In practice, these warm-starting and presolving strategies obtained via Section 2's relaxations allow MIQCP solvers to obtain high-quality solutions in a reasonable amount of time for $pr \leq 50$ (Section 4.2).

As spatial branch-and-bound technology improves over time, we believe that it should be possible to solve Problem (14) exactly at larger problem sizes. Indeed, recent works like Dong and Luo (2018), Gupta et al. (2022) solve some quadratically constrained problems with up to 50 variables to optimality using custom branch-and-bound solvers, and Gupta et al. (2022) further reports that `Gurobi`'s off-the-shelf QCQP solver has achieved a machine independent speedup factor of 67.5 in less than two years, which suggests that larger instances of (14) (or (13)) may soon be within reach. Unfortunately, we observe in our experiments in Section 4.1 that, due to numerical instability, the upper bounds obtained are currently not reliable for high values of (r, k) even when $p = 13$. Therefore, we use (14) primarily to obtain high-quality feasible solutions in this paper.

3.3. An Iterative Deflation Heuristic

In this section, we propose a local improvement technique which identifies near-optimal and near-feasible solutions to Problem (3). The technique is based on the theory of Lagrangean relaxations (see Geoffrion 1974), which argues that if a non-convex problem is decomposable as a sum of easier (but still non-convex) subproblems with a coupling constraint, a good strategy is often to penalize the coupling constraint in the objective and iteratively solve the non-convex subproblems with different penalty multipliers on the coupling constraint. For example, Lu and Zhang (2012) propose an augmented Lagrangian method for solving an ℓ_1 relaxation of the sparse PCA, where the sparsity constraints are replaced by an ℓ_1 penalty in the objective.

To apply this perspective to our sparse PCA problem with multiple PCs, let us first rewrite (7) without the variable $\boldsymbol{Y} = \sum_{t\in[r]} \boldsymbol{Y}^t$ as:

$$\max_{\substack{\boldsymbol{Z}\in\{0,1\}^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle \le k}} \max_{\boldsymbol{Y}^t\in\mathcal{S}^p_+} \sum_{t\in[r]} \langle \boldsymbol{Y}^t, \boldsymbol{\Sigma}\rangle \text{ s.t. } \begin{array}{l} \sum_{t\in[r]} \boldsymbol{Y}^t \preceq \mathbb{I} \\ \text{tr}(\boldsymbol{Y}^t) = 1,\ \forall t\in[r], \\ Y^t_{i,j} = 0 \text{ if } Z_{i,t} = 0\ \forall t\in[r], i,j\in[p], \\ \text{Rank}(\boldsymbol{Y}^t) = 1 \forall t\in[r]. \end{array} \tag{17}$$

Hence, Problem (17) is the sum of r rank-1 sparse PCA problems coupled via linear constraints on their respective supports $\boldsymbol{Z}_t$ and the semidefinite constraint $\sum_{t\in[r]} \boldsymbol{Y}^t \preceq \mathbb{I}$. Therefore, if the coupling constraint can be handled appropriately then this problem can be addressed via scalable methods for rank-one sparse PCA, as reviewed in the introduction. To this end, we assume that a sparsity budget k_t is imposed for each component as in Section 2.3, i.e., $\sum_{i\in[p]} \boldsymbol{Z}_{i,t} \le k_t,\ \forall t\in[r]$, and invoke Proposition 1 to replace the orthogonality constraint with $r(r-1)/2$ bilinear scalar constraints $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'}\rangle = 0,\ \forall t'\neq t$. Finally, since $\langle \boldsymbol{X}, \boldsymbol{W}\rangle \ge 0$ for any positive semidefinite matrices $\boldsymbol{X}, \boldsymbol{W}$ of the same size, we replace $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'}\rangle = 0,\ \forall t'\neq t$ with $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'}\rangle \le 0,\ \forall t'\neq t$ without loss of generality. Therefore, for any penalty $\lambda > 0$ we have the following valid Lagrangean relaxation:

$$\max_{\substack{\boldsymbol{Z}\in\{0,1\}^{p\times r}:\ \boldsymbol{Y}\in\mathcal{S}^p_+, \boldsymbol{Y}^t\in\mathcal{S}^p_+ \\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle \le k}} \sum_{t\in[r]} \langle \boldsymbol{Y}^t, \boldsymbol{\Sigma}\rangle - \lambda \sum_{t,t'\in[r]:t\neq t'} \langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'}\rangle \text{ s.t. } \begin{array}{l} \text{tr}(\boldsymbol{Y}^t) = 1,\ \forall t\in[r], \\ Y^t_{i,j} = 0 \text{ if } Z_{i,t} = 0\ \forall t\in[r], i,j\in[p], \\ \text{Rank}(\boldsymbol{Y}^t) = 1 \forall t\in[r]. \end{array} \tag{18}$$

Given an index t and a sparsity budget k_t, optimizing for $\boldsymbol{Y}^t$ (with all other $\boldsymbol{Y}_{t'}$, $t'\neq t$, and λ fixed) is equivalent to finding the leading k_t-sparse PC of the matrix $\boldsymbol{\Sigma} - \lambda\sum_{t'\neq t} \boldsymbol{Y}_{t'}$. Since $\text{tr}(\boldsymbol{Y}^t) = 1$, we can add a term $\lambda_{\text{offset}}\text{tr}(\boldsymbol{Y}^t)$ to the objective without impacting the optimal solution. Accordingly, in practice, we consider the matrix $\boldsymbol{\Sigma} - \lambda\sum_{t'\neq t} \boldsymbol{Y}_{t'} + \lambda_{\text{offset}}\mathbb{I}$, where the constant $\lambda_{\text{offset}} > 0$ is sufficiently large that the entire cost matrix is positive semidefinite, as required by most sparse PCA algorithms for $r = 1$. In our implementation, we solve these subproblems via the relax-and-round strategy from Section 3.1 in the special case $r = 1$ (analogous to Bertsimas et al. 2022). Algorithm 2 proceeds by optimizing for each $\boldsymbol{Y}^t$ sequentially and then iteratively increasing the penalty parameter $\lambda > 0$.

By increasing the penalty parameter λ, we can improve the orthogonality of the resulting PCs. Figure 1 illustrates the convergence of Algorithm 2 with the update rule $\lambda = \ell$ on the `pitprops` dataset with $r = 3$ and $k = (10, 10, 10)$ as explored in in Section 4. Namely, Algorithm 2 initially identifies a set of high-quality but not orthogonal PCs, and converts this into a feasible solution by iteratively increasing the cost of violating the orthogonality constraint. In Section 4, for correlation matrices $\boldsymbol{\Sigma}$, we use the update rule $\lambda_\ell = \ell$ for $\ell \le 50$ and $\lambda \leftarrow 50 + 5(\ell - 50)$ for $\ell > 50$, since, as

Algorithm 2 Lagrangean Alternating Minimization for Problem (3)

Require: Matrix $\mathbf{\Sigma}$, rank parameter r, sparsity parameters $k_1, \ldots, k_r$, number of iterations L

Require: Update scheme $\{\lambda_\ell\}_{\ell \in [L]}$

$\ell \leftarrow 1$

repeat

$t \leftarrow 1$

repeat

$\lambda_{\text{offset}} \leftarrow \epsilon - \lambda_{\min}\left(\mathbf{\Sigma} - \lambda_\ell \sum_{t' \in [r]: t' \neq t} \boldsymbol{Y}_{t'}\right)$

Compute $\boldsymbol{Y}^t$ approximate solution of

$$\max_{\substack{\boldsymbol{z} \in \{0,1\}^p: \\ \boldsymbol{e}^\top \boldsymbol{z} \leq k_t}} \max_{\boldsymbol{Y} \in \mathcal{S}^p_+} \left\langle \mathbf{\Sigma} - \lambda_\ell \sum_{t' \in [r]: t' \neq t} \boldsymbol{Y}_{t'} + \lambda_{\text{offset}} \mathbb{I}, \boldsymbol{Y} \right\rangle \text{ s.t. } \operatorname{tr}(\boldsymbol{Y}) = 1, \operatorname{Rank}(\boldsymbol{Y}) = 1,$$
$$Y_{i,j} = 0 \text{ if } z_i = 0 \ \forall i, j \in [p].$$

$t \leftarrow t + 1$

until $t > r$

$\ell \leftarrow \ell + 1$

until $\ell > L$

Return $\boldsymbol{Y} = \sum_{t \in [r]} \boldsymbol{Y}^t$

reflected in Figure 1, using $\lambda_\ell = \ell$ throughout the method sometimes results in slow convergence. For covariance matrices, we scale λ_ℓ by a factor $\frac{\langle \boldsymbol{U}\boldsymbol{U}^\top, \mathbf{\Sigma} \rangle}{p \|\boldsymbol{U}^\top \boldsymbol{U} - \mathbb{I}\|_1}$, where $\boldsymbol{U}$ is a solution obtained after one iteration with $\lambda = 0$, to account for the different magnitude of the objective values.

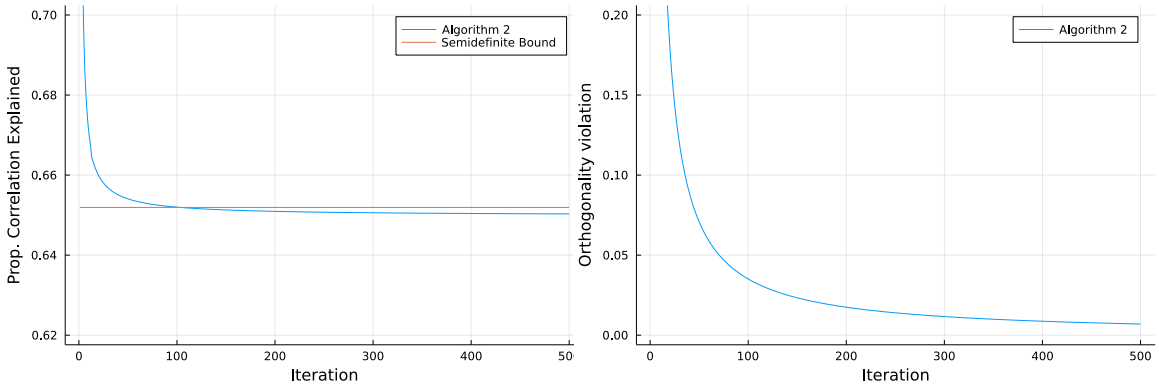

Figure 1 Convergence behavior of Algorithm 2 on the UCI pitprops dataset ($p = 13$) with $r = 3$ and $k = (10, 10, 10)$ with respect to: the proportion of correlation explained compared to the semidefinite upper bound (left) and the total orthogonality constraint violation $\|\boldsymbol{U}^\top \boldsymbol{U} - \mathbb{I}\|_1$ (right).

For futher scalability improvements, we also consider running Algorithm 2 only on indices i where $\sum_{t\in[r]} Z^\star_{i,t} > \epsilon$ in $\boldsymbol{Z}^\star$, a solution to the relaxation solved by Algorithm 1, as discussed in the previous section. We observe in our numerical results in Section 4 that this numerical strategy dramatically accelerates Algorithm 2 convergence at the price of a small loss in solution quality.

4. Numerical Results

In this section, we evaluate the algorithmic strategies derived in the previous two sections, implemented in `Julia` 1.7 using `JuMP.jl` 1.0.0, `Gurobi` version 9.5.1 to solve all non-convex quadratically constrained problems, and `Mosek` 9.1 to solve all conic relaxations. Except where indicated otherwise, all experiments were performed on a standard MacBook Pro laptop with a 2.9GHz 6-Core Intel i9 CPU, using 16 GB DDR4 RAM. For the purpose of averaging results across datasets with different p's, we report the proportion of variance explained whenever we report an objective value. For a correlation matrix, this corresponds to dividing by p, the number of features.

