Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

# Sparse Principal Component Analysis with Constraints

**Mihajlo Grbovic**
Department of Computer and Information Sciences, Temple University
mihajlo.grbovic@temple.edu

**Christopher R. Dance**
Xerox Research Centre Europe
chris.dance@xrce.xerox.com

**Slobodan Vucetic**
Department of Computer and Information Sciences, Temple University
slobodan.vucetic@temple.edu

## Abstract

The sparse principal component analysis is a variant of the classical principal component analysis, which finds linear combinations of a small number of features that maximize variance across data. In this paper we propose a methodology for adding two general types of feature grouping constraints into the original sparse PCA optimization procedure. We derive convex relaxations of the considered constraints, ensuring the convexity of the resulting optimization problem. Empirical evaluation on three real-world problems, one in process monitoring sensor networks and two in social networks, serves to illustrate the usefulness of the proposed methodology.

## Introduction

Sparse Principal Component Analysis (PCA) is an extension to the well-established PCA dimensionality reduction tool, which aims at achieving a reasonable trade-off between the conflicting goals of explaining as much variance as possible using near orthogonal vectors that are constructed from as few features as possible. There are several justifications for using Sparse PCA. First, regular principal components are, in general, combinations of all features and are unlikely to be sparse, thus being difficult to interpret. Sparse PCA greatly improves the relevance and interpretability of the components, and is more likely to reveal the underlying structure of the data. In many real-life applications, the features have a concrete physical meaning (e.g. genes, sensors, people) and interpretability, i.e. feature grouping based on correlation, is an important factor worth sacrificing some of the explained variance. Second, under certain conditions (Zhang and El Ghaoui 2011), sparse components can be computed faster. Third, Sparse PCA provides better statistical regularization.

Sparse PCA has been the focus of considerable research in the past decade. The first attempts at improving the interpretation of baseline PCA components were based on post-processing methods such as thresholding (Cadima and Jolliffe 1995) and factor rotation (Jolliffe 1995). Greedy heuristic Sparse PCA formulations have been investigated in (Moghaddam et al. 2006) and (d'Aspremont et al. 2008). More recent methods cast the problem into the optimization framework. Maximizing the explained variance along a normalized vector penalized for the number of non-zero elements of that vector, aims at simultaneous delivery of the aforementioned goals. Majority of these algorithms are based on non-convex formulations, including SPCA (Zou et al. 2006), SCoTLASS (Jolliffe et al. 2003), the Regularized SVD method (Shen and Huang 2008), and the Generalized Power method (Journée et al. 2010). Unlike these approaches, the $l_1$-norm based semidefinite relaxation DSPCA algorithm (d'Aspremont et al. 2007) guarantees global convergence and has been shown to provide better results than other algorithms, i.e. it produces sparser vectors while explaining the same amount of variance.

Interpreting Sparse PCA as feature grouping, where each component represents a group and group members correspond to non-zero component elements, we propose an extension of the DSPCA algorithm in which the user is allowed to add several types of constraints regarding the groups' structure. The idea is to limit the set of feasible solutions by imposing additional goals regarding the components' structure, which are to be reached simultaneously through optimization. An alternative way of handling these constraints is to post-process the solution by removing component elements to meet the constraints. However, this can lead to significant variance reduction. Unlike this baseline approach, the proposed solution is optimal, as it directly maximizes the variance subject to the constraints.

The first type of constraints we consider are the *distance constraints*. Let us consider an on-street parking problem, where features are on-street parking blocks and examples are hourly occupancies. For purposes of price management, the goal may be to group correlated parking blocks, such that the sums of geographic distances between blocks in a group are less than a specific value. Therefore, non-zero elements of each sparse component must satisfy this requirement.

The second type of constraints we consider are the *reliability constraints* that aim to maximize the overall *reliability* of the resulting groups. Assuming that each feature (e.g. sensor) has a certain reliability defined by its failure probability and that the entire component becomes temporarily suspended if any of its features fails, the use of features with low reliability can be costly. These constraints are especially important in industrial systems where we wish to group correlated sensors such that the groups are robust, in terms of maintenance. Another example can be found in social net-

works, when we wish to group correlated users making sure not to include users that do not spend enough time on-line.

The proposed Constrained Sparse PCA methodology was evaluated on several real-world problems. We illustrate its application to constrained decentralized fault detection in industrial processes using the Tennessee Eastman Process data. In addition, we demonstrate its application to constrained user grouping on Digg and Facebook social network data. In the experimental portion of the paper, we also illustrate how *link* and *do-not-link* constraints, which explicitly specify that some features should or should not be grouped together, could be handled within our framework.

## Preliminaries

We have multivariate observations in form of a data set $\mathbf{X}$ of size $m \times n$ with $n$ features and $m$ observations. Let us assume that the features are centered and denote the covariance matrix as $\mathbf{S} = \mathbf{X}^T\mathbf{X}$. Assuming that $\mathbf{S}_1 = \mathbf{S}$, the 1-st principal component $\mathbf{u}_1$ can be found as

$$\begin{aligned} &\text{maximize} && \mathbf{u}^T\mathbf{S}_1\mathbf{u} \\ &\text{subject to} && \mathbf{u}^T\mathbf{u} \leq 1. \end{aligned} \tag{1}$$

The second component can be found by updating the covariance matrix as $\mathbf{S}_2 = \mathbf{S}_1 - (\mathbf{u}_1^T\mathbf{S}_1\mathbf{u}_1)\mathbf{u}_1\mathbf{u}_1^T$ and applying (1). The iterative procedure stops after $K = \mathrm{rank}(\mathbf{S})$ iterations, since $\mathbf{S}_{\mathrm{rank}(\mathbf{S})+1}$ is empty.

