STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **3**, September 2015, pp 294–311.
Published online in International Academic Press (www.IAPress.org)

Two-Step Semidefinite Programming Approach to Clustering and Dimensionality Reduction

Eloísa Macedo*

Department of Mathematics, University of Aveiro, Portugal.

Received: 18 June 2015; Accepted: 25 August 2015
Editor: Yunhai Xiao

Abstract Inspired by the recently proposed statistical technique called clustering and disjoint principal component analysis (CDPCA), this paper presents a new algorithm for clustering objects and dimensionality reduction, based on Semidefinite Programming (SDP) models. The Two-Step-SDP algorithm is based on SDP relaxations of two clustering problems and on a K-means step in a reduced space. The Two-Step-SDP algorithm was implemented and tested in R, a widely used open source software. Besides returning clusters of both objects and attributes, the Two-Step-SDP algorithm returns the variance explained by each component and the component loadings. The numerical experiments on different data sets show that the algorithm is quite efficient and fast. Comparing to other known iterative algorithms for clustering, namely, the K-means and ALS algorithms, the computational time of the Two-Step-SDP algorithm is comparable to the K-means algorithm, and it is faster than the ALS algorithm.

Keywords: Data Mining, Clustering, PCA, Semidefinite Programming

AMS 2010 subject classifications: 62H25, 62H30, 90C22, 90C59

DOI: 10.19139/soic.v3i3.145

1. Introduction

The advances of computer technology have enabled to store large databases, such as, for example, data sets of sequenced genomes. When dealing with real data sets, it is often needed not only to reduce the dimension of the attribute space (dimensionality reduction), but also to reveal some patterns among the objects (clustering). Clustering methods and Principal Component Analysis (PCA) are extremely important for data visualization. These techniques have been widely studied and applied to many real-life data sets, and in areas such as Statistics, Machine Learning, Pattern Recognition, Engineering, Computational Biology and Image Processing [3, 18, 34].

The reduction of the object space is usually done by applying a clustering method to the data set. Clustering is an unsupervised learning technique. Clustering consists in division of a given set of objects into groups, called clusters, based on some similarity criterion in such a way that the objects within a cluster are more similar to one another, than the objects belonging to different clusters. There are different types of clustering problems. In this paper, we focus on the partitional clustering problem, where nonoverlapping clusters are constructed. In such a problem, each object can be considered as a point in a n-dimensional space and each cluster can be identified by its center, called centroid, a non-observable object calculated by taking the mean of all the objects assigned to this cluster [18, 32, 34]. To express similarity between objects, *i.e.*, homogeneity inside a cluster, several similarity measures have been proposed, such as a metric defined on the data set [2, 7]. One of the most used (dis)similarity measures is the squared Euclidean distance [3, 4, 14, 18, 34]. Given a number of objects, we will minimize the sum of the squared distances between each object and the corresponding cluster centroid. The resulting problem is called in the literature the minimum sum-of-squares clustering (MSSC) problem (see *e.g.*, [2, 4, 6, 14, 18, 26, 33, 34]). The MSSC problem is usually formulated as a Binary Integer Programming problem [2, 25, 26, 33], but it can also be rewritten either as a (0,1) - Semidefinite Programming (SDP) problem [19, 25, 26],

*Correspondence to: Eloísa Macedo (Email: macedo@ua.pt). Department of Mathematics, University of Aveiro. Campus Universitário de Santiago, 3810-193 Aveiro, Portugal.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

or as an unconstrained nonsmooth and nonconvex nonlinear problem [4, 6, 7, 33]. The MSSC problem is NP-hard [4, 15]. Many clustering algorithms have been developed to solve it, the most popular of them being, by far, the K-means algorithm [3, 6, 14, 18, 34]. In this paper, we focus on a SDP model for MSSC that was proposed in [25, 26].

PCA is a statistical technique for reducing the data dimensionality by finding linear combinations of all the original attributes, called components or principal components, that are able to explain the maximum variability of the data, *i.e.*, the data compression based on correlated attributes is done with minimum information loss [9, 16]. PCA is an orthogonal projection of the data onto a lower dimensional space along the direction where the data present the highest variability. Conventional PCA can be performed as either an eigendecomposition of the data covariance (or correlation) matrix, or a Singular Value Decomposition (SVD) of the column-centred (or standardized) data matrix. The resulting components are mutually uncorrelated and can be ordered by variance [14]. Usually, the coefficients of the principal components, also called loadings or weights, are nonzero. This can be considered a shortcoming for interpretation. Various PCA-based methodologies have been proposed to obtain disjoint or sparse components, *i.e.*, with zero loadings (see *e.g.*, [3, 9, 10, 17, 21, 32, 37]). Some of these approaches to get more interpretable components involve an attribute clustering (see *e.g.*, [3, 10, 32]).

A new methodology called Clustering and Disjoint Principal Component Analysis (CDPCA) was recently proposed by Vichi and Saporta in [32]. CDPCA permits to cluster the objects along a set of centroids and, at the same time, partition the attributes into a reduced set of components, in order to maximize the between cluster deviance. The resulting problem is formulated as a Quadratic Mixed Integer Programming problem and an alternating least-squares (ALS) algorithm is proposed to solve it in [32]. The ALS algorithm can be considered as a heuristic that guarantees only local solutions. In [22], the ALS algorithm to perform CDPCA was implemented in R [28]. Recently, in [23], a new scheme of the ALS algorithm was developed to improve the implementation of the CDPCA model in terms of estimating the parameters.

In this paper, we propose a new SDP-based approach to clustering and dimensionality reduction. Inspired by recent work [23, 25, 26, 32], we have developed the Two-Step-SDP algorithm that not only permits to cluster objects, but also attributes. The new algorithm is based on solving SDP relaxations of MSSC in an approximation algorithmic framework. The main purpose of the present work is to use SDP models and approximation algorithms to improve the ALS algorithm proposed by [32]. The Two-Step-SDP algorithm is implemented in a easy-to-use software application using R, which is available from the author upon request.

The paper is organized as follows. In Section 2, some notation and definitions used in this paper are introduced. In Section 3, we formulate the clustering problem and provide a brief description of the statistical technique CDPCA, as well as the underlying ALS algorithm. In Section 4, the clustering problem is reformulated as a SDP-based model and an approximation algorithm is described to solve it. As an alternative to CDPCA, a modification on the ALS algorithm based on SDP modelling, which is called Two-Step-SDP, is proposed. The implementation details of the new Two-Step-SDP algorithm are also described. Finally, in Section 5, we provide numerical experiments with the Two-Step-SDP algorithm on several data sets, including gene expression data sets. The concluding remarks are made in Section 6.

2. Notations and basic definitions

Consider a $(m \times n)$ real data matrix $\mathbf{D} = [d_{ij}]$ corresponding to a set of m objects (rows) described by n attributes (columns). Each object i is characterized by a n-dimensional row vector $\mathbf{d}_i$, $i = 1, ..., m$.

Clustering is to partition the objects of a data matrix into nonempty and nonoverlapping clusters C_j, $j = 1, ..., p$, where $2 \leq p < m$ is a given integer, in such a way that the sum of the squared distances between each object and the corresponding cluster centroid is minimized.

The assignment of objects into clusters can be stored in a $(m \times p)$ binary matrix $\mathbf{U} = [u_{ij}]$ where

$$u_{ij} = \begin{cases} 1, \text{ if object } i \in C_j, \\ 0, \text{ otherwise,} \end{cases} \tag{1}$$

and $\mathbf{U}$ is called *assignment* matrix. By construction, $\mathbf{U}$ is a binary row stochastic matrix satisfying the condition $\mathbf{U}\mathbf{e}^p = \mathbf{e}^m$, where $\mathbf{e}^k \in \mathbb{R}^k$ is a vector with all entries equal to one, $k \in \mathbb{N}$. Notice that $\mathrm{rank}(\mathbf{U}) = p$.