4.1. Performance of Upper Bounds

In this section, we compare the upper bounds from our disjoint and permutation invariant semidefinite relaxations—(9) with (10)-(11), hereafter "Disjoint-Ineq", and (12) with (10)-(11), hereafter 'Perm-Ineq"—against the bound obtained from the exact formulation (14) strengthened with the ℓ_1 valid inequalities (15)-(16) after a time limit of $3,600$s, hereafter "Non-Convex". We also explore the scalability of these formulations, and their second-order cone relaxations.

Benchmarking on Pitprops Data We first compare the bounds generated by each method—in terms of proportion of correlation explained—on the UCI `pitprops` dataset ($p = 13$) as we vary $r \in \{2, 3\}$ and $k \in \{2r, 3r, 4r, 5r\}$, in Table 1. We consider both imposing an overall sparsity budget alone (denoted by "$k, -$") and imposing a seperate budget for each PC, denoted by "$k, (k_1, \ldots, k_t)$". We impose a relative optimality tolerance of 10^{-6} for the exact formulation, set the parameters `FuncPieceError` and `FuncPieceLength` to 10^{-6} and 10^{-5} respectively (their minimum possible values), maintain an elementwise feasibility tolerance of 10^{-8} for the orthogonality constraint, and report the upper bound at the time limit if branch-and-bound does not terminate.

On the instances presented in Table 1, we observe that spatial branch-and-bound on our non-convex formulation (14) terminates within minutes and provides the tightest (i.e., smallest) upper bound. In comparison, the conic relaxations terminate in less than a second, while providing upper bounds that are only weaker at the third decimal. Unfortunately however, for $r = 3$ and $r > 9$, we started experiencing instability issues with spatial branch-and-bound. For example, for $r = 3$ and $k = 12$, it returned a global upper bound of 0.5440 after 1 hour, while a feasible solution with objective 0.5738 existed. Hence, with the current state of technology, the upper bounds obtained

Rank (r)	Sparsity (k, k_t)	Disjoint-Ineq		Perm-Ineq		Non-Convex		
		UB	T(s)	UB	T(s)	UB	Nodes	T(s)
2	4, -	0.297	0.86	-	-	0.295	4,600	17.32
2	4, (1, 3)	**0.267**	0.23	**0.267**	1.59	**0.267**	2,500	1.04
2	4, (2, 2)	**0.295**	0.17	**0.295**	0.31	**0.295**	3,000	7.45
2	6, -	0.384	0.63	-	-	**0.371**	12,200	52.73
2	6, (1, 5)	**0.339**	0.24	**0.339**	0.22	**0.339**	4,900	6.32
2	6, (2, 4)	**0.371**	0.26	**0.371**	0.33	**0.371**	5,200	14.34
2	6, (3, 3)	0.361	0.26	**0.360**	0.39	**0.360**	10,900	19.84
2	8, -	0.451	0.59	-	-	0.435	53,600	256.1
2	8, (1, 7)	**0.384**	0.25	**0.384**	0.21	**0.384**	22,900	7.41
2	8, (2, 6)	**0.435**	0.28	**0.435**	0.31	**0.435**	16,200	8.65
2	8, (3, 5)	0.420	0.28	**0.418**	0.41	**0.418**	25,600	24.03
2	8, (4, 4)	0.412	0.31	0.408	0.33	**0.404**	77,000	93.73
2	10, -	0.490	0.65	-	-	**0.458**	388,500	796.74
2	10, (1, 9)	**0.395**	0.32	**0.395**	0.21	**0.395**	86,900	41.41
2	10, (2, 8)	**0.457**	0.36	**0.457**	0.32	**0.457**	67,400	42.77
2	10, (3, 7)	0.461	0.3	0.459	0.31	**0.458**	116,400	56.46
2	10, (4, 6)	0.458	0.28	0.455	0.39	**0.451**	265,000	241.72
2	10, (5, 5)	0.453	0.3	0.449	0.32	**0.426**	78,400	>3600
3	6, -	0.443	0.82	-	-	0.435	74,200	2544.9
3	6, (1, 1, 4)	**0.380**	0.74	**0.380**	1.05	**0.380**	5,500	9.14
3	6, (1, 3, 2)	**0.412**	0.34	**0.412**	0.4	**0.412**	7,000	13.54
3	6, (2, 2, 2)	**0.435**	0.22	**0.435**	0.35	**0.435**	16,600	34.83
3	9, -	0.570	0.76	-	-	0.539	237,800	>3600
3	9, (1, 1, 7)	**0.461**	0.36	**0.461**	0.35	**0.461**	186,100	74.06
3	9, (1, 2, 6)	**0.512**	0.34	**0.512**	0.42	**0.512**	15,400	19.29
3	9, (1, 3, 5)	0.497	0.32	**0.495**	0.41	**0.495**	48,500	66.15
3	9, (1, 4, 4)	0.489	0.34	0.485	0.45	**0.481**	125,000	173.31
3	9, (2, 2, 5)	**0.539**	0.44	**0.539**	0.41	**0.539**	56,600	138.34
3	9, (2, 3, 4)	0.532	0.44	0.531	0.46	**0.530**	95,500	313.76
3	9, (3, 3, 3)	0.520	0.34	0.512	0.45	**0.511**	463,500	1316.15

Table 1 Performance of upper bounds on the pitprops dataset ($p=13$), as we vary the overall sparsity (k), the number of PCs (r) and the allocation of a sparsity budget to the different PCs. We denote the best performing solution (least upper bound) in bold. Note that all results are normalized by dividing by the trace of $\mathbf{\Sigma}$, i.e., p, the number of features, to report results in terms of the proportion of variance explained.

by solving (14) may not be reliable as k or r increases or for $p>13$. Accordingly, in the rest of this paper, we only use our convex relaxations to certify a solution's quality.

Regarding the two semidefinite relaxations "Disjoint-Ineq" and "Perm-Ineq", we should remind the reader that Perm-Ineq cannot compute a bound if we specify an overall sparsity budget k only. Disjoint-Ineq's bound for a given k is much weaker than the worst-case bound over all possible allocations of k_t's that sum to k. This can be explained by the strong second-order cone inequalities (10)-(11) that can be added in the latter case. Therefore, time permitting, we recommend computing the lower bound by solving the relaxations for all possible allocations of k_t and taking the worst-case bound. Furthermore, when individual sparsity budgets k_t are given, we observe that Perm-Ineq provides uniformly and sometimes significantly tighter bounds than Disjoint-Ineq. Ac-

cordingly, in the rest of the paper, we only consider instances of Problem (3) where we know both k and k_t and consider Perm-Ineq and its second-order cone relaxations, but not Disjoint-Ineq.

For concision, we will mainly report the results for symmetric sparsity allocations, i.e., $k_t = k/r$, only. In Table 1, we observe that for each value of (r,k), the worst-case upper bound is often achieved with asymmetric sparsity allocation, which suggests that sparse PCs with asymmetric sparsity allocation could explain more variance under a fixed sparsity budget. We investigate this matter in detail in Section 4.4.

Benchmarking on Larger-Scale Datasets We now investigate the scalability of the relaxation Perm-Ineq and its second-order cone relaxations on larger UCI datasets. Table EC.1 (see Section EC.5.1) compares the performance of the original semidefinite formulation (12) with (10)-(11) (PSD), its second-order cone relaxation (EC.5) with (10)-(11) and up to 50 PSD cuts (SOC-Cuts) as in Remark 4, and the second-order cone relaxation alone (EC.5) with (10)-(11) (SOC). Due to high memory or time requirements, some instances (denoted by a $*$) are solved on Intel Xeon E5–2690 v4 2.6GHz CPU machines with 256 GB RAM. A "-" indicates that the required memory exceeds 256 GB.

We observe that the PSD relaxation can be solved within a few minutes for $p \approx 50$ but quickly requires a prohibitive amount of memory and time at higher dimensions. The SOC-Cuts relaxation scales up to $p \approx 300$ and provides a high-quality upper bounds, within $1-2\%$ of PSD. Without the additional cuts, the SOC relaxation alone is weak when (k,r) are large relative to p. For example, for the pitprops dataset, when $k_t = (10,10,10)$, SOC returns an upper bound of 1.007 while the proportion of correlation explain can trivially not exceed 1. However, SOC is still preferable in high-dimensional settings, where the semidefinite formulation cannot be solved via an interior point method due to excessive memory requirements, and the formulation SOC-Cuts provides a small improvement in the upper bound at the price of significantly more runtime.

4.2. Performance of Feasible Methods

In this section, we numerically evaluate the quality of the three methods developed in Section 3 in terms of their ability to recover approximately orthogonal high-quality principal components on real-world datasets. We first validate that the variable fixing strategy proposed in Section 3.2 effectively reduces the number of variable without damaging the quality of the obtained solutions. We then compare our algorithms with state-of-the-art techniques on seven UCI datasets. Since Algorithm 2 and all three benchmarked algorithms from the literature require that the sparsity of each PC is specified separately, we consider this formulation in this section (and fix $k_t = k/r$).

Each of the algorithms proposed in Section 3 involves solving one of Section 2's convex relaxations. Based on the scalability results presented in the previous section, we use different convex

relaxations depending on the dimensionality of the problem. Namely, for Algorithm 1, we use the full semidefinite relaxation (12) if $p \leq 50$, the SOC relaxation (EC.5) with 50 PSD cuts if $p \leq 200$, and the SOC relaxation (EC.5) otherwise. The exact method from Section 3.2 uses the solution from Algorithm 1 as a warm-start. We impose a two hour time limit for branch-and-bound and report the best solution found at the time limit. For Algorithm 2, we use the full semidefinite relaxation (12) when rounding each subproblem if $p \leq 50$, and the SOC relaxation (EC.5) otherwise. We also impose a limit of 100 iterations for $p \leq 200$ or 200 iterations for $p > 200$.

Effectiveness of Variable Fixing: We first apply our three methods on four small and medium-sized UCI datasets (`Pitprops`, `Wine`, `Ionosphere`, and `Geography`) as we vary $r \in \{2, 3\}$ and $k \in \{5, 10, 20\} : k \leq p$. Summary results are reported in Table 2 (see also Table EC.3 in Section EC.5.3 for instance-wise results). We compute the relative gap between the solutions computed via each method and Algorithm 1's upper bound. For both Algorithm 2 and the spatial branch-and-bound method, we compare their naive implementation with the variable fixing heuristic proposed in Section 3.2. We denote these variants "Algorithm 2 (fixed)" and "branch-and-bound (fixed)" respectively. For these two variants, the computational time does not include the time required to solve a convex relaxation, which is approximately equal to the time needed by Algorithm 1.

Method	Obj.	Rel. gap (%)	Viol.	Obj. minus Viol.	T(s)	Nodes
Algorithm 1	0.366	13.25%	0	0.366	234.3	n/a
Algorithm 2	0.391	9.04%	0.009	0.382	1618	n/a
Algorithm 2 (fixed)	0.410	4.83%	0.017	0.393	1516	n/a
Branch-and-bound	0.384	10.16%	0	0.384	7031	162200
Branch-and-bound (fixed)	0.387	9.40%	0	0.387	7076	206900

Table 2 Average performance of methods across the UCI pitprops, wine, ionosphere and geographical datasets with $k \in \{5, 10, 20\}, r \in \{2, 3\} : k \leq p$. Note that the objective value is $\frac{1}{p}\left\langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma} \right\rangle$, the constraint violation is $\|\boldsymbol{U}^\top\boldsymbol{U} - \mathbb{I}\|_1$, and the objective value minus the constraint violation is $\frac{1}{p}\left\langle \boldsymbol{U}\boldsymbol{U}^\top, \boldsymbol{\Sigma} \right\rangle - \|\boldsymbol{U}^\top\boldsymbol{U} - \mathbb{I}\|_1$.