### A Direct Formulation of Sparse PCA (DSPCA)

The objective of Sparse PCA is to decompose the covariance matrix $\mathbf{S}$ into near orthogonal principal components $[\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_K]$, while constraining the number of non-zero elements (cardinality) of each $\mathbf{u}_k$ to $r \in \mathbb{N}$, where $r$ is a user defined parameter that controls the sparsity.

The problem of maximizing the variance of component $\mathbf{u}$ with a constraint on its cardinality is defined as

$$\begin{aligned} &\text{maximize} && \mathbf{u}^T\mathbf{S}\mathbf{u} \\ &\text{subject to} && ||\mathbf{u}||_2 = 1, \mathrm{card}(\mathbf{u}) \leq r, \end{aligned} \tag{2}$$

where $\mathrm{card}(\mathbf{u})$ is the number of non-zero elements of $\mathbf{u}$.

Due to the cardinality constraint, this problem is NP-hard. (d'Aspremont et al. 2007) proposed an approximate solution to this problem based on a convex relaxation of (2) in which the vector $\mathbf{u}$ is replaced by a matrix $\mathbf{U} = \mathbf{u}\mathbf{u}^T$. They first form the following problem which is equivalent to (2),

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{S}\mathbf{U}) \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{Rank}(\mathbf{U}) = 1 \\ & && \mathrm{card}(\mathbf{U}) \leq r^2, \mathbf{U} \succeq 0, \end{aligned} \tag{3}$$

where $\mathrm{card}(\mathbf{U})$ is the number of non-zero elements of $\mathbf{U}$ and $\mathbf{U} \succeq 0$ means that $\mathbf{U}$ is positive semidefinite. Problems (2) and (3) are equivalent. If $\mathbf{U}$ is a solution to the above problem, then $\mathbf{U} \succeq 0$ and $\mathbf{Rank}(\mathbf{U}) = 1$ mean that we have $\mathbf{U} = \mathbf{u}\mathbf{u}^T$, while $\mathbf{Tr}(\mathbf{U}) = 1$ means that $||\mathbf{u}||_2 = 1$. Finally, if $\mathbf{U} = \mathbf{u}\mathbf{u}^T$, then $\mathrm{card}(\mathbf{U}) \leq r^2 \equiv \mathrm{card}(\mathbf{u}) \leq r$.

Since (3) is nonconvex, both rank and cardinality constraints are relaxed to obtain an approximate convex problem. This is achieved by dropping the rank constraint and replacing the nonconvex constraint $\mathrm{card}(\mathbf{U}) \leq r^2$ by a weaker but convex constraint $\mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r$, where $\mathbf{1}$ is a vector of ones and matrix $|\mathbf{U}|$ contains absolute values of $\mathbf{U}$ elements,

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{S}\mathbf{U}) \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r, \mathbf{U} \succeq 0. \end{aligned} \tag{4}$$

Let us denote the solution to (4) as $\mathbf{U}^1$. Since $\mathbf{U}$ might not be rank 1, the first sparse component $\mathbf{u}_1$ is obtained as the most dominant eigenvector of $\mathbf{U}^1$. Similarly to PCA, this process iterates by updating $\mathbf{S}$, $\mathbf{S}_2 = \mathbf{S}_1 - (\mathbf{u}_1^T\mathbf{S}_1\mathbf{u}_1)\mathbf{u}_1\mathbf{u}_1^T$ to obtain the second component.

An alternative interpretation penalizes the cardinality,

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{S}\mathbf{U}) - \rho\mathbf{1}^T|\mathbf{U}|\mathbf{1} \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{U} \succeq 0, \end{aligned} \tag{5}$$

where parameter $\rho$ controls the magnitude of the penalty. In practice, a range of $\rho$ values needs to be explored to obtain a solution with the desired cardinality. Formulation (5) has an advantage of yielding a dual form. We can rewrite (5) as

$$\max_{\mathbf{Tr}(\mathbf{U})=1, \mathbf{U}\succeq 0} \; \min_{|\mathbf{V}_{ij}| \leq \rho} \mathbf{Tr}(\mathbf{U}(\mathbf{S} + \mathbf{V}))$$

in variables $\mathbf{U}$ and $\mathbf{V}$, which is upper bounded by its dual

$$\begin{aligned} &\text{minimize} && \lambda^{\max}(\mathbf{S} + \mathbf{V}) \\ &\text{subject to} && |\mathbf{V}_{ij}| \leq \rho, i, j = 1, ..., n, \end{aligned} \tag{6}$$

where $\lambda^{\max}(\mathbf{S} + \mathbf{V})$ is the maximum eigenvalue of $\mathbf{S} + \mathbf{V}$.

A recent result (Zhang and El Ghaoui 2011), shows that feature $i$ is guaranteed to be absent from $\mathbf{U}$ if its variance is less than $\rho$. Therefore, features with variances lower than $\rho$ can be safely eliminated before applying DSPCA, which can dramatically reduce the computational cost.

## Constrained Sparse PCA

Sparse PCA can be used to produce components of desired cardinality. This section shows how additional constraints can be incorporated into the optimization procedure. Without loss of generality, we only consider the 1-st component.

### Distance constraint

The *distance constraint* is motivated by the fact that there is a certain cost associated with grouping features together, determined by prior knowledge about the domain (e.g. some features might be highly correlated but expensive to group).