Each cluster can be identified by its cluster center, called centroid [18, 32, 34]. Thus, if the objects are assigned with the assignment matrix $\mathbf{U}$, then the centroid $\mathbf{c}_j \in \mathbb{R}^n$ of the cluster C_j, for $j = 1, ..., p$, can be defined as

$$\mathbf{c}_j = \frac{1}{\sum_{t=1}^{m} u_{tj}} \sum_{t=1}^{m} u_{tj}\mathbf{d}_t. \tag{2}$$

The $(p \times n)$ object cluster centroid matrix, whose rows correspond to the clusters presented by their centroids $\mathbf{c}_j$, is denoted here by $\bar{\mathbf{D}}$. In [32], this matrix is written as $\bar{\mathbf{D}} = \left(\mathbf{U}^T\mathbf{U}\right)^{-1}\mathbf{U}^T\mathbf{D}$.

Given a $(m \times p)$ matrix $\mathbf{M}$ whose p columns are linearly independent, the matrix $\mathbf{P}$ of the orthogonal projection of the space $\mathbb{R}^m$ onto the subspace spanned by the columns of $\mathbf{M}$ has the form $\mathbf{P} = \mathbf{M}(\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T$ and satisfies the following properties ([1, 35]):

$$\mathbf{P}^2 = \mathbf{P}, \tag{3}$$

$$\mathbf{P}^T = \mathbf{P}, \tag{4}$$

$$\mathrm{rank}\,(\mathbf{P}) = \mathrm{tr}\,(\mathbf{P}), \tag{5}$$

$$\mathrm{rank}\,(\mathbf{P}) = p, \tag{6}$$

$$\mathbf{P} \text{ has eigenvalues either } 0 \text{ or } 1. \tag{7}$$

From the last property, it follows that any orthogonal projection matrix $\mathbf{P}$ is positive semidefinite, denoted by $\mathbf{P} \succeq 0$.

Given any square matrices $\mathbf{A}$ and $\mathbf{B}$, $\mathbf{A} \succeq \mathbf{B}$ means that $\mathbf{A} - \mathbf{B}$ is a positive semidefinite matrix.

Given an assignment matrix $\mathbf{U}$, denote by $\mathcal{C}(\mathbf{U})$ the subspace of $\mathbb{R}^m$ spanned by its columns and consider a $(m \times m)$ matrix $\mathbf{Z} = [z_{ij}]$ in the following form ([25, 26]):

$$\mathbf{Z} = \mathbf{U}(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T. \tag{8}$$

It is easy to verify that $\mathbf{Z}$ has nonnegative elements and is the orthogonal projection matrix of the space $\mathbb{R}^m$ onto the space $\mathcal{C}(\mathbf{U})$. Hence, $\mathbf{Z}$ satisfies the properties (3)-(7).

Now, consider a partition of a set of n attributes into k, $2 \leq k < n$, disjoint subsets S_j, $j = 1, ..., k$, called components. The assignment of attributes can be stored in a $(n \times k)$ binary matrix $\mathbf{V} = [v_{ij}]$ where

$$v_{ij} = \begin{cases} 1, \text{ if attribute } i \in S_j, \\ \\ 0, \text{ otherwise.} \end{cases} \tag{9}$$

Notice that $\mathbf{V}$ is a binary row stochastic matrix satisfying $\mathbf{V}\mathbf{e}^k = \mathbf{e}^n$ and $\mathrm{rank}(\mathbf{V}) = k$. Let $\mathcal{C}(\mathbf{V})$ be the subspace of $\mathbb{R}^n$ spanned by the columns of the matrix $\mathbf{V}$. The $(n \times n)$ matrix $\mathbf{H} = [h_{ij}]$ defined as

$$\mathbf{H} = \mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T \tag{10}$$

is an orthogonal projection matrix of $\mathbb{R}^n$ onto $\mathcal{C}(\mathbf{V})$, and satisfies the above properties (3)-(7).

Given a data matrix $\mathbf{D}$, and assignment matrices $\mathbf{U}$ and $\mathbf{V}$, we introduce the following matrices:

- $\mathbf{B} = \sum_{q=1}^{k} \mathrm{diag}(\mathbf{v}_q)\mathrm{diag}(\mathbf{c}_q)$ – a diagonal matrix of order n specifying the loadings of each component, where $\mathbf{v}_q$ is the q-th column of $\mathbf{V}$ and $\mathbf{c}_q$ is the eigenvector associated to the largest eigenvalue of the matrix $(\mathbf{U}\bar{\mathbf{D}}\mathrm{diag}(\mathbf{v}_q))^T\mathbf{U}\bar{\mathbf{D}}\mathrm{diag}(\mathbf{v}_q)$ (here, $\mathrm{diag}(\mathbf{w})$ is a diagonal matrix whose entries are the elements of vector $\mathbf{w}$),
- $\mathbf{A} = \mathbf{B}\mathbf{V}$ – a $(n \times k)$ component loading matrix with a unique nonzero element per row, and $\mathrm{rank}(\mathbf{A}) = k$ and $\mathbf{A}^T\mathbf{A} = \mathbf{I}_k$,
- $\mathbf{Y} = \mathbf{D}\mathbf{A}$ – a component score matrix where y_{iq} is the value of the i-th object for the q-th component,
- $\bar{\mathbf{Y}} = \bar{\mathbf{D}}\mathbf{A}$ – an object centroid matrix in the reduced space of the components.

3. Clustering and dimensionality reduction

In this section, we formulate the clustering problem and describe a standard algorithm to solve it, the K-means algorithm. Then, we describe a statistical technique called CDPCA which permits to reduce not only the object space, but also the attribute space.

3.1. The clustering problem

Consider the clustering problem introduced in the previous section. The sum of the squared distances between each object and the centroid of the cluster to which it belongs is equal to $\sum_{j=1}^{p}\sum_{i=1}^{m} u_{ij}\|\mathbf{d}_i - \mathbf{c}_j\|_2^2$, where the cluster centroid $\mathbf{c}_j$ is defined in (2). Thus, the minimum sum-of-squares clustering (MSSC) problem can be formulated as follows:

$$\begin{aligned} \min_{u_{ij}} \quad & \sum_{j=1}^{p}\sum_{i=1}^{m} u_{ij}\left\|\mathbf{d}_i - \frac{\sum_{t=1}^{m} u_{tj}\mathbf{d}_t}{\sum_{t=1}^{m} u_{tj}}\right\|_2^2 \\ \text{s.t.} \quad & \sum_{j=1}^{p} u_{ij} = 1, \quad i = 1, ..., m, \\ & \sum_{i=1}^{m} u_{ij} \geq 1, \quad j = 1, ..., p, \\ & u_{ij} \in \{0, 1\}. \end{aligned} \tag{11}$$

The first constraint in (11) ensures that each object is assigned to a single cluster and the second constraint ensures that each cluster has at least one object assigned. Any feasible solution of problem (11) is an assignment matrix $\mathbf{U}$.

Problem (11) is a Binary Integer Programming problem with discrete variables and nonlinear and nonconvex objective function. This problem is known to be NP-hard and hence, very difficult to solve [2, 4, 15, 26]. Therefore, many approaches have been proposed to solve it either by exact algorithms, or heuristics (see *e.g.*, [2, 4, 19, 25, 26]). By far, the most popular algorithm to solve the MSSC problem (11) is the K-means algorithm, whose main steps can be outlined as follows ([15]):

Algorithm 1 K-means

1: Choose an initial partition of p clusters (or generate it randomly) and find the corresponding centroids,
2: Assign each object i, $i = 1, ..., m$, to the closest centroid,
3: Update the centroids using the current assignments.

Steps 2 and 3 are repeated until the within cluster sum of squares is no longer reduced. Numerical tests show that the K-means algorithm returns well-separated clusters having a convex-shaped geometry, *i.e.*, spherical or elliptical [14, 15].

It is worthwhile mentioning that the K-means algorithm should be used on scaled data, since it relies on the Euclidean distances. Moreover, the K-means algorithm returns a local optimum and depends on the initial choice of the centroids [6, 15, 18]. To overcome this drawback, it is recommended to consider several random initializations [15]. For example, in [32], a "tandem analysis" (*i.e.*, PCA followed by applying the K-means algorithm using only the first few components) was carried out with 10000 random starts of the K-means algorithm using the first two principal components of a particular data set. The data set consists of 20 objects and 6 attributes, and the aim was to obtain 3 clusters of objects.

3.2. The Clustering and Disjoint Principal Component Analysis (CDPCA)

A new methodology called CDPCA was recently proposed by Vichi and Saporta in [32] for obtaining not only nonoverlapping clusters of objects, but also a partition of the attribute space into disjoint subsets. This is important for visualization and interpretation purposes of (large) data sets.