For Algorithm 2, the variable fixing heuristic reduces computational time by nearly 10% and surprisingly improves the average objective from 0.391 to 0.410, thus halving the average relative optimality gap. By inspecting the instance-wise results in Table EC.3-EC.4, we observe that this gain happens mostly for the `Geography` dataset with $p = 68$ features. Recalling that we use a PSD relaxation for $p \leq 50$ and a SOC relaxation otherwise, it is clear that the variable fixing heuristic allows the effective dimension of the `Geography` dataset to fall below 50, enabling "Algorithm 2 (fixed)" to use PSD relaxations, which are tighter than the SOC relaxation used by "Algorithm 2" for the same data. For the branch-and-bound approach, the scalability gains translate into a higher number of nodes explored (+25%) within the same time limit, although most instances are still not solved to optimality.

Comparing against the semidefinite bounds computed in the previous section, we observe that all methods return solutions which are at most 15% suboptimal on average and that Algorithm 2 (fixed) provides the best average performance ($\leq 5\%$ optimality gap on average).

Benchmarking on Pitprops Data: Lu and Zhang (2012) extensively compared the performance of existing sparse PCA methods on the `pitprops` dataset with $r = 6$ PCs. All six methods they benchmarked required an overall sparsity of around 45–60 to explain less than 70% of the variance. The best performing method could explain 69.55% of the variance with an overall sparsity of 46 (Lu and Zhang 2012, Table 11). They concluded that "there do not exist six highly sparse, nearly orthogonal and uncorrelated PCs while explaining most of variance". The three methods we developed in Section 3 strongly challenge their conclusions. As reported in Table EC.2, with $r = 6$ PCs and $k_t = 2$, hence an overall sparsity of 12, solutions returned by *any* of our methods explain 73%–75% of the variance. Algorithm 2 even provides a solution that explains 81% of the variance with an overall sparsity of $6 \times 4 = 24$. That is to say, what was previously considered by the community to be impossible can be done with the techniques in this paper, in seconds.

Benchmarking on Larger-Scale Datasets We now investigate the performance of Algorithm 1, Algorithm 2 (fixed) and branch-and-bound (fixed) on seven UCI datasets, whose dimension range from $p = 13$ (`pitprops`) to $p = 1300$ (`micromass`). We compare them with three state-of-the-art methods from the literature. Namely,

- The branch-and-bound method of Berk and Bertsimas (2019) for optimally computing one sparse PC, combined with the deflation scheme of Mackey (2008) to obtain multiple PCs, implemented in `Julia` and made available at `github.com/lauren897/Optimal-SPCA`. According to Berk and Bertsimas (2019), this method outperformed four others across three UCI datasets ($r = 3, k = 5$).
- The deflation method of Hein and Bühler (2010), using the custom deflation method developed in Bühler (2014), implemented in `Matlab` and made publicly available at `github.com/tbuehler/sparsePCA`, using default parameters. This approach was found by Berk and Bertsimas (2019, Table 9) to be second-best of the methods in their comparison.
- The `Lasso`-inspired method of Zou et al. (2006), using the `spca` function in the `elasticnet` package version 1.3, using default parameters. This approach is perhaps the most commonly used one in practice, since it is distributed via the ubiquitous `elasticnet` package.

We report summary results in Table 3 (see also Tables EC.4-EC.5 in Section EC.5.4 for instance-wise results). We remind the reader that the methods from the literature we benchmark against do not provide an upper bound in and of themselves; we need Algorithm 1 to compute optimality gaps. Also, when the returned solution violates the orthogonality condition, its objective value is not necessarily a valid bound on (3)'s objective and the reported gap is an optimistic estimate.

Method	Obj.	Rel. gap (%)	Viol.	Obj. minus Viol.	T(s)
Algorithm 1	0.232	19.19%	0	0.232	16740
Algorithm 2 (fixed)	0.249	18.25%	0.013	0.236	1580
Branch-and-bound (fixed)	0.241	18.61%	0	0.241	>7200
Berk and Bertsimas (2019)	0.266	3.22%	0.045	0.221	40.89
Hein and Bühler (2010)	0.233	28.68%	0.033	0.200	0.26
Zou et al. (2006)	0.069	78.16%	1.599	-1.530	4.26

Table 3 Average performance of methods across the UCI pitprops, wine, ionosphere, geographical, communities, arrhythmia and micromass datasets with $k \in \{5, 10, 20\}, r \in \{2, 3\} : k \leq p$

.

In terms of the average proportion of correlation explained minus the average orthogonality constraint violation, we observe that Algorithm 2 (fixed) and branch-and-bound (fixed) perform the best overall, with scores of 0.236 and 0.241 respectively, which compares favorably with the average upper bound computed by Algorithm 1 using the conic relaxations developed in Section 2 (0.276). Of the remaining methods, Algorithm 1 and the method of Berk and Bertsimas (2019) perform the next best, with respective scores of 0.232 and 0.221. Finally, the methods of Hein and Bühler (2010) and Zou et al. (2006) have the lowest respective scores, 0.200 and -1.530. One could also score the methods using the relative optimality gap alone; we avoid doing so here since some methods, e.g. the method of Berk and Bertsimas (2019) or Zou et al. (2006), often violate the orthogonality constraint significantly, thus potentially achieving artificially low optimality gaps.

We remark that no one method performs best on every instance. The method of Berk and Bertsimas (2019) performs best on instances where k is very small and where disjoint solutions are nearly optimal, in which case the orthogonality constraint can essentially be ignored (avg. relative gap of 3.38% and avg. constraint violation of 0.00875 over instances where $p \geq 101$). Algorithm 2 (fixed), on the other hands, performs best on instances where k is large relative to p and the orthogonality constraint is important to account for (avg. relative gap of 2.55% and avg. constraint violation of 0.00614 over instances where $p \leq 34$). These results suggest that both k and the amount of overlap between the optimal PCs impact the performance of each method, and motivate a comparison on synthetic data, where we control the ground truth, in the next section.

Finally, we note that while branch-and-bound (fixed) performs well on instances where $p \leq 34$, Table EC.4 reveals that, for larger instances it does not improve upon the warm-start provided by Algorithm 1 within 7200s. Therefore, we do not consider the use of branch-and-bound any further in this paper. This situation should be revisited as global non-convex solvers improve.

4.3. Statistical Recovery Properties of Method

To evaluate the support recovery ability of each method, we now generate synthetic data according to a spiked covariance matrix model and compare the performance of Algorithms 1 and 2 (fixed)

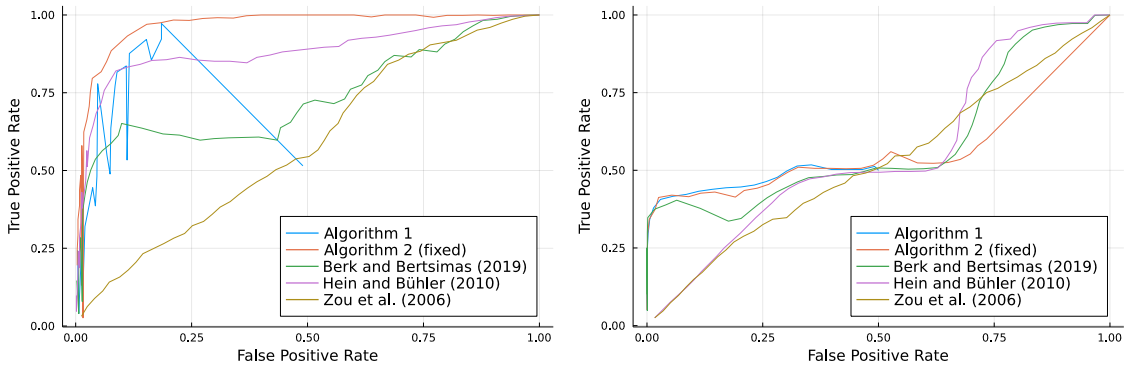

Figure 2 ROC curve over 20 synthetic instances where $p = 50$, $r = 2$, $k_{\text{true}} = (20, 20)$, and the proportion of overlap in the ground truth, $q = 0.05$ (left), $q = 0.95$ (right) is unspecified.

against the same three methods from the literature. We use an experimental setup inspired by d'Aspremont et al. (2008). Namely, we generate a test matrix $\mathbf{\Sigma} \in \mathcal{S}_+^p$, where $\mathbf{\Sigma} = \boldsymbol{U}^\top \boldsymbol{U} + \sigma \boldsymbol{x}_1 \boldsymbol{x}_1^\top + \sigma \boldsymbol{x}_2 \boldsymbol{x}_2^\top$, $\boldsymbol{U} \in [0,1]^{p \times p}$ is a noisy matrix with i.i.d. uniform entries, $\sigma = 2$ is the signal-to-noise ratio, $p = 50$, and $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are k_{true}-sparse binary vectors (we fix $k_{true} = 20$). We also control the proportion of overlap between the supports of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, $q := \frac{\|\boldsymbol{x}_1 \odot \boldsymbol{x}_2\|_0}{\|\boldsymbol{x}_1\|_0}$, where $\odot$ denotes elementwise vector multiplication. Note that $q = 0$ and $q = 1$ correspond to the special cases of sparse PCA with disjoint support and row-sparsity discussed in Section 1.2. We impose a limit of 200 iterations for Algorithm 2.

Figure 2 depicts the ROC curves (true positive rate vs. false positive rate for recovering the support of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$) as we vary k, averaged over 20 random instances, for low and high overlap ($q \in \{\frac{1}{20}, \frac{19}{20}\}$). Note that, since Algorithm 1 cannot allow any overlap between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, it converges (as $k \to p$) towards setting $Z_{i,t} = 1 \ \forall i, t$ in the relaxation and randomly setting each $Z_{i,t} = 0$ with probability 1/2 in the rounding step, leading the ROC curve to end at $(1/2, 1/2)$. First, we observe high that overlap proportion negatively impacts the support recovery ability of all methods, since all ROC curves are noticeably closer to the diagonal line (random classifier) when the overlap proportion q increases. For low overlap ($q = \frac{1}{20}$), Algorithm 2 appears to strictly dominate all other methods. We also observe that the methods of Hein and Bühler (2010), Berk and Bertsimas (2019) and Algorithm 1 behave almost identically for $k \leq 10$, which aligns with our finding in Section 4.2 that these methods are comparable when k is small.

We confirm these findings with Table 4 that reports the Areas Under the ROC Curve (AUC) and the true positive and false positive rates (TPR, FPR) at the operator point $k = k_{\text{true}} = 20$, for each method. Note that there is no clear correspondence between $\boldsymbol{U}_1$, $\boldsymbol{U}_2$ and $\boldsymbol{x}_1, \boldsymbol{x}_2$, since any permutation of the columns of $\boldsymbol{U}$ gives an equivalent solution. Therefore, we compute the better true/false positive rates over both permutations. Moreover, to compute the AUC for Algorithm 1, we augment its ROC curve with the point $(1, 1)$.