We model the cost as a distance metric, through a zero-diagonal symmetric matrix $\mathbf{D}$. Therefore, we define the total cost associated with the principal component as

$$C = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} I(\mathbf{U}_{ij} \neq 0)\mathbf{D}_{ij}, \tag{7}$$

where $I$ is an indicator function, $\mathbf{D}_{ij}$ is the distance between features $i$ and $j$ and $\mathbf{U}$ is the solution to problem (4) .

We constrain the set of possible solutions for $\mathbf{U}$ by introducing the *distance constraint* $C \leq d$, where $d$ is user-specified threshold.

For example, in sensor networks, the within-group communication cost is proportional to the length of the communication channels between the sensors that form the group.

In that case, $d$ is a limit on the amount of allowed communication within a group.

By introducing the distance constraint, the optimization problem becomes,

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{SU}) \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r \\ & && C \leq d,\ \mathbf{U} \succeq 0, \end{aligned} \tag{8}$$

Since $C$ is non-convex, we consider replacing it with the convex function $C^*$, defined as

$$C^* = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}|\mathbf{U}_{ij}|\mathbf{D}_{ij}.$$

Proposition 1 establishes a connection between $C$ and $C^*$. We begin with Lemma 1.

**Lemma 1.** *For matrix $\mathbf{U}$ of size $n \times n, n > 1, \mathbf{Tr}(\mathbf{U}) = 1$ and $\mathbf{U} \succeq 0$, it holds that $|\mathbf{U}_{ij}| \leq 0.5$ for $r(r-1)$ non-zero elements outside the diagonal, $i \neq j$.*

*Proof.* $|\mathbf{U}_{ij}| \leq \sqrt{\mathbf{U}_{ii}\mathbf{U}_{jj}} \leq 0.5(\mathbf{U}_{ii} + \mathbf{U}_{jj}) \leq 0.5$, where the first two inequalities follow from $\mathbf{U} \succeq 0$ (Horn and Johnson 1985) and the third inequality follows from $\mathbf{Tr}(\mathbf{U}) = 1$. □

**Proposition 1.** *Suppose $n > 1$ and $1 \leq i, j \leq n$,*

- $(a.1)$ *$\mathbf{D}_{ij}$ is a distance metric ($\mathbf{D}_{ij} \geq 0, \mathbf{D}$ is symmetric zero-diagonal, and satisfies triangle inequality)*
- $(a.2)$ *$\mathbf{U}$ satisfies Lemma 1*

*Then $C^* \leq 0.5 \cdot C$. Furthermore, the inequality is tight in the sense that there exist $\mathbf{D}$ and $\mathbf{U}$ satisfying $(a.1)$ and $(a.2)$, such that $C^* = 0.5 \cdot C$.*

*Proof.*

$$\begin{aligned} \sum_{i,j}|\mathbf{U}_{ij}|\mathbf{D}_{ij} &\leq \max_{i,j}\{|\mathbf{U}_{ij}|\}\sum_{i,j}I(\mathbf{U}_{ij} \neq 0)\mathbf{D}_{ij} \ \text{ by Hölder} \\ &\leq 0.5 \cdot C \ \text{ by Lemma 1.} \end{aligned}$$

To show that the bound is tight, let $|\mathbf{u}(i)| = 2^{-1/2}$ for $i \in \{1, 2\}$ and $\mathbf{u}(i) = 0$ otherwise. Then, $C^* = \sum_{i,j}|\mathbf{U}_{ij}|\mathbf{D}_{ij} = \frac{1}{2}\sum_{i,j}I(\mathbf{U}_{ij} \neq 0)\mathbf{D}_{ij} = \frac{1}{2}C$. □

The proven inequality is guaranteed in the worst case. In practice, we could assume that $\mathbf{U}$ components have the maximum value of $1/r$. From there it follows that constraints such as $C^* \leq C/r$ could be considered as well.

Motivated by Proposition 1, we propose to use convex relaxation of (8) obtained by replacing $C \leq d$ with $2C^* \leq d$. The resulting constrained Sparse PCA problem is

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{SU}) \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r \\ & && \mathbf{1}^T|\mathbf{U} \circ \mathbf{D}|\mathbf{1} \leq d,\ \mathbf{U} \succeq 0, \end{aligned} \tag{9}$$

where $\circ$ is the Hadamard product. Similarly to (5), to tackle larger problems, we can replace (9) with its penalized form,

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{SU}) - \rho\mathbf{1}^T|\mathbf{U}|\mathbf{1} - \rho_d\mathbf{1}^T|\mathbf{U} \circ \mathbf{D}|\mathbf{1} \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{U} \succeq 0, \end{aligned} \tag{10}$$

where parameter $\rho_d$ controls the magnitude of the penalty on distance. Rewriting this problem as

$$\max_{\mathbf{Tr}(\mathbf{U})=1, \mathbf{U}\succeq 0}\ \min_{|\mathbf{V}_{ij}| \leq \rho + \rho_d\mathbf{D}_{ij}} \mathbf{Tr}(\mathbf{U}(\mathbf{S} + \mathbf{V})), \tag{11}$$

leads to the following dual,

$$\begin{aligned} &\text{minimize} && \lambda^{\max}(\mathbf{S} + \mathbf{V}) \\ &\text{subject to} && |\mathbf{V}_{ij}| \leq \rho + \rho_d\mathbf{D}_{ij},\ i, j = 1, ..., n, \end{aligned} \tag{12}$$

where the Karush-Kuhn-Tucker (KKT) conditions are

$$\begin{cases} (\mathbf{S} + \mathbf{V})\mathbf{U} = \lambda^{\max}(\mathbf{S} + \mathbf{V})\mathbf{U} \\ \mathbf{V} \circ \mathbf{U} = (\rho + \rho_d\mathbf{D}_{ij})|\mathbf{U}| \\ \mathbf{Tr}(\mathbf{U}) = 1,\ \mathbf{U} \succeq 0 \\ |\mathbf{V}_{ij}| \leq \rho + \rho_d\mathbf{D}_{ij},\ i, j = 1, ..., n. \end{cases} \tag{13}$$

## Reliability constraint

The *reliability constraint* is inspired by the maintenance scheduling problem (Ben-Daya et al. 2000) in industrial sensor networks. Assuming knowledge of the sensors' reliability and considering that an entire sensor group must go offline during the maintenance of any one sensor, the goal is to make groups of correlated sensors such that their group reliability is above a certain threshold.