Given a data set, CDPCA groups m objects into p, $2 \leq p < m$, clusters identified by their centroids and, simultaneously, partitions n attributes into k, $2 \leq k < n$, disjoint subsets of attributes.

In what follows, we briefly describe the CDPCA methodology (see [23, 32] for further details). A given data matrix $\mathbf{D}$ can be modelled as follows:

$$\mathbf{D} = \mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T + \mathbf{E}, \tag{12}$$

where $\mathbf{U}$ is the object assignment matrix, $\bar{\mathbf{Y}}$ is the object centroid matrix in the reduced space of the components, $\mathbf{A}$ is the component loading matrix and $\mathbf{E}$ is a $m \times n$ error matrix.

The idea is to minimize the error in the CDPCA model (12), *i.e.*, minimize the norm of the error matrix: $\|\mathbf{D} - \mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T\|_2^2$. It is easy to see that this problem is equivalent to maximizing $\|\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T\|_2^2$. Since the columns $\mathbf{A}_i$, $i = 1, ..., k$, of the matrix $\mathbf{A}$ are orthogonal, *i.e.*, $\mathbf{A}^T\mathbf{A} = \mathbf{I}_k$, the matrix $\bar{\mathbf{Y}}$ is given by $\bar{\mathbf{Y}} = \bar{\mathbf{D}}\mathbf{A}$, and the trace of a square matrix equals the trace of its transpose, then maximizing $\|\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T\|_2^2$ is the same as maximizing $\|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2$, the between cluster deviance in the reduced space. Indeed, $\|\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T\|_2^2 = \operatorname{tr}\left((\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T)(\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T)^T\right) = \operatorname{tr}\left(\mathbf{U}\bar{\mathbf{Y}}\mathbf{A}^T\mathbf{A}\bar{\mathbf{Y}}^T\mathbf{U}^T\right) = \operatorname{tr}\left(\mathbf{U}\bar{\mathbf{Y}}\bar{\mathbf{Y}}^T\mathbf{U}^T\right) = \operatorname{tr}\left(\mathbf{U}(\bar{\mathbf{D}}\mathbf{A})(\bar{\mathbf{D}}\mathbf{A})^T\mathbf{U}^T\right) = \operatorname{tr}\left((\mathbf{U}\bar{\mathbf{D}}\mathbf{A})(\mathbf{U}\bar{\mathbf{D}}\mathbf{A})^T\right) = \|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2$. Hence, we get the optimization problem

$$\max_{\mathbf{U},\bar{\mathbf{D}},\mathbf{A}} \quad \|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2. \tag{13}$$

In [32], to solve the problem (13), it is proposed to consider a decomposition of the matrix $\mathbf{A}$ in order to include a binary and row stochastic matrix $\mathbf{V}$, specifying the partition of n attributes into k disjoint subsets. The positions of the nonzero elements in the matrix $\mathbf{A}$ are identified by the positions of the unit elements in the matrix $\mathbf{V}$. Hence, the CDPCA problem can be formulated as the following Quadratic Mixed Integer Programming problem:

$$\begin{array}{lll}
\max\limits_{\mathbf{U},\mathbf{V},\mathbf{A}} & \|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2 & \\
\text{s.t.} & u_{ij} \in \{0,\ 1\}, & i = 1, ..., m;\ j = 1, ..., p \\
& \sum\limits_{j=1}^{p} u_{ij} = 1, & i = 1, ..., m \\
& v_{ij} \in \{0,\ 1\}, & i = 1, ..., n;\ j = 1, ..., k \\
& \sum\limits_{j=1}^{k} v_{ij} = 1, & i = 1, ..., n \\
& \sum\limits_{i=1}^{n} a_{ij}^2 = 1, & j = 1, ..., k \\
& \sum\limits_{i=1}^{n} (a_{ij}a_{ir})^2 = 0, & j = 1, ..., k-1;\ r = j+1, ..., k.
\end{array} \tag{14}$$

The first two constraints correspond to the assignment of m objects into p clusters and the next two constraints represent the assignment of n attributes into k disjoint components. The remaining constraints are associated to the PCA implementation. The objective function value given by $\|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2$ can be computed either by $\operatorname{tr}\left(\mathbf{U}\bar{\mathbf{D}}\mathbf{A}(\mathbf{U}\bar{\mathbf{D}}\mathbf{A})^T\right)$, which corresponds to the between cluster distances, or by $\operatorname{tr}\left((\mathbf{U}\bar{\mathbf{D}}\mathbf{A})^T\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\right)$, which represents the total variance of the data in the reduced space, where the objects are identified by their centroids.

3.3. Alternating Least-Squares (ALS) algorithm

Problem (14) is difficult to solve due to the presence of discrete variables. In [32], an Alternating Least-Squares (ALS) algorithm is proposed to solve this problem.

The ALS algorithm starts from a randomly chosen assignment of the objects, specified by the matrix $\mathbf{U}$, and an assignment of the attributes, specified by $\mathbf{V}$. Then, it repeats the following four steps at each iteration:

1. update the assignment of attributes $\mathbf{V}$,
2. apply PCA to get the component loadings,
3. update the assignment of objects $\mathbf{U}$,
4. update the centroid matrix $\bar{\mathbf{D}} = \left(\mathbf{U}^T\mathbf{U}\right)^{-1}\mathbf{U}^T\mathbf{D}$.

In [23], we showed that each iteration of the ALS algorithm can be summarily described by two basic steps: the assignment of objects via K-means, and the reduction of the attribute space via application of PCA to the resulting centroids. The simplified version of the ALS algorithm is presented below.

Algorithm 2 Alternating Least-Squares (Two-Step version of ALS) ([23])

input: $\mathbf{D}$, data matrix of m objects and n attributes;
p and k, the desired numbers of clusters of objects and attributes, respectively;
ε, numerical tolerance.
output: $\mathbf{U}$, object assignment matrix;
$\mathbf{V}$, attribute assignment matrix;
$\mathbf{A}$, component loading matrix;
$\|\mathbf{U}\bar{\mathbf{D}}\mathbf{A}\|_2^2$, the between cluster deviance in the reduced space of the components.

repeat

1. K-means step (for the objects):
   - assign m objects into p clusters, obtaining matrix $\mathbf{U}$,
   - calculate the centroids in the space of the observed attributes and find matrix $\bar{\mathbf{D}}$,
   - identify the objects by their cluster centroids in the space of the observed attributes and get $\mathbf{U}\bar{\mathbf{D}}$.
2. PCA step (for the attributes):
   - assign n attributes into k subsets and obtain matrix $\mathbf{V}$,
   - obtain the loadings of the CDPCA components and get matrix $\mathbf{A}$,
   - calculate the centroids in the reduced space of k CDPCA components and define matrix $\bar{\mathbf{Y}} = \bar{\mathbf{D}}\mathbf{A}$,
   - identify the objects in the reduced space of k CDPCA components and calculate matrix $\mathbf{Y} = \mathbf{D}\mathbf{A}$.

until the difference between two consecutive computations of the between cluster deviance in the reduced space is smaller than ε.

Since the objective function of problem (14) is bounded above, the algorithm converges to a stationary point, which is a local maximum of problem [32]. This procedure can be considered as a heuristic and thus, to increase the possibility to achieve the global maximum, it has been suggested to run the algorithm several times for different initial assignment matrices $\mathbf{U}$ and $\mathbf{V}$, randomly chosen at the beginning of each run.

The difference between the four step ALS algorithm proposed in [32] and its simplified version described in [23], and summarized in Algorithm 2, consists in the way matrices $\mathbf{V}$ and $\mathbf{A}$ are updated. In the general iteration of the Two-Step version of ALS, these matrices are sequentially constructed by an iterative procedure applied to their rows and columns, in order to maximize the between cluster deviance in the reduced space, while in the four step ALS algorithm the matrices $\mathbf{V}$ and $\mathbf{A}$ are updated separately. Numerical experiments show that the ALS algorithm in its simplified Two-Step version 2 is faster than the four step version.

In the following section, we will present a new algorithm for clustering and dimensionality reduction, based on SDP models.

4. Two-Step-SDP: A new version of clustering and dimensionality reduction

In this section, the clustering problem is reformulated in the form of a nonlinear SDP-based model. To solve this model, an approximation algorithm based on a linear SDP relaxation is described. The new Two-Step-SDP algorithm based on a modification of the ALS algorithm to perform CDPCA is presented. Some differences between the Two-Step-SDP and the ALS algorithms are discussed.