Method	$q=\frac{1}{20}$			$q=\frac{1}{2}$			$q=\frac{19}{20}$		
	AUC	TPR	FPR	AUC	TPR	FPR	AUC	TPR	FPR
Algorithm 1	0.7377	0.8363	0.1092	0.6512	0.6287	0.2475	0.6068	**0.5137**	**0.3242**
Algorithm 2	0.9663	**0.8850**	**0.0767**	0.7276	**0.6785**	**0.2142**	0.5687	0.5100	0.3267
Berk and Bertsimas (2019)	0.7359	0.5975	0.2683	0.6542	0.5075	0.3283	0.5879	0.4763	0.3492
Hein and Bühler (2010)	0.8825	0.8325	0.1117	0.5954	0.3863	0.4012	0.5541	0.4725	0.3517
Zou et al. (2006)	0.5816	0.4438	0.3708	0.4693	0.3813	0.4125	0.5231	0.4287	0.3808

Table 4 Area under the Receiver Operator Curve (AUC) depicted in Figure 2 and True Positive Rate (TPR), False Positive Rate (FPR) at the operator point $k=k_{\text{true}}=20$ for each method, as we vary q. For each q, we denote the best performing method in terms of TPR/FPR in bold.

Across the three degrees of overlap, we observe that the method of Zou et al. (2006) is the least accurate, with a substantially larger false detection rate than the remaining methods.

For $q=\frac{1}{20},\frac{1}{2}$ (low and medium overlap), Algorithm 2 achieves the highest AUC (0.9663 and 0.7276 respectively) and dominates all other methods (highest TPR, lowest FPR) at the operator point. This constitutes a substantial 7%-8% improvement in AUC compared to the better of the method of Berk and Bertsimas (2019) (AUC: $0.7359, 0.6542$) and the method of Hein and Bühler (2010) (AUC: $0.8825, 0.5954$). Algorithm 1 achieves a similar AUC than Berk and Bertsimas (2019), hence is outperformed by (resp. outperforms) Hein and Bühler (2010) for $q=\frac{1}{20}$ (resp. $q=\frac{1}{2}$). In terms of (TPR, FPR), Algorithm 1 actually dominates both Berk and Bertsimas (2019) and Hein and Bühler (2010), for $q=\frac{1}{20},\frac{1}{2}$.

For $q=\frac{19}{20}$ (high overlap), Algorithm 1 and the method of Berk and Bertsimas (2019) obtain the best overall AUC. Algorithms 1 and 2 dominate the other methods in terms of (TPR, FDR). However, as already observed in Figure 2, all AUCs are in the $0.5-0.6$ range and the differences between methods are less acute than in the low- to medium-overlap cases.

Overall, Algorithm 2 dominates current state-of-the-art methods, in terms of (FDR, TPR) across all overlap regimes, and in terms of AUC in two out of three cases. Although it only returns PCs with disjoint support, Algorithm 1 performs is one of the best performing methods, achieving similar AUC as Berk and Bertsimas (2019).

4.4. Specifying the Sparsity Pattern: The Benefits of Asymmetry

While Problem (3) only requires a bound on the total sparsity, thus allowing flexibility on how this budget is allocated across PCs, the worst-case semidefinite upper bound over all sparsity patterns $(k_1,\ldots,k_r):\sum_{t\in[r]}k_t=k$ is often significantly tighter than Problem (3)'s semidefinite relaxation with a sparsity budget of k alone, as demonstrated in Section 4.1. Moreover, Algorithm 2, which as demonstrated in Sections 4.2–4.3 is currently the best performing method for obtaining feasible solutions to Problem (3), requires that $(k_1,\ldots,k_t)$ are individually specified. Collectively, these

observations suggest that it may be necessary to enumerate all allocations of the sparsity budget k, which could be expensive. In our experiments, as is often done in practice, we restricted our search to symmetric allocations. In this section, we revisit the symmetry assumption, investigate when it is justified, and study the relative benefits of asymmetric sparsity budget allocations in terms of obtaining equally sparse sets of PCs that explain more variance.

We consider the `pitprops`, `ionosphere`, `geographical` and `communities` UCI datasets with a fixed number of PCs $r=3$ and a given overall sparsity budget $k \in \{15, 30\}$. Accordingly, in Figures 3 and EC.1 of Section EC.5.5, we depict the relationship between the proportion of correlation explained in the data for each possible allocation of the sparsity budget $(k_1, k_2, k_3) : k_1+k_2+k_3 = k, p \geq k_1 \geq k_2 \geq k_3 \geq 1$, as computed by Algorithm 2 with a limit of 200 iterations and the same setup as in Section 4.2 (and the corresponding upper bound computed by Algorithm 1), against the relative asymmetry in the sparsity budget, as measured by

$$\frac{\mathrm{KL}((k_1,k_2,k_3)||(k/3,k/3,k/3))}{\max_{p\geq k_1\geq k_2\geq k_3 : k_1+k_2+k_3=k} \mathrm{KL}((k_1,k_2,k_3)||(k/3,k/3,k/3))},$$

where $KL(p,q) := \sum_i p_i \log(p_i/q_i)$ denotes the KL divergence.

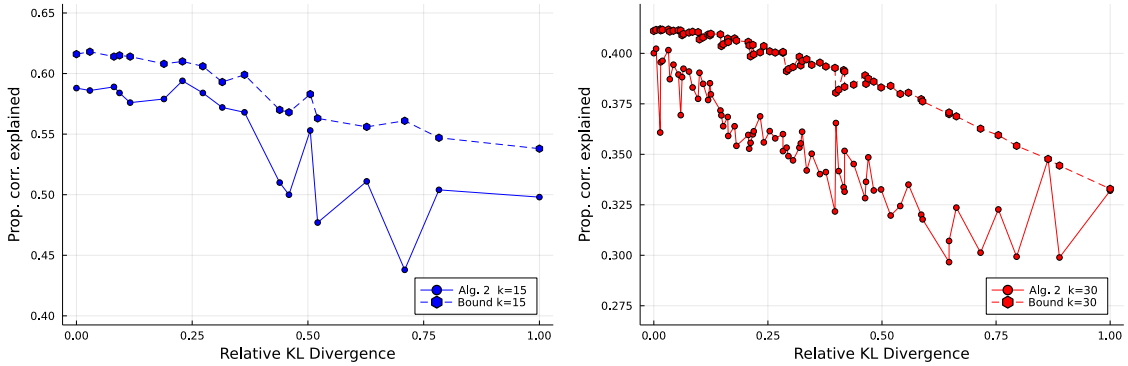

Figure 3 Symmetry of sparsity budget allocation vs. proportion of correlation in the dataset explained for the pitprops $k=15$ (left) and ionosphere $k=30$ (right) where $r=3$. Note that we normalize the KL divergence for $k=15$ and $k=30$ separately.

We observe a general trend that more symmetric sparsity budget allocations tend to explain more of the correlation in the data (both in terms of actual correlation explained by a solution from Algorithm 2 and in terms of the upper bound). This suggests that, when time is a concern, requiring that all PCs are equally sparse is a reasonable approach.

Table 5 compares the quality of Algorithm 2's solution (a) when all PCs have the sparsity budget of k/r and (b) the maximum possible correlation explained over all feasible allocations of the sparsity budget k (computed by enumerating all possible allocations of the sparsity budget-18 such allocations for $k=15, p \geq k$ and 74 allocations for $k=30, p \geq 30$), together with the upper bound on

the proportion of correlation explained obtained in each case. We observe that in several instances a perfectly symmetric allocation of the sparsity budget yields the highest quality solution, and in all instances a perfectly symmetric allocation is within 10% in the worst-case (and within 1.65% in average) of the best solution. In the enumerated case, we also compute the optimality gap between the worst-case upper bound over all sparsity budget allocations, and the best solution found, and observe that on average it is less than 1% over the instances considered. Note that this is a different gap to the one reported in Section 4.2, where the upper bound is computed after assuming that all PCs are equally sparse.

Dataset	p	r	k	Symmetric			Enumerated					Improvement (%)
			k	UB	Obj.	Viol.	UB	k_t	Obj.	Rel. gap (%)	Viol.	
Pitprops	13	3	15	0.616	0.591	0.003	0.618	(7, 6, 2)	0.595	3.67%	0.002	0.45%
			30	0.652	0.650	0.005	0.652	(10, 10, 10)	0.650	0.28%	0.005	0%
Ionosphere	34	3	15	0.297	0.291	0	0.299	(7, 6, 2)	0.299	0%	0	2.84%
			30	0.411	0.400	0	0.412	(11, 10, 9)	0.402	2.35%	0	0.54%
Geographical	68	3	15	0.221	0.219	0	0.221	(6, 5, 4)	0.221	0%	0	0.90%
			30	0.410	0.389	0	0.420	(12, 12, 6)	0.420	0%	0	7.29%
Communities	101	3	15	0.141	0.141	0	0.142	(6, 5, 4)	0.142	0.01%	0	0.42%
			30	0.246	0.245	0	0.247	(11, 11, 8)	0.247	0.17%	0	0.78%

Table 5 Comparison of symmetric and enumerated solution computed by Algorithm 2 for a given sparsity budget k_{total} and given number of PCs ($r = 3$). We report the largest upper bound over all sparsity budget allocations as our enumerated upper bound, and report the relative optimality gap between the best solution and the worst-case bound. On average, considering asymmetric sparsity budget allocations improves the proportion of correlation explained by 1.65%.

All in all, more symmetric allocations of the sparsity budget tend to perform better on average. Therefore, for a given sparsity budget k, a reasonable strategy could be to (a) run Algorithm 2 to compute a perfectly symmetric allocation, (b) compute Problem (12)'s upper bound across all possible allocations, and (c) run Algorithm 2 only on the asymmetric allocations for which the upper bound from (b) allows for a significant potential improvement upon the symmetric solution.

As a robustness check, we also report the quality of the solution and bound obtained by Algorithm 1 where we specify k but not k_t in Section EC.5.6. We observe that the best solution obtained by Algorithm 1 without specifying k_t is significantly worse than the solution obtained by Algorithm 2 with $k_1 = k_2 = k_3$, and moreover there is little correlation between the k_t's identified by Algorithm 1 and the best k_t's obtained by Algorithm 2, which validates the idea that running Algorithm 2 with a perfectly symmetric sparsity allocation is a good idea in practice.

4.5. Summary and Guidelines From Numerical Experiments

In summary, our main findings from our numerical experiments are as follows:

- The sparsity constraint on $\boldsymbol{U}$ can either be imposed via an overall sparsity budget of $\|\boldsymbol{U}\|_0 \leq k$ or with component-specific sparsity constraints $\|\boldsymbol{U}_t\|_0 \leq k_t$. As reflected in Section 4.1, constraining each column separately yields substantially tighter conic relaxations. Moreover, some of the best performing algorithms for obtaining feasible solutions, including our Algorithm 2, explicitly require a constraint on the non-zero entries in each column. Therefore, constraining the sparsity of each column separately should be preferred wherever feasible.
- If practitioners have an overall sparsity budget but are agnostic about the sparsity of each column, a reasonable strategy is to require that all columns are equally sparse (i.e., set $k_t = k/r$), as shown in Section 4.4. In our experiments, considering asymmetrically sparse sets of PCs increases the amount of variance explained by around 2% on average, at the price of increasing the total runtime by an order of magnitude.
- Under a separate sparsity constraint on each column of $\boldsymbol{U}$, our proposed combination of solving a conic relaxation and either rounding to obtain a disjoint solution or running Algorithm 2 yields certifiably near optimal solutions within minutes (resp. hours) for problems with hundreds (resp. thousands) of features. Moreover, it substantially outperforms existing methods for sparse PCA with multiple PCs in terms of obtaining higher-quality solutions (Section 4.2) with a lower false discovery rate (Section 4.3). Therefore, it should be considered as a viable and more accurate alternative for sparse PCA problems with multiple PCs.