Let us denote by $\mathbf{l}$ a feature reliability vector, where $\mathbf{l}_i \in [0, 1)$ is a probability that sensor $i$ will need maintenance during a certain time period, e.g. one month. Then, the reliability of a group of features defined by sparse component $\mathbf{u}$ is defined as

$$R = \prod_{i=1}^{n}(1 - I(\mathbf{u}_i \neq 0)\mathbf{l}_i), \tag{14}$$

In the context of constrained Sparse PCA, it is convenient to rewrite the reliability of component $\mathbf{u}$ as

$$R = \prod_{i=1}^{n}\prod_{j=1}^{n}\left(1 - I(\mathbf{U}_{ij} \neq 0)\mathbf{L}_{ij}\right)^{\frac{1}{r-1}}, \tag{15}$$

where matrix $\mathbf{L}$ is obtained by making all of its columns equal to $\mathbf{l}$, and then setting all of its diagonal elements to 0.

Let us consider constraining the set of possible solutions for $\mathbf{U}$ by requiring $R \geq l$, where $0 < l \leq 1$ is some user specified threshold, representing the lowest allowed component reliability. After taking the logarithm, we can rewrite the constraint as

$$\frac{1}{r-1} \cdot \sum_{i=1}^{n}\sum_{j=1}^{n}\log(1 - I(\mathbf{U}_{ij} \neq 0)\mathbf{L}_{ij}) \geq \log l. \tag{16}$$

By introducing the reliability constraint, the optimization problem becomes,

$$\begin{aligned} &\text{maximize} && \mathbf{Tr}(\mathbf{SU}) \\ &\text{subject to} && \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r, \mathbf{U} \succeq 0 \\ & && \log R \geq \log l. \end{aligned} \tag{17}$$

Since $\log R$ is neither concave nor convex, we consider replacing $\log R$ with convex function $\log R^*$, defined as

$$\log R^* = \frac{1}{r-1} \cdot \sum_{i=1}^{n}\sum_{j=1}^{n}|\mathbf{U}_{ij}|\log(1 - \mathbf{L}_{ij}). \tag{18}$$

Using the Proposition 1 arguments, it can be determined that $\log R$ and $\log R^*$ are related as $\log R^* \geq \frac{1}{2}\log R$. Therefore, a convex relaxation of (17) can be obtained by replacing $\log R \geq \log l$ with $\log R^* \geq \frac{1}{2}\log l$.

Alternatively, we consider another relaxation by replacing $\log R$ with function $\log R^+$, defined as

$$\log R^+ = \frac{1}{r-1} \cdot \sum_{i=1}^{n}\sum_{j=1}^{n} \log(1 - |\mathbf{U}_{ij}|\mathbf{L}_{ij}). \quad (19)$$

Proposition 2 establishes a relationship between $R$ and $R^+$.

**Proposition 2.** *Suppose $n > 1, 1 \leq i, j \leq n$, $\mathbf{L}$ is a zero-diagonal reliability matrix and that $\mathbf{U}$ satisfies Lemma 1. Then,*

$$\log R^+ \geq r(r-1)\log((1 + R^{\frac{1}{r(r-1)}})/2). \quad (20)$$

*Furthermore, this inequality is tight as there exist $\mathbf{L}, \mathbf{U}$ for which the equality holds and the conditions are satisfied.*

*Proof.* First, using Lemma 1 it follows that

$$\prod_{i,j}(1 - |\mathbf{U}_{ij}|\mathbf{L}_{ij}) \geq \prod_{i,j:\mathbf{U}_{ij}\neq 0}(1 - \frac{1}{2}\mathbf{L}_{ij}). \quad (21)$$

Next, we show that the following inequality holds

$$\prod_{i,j:\mathbf{U}_{ij}\neq 0}(1-\frac{1}{2}\mathbf{L}_{ij})^{\frac{1}{r(r-1)}} \geq \frac{1 + \prod_{i,j:\mathbf{U}_{ij}\neq 0}(1 - \mathbf{L}_{ij})^{\frac{1}{r(r-1)}}}{2}, \quad (22)$$

where $r(r-1)$ is the number of product elements. By denoting $\mathbf{x}_{ij} = 1 - \mathbf{L}_{ij}$, we can restate (22) as

$$\prod_{i,j:\mathbf{U}_{ij}\neq 0}(1 + \mathbf{x}_{ij})^{\frac{1}{r(r-1)}} \geq 1 + \prod_{i,j:\mathbf{U}_{ij}\neq 0}\mathbf{x}_{ij}^{\frac{1}{r(r-1)}}. \quad (23)$$

To prove (23), let us take the logarithm of the left hand side,

$$g := \frac{\sum_{i,j:\mathbf{U}_{ij}\neq 0}\log(1 + \mathbf{x}_{ij})}{r(r-1)} = \frac{\sum_{i,j:\mathbf{U}_{ij}\neq 0}\log(1 + e^{\mathbf{y}_{ij}})}{r(r-1)},$$

where $\mathbf{x}_{ij} =: \exp \mathbf{y}_{ij}$. By convexity it follows,

$$g \geq \log\big(1 + e^{\frac{\sum_{i,j:\mathbf{U}_{ij}\neq 0}\mathbf{y}_{ij}}{r(r-1)}}\big) = \log\big(1 + \prod_{i,j:\mathbf{U}_{ij}\neq 0}\mathbf{x}_{ij}^{\frac{1}{r(r-1)}}\big),$$

which proves (23), since $g$ is the arithmetic mean of a convex function $\log(1 + \exp \mathbf{y})$ and $\log$ is an increasing function. This concludes the proof as (20) follows from (22) and (21).