4.1. SDP-based formulation for the clustering problem

In [25, 26], another formulation for the clustering problem (11), called 0-1 SDP model, is proposed. This formulation involves an unknown orthogonal projection matrix $\mathbf{Z}$ defined in (8). It is shown that the objective function in (11) is equal to the function $\mathrm{tr}\left(\mathbf{D}\mathbf{D}^T(\mathbf{I_m} - \mathbf{Z})\right)$ and the 0-1 SDP model is formulated as follows:

$$\begin{array}{ll} \min\limits_{\mathbf{Z}} & \mathrm{tr}\left(\mathbf{D}\mathbf{D}^T(\mathbf{I_m}-\mathbf{Z})\right) \\ \text{s.t.} & \mathbf{Z}^T=\mathbf{Z}, \\ & \mathbf{Z}^2=\mathbf{Z}, \\ & \mathrm{tr}(\mathbf{Z})=p, \\ & \mathbf{Z}\mathbf{e}^m=\mathbf{e}^m, \\ & z_{ij}\geq 0, \forall i,j=1,2,...,m. \end{array} \tag{15}$$

The first two constraints in (15) imply that $\mathbf{Z}$ is an orthogonal projection matrix and the next constraint ensures that there are exactly p clusters. The last but one constraint in problem (15) means that each object is assigned to a single cluster, *i.e.*, each row of the matrix $\mathbf{Z}$ has sum equal to 1. The last constraint ensures that $\mathbf{Z}$ has nonnegative elements.

In [26], it is proved that problem (15) is equivalent to (11). Problem (15) is called 0-1 SDP model in [25, 26], since the first two constraints imply that $\mathbf{Z}$ is positive semidefinite, with eigenvalues 0 or 1. Nevertheless, this model does not have a standard SDP form, therefore, in this paper, we will call it the SDP-based model of the clustering problem (11).

Since minimizing $\mathrm{tr}\left(\mathbf{D}\mathbf{D}^T(\mathbf{I_m}-\mathbf{Z})\right)$ is equivalent to maximizing $\mathrm{tr}\left(\mathbf{D}\mathbf{D}^T\mathbf{Z}\right)$, we can rewrite problem (15) as

$$\begin{array}{ll} \max\limits_{\mathbf{Z}} & \mathrm{tr}(\mathbf{D}\mathbf{D}^T\mathbf{Z}) \\ \text{s.t.} & \mathbf{Z}^T=\mathbf{Z}, \\ & \mathbf{Z}^2=\mathbf{Z}, \\ & \mathrm{tr}(\mathbf{Z})=p, \\ & \mathbf{Z}\mathbf{e}^m=\mathbf{e}^m, \\ & z_{ij}\geq 0, \forall i,j=1,2,...,m. \end{array} \tag{16}$$

Problem (16) is very difficult to solve due to the nonlinearity of the second constraint. In [25], an approximation algorithm based on a linear SDP relaxation is proposed to obtain its solution.

4.2. SDP-based approximation algorithm

Approximation algorithms are widely used to solve hard optimization problems (see *e.g.*, [4, 12, 13, 26]). The general idea of an approximation algorithm is to first solve a relaxation of the hard problem, and then apply some rounding procedure to the obtained solution to finally find a feasible solution of the original problem. It is known that SDP relaxations provide good approximations. The first attempt to use a SDP-based approximation algorithm dates back to 1995, when Goemans and Williamson [13] suggested the randomized approximation algorithm for the max-cut problem, a well known NP-hard problem [20]. Since then, SDP has been successfully applied in the development of approximation algorithms for several classes of hard combinatorial optimization problems. It is also known that standard interior point SDP methods, such as the primal-dual interior point methods [5, 12, 30, 36], are not so efficient for large scale problems and thus, different strategies have been developed [11, 19, 25].

In [25], an approximation algorithm that uses a linear SDP relaxation is proposed to obtain an approximate solution of (16). First, the SDP relaxation is solved using a procedure based on a characterization introduced in [24]. Then, a rounding procedure is used to extract a feasible solution of (16). In what follows, we consider two steps of the approximation algorithm based on a SDP relaxation proposed in [25].

4.2.1. Linear SDP relaxation and its solution

The SDP-based model (16) can be relaxed to a simpler problem by removing the nonnegative requirement on the entries of $\mathbf{Z}$ and replacing its first two constraints by the constraint $\mathbf{I}_m \succeq \mathbf{Z} \succeq 0$, yielding the following convex SDP problem:

$$\begin{array}{ll} \max\limits_{\mathbf{Z}} & \mathrm{tr}(\mathbf{D}\mathbf{D}^T\mathbf{Z}) \\ \text{s.t.} & \mathrm{tr}(\mathbf{Z})=p, \\ & \mathbf{Z}\mathbf{e}^m=\mathbf{e}^m, \\ & \mathbf{I}_m \succeq \mathbf{Z} \succeq 0. \end{array} \tag{17}$$

Notice that the last constraint ensures that all the eigenvalues of the positive semidefinite matrix $\mathbf{Z}$ are less or equal to 1.

It is worthwhile mentioning that the solution of the relaxed problem (17) may not coincide with the solution to the SDP-based problem (16).

Since standard SDP methods are not so efficient for large scale problems, in [25], an alternative procedure to solve the SDP problem (17) is suggested. This procedure is based on the characterization of the sum of the largest eigenvalues of a symmetric matrix introduced in [24].

First, notice that one of the eigenvalues of any feasible solution $\mathbf{Z}$ of problem (17) is equal to 1 and the corresponding eigenvector is $\frac{1}{\sqrt{m}}\mathbf{e}^m$. Moreover, any matrix $\mathbf{Z}$ can be written as

$$\mathbf{Z} = \mathbf{Z}_1 + \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T, \tag{18}$$

where $\mathbf{Z}_1$ is a column and row centred matrix by the orthogonal projection matrix $\left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right)$, *i.e.*,

$$\mathbf{Z}_1 = \left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right)\mathbf{Z} = \left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right)\mathbf{Z}\left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right). \tag{19}$$

Notice that $\mathrm{tr}(\mathbf{Z}_1) = \mathrm{tr}(\mathbf{Z}) - 1 = p - 1$.

Applying the same transformation on $\mathbf{D}\mathbf{D}^T$, we get the matrix $\mathbf{W}_1$, which is given by

$$\mathbf{W}_1 = \left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right)\mathbf{D}\mathbf{D}^T\left(\mathbf{I}_m - \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T\right). \tag{20}$$

Second, notice that maximizing the objective function in (17) is equivalent to maximizing $\mathrm{tr}(\mathbf{D}\mathbf{D}^T\mathbf{Z}_1)$, which in turn is equal to $\mathrm{tr}(\mathbf{W}_1\mathbf{Z}_1)$. Then, problem (17) can be rewritten in the form of the following problem w.r.t. the matrix variable $\mathbf{Z}_1$:

$$\begin{array}{ll} \max\limits_{\mathbf{Z}_1} & \mathrm{tr}(\mathbf{W}_1\mathbf{Z}_1) \\ \text{s.t.} & \mathrm{tr}(\mathbf{Z}_1) = p-1, \\ & \mathbf{I}_m \succeq \mathbf{Z}_1 \succeq 0. \end{array} \tag{21}$$

Based on the results from [24] and [25], the optimal solution of the SDP problem (21) can be obtained if and only if

$$\mathrm{tr}(\mathbf{W}_1\mathbf{Z}_1) = \sum_{i=1}^{p-1}\lambda_i,$$

where $\lambda_1, ..., \lambda_m$ are the eigenvalues of the matrix $\mathbf{W}_1$ listed in decreasing order.

Any solution $\mathbf{Z}_1$ of the SDP problem (21) has the form $\mathbf{Z}_1 = \mathbf{F}\mathbf{F}^T$, where $\mathbf{F}$ is the $(m \times (p-1))$ matrix whose columns are the eigenvectors associated to the $p-1$ largest eigenvalues of the matrix $\mathbf{W}_1$ [20, 24, 25].

Hence, after obtaining a solution $\mathbf{Z}_1$ of the SDP problem (21), we replace it in (18) to get the solution of the SDP relaxed problem (17).