5. Conclusion

In this paper, we studied the problem of selecting a set of mutually orthogonal sparse principal components and proposed techniques which, for the first time, allow this problem to be solved to certifiable (near) optimality with 100s or 1000s of features in minutes or hours. In particular, we proposed a strong semidefinite relaxation (Section 2) which provides high-quality upper bounds on the amount of variance explainable by any set of sparse and mutually orthogonal components, and a suite of numerically efficient algorithms (Section 3) which, as demonstrated in Section 4, recover components that nearly match these upper bounds. This contributes towards an ever-growing body of work demonstrating that computationally challenging non-convex optimization problems can often be solved to provable (near) optimality in practice.

References

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).

Asteris M, Papailiopoulos D, Kyrillidis A, Dimakis AG (2015) Sparse PCA via bipartite matchings. *Advances in Neural Information Processing Systems* 28.

Atamtürk A, Gomez A (2019) Rank-one convexification for sparse regression. *arXiv preprint arXiv:1901.10334* .

Benidis K, Sun Y, Babu P, Palomar DP (2016) Orthogonal sparse PCA and covariance estimation via procrustes reformulation. *IEEE Transactions on Signal Processing* 64(23):6211–6226.

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

Berthet Q, Rigollet P (2013) Optimal detection of sparse principal components in high dimension. *The Annals of Statistics* 41(4):1780–1815.

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

Bertsimas D, Cory-Wright R, Pauphilet J (2021a) Mixed-projection conic optimization: A new paradigm for modeling rank constraints. *Operations Research* URL `http://dx.doi.org/10.1287/opre.2021.2182`.

Bertsimas D, Cory-Wright R, Pauphilet J (2021b) A unified approach to mixed-integer optimization problems with logical constraints. *SIAM Journal on Optimization* 31(3):2340–2367.

Bertsimas D, Cory-Wright R, Pauphilet J (2022) Solving large-scale sparse PCA to certifiable (near) optimality. *Journal of Machine Learning Research* 23(13):1–35.

Bertsimas D, Digalakis V (2022) The backbone method for ultra-high dimensional sparse machine learning. *Machine Learning* 1–52.

Bertsimas D, King A, Mazumder R (2016) Best subset selection via a modern optimization lens. *The Annals of Statistics* 44(2):813–852.

Bertsimas D, Kitane DL (2022) Sparse PCA: A geometric approach. *Working paper* URL `dbertsim.mit.edu/pdfs/Papers2022/Sparse%20PCA%20a%20Geometric%20Approach.pdf`.

Bertsimas D, Pauphilet J, Van Parys B (2020) Sparse regression: Scalable algorithms and empirical performance. *Statistical Science* 35(4):555–578.

Boutsidis C, Drineas P, Magdon-Ismail M (2011) Sparse features for PCA-like linear regression. *Advances in Neural Information Processing Systems* 24.

Bühler T (2014) *A flexible framework for solving constrained ratio problems in machine learning*. Ph.D. thesis, Saarland University.

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

d'Aspremont A, El Ghaoui L, Jordan MI, Lanckriet GR (2007) A direct formulation for sparse PCA using semidefinite programming. *SIAM Review* 49(3):434–448.

Del Pia A (2022) Sparse PCA on fixed-rank matrices. *Mathematical Programming* 1–19.

Dey SS, Mazumder R, Wang G (2022a) Using ℓ_1-relaxation and integer programming to obtain dual bounds for sparse PCA. *Operations Research* 70(3):1914–1932.

Dey SS, Molinaro M, Wang G (2022b) Solving sparse principal component analysis with global support. *Mathematical Programming* 1–39.

Dong H, Luo Y (2018) Compact disjunctive approximations to nonconvex quadratically constrained programs. *arXiv preprint arXiv:1811.08122* .

Eckart C, Young G (1936) The approximation of one matrix by another of lower rank. *Psychometrika* 1(3):211–218.

Fischetti M, Monaci M (2013) Backdoor branching. *INFORMS Journal on Computing* 25(4):693–700.

Gally T, Pfetsch ME (2016) Computing restricted isometry constants via mixed-integer semidefinite programming. *Optimization Online* .

Gamarnik D (2021) The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences* 118(41).

Gamarnik D, Jagannath A, Sen S (2021) The overlap gap property in principal submatrix recovery. *Probability Theory and Related Fields* 181(4):757–814.

Geoffrion AM (1974) Lagrangean relaxation for integer programming. *Approaches to Integer Programming*, 82–114 (Springer).

Goodman B, Flaxman S (2017) European Union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine* 38(3):50–57.

Günlük O, Linderoth J (2010) Perspective reformulations of mixed integer nonlinear programs with indicator variables. *Mathematical Programming* 124(1):183–205.

Gupta SD, Van Parys BP, Ryu EK (2022) Branch-and-bound performance estimation programming: A unified methodology for constructing optimal optimization methods. *arXiv preprint arXiv:2203.07305* .

Hein M, Bühler T (2010) An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA. *Advances in Neural Information Processing Systems* 23.

Horn RA, Johnson CR (1990) *Matrix Analysis* (Cambridge University Press).

Hotelling H (1933) Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology* 24(6):417.

Jolliffe IT, Trendafilov NT, Uddin M (2003) A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics* 12(3):531–547.

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

Kim J, Tawarmalani M, Richard JPP (2021) Convexification of permutation-invariant sets and an application to sparse principal component analysis. *Mathematics of Operations Research* .

Li Y, Xie W (2020) Exact and approximation algorithms for sparse PCA. *arXiv preprint arXiv:2008.12438* .

Li Y, Xie W (2021) Beyond symmetry: Best submatrix selection for the sparse truncated SVD. *arXiv preprint arXiv:2105.03179* .

Lu Z, Zhang Y (2012) An augmented Lagrangian approach for sparse principal component analysis. *Mathematical Programming* 135(1):149–193.

Mackey L (2008) Deflation methods for sparse PCA. *Advances in Neural Information Processing Systems* 21.

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).

Overton ML, Womersley RS (1992) On the sum of the largest eigenvalues of a symmetric matrix. *SIAM Journal on Matrix Analysis and Applications* 13(1):41–45.

Pearson K (1901) Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2(11):559–572.

Probel CJ, Tropp JA (2011) Large-scale PCA with sparsity constraints.

Rudin C, Chen C, Chen Z, Huang H, Semenova L, Zhong C (2022) Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys* 16:1–85.

Tan KM, Petersen A, Witten D (2014) Classification of RNA-seq data. *Statistical Analysis of Next Generation Sequencing Data*, 219–246 (Springer).

Tropp JA, Yurtsever A, Udell M, Cevher V (2017) Practical sketching algorithms for low-rank matrix approximation. *SIAM Journal on Matrix Analysis and Applications* 38(4):1454–1485.

Udell M, Horn C, Zadeh R, Boyd S (2016) Generalized low rank models. *Foundations and Trends® in Machine Learning* 9(1):1–118.

Vu VQ, Cho J, Lei J, Rohe K (2013) Fantope projection and selection: A near-optimal convex relaxation of sparse pca. *Advances in Neural Information Processing Systems* 26.

Vu VQ, Lei J (2013) Minimax sparse principal subspace estimation in high dimensions. *The Annals of Statistics* 41(6):2905–2947.

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

Supplementary Material

EC.1. Proof of Proposition 1

Proof of Proposition 1 We decompose each matrix $\boldsymbol{Y}^t$ into $\boldsymbol{Y}^t = \boldsymbol{u}_t \boldsymbol{u}_t^\top$ with $\|\boldsymbol{u}_t\|^2 = \text{tr}(\boldsymbol{Y}^t) = 1$. Hence, for any pair (t, t'), $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'} \rangle = \boldsymbol{u}_t^\top \boldsymbol{Y}^{t'} \boldsymbol{u}_t = (\boldsymbol{u}_t^\top \boldsymbol{u}_{t'})^2 \geq 0$.

($\Rightarrow$) If $\boldsymbol{Y} := \sum_{t' \in [r]} \boldsymbol{Y}^{t'} \preceq \mathbb{I}$, then, for any $t \in [r]$, $\boldsymbol{u}_t^\top \boldsymbol{Y} \boldsymbol{u}_t \leq \|\boldsymbol{u}_t\|^2 = 1$. However, $\boldsymbol{u}_t^\top \boldsymbol{Y} \boldsymbol{u}_t = 1 + \sum_{t' \neq t} \langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'} \rangle$. Hence, for all $t' \neq t$, we must have $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'} \rangle = 0$.

($\Leftarrow$) If $\langle \boldsymbol{Y}^t, \boldsymbol{Y}^{t'} \rangle = 0$ for all $t' \neq t$, then $\{\boldsymbol{u}_t\}_{t \in [r]}$ is an orthonormal family that can be completed to form an orthonormal basis $\{\boldsymbol{u}_t\}_{t \in [p]}$. For any $t \in [p], t' \in [r]$, $\boldsymbol{u}_t^\top \boldsymbol{Y}^{t'} \boldsymbol{u}_t = 1$ if $t = t'$, 0 otherwise so for any $t \in [p]$, $\boldsymbol{u}_t^\top \boldsymbol{Y} \boldsymbol{u}_t \leq \|\boldsymbol{u}_t\|^2$ and $\boldsymbol{Y} = \sum_{t' \in [r]} \boldsymbol{Y}^{t'} \preceq \mathbb{I}$. □

EC.2. A derivation of some valid inequalities

In this section, we derive the valid inequalities introduced in Theorem 2 from first principles. We proceed in two ways. First, we derive inequalites which hold for each $\boldsymbol{Y}^t$ separately. Second, we observe that these inequalities can be generalized to also apply for $\boldsymbol{Y} = \sum_{t \in [r]} \boldsymbol{Y}^t$, and also derive new inequalities which reflect the interactionof the sparsity and rank constraints.