To show that the bound is tight, let $|\mathbf{u}(i)| = 2^{-1/2}$ for $i \in \{1, 2\}$ and $\mathbf{u}(i) = 0$ otherwise. Further, let $\mathbf{L}_{ij} = \mathbf{L}_{ji} = L^*$. Then, $r - 1 = 1$, $R = (1 - L^*)^{r(r-1)}$ and $\log R^+ = \frac{1}{r-1} \cdot \sum_{i,j:\mathbf{U}_{ij}\neq 0}\log(\frac{1+(1-L^*)}{2}) = r(r-1)\log(\frac{1+R^{\frac{1}{r(r-1)}}}{2})$. □

Motivated by Proposition 2, we propose to use convex relaxation of (17) obtained by replacing $\log R \geq \log l$ with $\log R^+ \geq r(r-1)\log\frac{1+l^{\frac{1}{r(r-1)}}}{2}$.

$$\begin{aligned} \text{maximize} \quad & \mathbf{Tr}(\mathbf{SU}) \\ \text{subject to} \quad & \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{1}^T|\mathbf{U}|\mathbf{1} \leq r, \mathbf{U} \succeq 0, \\ & \mathbf{1}^T \log(1 - |\mathbf{U}| \circ \mathbf{L})\mathbf{1} \geq \log(\frac{1 + l^{\frac{1}{r(r-1)}}}{2})^{r(r-1)^2}. \end{aligned} \quad (24)$$

Alternatively, we can use the penalized optimization form,

$$\begin{aligned} \text{maximize} \quad & \mathbf{Tr}(\mathbf{SU}) - \rho\mathbf{1}^T|\mathbf{U}|\mathbf{1} + \rho_l\mathbf{1}^T\big(|\mathbf{U}| \circ \log(1 - \mathbf{L})\big)\mathbf{1} \\ \text{subject to} \quad & \mathbf{Tr}(\mathbf{U}) = 1, \mathbf{U} \succeq 0, \end{aligned} \quad (25)$$

where parameter $\rho_l$ controls the reliability of the component. Finally, (25) leads to dual form,

$$\begin{aligned} \text{minimize} \quad & \lambda^{\max}(\mathbf{S} + \mathbf{V}) \\ \text{subject to} \quad & |\mathbf{V}_{ij}| \leq \rho - \rho_l \log(1 - \mathbf{L}_{ij}),\ i, j = 1, ..., n. \end{aligned} \quad (26)$$

The KKT conditions for problem (25) and (26) are

$$\begin{cases} (\mathbf{S} + \mathbf{V})\mathbf{U} = \lambda^{\max}(\mathbf{S} + \mathbf{V})\mathbf{U} \\ \mathbf{V} \circ \mathbf{U} = (\rho - \rho_l \log(1 - \mathbf{L}_{ij}))|\mathbf{U}| \\ \mathbf{Tr}(\mathbf{U}) = 1,\ \mathbf{U} \succeq 0 \\ |\mathbf{V}_{ij}| \leq \rho - \rho_l \log(1 - \mathbf{L}_{ij}),\ i, j = 1, ..., n. \end{cases} \quad (27)$$

## Empirical Studies

Empirical studies were conducted on two types of problems; one is the grouping of sensors in industrial networks to perform decentralized fault detection, the other is the grouping of people in social networks based on their activity. For small problems, the semidefinite programs (9) and (24) were solved using interior-point solver SEDUMI (Sturm et al. 1999). Larger-scale problems were solved using an optimal first-order method (Nesterov et al. 1983) with an approximate gradient (d'Aspremont et al. 2008), specialized for problems such as (12) and (26). The code is written in C, with partial eigenvalue decompositions that are computed using the ARPACK package.

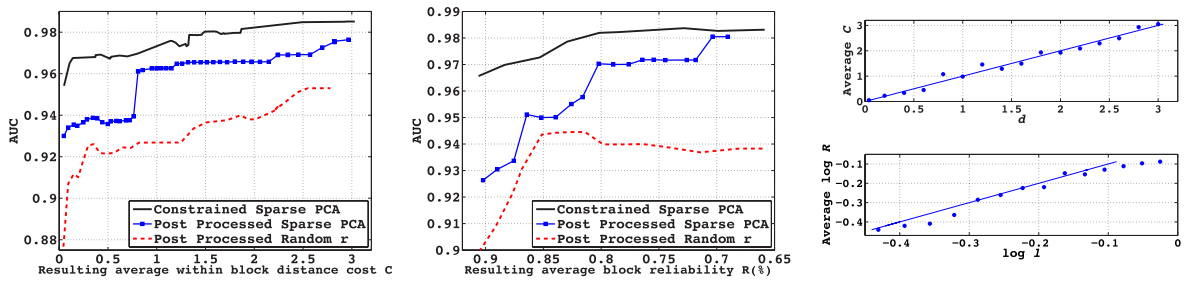
In addition, the following baselines were used:

1) **Post-processed Random r** starts by forming $K$ groups of randomly selected features of size $r$. To satisfy the distance or reliability constraints, features are removed one by one from a group that breaks the constraint. This is done in a greedy manner that removes the feature with the biggest impact on distance cost $C$ or reliability cost $R$.