Therefore, a procedure to obtain a solution $\mathbf{Z}$ of the relaxed problem (17) can be outlined as follows ([25]):

Algorithm 3 Solving the SDP relaxed problem (17)

1: Compute the matrix $\mathbf{W}_1$ by using (20),
2: Apply SVD to obtain the $p-1$ largest eigenvalues of the matrix $\mathbf{W}_1$ and the associated eigenvectors, $\mathbf{v}^1, ..., \mathbf{v}^{p-1}$, and construct the matrix $\mathbf{F}$, whose columns are these eigenvectors,
3: Set $\mathbf{Z} = \mathbf{F}\mathbf{F}^T + \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T$.

The solution of the SDP problem (17), obtained using the Algorithm 3, can be used as an approximate solution of the SDP-based problem (16). This solution is not necessarily feasible. To find a feasible solution of (16), a rounding procedure is needed. In what follows, we will describe a rounding procedure that was proposed in [26].

4.2.2. Rounding the approximate solution

It can be observed that if $\mathbf{Z}$ is a solution of problem (16), then $\mathbf{ZD}$ has at least p different rows. Notice that $\mathbf{ZD}$ can be considered as an object centroid based matrix, where each object is identified by the cluster centroid to which it belongs. Based on this observation, in [26], the following rounding procedure is suggested.

Algorithm 4 Rounding procedure for the SDP-based problem (16)

1: Given the data matrix $\mathbf{D}$ and the solution $\mathbf{Z}$ of the relaxed problem (17), select p different rows of $\mathbf{ZD}$ and define the initial centroids,
2: Apply K-means to problem (11) using the initial centroids to obtain $\mathbf{U}$,
3: Set $\mathbf{Z} = \mathbf{U}(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T$.

Notice that the SDP-based approximation algorithm, described by the Algorithms 3 and 4, performs a PCA (Principal Component Analysis) step, by reducing the problem into (21) and solving it using SVD, and a K-means step. Motivated by this and the idea of the CDPCA methodology, in the following section, we propose a new approach based on a Two-Step-SDP clustering scheme, that uses approximation algorithms based on SDP relaxations for clustering and dimensionality reduction.

4.3. A new approach: Two-Step-SDP

Motivated by the works [23, 25, 26] and [32], we propose a new approach to CDPCA by combining SDP models and the CDPCA methodology. The new approach is called Two-Step-SDP, since two clustering problems are considered and solved using a SDP-based approximation algorithmic framework, and since we modify and improve the ALS algorithm in its Two-Step version summarized in Algorithm 2 in order to use SDP models for clustering objects and attributes. There are two main modifications on the Algorithm 2. In the first modification, we suggest to apply the SDP-based approximation algorithm described in Section 4.2 to construct the initial matrices $\mathbf{U}$ and $\mathbf{V}$, instead of a random choice. It is expected that this approximation algorithm returns *almost optimal solutions* quite fast. The second modification on the Algorithm 2 consists in updating the component loading matrix $\mathbf{A}$ using the current assignment matrix $\mathbf{V}$. Both modifications improve the Algorithm 2 not only in terms of computational time, but also in efficiency.

The idea of the Two-Step-SDP approach is to obtain not only a solution of the SDP-based problem defined in (16) for clustering objects, but also a solution of a SDP-based problem for clustering attributes into disjoint subsets, using an approximation algorithm based on the framework described in the previous section. The solutions of such problems are computed in order to maximize the between cluster deviance in the reduced space of the components. Therefore, the Two-Step-SDP approach can be considered as an alternative to CDPCA.

Consider the SDP-based model (16) for the clustering problem. Recall that for clustering the objects into groups, in (16) we use the matrix $\mathbf{Z}$ defined in (8) as an unknown orthogonal projection matrix defined by the assignment matrix $\mathbf{U}$. For clustering attributes into disjoint subsets, we can use the matrix $\mathbf{V}$ and an unknown orthogonal projection matrix $\mathbf{H}$ defined in (10). Using the model (16) with the matrix variable $\mathbf{H}$, we get a SDP-based problem.

As it was mentioned above, the SDP-based models of the form (16) are difficult to solve. Let us relax the nonlinear constraints of the SDP-based problems using the approach described in the previous section. Thus, for the SDP-based problem for clustering objects, we get the relaxed SDP problem (17),

$$\begin{array}{ll} \max\limits_{\mathbf{Z}} & \mathrm{tr}(\mathbf{D}\mathbf{D}^T\mathbf{Z}) \\ \text{s.t.} & \mathrm{tr}(\mathbf{Z}) = p, \\ & \mathbf{Z}\mathbf{e}^m = \mathbf{e}^m, \\ & \mathbf{I}_m \succeq \mathbf{Z} \succeq 0, \end{array}$$

and for the SDP-based problem for clustering attributes, we get the following relaxation:

$$\begin{array}{ll} \max\limits_{\mathbf{H}} & \mathrm{tr}(\mathbf{D}^T\mathbf{D}\mathbf{H}) \\ \text{s.t.} & \mathrm{tr}(\mathbf{H}) = k, \\ & \mathbf{H}\mathbf{e}^n = \mathbf{e}^n, \\ & \mathbf{I}_n \succeq \mathbf{H} \succeq 0. \end{array} \tag{22}$$

To solve the linear SDP problems (17) and (22) one can use the approach described in Section 4.2, and the solution of the original SDP-based models can be obtained using a rounding procedure.

The rounding procedure used in the Two-Step-SDP approach can be summarized as follows. Given assignment matrices $\mathbf{U}$ and $\mathbf{V}$, apply the CDPCA methodology to obtain the component loading matrix $\mathbf{A}$, as well as the component score matrix $\mathbf{Y}$, and the object centroid matrix in the reduced space, $\bar{\mathbf{Y}}$. Next, perform the K-means algorithm for clustering the objects in the reduced space, *i.e.*, apply K-means to the matrix $\mathbf{Y}$ and use $\bar{\mathbf{Y}}$ as initial centroids. Finally, compute the between cluster deviance in the reduced space of the components. The algorithm stops when the between cluster deviance is no longer increased.

4.3.1. Two-Step-SDP algorithm

The algorithmic scheme of the Two-Step-SDP algorithm for clustering and dimensionality reduction can be described as follows.

Algorithm 5 Two-Step-SDP

input: $\mathbf{D}$, data matrix of m objects and n attributes;
p and k, the desired number of clusters of objects and attributes, respectively;
ε, numerical tolerance.

output: $\mathbf{U}$, object assignment matrix;
$\mathbf{V}$, attribute assignment matrix;
$\mathbf{A}$, component loading matrix;
$\|\mathbf{ZDA}\|_2^2$, the between cluster deviance in the reduced space of the components.

1: Solve approximately the SDP-based model for clustering objects:

a) Considering the relaxed model (17), compute the matrix $\mathbf{W}_1$ by using (20),
b) Use SVD to obtain the $p-1$ largest eigenvalues of the matrix $\mathbf{W}_1$ and the associated eigenvectors, $\mathbf{v}^1, ..., \mathbf{v}^{p-1}$, and construct the matrix $\mathbf{F}$, whose columns are these eigenvectors,
c) Set the approximate solution: $\bar{\mathbf{Z}} = \mathbf{F}\mathbf{F}^T + \frac{1}{m}\mathbf{e}^m(\mathbf{e}^m)^T$.

2: Solve approximately the SDP-based model for clustering attributes:

a) Considering the relaxed model (22), compute the matrix $\mathbf{W}_2$ by using (20) on the matrix $\mathbf{D}^T\mathbf{D}$,
b) Use SVD to obtain the $k-1$ largest eigenvalues of the matrix $\mathbf{W}_2$ and the associated eigenvectors, $\mathbf{w}^1, ..., \mathbf{w}^{k-1}$, and construct the matrix $\mathbf{G}$, whose columns are these eigenvectors,
c) Set the approximate solution: $\bar{\mathbf{H}} = \mathbf{G}\mathbf{G}^T + \frac{1}{n}\mathbf{e}^n(\mathbf{e}^n)^T$.