Rank-One Valid Inequalities First, inspired by Bertsimas and Cory-Wright (2020), we observe that in a feasible solution to Problem (8) the 2×2 minors of $\boldsymbol{Y}^t$ are certainly non-negative, i.e.,

$$Y^t_{i,j} \leq Y^t_{i,i} Y^t_{j,j}, \ \forall i, j \in [p].$$

These constraints are implied by $\boldsymbol{Y}^t \succeq \boldsymbol{0}$ and hence redundant in-and-of-themselves. However, we can sum over all such constraints $i \in [p]$ and use $\text{tr}(\boldsymbol{Y}^t) = 1$, to obtain the constraint

$$\sum_{i \in [p]} \left(Y^t_{i,j}\right)^2 \leq Y^t_{j,j}, \ \forall j \in [p].$$

This constraint is a sum of redundant constraints and hence redundant. However, we can strengthen it, by noting that it is a separable convex quadratic inequality under logical constraints. Indeed, by Lemma 2, its convex closure under the logical constraints $Y^t_{i,j} = 0$ if $Z_{i,t} = 0$ is given by:

$$\sum_{j=1}^{p} \left(Y^t_{i,j}\right)^2 \leq Y^t_{i,i} Z_{i,t}, \ \forall i \in [p], t \in [k]. \tag{EC.1}$$

Rank-r Valid Inequalities In the same spirit as in the rank one case, we can obtain strong valid inequalities by summing the 2×2 minors of $Y_{i,i} = \sum_{t=1}^{r} Y^t_{i,i}$. Indeed, since $\boldsymbol{Y}$ is positive semidefinite, summing its 2×2 minors implies that:

$$\sum_{j=1}^{p} \left(Y_{i,j}\right)^2 \leq r Y_{i,i}.$$

Moreover, since $Y_{i,j} = \sum_{t=1}^r \boldsymbol{Y}_{i,j}^t$ is a rank-one quadratic under logical constraints $Y_{i,j}^t = 0$ if $Z_{i,t} = 0$, invoking Lemma 3 reveals that this quadratic constraint's convex closure under these logical constraints is given by the strengthened inequality:

$$\sum_{j=1}^p (Y_{i,j})^2 \leq r Y_{i,i} \min\left(1, \sum_{t=1}^r Z_{i,t}\right), \ \forall i \in [p]. \tag{EC.2}$$

Second, in any feasible solution we have:

$$|Y_{i,j}| \leq \sum_{t=1}^r |Y_{i,j}^t| = \sum_{t=1}^r |U_{i,t}||U_{j,t}|.$$

Let us denote by k_t the sparsity of the tth column of $\boldsymbol{U}$, $\boldsymbol{U}_t$. Then, it is well known that $\|\boldsymbol{U}_t\|_1 \leq \sqrt{k_t}$. Therefore:

$$\sum_{j=1}^p |Y_{i,j}| \leq \sum_{j=1}^p \left(\sum_{t=1}^r |U_{i,t}||U_{j,t}|\right) \leq \sum_{t=1}^r \sqrt{k_t}|U_{i,t}|.$$

Next, squaring both sides and invoking the Cauchy-Schwarz inequality reveals that

$$\left(\sum_{j=1}^p |Y_{i,j}|\right)^2 \leq \left(\sum_{t=1}^r U_{i,t}^2\right)\left(\sum_{t=1}^r k_t\right) = k Y_{i,i}.$$

Finally, noting that the expression $\left(\sum_{j=1}^p |Y_{i,j}|\right)^2 \leq k Y_{i,i}$ is a convex quadratic under logical constraints $Y_{i,j}^t = 0$ if $Z_{i,t} = 0$ and invoking Lemma 3 to obtain its convex closure yields the strengthened second-order cone inequality

$$\left(\sum_{j=1}^p |Y_{i,j}|\right)^2 \leq k Y_{i,i} \min\left(1, \sum_{t\in[r]} Z_{i,t}\right), \ \forall i \in [p], t \in [k]. \tag{EC.3}$$

Third, in the same spirit, the 2×2 minors of $\boldsymbol{Y} \preceq \text{Diag}\left(\min\left(\boldsymbol{e}, \sum_{t\in[r]} \boldsymbol{Z}_t\right)\right)$ are

$$\left(\min\left(1, \sum_{t\in[r]} Z_{i,t}\right) - Y_{i,i}\right)\left(\min\left(1, \sum_{t\in[r]} Z_{j,t}\right) - Y_{j,j}\right) \geq \left(\min\left(1, \sum_{t\in[r]} Z_{i,t}\right)\delta_{i,j} - Y_{i,j}\right)^2,$$

where $\delta_{i,j} = \mathbf{1}\{i = j\}$ is an indicator denoting whether $i = j$. Summing these constraints over all indices $i \neq j$ and using $k - r + 1$ as an upper bound on $\sum_{i\in[p]:i\neq j} \sum_{t\in[r]} Z_{i,t} - Y_{i,i}$ then yields

$$(k - r + 1)\left(\min\left(1, \sum_{t\in[r]} Z_{j,t}\right) - Y_{j,j}\right) \geq \sum_{i\in[p]:i\neq j} Y_{i,j}^2, \ \forall j \in [p].$$

Finally, we recognize the right hand side as a sum of rank-one quadratic terms $(\sum_{t=1}^r Y_{i,j}^t)^2$ under logical constraints $Y_{i,j}^t = 0$ if $Z_{j,t} = 0$ and invoke Lemma 3 to obtain the convex closure, giving:

$$(k - r + 1)\min\left(1, \sum_{t\in[r]} Z_{j,t}\right)\left(\min\left(1, \sum_{t\in[r]} Z_{j,t}\right) - Y_{j,j}\right) \geq \sum_{i\in[p]:i\neq j} Y_{i,j}^2 \ \forall j \in [p]. \tag{EC.4}$$

EC.3. Proof of Proposition 2

Proof of Proposition 2 First, let us observe that if $\sum_{i\in[p]} Z_{i,t} \leq k$ and $\boldsymbol{Y}^t = \boldsymbol{U}_t\boldsymbol{U}_t^\top$ is a rank-one matrix such that $\|\boldsymbol{U}\|_2 = 1$ then we have $\|\boldsymbol{U}\|_1 \leq \sqrt{k_t}$ by norm equivalence. Therefore

$$\sum_{j=1}^{p} |Y_{i,j}^t| \leq \sum_{j=1}^{p} |U_{i,t}||U_{j,t}| \leq \sqrt{k_t}|U_{i,t}|.$$

Squaring both sides of this inequality then yields

$$\left(\sum_{j=1}^{p} |Y_{i,j}^t|\right)^2 \leq k_t Y_{i,i}^t,$$

and combining Lemma 2 with this inequality yields (10).

Second, in the same spirit, since $\boldsymbol{U}_t\boldsymbol{U}_t^\top$ is only supported on indices where $\boldsymbol{Z}_t$ is non-zero, we have that $\boldsymbol{Y}^t \preceq \text{Diag}(\boldsymbol{Z}_t)$. This constraint implies the following 2×2 minors are non-negative

$$(Z_{i,t} - Y_{i,i}^t)(Z_{j,t} - Y_{j,j}^t) \geq (\delta_{i,j} - Y_{i,j}^t)^2, \ \forall i,j \in [p],$$

where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. Summing these inequalities over indices $i \neq j$ and setting $k_t - 1$ as a valid upper bound on $\sum_{i\in[p]:i\neq j} Z_{i,t} - Y_{i,i}^t$ whenever $Z_{j,t} = 1$ (as $Y_{i,j}^t = 0$ if $Z_{j,t} = 0$) gives

$$(k_t - 1)(Z_{j,t} - Y_{j,j}^t) \geq \sum_{i\in[p]:i\neq j} {Y_{i,j}^t}^2, \ \forall j \in [p].$$

Finally, using Lemma 2 to take the convex closure of this inequality under the logical constraints $Y_{i,j}^t = 0$ if $Z_{j,t} = 0$ gives Equation (11). $\square$

EC.4. A Second-Order Cone Relaxation

We now supply the full second-order cone relaxation we alluded to in Section 2:

$$\max_{\substack{\boldsymbol{Z}\in[0,1]^{p\times r}:\\ \langle \boldsymbol{E},\boldsymbol{Z}\rangle \leq k, \boldsymbol{w}\in[0,1]^p}} \ \max_{\substack{\boldsymbol{Y}\in\mathcal{S}^p, \boldsymbol{Y}^t, \boldsymbol{F}_t, \boldsymbol{G}_t\in\mathcal{S}^p,\\ \boldsymbol{T}_t\in\mathbb{R}_+^{p\times p} \ \forall t\in[k],\\ \boldsymbol{r}^{t,D}\in\mathbb{R}^{p-1}, \boldsymbol{t}^{t,D}\in\mathbb{R}_+^{p\times p-1}}} \langle \boldsymbol{Y}, \boldsymbol{\Sigma}\rangle \tag{EC.5}$$

$$\begin{aligned}
\text{s.t.} \quad & \boldsymbol{Y} = \sum_{t=1}^{k} \boldsymbol{Y}^t, \ \text{tr}(\boldsymbol{Y}^t) = 1, \boldsymbol{w} \leq \boldsymbol{Z}\boldsymbol{e} && \forall t \in [r],\\
& {Y_{i,j}^t}^2 \leq Y_{i,i}^t Y_{j,j}^t && \forall i,j \in [p], \forall t \in [r],\\
& (\delta_{i,j} - Y_{i,j})^2 \leq (w_i - Y_{i,i})(w_j - Y_{j,j}) && \forall i,j \in [p], \forall t \in [r],\\
& \sum_{j=1}^{p} {Y_{i,j}}^2 \leq r Y_{i,i} w_i && \forall i \in [p]\\
& \left(\sum_{j=1}^{p} |Y_{i,j}|\right)^2 \leq k Y_{i,i} w_i, \ \pm \boldsymbol{Y}^t \leq \boldsymbol{F}_t && \forall i \in [p], \ \forall t \in [r],\\
& \sum_{i\in[p]:i\neq j} Y_{i,j}^2 \leq (k - r + 1) w_j (w_j - Y_{j,j}) && \forall j \in [p],\\
& Y_{i,i}^t \leq t_{i,j}^{t,D} + r_j^{t,D} && \forall i \in [p], j \in [p-1], t \in [r],
\end{aligned}$$

$$\begin{aligned}
&\mathrm{tr}(\boldsymbol{Y}^t) = \mathrm{tr}(\boldsymbol{G}^t) = \mathrm{tr}(\boldsymbol{F}_t) = 1 && \forall t \in [r],\\
&\langle \boldsymbol{E}, \boldsymbol{G}_t - \boldsymbol{F}_t \rangle = 0 && \forall t \in [r],\\
&{G^t_{i,j}}^2 \leq G^t_{i,i} G^t_{j,j} && \forall i,j \in [p], \forall t \in [r],\\
&G^t_{i,1} \geq G^t_{i,2} \geq \ldots \geq G^t_{i,k_t} && \forall i \in [k_t],\ \forall t \in [r],\\
&G^t_{i,j} = 0 && \forall i > k_t \text{ or } j > k_t\ \forall t \in [r],\\
&\sum_{i=1}^{j} G^t_{i,i} \geq j r^{D,t}_j + \sum_{j=1}^{n} t^{t,D}_{i,j} && \forall j \in [p-1],\ t \in [r],\\
&{F^t_{i,j}}^2 \leq T^t_{i,j} T^t_{j,i},\ T^t_{i,i} = F^t_{i,i} && \forall i \in [p], j \in [i-1],\ t \in [r],\\
&\sum_{j \in [p]} T^t_{i,j} = Z_{i,t},\ \sum_{i \in [p]} T^t_{i,j} = k_t F^t_{j,j} && \forall i \in [p], \forall j \in [p], t \in [r],\\
&0 \leq T^t_{i,j} \leq F^t_{i,j}, {F^t_{i,j}}^2 \leq F^t_{i,i} F^t_{j,j} && \forall i,j \in [p], \forall t \in [r].
\end{aligned}$$

EC.5. Supplementary Numerical Results

In this section, we provide supplementary results which support the numerical experiments performed in Section 4.