2) **Post-processed Sparse PCA** starts by using Sparse PCA to create the first $K$ principal components of maximal cardinality $r$. Then, for each component that breaks the distance or reliability constraints, features are removed in a greedy manner as described above.

### Application to Industrial Sensor Networks

Timely fault detection in complex manufacturing systems is critical for effective operation of plant equipment. For small-scale sensor networks, it is reasonable to assume that all measurements are sent to some central location (sink) where fault predictions are made. This is known as a centralized fault detection approach. For large scale networks, a decentralized approach is often used (Grbovic and Vucetic 2012), in which the network is decomposed into potentially overlapping blocks, where each block provides local decisions that are fused at the sink. The appealing properties of the decentralized approach include fault tolerance, scalability, and reusability. For example, when one or more blocks go offline due to maintenance of their sensors, the predictions can still be made using the remaining blocks.




(a) *distance constr.* ($r = 4, K = 20$) (b) *reliability constr.* ($r = 4, K = 20$) (c) imposed constraints vs. resulting costs

Figure 1: TEP decentralized fault detection performance results in terms of Area Under the Curve (AUC)

**Data**. The Tennessee Eastman Process (TEP) (Downs and Vogel 1993) is a chemical process with 53 variables (pressure, temperature, etc.), measured by 53 sensors at distributed locations and 20 known fault types. The available data includes a training set $\mathbf{X}$ of $50,000$ normal operation examples and a test set that consists of $150,000$ normal operation examples and $150,000$ examples of faulty process operation. The sensor layout in TEP can be described by a fully connected undirected graph, where the edges represent the distances between physically connected sensors. This information was stored in matrix $\mathbf{D}$. The average edge length is $0.022$ and the average distance between two sensors is $0.24$. Different sensor types have different reliabilities and we have assigned them to vector $\mathbf{l}$. We set $\mathbf{l}$ within the range $0.8$ - $0.98$ with an average reliability of $0.93$.

**Baseline network decomposition approaches**. We considered a completely decentralized approach in which each sensor represents one block. We also considered several approaches that decompose sensors into $K$ overlapping blocks of size up to $r$ sensors. Mainly, a topology based approach (Zhang et al. 2010) in which we grouped sensors based on domain knowledge; randomly splitting sensors into overlapping groups; and baseline Sparse PCA where each of the first $K$ principal components defines a block of size up to $r$.

**Constrained network decomposition**. We considered Sparse PCA with communication constraints (9), with reliability constraints (24), and with both communication and reliability constraints where both convex relaxation constraints were added to (4). We also evaluated the post-processing approaches to maintaining the constraints that greedily remove sensors until the constraints are satisfied.

**Local predictors**. To create fault detectors for the $k$-th sensor group, we performed PCA using local training data $\mathbf{X}_k \in \mathbb{R}^{n_k \times m}$, where $n_k$ is the $k$-th group size. We calculated the amount of variation in the residual subspace (SPE values) spanned by $a = \max(1, n_k - \tau_{99})$ smallest principal components, where $\tau_{99}$ is the number of principal components that explain $99\%$ variance. Then, we determined an SPE threshold such that it is exceeded $p\%$ of the time on normal operation training data. If for a new example the threshold is exceeded, the $k$-th local detector reports an alarm.

**Decision Fusion**. Local predictions are sent to the sink. An alarm is signaled if any of the local detectors reports it.

| Baselines (no constraints) | $K$ | $r$ | Cost | Reliability | AUC |
|---|---|---|---|---|---|
| Centralized | 1 | 53 | 122.9 | 8.72% | .988 |
| Completely Decentralized | 53 | 1 | $\approx 0$ | 93% | .919 |
| Topology Based | 10 | 6 | .347 | 76.9% | .947 |
| Random $r$ | 20 | 4 | 1.54 | 71.6% | .932 |
| Sparse PCA | 20 | 4 | 3.6 | 68.6% | .984 |
| **Both const. $d = 1.5, l = .8$** | $K$ | $r$ | Cost | Reliability | AUC |
| Post-processed Sparse PCA | 20 | 4 | 1.49 | 79.0% | .966 |
| Post-processed Random $r$ | 20 | 4 | 1.38 | 75.6% | .913 |
| Constrained Sparse PCA | 20 | 4 | 1.49 | 79.2% | .981 |

Table 1: TEP fault detection performance results

**Results**. Test data were used to evaluate fault detection abilities of various decomposition schema in terms of AUC calculated by considering thresholds $p$ from $0.1$ to $99.9$.

Table 1 (top) compares 5 different network decomposition methods. The resulting average block communication cost ($\bar{C}$), reliability ($\bar{R}$) and AUC are reported. We can observe that the Sparse PCA model performed better than other decentralized models and similarly to the centralized model, while providing increased flexibility and fault tolerance.

In Figure 1 and Table 1 (bottom), we compare constrained network decomposition methods. Figure 1.a shows the fault detection performance (AUC) versus the resulting average within-block distance cost ($\bar{C}$). The performance curves were obtained by changing the distance threshold $d$ (the average distance between sensors in a block) from $0.05$ to $3$.

Figure 1.b shows the fault detection performance results (AUC) versus the resulting average block reliability ($\bar{R}$). The performance curves were obtained by changing the reliability threshold $l$ (the failure probability of each block is less than $l \cdot 100\%$) from $0.65$ to $0.99$.