3: Rounding procedure:

Initialization: randomly select p different rows from $\bar{\mathbf{Z}}\mathbf{D}$, and k different rows from $\bar{\mathbf{H}}\mathbf{D}^T$, as initial centroids on the K-means algorithm to get the initial matrices $\mathbf{U}$, and $\mathbf{V}$, respectively.

a) Compute $\mathbf{Z}^* = \mathbf{U}(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T$ and $\bar{\mathbf{D}} = (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{D}$.
b) Compute $\mathbf{A}$ by fixing each column $\mathbf{v}_q$, $q = 1, ..., k$, of $\mathbf{V}$ and replacing the nonzero elements in $\mathbf{v}_q$ by the elements of the eigenvector associated to the largest eigenvalue of the matrix $(\mathbf{Z}^*\mathbf{D}[\mathbf{c}])^T\,\mathbf{Z}^*\mathbf{D}[\mathbf{c}]$, where $\mathbf{c}$ is a row vector containing the row indices of nonzero elements of $\mathbf{v}_q$, *i.e.*, the attributes that contribute to the component q, and $[\mathbf{c}]$ means that only the columns specified in $\mathbf{c}$ are used.
c) Update $\mathbf{U}$ by performing the K-means algorithm in the reduced space, *i.e.*, applying it to the matrix $\mathbf{Y} = \mathbf{DA}$ with initial centroids $\bar{\mathbf{Y}} = \bar{\mathbf{D}}\mathbf{A}$. Randomly select k different rows from $\bar{\mathbf{H}}\mathbf{D}^T$ as initial centroids on K-means and get $\mathbf{V}$.
d) Compute the between cluster deviance in the reduced space, $\|\mathbf{Z}^*\mathbf{DA}\|_2^2$.

Stopping criterion: The rounding procedure stops when the difference between two consecutive computations of the between cluster deviance in the reduced space is smaller than the pre-specified tolerance value ε.

Since the Two-Step-SDP algorithm uses the approximate solutions of the nonlinear SDP-based problems for clustering objects and attributes, which are close to the final solutions, the clustering process is expected to be finite.

The main feature of the Two-Step-SDP algorithm is that it constructs the clusters of attributes and use them to cluster the objects by maximizing the between cluster deviance in the reduced space of the components, *i.e.*, the K-means algorithm is performed in lower dimensions. Moreover, the Two-Step-SDP algorithm provides not only the allocation of objects and attributes into clusters, but also the component loadings of such allocation of attributes. The quality of the clustering can be measured by considering the between cluster deviance in the reduced space of the components, or the within sum of squared distances between each object and the cluster centroids.

4.3.2. Two-Step-SDP and ALS

Notice that both the Two-Step-SDP and the ALS algorithms use iterative schemes to obtain the best solution. Nevertheless, there are some differences. In particular, in the ALS algorithm, the update of the matrices $\mathbf{V}$ and $\mathbf{A}$ is done using an alternating procedure that works row-by-row and column-by-column on these matrices by obtaining components that explain the largest variance in order to maximize the between cluster deviance in the reduced space (see all the details in [23]). Such alternating procedure has to be performed nk times at each iteration of the main algorithm. Thus, it may be quite expensive in terms of computation time and memory. In the Two-Step-SDP algorithm, the matrix $\mathbf{V}$ is obtained using the solution of the SDP relaxation problem (22), and therefore, it is expected to be close to the optimal assignment. Hence, at each iteration of the rounding procedure, the update of the matrix $\mathbf{A}$ requires only one step. Notice that in the Two-Step-SDP algorithm, the dimensionality reduction is done by finding a partition of the attributes specified by $\mathbf{V}$, and then the component loadings specified in the matrix $\mathbf{A}$ for this particular partition are computed. Thus, the dimensionality reduction may not explain the largest variance. Therefore, the Two-Step-SDP algorithm can be regarded as a simplified version of the ALS algorithm.

5. Computational results

This section is devoted to present our numerical experiments using the statistical software R, which is an open source software widely used by the statistical community. This software has a lot of specific routines for different kinds of problems. Details on how to use our implementation of the Two-Step-SDP algorithm, as well as some examples are presented. A comparison of the results obtained using the proposed Two-Step-SDP algorithm and other existing techniques is also reported.

5.1. The R *function* `TwostepSDPClust`

We have implemented the Two-Step-SDP algorithm in R. In order to solve the problems more efficiently, we used two strategies concerning computation of eigenvalues and eigenvectors. In the steps of the algorithm, we need to compute eigenvalues and eigenvectors of possibly huge covariance matrices of the form $\mathbf{M}^T\mathbf{M}$, where $\mathbf{M}$ is a matrix with more columns than rows. In these cases, we used the approach suggested in [27] and considered the smaller matrix $\mathbf{M}\mathbf{M}^T$. It is shown that the nonzero eigenvalues of $\mathbf{M}^T\mathbf{M}$ and $\mathbf{M}\mathbf{M}^T$ coincide, and each unit eigenvector $\mathbf{w}$ of $\mathbf{M}^T\mathbf{M}$ associated to the eigenvalue λ is related to the unit eigenvector $\mathbf{v}$ of the smaller matrix $\mathbf{M}\mathbf{M}^T$ by $\mathbf{w} = \mathbf{M}^T\frac{\mathbf{v}}{\sqrt{\lambda}}$. In the rounding procedure step of the Two-Step-SDP algorithm, we need to compute the largest eigenvalue and the corresponding eigenvector of a matrix. Here, we have implemented the Power Method [8], which proved to be faster than the standard methods for computing eigenvalues and eigenvectors.

The Two-Step-SDP algorithm was implemented in R by the function `TwostepSDPClust`, and is available from the author upon request. This function is suitable for data matrices with numeric elements and starts by standardizing the data (the description of the standardize step can be found in [23]). The user needs to input the following arguments in `TwostepSDPClust`:

- `data` – the numeric data matrix
- `p` – the number of clusters of objects

- `k` – the number of clusters of attributes
- `class` – the vector of integers with the true classification of objects, or 0
- `tol` – a small convergence tolerance value
- `maxit` – the maximum number of iterations

The routine `TwostepSDPClust` provides much more information than the standard K-means function in R, *e.g.*, besides returning the clusters of both objects and attributes, it returns the loadings of each original attribute to each component in the reduced space, and, when the real objects classification is known, a pseudo-confusion matrix. A pseudo-confusion matrix is important to show the (mis)classification of the objects.

The R function `TwostepSDPClust` returns the following information:

- `Dscale` – the scaled data matrix
- `U` – the object assignment matrix
- `cluster` – the vector of integers identifying the clusters of objects
- `Usizes` – the vector whose coordinates are the sizes of each cluster of objects
- `Z0` – the approximate solution of the SDP-based problem for clustering objects
- `Z` – the solution to the SDP-based problem for clustering objects
- `bcdr` – the between cluster deviance in the reduced space of the k components
- `bcdrp` – the between cluster deviance of the total variance
- `wssd` – the within cluster deviance for clustering of objects
- `ofrelaxZ` – the value of the objective function of the SDP problem (17)
- `iter` – the number of iterations in the refinement solution step
- `V` – the attribute assignment matrix
- `H0` – the approximate solution of the SDP-based problem for clustering attributes
- `H` – the solution of the SDP-based problem for clustering attributes
- `Vsizes` – the vector with the sizes of each cluster of attributes
- `wssdat` – the within cluster deviance for clustering of attributes
- `bcdat` – the between cluster deviance for clustering of attributes
- `Ybar` – the object centroid matrix in the reduced space
- `Dbar` – the object centroid matrix
- `Y` – the component score matrix
- `A` – the component loading matrix
- `explvar` – the vector of explained variance of each component
- `E` – the error associated to the CDPCA model
- `compt` – the computational time
- `pcm` – the pseudo-confusion matrix, when the true classes are known

In the R environment, after defining the input arguments, the user can use the following command to apply the function `TwostepSDPClust`:

```
> example <- TwostepSDPClust(data,p,k,class,tol,maxit)
```

5.2. Numerical experiments

To test the efficiency of the Two-Step-SDP algorithm implemented in the R function `TwostepSDPClust`, several numerical experiments have been carried out on a computer with an Intel Core i7-2630QM processor CPU@2.0GHz,, with Windows 7 (64 bits) and 12GB RAM, using R version 3.1.2 (2014). We compare the results of the Two-Step-SDP algorithm with other freely-available R functions. One such function is `kmeans`, the standard K-means algorithm which is available in R. The other function is our R implementation of the ALS algorithm, under the name `CDpca`, firstly implemented in [22] and recently improved in [23].