EC.5.1. Performance of Conic Relaxations

EC.5.2. Preliminary Experiments With Pitprops Dataset

Dataset	Dim. (p)	Rank (r)	Sparsity (k, k_t)	PSD		SOC-Cuts		SOC	
				UB	T(s)	UB	T(s)	UB	T(s)
Pitprops	13	2	10, (5, 5)	0.449	0.94	0.450	11.86	0.524	0.21
		2	20, (10, 10)	0.507	0.35	0.515	12.90	0.672	0.27
		3	15, (5, 5, 5)	0.616	0.46	0.619	24.70	0.761	0.30
		3	30, (10, 10, 10)	0.652	0.46	0.678	23.77	1.007	0.33
Wine	13	2	10, (5, 5)	0.458	0.35	0.459	13.89	0.529	0.21
		2	20, (10, 10)	0.554	0.37	0.560	13.57	0.722	0.22
		3	15, (5, 5, 5)	0.632	0.45	0.634	20.21	0.762	0.28
		3	30, (10, 10, 10)	0.665	0.44	0.689	25.16	1.083	0.32
Ionosphere	34	2	10, (5, 5)	0.209	6.52	0.209	97.16	0.221	1.41
		2	20, (10, 10)	0.305	7.95	0.310	103.3	0.363	1.43
		2	40, (20, 20)	0.378	8.84	0.390	121.5	0.504	2.88
		3	15, (5, 5, 5)	0.297	8.62	0.298	215.4	0.331	2.42
		3	30, (10, 10, 10)	0.411	8.31	0.420	164.2	0.545	2.25
		3	60, (20, 20, 20)	0.464	8.72	0.495	158.0	0.757	2.96
Geographical	68	2	10, (5, 5)	0.147	85.35	0.147	53.26	0.147	15.36
		2	20, (10, 10)	0.294	89.31	0.294	57.33	0.294	14.80
		2	40, (20, 20)	0.432	101.1	0.433	652.9	0.567	13.08
		3	15, (5, 5, 5)	0.221	117.1	0.221	78.87	0.221	28.49
		3	30, (10, 10, 10)	0.410	184.2	0.410	1107	0.441	27.77
		3	60, (20, 20, 20)	0.520	142.0	0.529	880.1	0.852	23.72
Communities	101	2	10, (5, 5)	0.095	114300 (*)	0.095	773.9	0.096	34.79
		2	20, (10, 10)	0.169	148100 (*)	0.169	1502	0.175	28.93
		2	40, (20, 20)	0.263	151300 (*)	0.268	1760	0.286	33.61
		3	15, (5, 5, 5)	0.141	140900 (*)	0.141	970.9	0.144	49.61
		3	30, (10, 10, 10)	0.245	255600 (*)	0.246	3114	0.262	36.50
Arrhythmia	274	2	10, (5, 5)	-	-	0.031	27720 (*)	0.031	359.8
		2	20, (10, 10)	-	-	0.052	35810 (*)	0.055	350.6
		2	40, (20, 20)	-	-	0.080	41030 (*)	0.086	671.7
		3	15, (5, 5, 5)	-	-	0.046	68570 (*)	0.047	769.3
		3	30, (10, 10, 10)	-	-	0.076	78620 (*)	0.083	736.1
		3	60, (20, 20, 20)	-	-	0.118	76800 (*)	0.129	1196
Micromass	1300	2	10, (5, 5)	-	-	-	-	0.008	41900 (*)
		2	20, (10, 10)	-	-	-	-	0.014	47000 (*)
		2	40, (20, 20)	-	-	-	-	0.023	50900 (*)
		3	15, (5, 5, 5)	-	-	-	-	0.011	197700 (*)
		3	30, (10, 10, 10)	-	-	-	-	0.021	154400 (*)
		3	60, (20, 20, 20)	-	-	-	-	0.034	157900 (*)

Table EC.1 Performance of bounds across UCI datasets. All bounds are normalized by dividing by $p = \text{tr}(\mathbf{\Sigma})$, i.e., the number of features, to report in terms of the proportion of correlation explained. The notation "$(*)$" denotes that an instance could not be solved with 16 GB RAM and was instead solved using a high-performance computing environment with 256 GB RAM. The notation "$-$" denotes that neither environment could solve the instance.

r	k_t	Alg. 1			Alg. 2			Branch-and-Bound			
		Obj.	Viol.	T(s)	Obj.	Viol.	T(s)	Obj.	Viol.	Nodes	T(s)
2	2	0.204	0	1.00	**0.295**	0	1.40	0.295	0	4800	5.58
	4	0.333	0	0.32	**0.404**	0	3.19	0.404	0	38500	56.38
	6	0.429	0	0.28	**0.461**	0.010	63.54	**0.452**	0	354400	>600
	8	0.381	0	0.45	**0.490**	0.010	74.51	0.474	0	237500	>600
	10	0.380	0	0.39	0.502	0.015	69.54	**0.491**	0	152100	>600
3	2	**0.435**	0	0.50	**0.435**	0	2.51	**0.435**	0	25700	20.18
	4	**0.555**	0	0.55	0.542	0	100.7	0.533	0.001	147900	>600
	6	0.483	0	0.46	**0.617**	0.032	99.35	0.563	0.001	167400	>600
	8	0.488	0	0.52	**0.639**	0.043	118.2	0.593	0.001	125100	>600
	10	0.511	0	0.51	**0.652**	0.035	105.2	0.608	0.001	110400	>600
4	2	**0.554**	0	0.60	**0.554**	0	3.12	**0.554**	0	322400	>600
	4	0.563	0	0.60	0.665	0.061	228.68	**0.607**	0.001	103000	>600
	6	0.544	0	1.24	0.696	0.077	198.31	**0.644**	0.002	120900	>600
	8	0.530	0	0.67	**0.718**	0.069	195.09	0.645	0.002	82900	>600
	10	0.548	0	0.85	0.737	0.072	192.34	**0.681**	0.002	56000	>600
5	2	0.644	0	0.98	0.647	0	12.68	**0.656**	0	222900	>600
	4	0.548	0	0.70	**0.754**	0.019	198.9	0.636	0.002	72000	>600
	6	0.619	0	0.82	**0.782**	0.090	189.49	0.661	0.002	56600	>600
	8	0.619	0	0.85	**0.804**	0.089	183.26	0.672	0.003	62300	>600
	10	0.586	0	0.71	**0.809**	0.094	180.8	0.688	0.003	44100	>600
6	2	**0.749**	0	1.01	0.734	0	23.31	0.740	0.001	65600	>600
	4	0.666	0	0.90	**0.814**	0.085	287.34	0.718	0.002	39200	>600
	6	0.686	0	1.54	0.836	0.153	336.99	**0.723**	0.003	39600	>600
	8	0.673	0	1.89	0.854	0.122	293.38	**0.745**	0.003	45700	>600
	10	0.645	0	1.34	0.874	0.115	206.93	**0.790**	0.004	25200	>600
Avg.		0.535	0	0.79	**0.653**	0.048	134.75	0.600	0.001	108900	>600

Table EC.2 Performance of our approaches on pitprops dataset ($p=13$) using the experimental setup laid out in Section 4.2, except we use the update rule $\lambda_l = l\ \forall l$ for Algorithm 2 and a time limit of 600s for branch-and-bound. k_t denotes the sparsity of each individual component, meaning a set of r PCs have a collective sparsity budget of $k_t r$. We denote the best performing solution (in terms of proportion of variance explained minus the total orthogonality constraint violation) in bold. We use >600 to denote an instance where branch-and-bound terminates at the 600s time limit. Note that all objective values are reported in terms of the proportion of variance explained by dividing by p, the number of features.

EC.5.3. Instance-Wise Results on UCI Datasets

Next, we provide an instance-by-instance account of the results summarized in Table 2, in Table EC.3. Note that we do not report on Algorithm 1, Algorithm 2 (fixed) or branch-and-bound (fixed) here, as these results are already reported in Table EC.4.

Dataset	p	r	k_t	Alg. 2			Branch-and-bound		
				Obj.	Viol.	T(s)	Obj.	Viol.	T(s)
Pitprops	13	2	5	0.439	0	10.43	0.439	0	4864.7
		2	10	0.501	0.004	53.26	0.490	0	>7200
		3	5	0.591	0.006	87.06	0.560	0	>7200
		3	10	0.650	0.009	83.61	0.594	0	>7200
Wine	13	2	5	0.431	0	14.79	0.448	0	6023
		2	10	0.545	0.006	69.85	0.525	0	>7200
		3	5	0.614	0.001	74.08	0.559	0	>7200
		3	10	0.661	0.01	94.52	0.600	0	>7200
Ionosphere	34	2	5	0.205	0	254.7	0.201	0	>7200
		2	10	0.294	0	2904	0.270	0	>7200
		2	20	0.360	0.010	5245	0.311	0	>7200
		3	5	0.286	0	2385	0.287	0	>7200
		3	10	0.400	0	5257	0.312	0	>7200
		3	20	0.458	0.019	7773	0.350	0	>7200
Geography	68	2	5	0.147	0	77.22	0.147	0	>7200
		2	10	0.294	0	32.58	0.292	0	>7200
		2	20	0.298	0.047	2733	0.343	0	>7200
		3	5	0.218	0	50.42	0.218	0	>7200
		3	10	0.149	0.044	1288	0.341	0	>7200
		3	20	0.273	0.026	3868	0.386	0	>7200

Table EC.3 Supplementary: Performance of methods on UCI datasets, as averaged in Table 2.

EC.5.4. Instance-Wise Results on Larger UCI Datasets

Next, we provide an instance-by-instance account of the results summarized in Table 3, in Tables EC.4-EC.5.

Dataset	p	r	k_t	Alg. 1				Alg. 2 (fixed)			Branch-and-bound (fixed)		
				UB	Obj.	Viol.	T(s)	Obj.	Viol.	T(s)	Obj.	Viol.	T(s)
Pitprops	13	2	5	0.449	0.429	0	0.99	**0.439**	0	8.06	**0.439**	0	4702
		2	10	0.507	0.380	0	0.34	**0.501**	0.005	60.82	0.490	0	>7200
		3	5	0.616	0.541	0	0.44	**0.591**	0.008	85.11	0.539	0	>7200
		3	10	0.652	0.511	0	0.46	**0.651**	0.012	99.81	0.594	0	>7200
Wine	13	2	5	0.458	0.401	0	0.35	0.431	0	17.57	0.440	0	>7200
		2	10	0.554	0.508	0	0.31	**0.546**	0.007	69.58	0.525	0	>7200
		3	5	0.632	0.446	0	0.48	**0.614**	0.001	108.2	0.559	0	>7200
		3	10	0.665	0.528	0	0.46	**0.662**	0.014	115.7	0.600	0	>7200
Ionosphere	34	2	5	0.209	0.203	0	5.97	**0.205**	0	3.46	0.204	0	>7200
		2	10	0.305	0.265	0	8.72	**0.294**	0	2280	0.279	0	>7200
		2	20	0.378	0.286	0	23.31	**0.360**	0.014	5422	0.304	0	>7200
		3	5	0.297	0.287	0	31.04	0.286	0	1074	**0.287**	0	>7200
		3	10	0.411	0.305	0	39.00	**0.400**	0	6206	0.305	0	>7200
		3	20	0.464	0.390	0	10.26	**0.459**	0.025	6300	0.350	0	>7200
Geography	68	2	5	0.147	0.145	0	99.45	**0.147**	0	112.7	**0.147**	0	>7200
		2	10	0.294	0.290	0	107.3	**0.294**	0	58.61	**0.294**	0	>7200
		2	20	0.433	0.393	0	1213	0.296	0.062	1398	0.391	0	>7200
		3	5	0.221	0.213	0	119.3	0.218	0	73.91	**0.221**	0	>7200
		3	10	0.410	0.342	0	1453	0.359	0.09	1285	0.341	0	>7200
		3	20	0.530	**0.457**	0	1572	0.447	0.094	5535	0.422	0	>7200
Communities	101	2	5	0.095	**0.095**	0	484.0	**0.095**	0	709.5	**0.095**	0	2534
		2	10	0.169	**0.169**	0	1327	**0.169**	0	1600	0.168	0	>7200
		2	20	0.268	0.219	0	2438	0.157	0.032	2513	0.193	0	>7200
		3	5	**0.141**	**0.141**	0	979.7	**0.141**	0	881.3	**0.141**	0	>7200
		3	10	0.246	0.242	0	3553	**0.245**	0	4252	0.243	0	>7200
		3	20	0.385	0.267	0	3231	0.165	0.039	4779	0.257	0	>7200
Arrhythmia	274	2	5	0.031	0.021	0	583.7	0.023	0.006	386.2	0.023	0	>7200
		2	10	0.055	0.035	0	555.4	0.038	0.009	469.6	0.038	0	>7200
		2	20	0.086	0.067	0	622.9	0.048	0.005	2592	0.065	0	>7200
		3	5	0.047	0.031	0	1423	0.030	0.015	839.6	0.029	0	>7200
		3	10	0.083	0.044	0	1086	0.041	0.02	1062	0.039	0	>7200
		3	20	0.129	0.083	0	1060	0.059	0.009	4664	0.071	0	>7200
Micromass	1300	2	5	0.008	0.005	0	46170	0.004	0	0.39	0.004	0	>7200
		2	10	0.014	0.008	0	44520	0.007	0	12.15	0.007	0	>7200
		2	20	0.023	0.018	0	49780	**0.023**	0	29.30	0.019	0	>7200
		3	5	0.011	0.008	0	173800	0.008	0	6.95	0.008	0	>7200
		3	10	0.021	0.009	0	135300	0.007	0	1.05	0.007	0	>7200
		3	20	0.034	0.029	0	164400	0.014	0.022	4897	0.025	0	>7200
Average				0.276	0.232	0	16740	0.249	0.013	1580	0.241	0	>7200