Figure 1.c shows that the proposed relaxations successfully produce models with desired costs. The use of $2d/r$ in problem (9) and $r(r-1)\log((1+l^{1/r(r-1)})/2)$ in (24) shows good results in practice, as the resulting $C$ corresponds to $d$ and the resulting $\log R$ corresponds to $\log l$.

Finally, Table 1 (bottom) shows the results when both constraints were used. We set the distance threshold to $d = 1.5$ and the reliability constraint to $l = 0.8$.

In all cases constrained Sparse PCA produced models with desired cost, while performing better than the baselines.

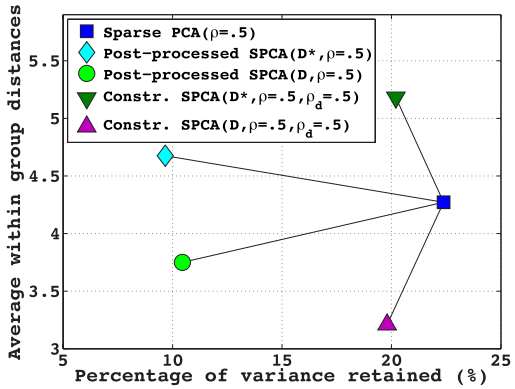



Figure 2: Digg social network results (distance vs. variance)

| | Sparse Principal Component | | |
|---|---|---|---|
| Strategy | 1 | 2 | 3 |
| DSPCA, $\rho = 0.5$ | **109 259** 277 | **183 188** | 527 221 |
| | 309 312 495 | 317 379 181 | 336 482 |
| | 159 180 224 257 | 385 418 138 | 31 41 275 |
| Constrained DSPCA | 109 259 277 | 138 **188** | **183** 221 |
| **do-not-link(183,188)** | 309 312 495 | 385 418 | 31 121 |
| $\rho = 0.5, \rho_d = 0.5$ | 159 180 224 257 | 379 | 129 |
| **remove-tags(109,259)** | **109** 565 277 585 | 183 188 | 527 221 |
| + | 309 159 180 312 | 317 379 181 | 336 482 |
| DSPCA, $\rho = 0.5$ | 224 257 629 251 | 385 418 138 | 31 41 275 |
| **remove-tags(109,259)** + | **109 259** 277 | 183 188 | 527 221 |
| DSPCA with Prior | 309 312 495 | 317 379 181 | 336 482 |
| **link(109,259)**, $\rho = 0.5$ | 159 180 224 257 | 385 418 138 | 31 41 275 |

Table 2: Facebook social network - resulting user groups

## Application to Social Networks

**The Digg social network** consists of $208,926$ users connected with $1,164,513$ edges that made $39,645$ "diggs" of $5,000$ news articles. Data matrix $\mathbf{X}$ was constructed as follows, $\mathbf{X}(i,j)$ was set to 1 if user $j$ diggs news $i$, and to 0 otherwise. In addition, there is an associated sparse friendship graph in which an edge indicates that the two users are friends on Digg. The distances $\mathbf{D}_{ab}$ between users $a$ and $b$ in the graph were calculated via Breath-first search. The maximum pairwise distance is 7 and the mean distance is $4.28$.

We used Sparse PCA with *distance constraint* (12) to find small groups of users with similar taste that are in addition:

1) Close in the network graph. The social network could benefit from such groups in the following manner. Instead of sending a notification of type "*your friend $x$ digged article $y$*" to all of your friends, it could notify only the friends that belong to the same interest group as you.

2) Far from each-other in the network graph (using (12) with $\mathbf{D}^*$, $\mathbf{D}^*_{ab} = 1/\mathbf{D}_{ab}$ , $\mathbf{D}^*_{aa} = 0$). Such groups would allow a social network to make smart friend suggestions to similar taste users located far from each other in the graph.

We used feature elimination to remove users with variances lower than $\rho$, which dramatically reduced the size of the problem, allowing us to work on covariance matrices of order at most $n = 2,000$, instead of the full order.

Figure 2 compares the baseline Sparse PCA result to the Constrained Sparse PCA and post-processed Sparse PCA results in the two scenarios. Default parameters were selected, $\rho = \rho_d = 0.5$. We report the average within-group distances in the first 8 groups versus the amount of explained variance.

Compared to Sparse PCA, the constrained and post-processed Sparse PCA groups had reduced average group distances in the 1-st, and increased average group distances in the 2-nd scenario. However, the constrained Sparse PCA components were more compact, and accounted for twice as much variance when compared to the baseline.

**The Facebook social network** consists of 953 friends of a single user that are tagged in $7,392$ photos. Data matrix $\mathbf{X}$ was constructed as follows, $\mathbf{X}(i,j)$ was set to 1 if friend $j$ was tagged in photo $i$, and to 0 otherwise.

The goal of this investigation was to test **link**$(a,b)$ and **do-not-link**$(a,b)$ constraints when grouping users based on tag information. We can think of this setup in terms of a Facebook application that would visualize clusters of user's friends that interact in real life (based on tags), and allow the user to refine the results using **link** and **do-not-link**.

The **do-not-link**$(a,b)$ constraints are important because spam photos with incorrect tags can lead to the wrong groups. These constraints can be enforced using *distance constraints*, where we set $\mathbf{D}_{ab} = \mathbf{D}_{ab} = +\infty$, and 0 otherwise. If we do not absolutely insist, the fields can be set to some high value. Constraints such as **do-not-link**$(a,b) >$ **do-not-link**$(c,d)$ are also possible, by setting $\mathbf{D}_{ab} > \mathbf{D}_{cd}$.