The experiments were made using some standard problems from cluster analysis literature. Both data sets with more objects than attributes and with more attributes than objects have been selected. The following real-world data sets are available in the UCI repository [29], in the R package `plsgenomics` and in [31]:

- *OECD countries* – the data set of 20 objects and 6 attributes from [31], representing the short-term macroeconomic scenario (1999) of OECD countries; notice that, although the true classes are not known, the experiments reported in [31, 32] suggest to divide the objects into 3 clusters.
- *Breast Cancer* – the Winsconsin Breast Cancer data from UCI repository; contains 683 instances (originally, there were 699 instances, but 16 of them were excluded, since they contain missing values), each instance is described by 9 attributes with integer values in the range $1 - 10$ and a real binary class label, which divides the instances into two clusters, representing the type of tumor (benign or malignant).
- *Diabetes* – the Pima Indians Diabetes data set from UCI repository, contains 768 objects and 8 attributes, divided into two classes, representing the results on testing diabetes (positive or negative).
- *Colon* – the data set from microarray experiments on colon tissue samples, available in the the R package `plsgenomics`; contains 62 samples and 2000 genes, divided into 2 classes, representing the type of tumor (benign or malignant).
- *leukemia* – the leukemia data set, available in the package `plsgenomics`; contains 38 samples and 3051 genes, also divided into two classes, representing the type of tumor (benign or malignant).
- *SRBCT* – the gene expression data set from microarray experiments on small round blue cell tumors, also available in package `plsgenomics`; contains 83 objects (called samples) and 2308 attributes (genes), divided into 4 groups, representing four cancer variants.
- *Iris* – the Fishers Iris data set, available in the UCI repository, the most famous data set for clustering experiments; contains 150 objects described by 4 attributes, and the true classification of the objects in 3 clusters is known, representing the three species of the Iris flower.
- *Soybean* – the Soybean data set from UCI repository contains 47 objects described by 35 attributes, where the four true classes of objects are also known, and represent four soybean diseases.

For the numerical tests using the Two-Step-SDP algorithm we have set the tolerance value to 10^{-8} and the maximum number of iterations to 100. The K-means heuristic was executed using all the data sets previously scaled, the multiple random start was set to 1000, and the maximum number of iterations was 100000. The tolerance for the ALS algorithm was set to 10^{-5} and the maximum number of iterations was 100. To augment the efficiency of the ALS algorithm, the numerical tests were run 1000 times, with the exception of the gene expression data *Colon*, *leukemia* and *SRBCT*. For the data sets *Colon* and *leukemia*, 20 runs were performed, and for *SRBCT*, only 10 runs were performed, because the computation time of the ALS algorithm increased in an unreasonable way in these cases. For the *Soybean* data set, the standardize step on the experiments was not executed, since the data matrix presents null columns.

Following the analysis presented in [32] for the *OECD countries* data set, let us consider only two principal components for all the tested data sets, *i.e.*, $k = 2$. Table I summarizes the characteristics of the data sets. It also contains the desired number of clusters of objects, represented by p.

Table I. Summary of the characteristics of the data sets and the number of clusters of objects used in the experiments.

Dataset	Number of objects, m	Number of attributes, n	Desired number of clusters of objects, p
OECD countries	20	6	3
Breast Cancer	683	9	2
Diabetes	768	8	2
Colon	62	2000	2
leukemia	38	3051	2
SRBCT	83	2308	4
Iris	150	4	3
Soybean	47	35	4

For all the presented data, the true classes are known, with exception for the *OECD countries* data set.

The detailed numerical results are presented in Tables II, III and IV. The Tables II and III contain the results obtained using the Two-Step-SDP algorithm and the K-means algorithm, respectively, while Table IV contains the

results obtained using the ALS algorithm. The first column of Tables II, III and IV contains the name of the data set, and the second column contains the computational time in seconds. In these tables, *iter* is the number of iterations, *wssd* is the within sum of squared distances, *i.e.*, within cluster deviance, *bcd* is the between cluster deviance, *bcdr* is the between cluster deviance in the reduced space of the components, *bcdrp* is the between cluster deviance of the total variance, e is the error associated to the CDPCA model (12), *Usize* and *Vsize* represent the sizes of the clusters of objects and attributes, respectively, and *Exp.var.* is the explained variance of the components. It should be noticed that, according to the information provided by the algorithms, the Tables II, III and IV contain a different number of columns.

Table II. Numerical results using the Two-Step-SDP algorithm.

Data set	time	iter	wssd	bcd	bcdr	bcdrp	e	Usize	Vsize	Exp.var.
OECD countries	0.00	2	72.15	47.84	31.57	70.0	0.47	10	4	22.39
								3	2	17.17
								7		
Breast Cancer	1.19	2	2728.15	3418.85	3418.85	75.6	0.07	230	8	62.52
								453	1	11.13
Diabetes	1.22	2	5121.9	1022.09	1022.09	38.7	0.09	309	2	26.62
								459	6	16.45
Colon	14.66	2	84631.79	39368.21	39368.21	61.4	4.69	18	581	15.03
								44	1419	15.03
leukemia	60.78	2	102681.7	13256.27	13256.27	75.9	8.43	17	1374	7.83
								21	1677	7.63
SRBCT	28.97	2	156947.7	34616.26	19438.99	86.2	4.99	13	1293	6.28
								16	1015	5.63
								18		
								36		
Iris	0.03	2	141.03	458.96	454.5	80.5	0.08	50	1	69.60
								52	3	25.17
								48		
Soybean	0.05	6	453.12	2760.87	2557.18	99.6	0.54	11	10	1.51
								13	25	1.38
								9		
								14		

Table III. Numerical results using the K-means algorithm.

Data set	time	wssd	bcd	Usize	Vsize
OECD countries	0.12	72.15	47.84	10, 3, 7	4, 2
Breast Cancer	0.49	2728.15	3418.85	230, 453	8, 1
Diabetes	0.73	5121.9	1022.09	309, 459	2, 6
Colon	10.27	84631.79	39368.21	18, 44	581, 1419
leukemia	9.83	102681.7	13256.27	17, 21	1374, 1677
SRBCT	25.76	151603.6	39960.37	17, 18, 22, 26	1278, 1030
Iris	0.17	139.82	460.17	50, 47, 53	1, 3
Soybean	0.21	205.96	484.2	10, 13, 14, 10	23, 12

Table IV. Numerical results using the ALS algorithm.

Data set	time	loop	iter	bcdr	bcdrp	e	Usize	Vsize	Exp.var.
OECD countries	26.17	98	5	45.98	77.8	0.43	11	3	28.65
							3	3	23.18
							6		
Breast Cancer	107.62	329	4	3418.85	79.8	0.07	230	5	39.11
							453	4	30.67
Diabetes	276.54	7	14	1022.09	44.5	0.09	309	7	24.96
							459	1	12.52
Colon	3198.53	17	4	39368.21	71.1	4.69	18	1019	23.39
							44	981	21.96
leukemia	3388.19	13	5	13256.27	78.4	8.43	17	1559	7.75
							21	1492	7.23
SRBCT	12786.95	5	15	25873.22	90.4	4.90	16	1302	8.75
							17	1006	6.37
							24		
							26		
Iris	58.33	566	10	454.5	80.5	0.08	50	3	69.60
							52	1	25.17
							48		
Soybean	175.76	1	4	2736.31	99.3	0.46	10	14	13.11
							13	21	1.31
							11		
							13		

Comparing the results obtained using the Two-Step-SDP algorithm presented in the Table II and the standard K-means algorithm displayed in the Table III, we can conclude that in general, in terms of computational time and solution, both approaches are quite efficient. For the *leukemia* data set, the K-mean algorithm performs quite faster. Notice that the K-means had to be executed for clustering attributes and objects, thus, the computational time is the sum of the running times of both procedures. With respect to the clusters of attributes, the K-means algorithm does not provide further information on the variance explained by the resulting components, while the Two-Step-SDP algorithm does. It can be observed that in almost all cases, the values of the within and between cluster deviances obtained using the Two-Step-SDP or the K-means algorithms are quite similar. The major difference in the performance of these algorithms is for the *Soybean* data set. The Two-Step-SDP algorithm returned the within cluster deviance equal to 453.12 and the between cluster deviance equal to 2760.87, while the K-means algorithm returned the within cluster deviance equal to 205.96 and the between cluster deviance equal to 484.2. Regarding the clusters of the attributes, it can be observed that the Two-Step-SDP and the K-means algorithms return clusters of equal sizes, with exception for the *Soybean* and *SRBCT* data sets. With respect to the clusters of objects, these algorithms have returned different clusters for the *Iris*, *Soybean* and *SRBCT* data sets.