Table EC.4 Performance of methods on UCI datasets. k_t denotes the sparsity of each individual component, meaning a set of r PCs have a collective sparsity budget of $k_t r$. Note that all objective values are reported in terms of the proportion of correlation explained by dividing by p, the number of features.

Dataset	p	r	k_t	Berk and Bertsimas (2019)			Hein and Bühler (2010)			Zou et al. (2006)		
				Obj.	Viol.	T(s)	Obj.	Viol.	T(s)	Obj.	Viol.	T(s)
Pitprops	13	2	5	0.421	0.168	1.67	0.418	0	0.11	0.177	1.341	0.12
		2	10	0.502	0.008	0.14	0.502	0.008	0.01	0.139	1.827	0.22
		3	5	0.592	0.675	0.08	0.575	0.166	0.02	0.169	3.462	0.04
		3	10	0.648	0.073	0.07	0.647	0.084	0	0.181	3.771	0.36
Wine	13	2	5	**0.448**	0	0.04	0.422	0.004	0.01	0.127	0.315	0.04
		2	10	0.545	0.020	0.04	0.545	0.02	0	0.068	0.731	0.06
		3	5	0.610	0.019	0.06	0.559	0.092	0	0.225	2.830	0.05
		3	10	0.654	0.059	0.06	0.655	0.093	0	0.232	2.771	0.32
Ionosphere	34	2	5	**0.205**	0	0.08	0.153	0	0.08	0.078	0	0.02
		2	10	0.289	0	0.30	0.288	0	0.01	0.106	0	0.04
		2	20	0.369	0.058	4.45	0.370	0.01	0.17	0.147	0.305	0.12
		3	5	**0.291**	0	0.14	0.227	0	0.02	0.097	1.666	0.07
		3	10	0.392	0.109	0.38	0.365	0.255	0.01	0.100	1.909	0.12
		3	20	0.449	0.183	0.27	0.451	0.037	0.03	0.111	2.111	4.51
Geography	68	2	5	**0.147**	0	0.09	0.097	0	0.01	0.034	1.793	0.41
		2	10	**0.294**	0	0.08	0.164	0	0	0.068	1.939	0.66
		2	20	**0.395**	0	5.95	0.316	0.135	0.04	0.062	1.754	0.53
		3	5	**0.221**	0	0.13	0.122	0	0.01	0.061	2.720	0.57
		3	10	**0.389**	0	0.18	0.192	0	0.01	0.054	4.021	0.90
		3	20	0.484	0.273	23.66	0.387	0.261	0.06	0.090	5.009	1.40
Communities	101	2	5	**0.095**	0	0.73	0.093	0	0	0.032	0.576	0.05
		2	10	**0.169**	0	1.68	0.154	0	0	0.029	0.605	0.18
		2	20	**0.258**	0	120	**0.258**	0	0.07	0.027	0.090	1.49
		3	5	**0.141**	0	1.18	0.129	0	0.01	0.050	1.854	0.29
		3	10	**0.245**	0	2.85	0.181	0	0.02	0.044	1.504	1.76
		3	20	**0.361**	0.058	180.1	0.350	0.064	0.02	0.043	1.869	5.27
Arrhythmia	274	2	5	**0.031**	0	2.81	0.012	0	0.02	0.007	1.799	0.71
		2	10	**0.052**	0	61.25	0.011	0	0.03	0.007	1.143	1.08
		2	20	**0.077**	0	120.0	0.043	0.005	0.06	0.006	1.140	4.62
		3	5	**0.046**	0	5.35	0.016	0	0.02	0.012	1.076	0.53
		3	10	**0.074**	0	121.6	0.018	0	0.05	0.012	0.876	3.82
		3	20	**0.109**	0	180.0	0.074	0.005	0.07	0.012	0.694	10.65
Micromass	1300	2	5	**0.008**	0	45.4	0.004	0	0.77	0.002	0.014	18.05
		2	10	**0.014**	0	120.2	0.007	0	1.09	0.002	0.323	41.97
		2	20	**0.023**	0	120.2	0.012	0	1.34	0.002	0.361	2.64
		3	5	**0.011**	0	71.97	0.005	0	1.54	0.002	3.004	24.30
		3	10	**0.020**	0	180.3	0.008	0	1.80	0.002	2.301	31.89
		3	20	**0.034**	0	180.3	0.013	0	2.64	0.002	1.252	2.02
Average				0.266	0.045	40.89	0.233	0.033	0.26	0.069	1.599	4.26

Table EC.5 Performance of methods on UCI datasets (cont).

EC.5.5. Instance-Wise Plots of Symmetry vs. Proportion of Correlation Explained

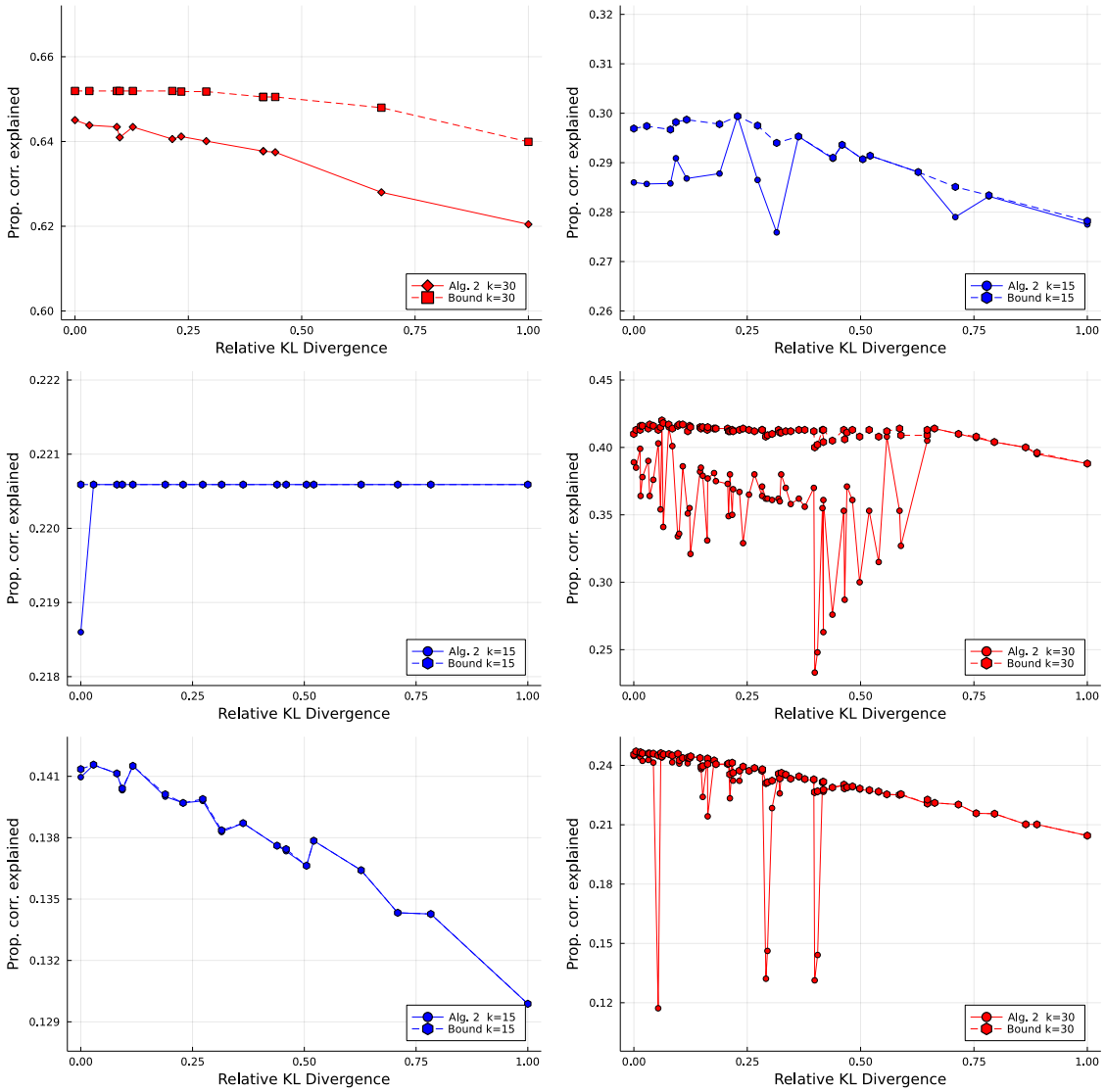

Figure EC.1 Symmetry of sparsity budget allocation vs. proportion of correlation in the dataset explained for pitprops $k = 30$ (top left), ionosphere $k = 15$ (top right), geographical $k = 15$ (middle left), geographical $k = 30$ (middle right), communities $k = 15$ (bottom left), and communities $k = 30$ (bottom right). Note that we normalize the KL divergence for $k = 15$ and $k = 30$ separately.

EC.5.6. A Robustness Check Via Algorithm 1

Dataset	p	r	k	UB	k_t	Obj.	Viol.	Rel. gap (%)
Pitprops	13	3	15	0.652	(7, 5, 1)	0.498	0	23.60%
			30	0.652	(11, 1, 1)	0.432	0	33.70%
Ionosphere	34	3	15	0.322	(9, 5, 1)	0.261	0	18.79%
			30	0.457	(25, 4, 1)	0.304	0	33.35%
Geographical	68	3	15	0.223	(8, 4, 3)	0.214	0	4.20%
			30	0.424	(22, 7, 1)	0.373	0	12.04%
Communities	101	3	15	0.147	(9, 4, 2)	0.080	0	45.50%
			30	0.268	(13, 11, 6)	0.164	0	38.74%

Table EC.6 Solution reported by Algorithm 1 for a given sparsity budget k_{total} and number of PCs ($r = 3$), with k_t not specified. We report the k_t identified by Algorithm 1's rounding mechanism. The sparsity of the solutions returned by Algorithm 1 on the pitprops dataset can be explained by the fact that $\sum_t k_t \leq p$ in Algorithm 1's solutions, since it returns disjoint solutions.