The **do-not-link**$(a,b)$ constraints were tested by repeating the following procedure many times: a) perform DSPCA b) randomly select $a$ and $b$, which belong to the same group c) repeat DSPCA with **do-not-link**$(a,b)$. We observed different outcomes: 1) exclusion of $a$ or $b$ from the group, and 2) the group split into subgroups that contain $a$ or $b$.

The **link**$(a,b)$ constraints are important due to missing tags. They cannot be directly enforced using the proposed framework. However, we have tested their soft enforcement through a covariance matrix prior, $\mathbf{S} = \mathbf{X}^T\mathbf{X} + \mathbf{P}$, where $\mathbf{P}$ is a symmetric matrix used to express user beliefs.

The **link**$(a,b)$ constraints were tested by removing tags from photos containing both $a$ and $b$, and attempting to compensate for the missing tags through prior knowledge, e.g. $\mathbf{P}_{ab} = 0.5$. In most cases the original groups were recovered, unless $\mathbf{P}_{ab}$ was set too high, which resulted in another group that contained only $a$ and $b$.

Table 2 shows an example of the results. The first column describes the applied strategy, while the remaining columns show the non-zero elements (i.e. user IDs) of the first 3 sparse components. The first row shows the resulting groups when baseline DSPCA with $\rho = 0.5$ was applied. For evaluation of our constrained Sparse PCA framework, we randomly selected two users from both the first and the second group (shown in bold).

The second row shows the results of DSPCA with a **do-not-link**$(188,183)$ constraint. It can be observed that the first group remained unchanged, while the second and third group changed in order to account for the imposed constraint. Hence, users 188 and 183 belong to separate groups.

The third and fourth row show the results of DSPCA with a **link**$(109, 259)$ constraint. Since current data already suggests that these two users should be grouped together, we first need to remove the evidence. This was done by deleting tags from photos containing both $109$ and $259$, i.e. setting their $1$ entries in corresponding $\mathbf{X}$ rows to $0$. To ensure that all evidence was removed, we repeated DSPCA on new data (third row). It can be observed that users $109$ and $259$ no longer belong to the same group. Next, we attempted to impute the missing information through an appropriate prior, by setting $\mathbf{P}_{109,259} = \mathbf{P}_{259,109} = 0.5$ and the remaining $\mathbf{P}$ entries to $0$. The resulting DSPCA decomposition with the **link**$(109, 259)$ constraint (fourth row) was able to recover the original components from the first row.

## Conclusion

We have presented a framework for adding constraints to the Sparse PCA problem. The constraints limit the set of possible solutions by imposing additional goals to be reached trough optimization along with the existing Sparse PCA goals. We have demonstrated the proposed framework on two real-world problems, industrial and social networks. The results show that Sparse PCA can utilize prior knowledge, which is not directly available in data, in order to produce desirable network partitions. All material used in the paper is available online[1].

## References


Amini, A., and Wainwright, M. 2009. High-dimensional analysis of semidefinite relaxations for sparse principal components. *The Annals of Statistics* 27(5B):2877–2921.

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

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

d'Aspremont, A.; El Ghaoui, L. E.; Jordan, M.; and Lanckriet., G. 2007. A direct formulation of sparse PCA using semidefinite programming. *SIAM Review* 49(3).

Downs, D., and Vogel, E. 1993. A plant-wide industrial process control problem. *Computers and Chemical Engineering* 17(3):245–255.

Jolliffe, I. T.; Trendafilov, N.; and Uddin, M. 2003. A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics* 12:531–547.

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

Journée, M.; Nesterov, Y.; Richtárik, P.; and Sepulchre, R. 2010. Generalized Power Method for Sparse Principal Component Analysis. *Journal of Machine Learning Research* 11:517–553.


[1] http://www.dabi.temple.edu/∼mihajlo/constrainedSPCA.htm


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

Qi, Z., and Davidson, I. 2009. A principled and flexible framework for finding alternative clustering. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

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

Wagsta, K.; Cardie, C.; Rogers, S.; and Schroedl, S. 2001. Constrained K-means Clustering with Background Knowledge. *International Conference on Machine Learning*, 577–584.

Wang, X., and Davidson, I. 2010. Flexible Constrained Spectral Clustering. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Zhang, Y., and El Ghaoui, L. E. 2011. Large-Scale Sparse Principal Component Analysis with Application to Text Data. *Advances in Neural Information Processing Systems*.

Zhang, Y.; d'Aspremont, A.; and El Ghaoui, L. E. 2011. Sparse PCA: Convex relaxations, algorithms and applications. *Handbook on Semidefinite, Cone and Polynomial Optimization: Theory, Algorithms, Software and Applications*.

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

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

Shahidehpour, M., and Marwali, M. 2000. Maintenance scheduling in restructured power systems. *Kluwer Academic Publisher*

Ben-Daya, M.; Salih, O. D.; and Raouf, A. 2000. Maintenance, modeling, and optimization. *Kluwer Academic Publisher*

Chiang, L. H.; Russell, E.; and Braatz R. D. 2001. Fault detection and diagnosis in industrial systems. *Springer*

Zhang, Y.; Zhou, H.; Qin, S. J.; and Chai, T. 2010. Decentralized Fault Diagnosis of Large-Scale Processes Using Multiblock Kernel Partial Least Squares. *IEEE Transactions on Industrial Informatics* 6:3–10.

Grbovic, M., and Vucetic, S. 2012. Decentralized fault detection and diagnosis via sparse PCA based decomposition and Maximum Entropy decision fusion. *Journal of Process Control* 22:738–750.

Sturm, J. 1999. Using SEDUMI 1.0x, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software* 11:625–653.

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

d'Aspremont, A. 2008. Smooth optimization with approximate gradient. *SIAM Journal on Optimization* 19:1171–1183

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