Comparing the results presented in Tables II and IV, we can conclude that the Two-Step-SDP algorithm is faster than the ALS algorithm. In a couple of iterations, it founds a solution that either yields the same value of the between cluster deviance in the reduced space as that returned by the ALS algorithm, or is very close to it. It can also be observed that the errors of solving the CDPCA model (12) using the Two-Step-SDP and the ALS algorithms are very similar. Regarding the results of the clustering of objects, it can be observed that for the *Breast Cancer*, *Diabetes*, *Colon*, *leukemia* and *Iris* data sets, the sizes of each cluster coincide for both approaches. With respect to the clusters of attributes, there are some differences, which induce differences in the variance explained by the obtained components. The ALS algorithm provides better results for the *Soybean*, *OECD countries* and *SRBCT* data sets in terms of the value of the between cluster deviance in the reduced space and in the explained variance by the components. It can also be observed that for the *Breast Cancer* and the *Colon* data sets, the variances presented by the first component are, respectively, 62.52% and 37.56% for the Two-Step-SDP algorithm, and 39.11% and 23.39% for the ALS algorithm.

In order to compare the quality of the object clusters obtained using the three algorithms, we present pseudo-confusion matrices, summarizing the (mis)classification of the objects when the true classes are known. In a

pseudo-confusion matrix, the rows correspond to the true classes and the columns correspond to the classes returned by algorithms. Notice that the order of the predicted classes may be different to that chosen for the real ones. The analysis of the performance of each algorithm can be complemented by computing its accuracy. Here, we measure the accuracy of an algorithm by the number of the correct predictions of objects divided by the total number of objects.

Table V. Pseudo-confusion matrices for classification of objects obtained using the R functions `TwostepSDPClust`, `kmeans` and `CDpca`. The *OECD countries* data is not included, since the true classes of this data set are unknown.

Data set: `TwostepSDPClust` | `kmeans` | `CDpca`

Breast Cancer

Real	Two-Step-SDP 1	2
1	10	434
2	220	19

Real	K-means 1	2
1	10	434
2	220	19

Real	CDPCA 1	2
1	434	10
2	19	220

Diabetes

Real	Two-Step-SDP 1	2
1	156	356
2	153	103

Real	K-means 1	2
1	156	356
2	153	103

Real	CDPCA 1	2
1	156	356
2	153	103

Colon

Real	Two-Step-SDP 1	2
1	6	16
2	12	28

Real	K-means 1	2
1	6	16
2	12	28

Real	CDPCA 1	2
1	16	6
2	28	12

leukemia

Real	Two-Step-SDP 1	2
1	16	11
2	1	10

Real	K-means 1	2
1	11	16
2	10	1

Real	CDPCA 1	2
1	11	16
2	10	1

SRBCT

Real	Two-Step-SDP 1	2	3	4
1	4	13	9	3
2	0	0	8	3
3	0	0	12	6
4	9	3	7	6

Real	K-means 1	2	3	4
1	12	8	6	3
2	0	4	4	3
3	0	3	9	6
4	5	7	7	6

Real	CDPCA 1	2	3	4
1	7	6	4	12
2	0	4	7	0
3	4	9	5	0
4	13	7	0	5

Iris

Real	Two-Step-SDP 1	2	3
1	50	0	0
2	0	41	9
3	0	11	39

Real	K-means 1	2	3
1	50	0	0
2	0	11	39
3	0	36	14

Real	CDPCA 1	2	3
1	0	0	50
2	9	41	0
3	39	11	0

Soybean

Real	Two-Step-SDP 1	2	3	4
1	0	4	0	6
2	8	0	2	0
3	0	1	5	4
4	3	8	2	4

Real	K-means 1	2	3	4
1	0	10	0	0
2	0	0	0	10
3	5	0	5	0
4	9	0	8	0

Real	CDPCA 1	2	3	4
1	5	0	5	0
2	4	0	6	0
3	1	4	0	5
4	0	9	0	8

Considering the pseudo-confusion matrices presented in the Table V, we can conclude that all the approaches to clustering provide the same results in terms of classification of the objects for the *Breast Cancer* data, and the corresponding accuracy is very high, 96%; the *Diabetes* data, presenting the accuracy equal to 66%; the *Colon* data, with the accuracy equal to 55% and the *leukemia* data, with the accuracy equal to 68%. For the *SRBCT* and *Soybean* data sets, all the approaches present different cluster structures. The K-means presents the lowest accuracy for the *SRBCT* data, which is 37%, while the ALS algorithm had the highest accuracy, 47%, that is quite

close to that obtained using the Two-Step-SDP algorithm, equal to 45%. For the *Soybean* data, the Two-Step-SDP algorithm yielded an accuracy of 57%, the ALS algorithm had an accuracy of 53% and the K-means algorithm presented the highest accuracy: 72%. For the *Iris* data set, the Two-Step-SDP and ALS algorithms obtained the same classifications, presenting an accuracy of 87%, while the K-means algorithm returned an accuracy of 83%.

6. Concluding remarks

In this paper, we have presented a new approach to clustering and dimensionality reduction. The Two-Step-SDP algorithm is presented for clustering both objects and attributes. The algorithm starts by solving particular SDP relaxed models, whose solutions are used to get the initial centroids for the clustering procedure. Then, it uses an iterative rounding scheme for clustering the objects by maximizing the between cluster deviance in the reduced space. Since the initial centroids are obtained from the solutions of the SDP relaxations, it is expected that the rounding procedure stops in a finite number of iterations. The new algorithm is implemented in a easy-to-use software application using R, and we have included tests that show the efficiency of the Two-Step-SDP algorithm. A comparison with other clustering algorithms, namely, the ALS and the K-means algorithms, was performed.

Based on the numerical experiments, we can conclude that the Two-Step-SDP algorithm finds solutions that are close to that obtained using the ALS and K-means heuristics. The experiments show that the Two-Step-SDP algorithm is comparable to the K-means algorithm in terms of the execution time, and that the Two-Step-SDP algorithm is significantly faster than the ALS algorithm. Although the computational time of K-means is slightly better than the Two-Step-SDP algorithm in several cases, the K-means algorithm ends up returning less information than the Two-Step-SDP algorithm. In particular, K-means does not provide further information on the partition of the attribute space, namely, about the component loadings and the variance explained by the components, while the Two-Step-SDP algorithm does. We can conclude that the Two-Step-SDP algorithm for clustering and dimensionality reduction can be considered as a modification or a simplified version of the ALS algorithm, since it provides almost the same information, but it is much faster in obtaining the assignments, and the loss on the information of the explained variance by the components is quite insignificant.

Acknowledgment

The author would like to thank the anonymous referees for the valuable comments that have helped to improve the paper. This work was partially supported by Portuguese funds through the CIDMA - Center for Research and Development in Mathematics and Applications, and the Portuguese Foundation for Science and Technology (FCT - Fundação para a Ciência e a Tecnologia), within project UID/MAT/04106/2013.

REFERENCES

1. Abadir, K.M. and Magnus, J.R., *Matrix Algebra*, Cambridge University Press, 2005.
2. Aloise, D. and Hansen, P., *A branch-and-cut sdp-based algorithm for minimum sum-of-squares clustering*, Pesquisa Operacional, 29(3), pp. 503–516, 2009.
3. Ames, Brendan P.W., *Guaranteed clustering and biclustering via semidefinite programming*, Mathematical Programming, 147(1-2), Springer Berlin Heidelberg, pp. 429–465, 2014.
4. Akteke-Ozturk, B., Weber, G.-W. and Kropat, E., *Continuous Optimization Approaches for Clustering via Minimum Sum of Squares*, Proc. 20th Mini-EURO Conf. Continuous Optimization and Knowledge-Based Technologies, 2007.
5. Anjos, M.F. and Lasserre, J.B. (Eds.), *Handbook of Semidefinite, Conic and Polynomial Optimization: Theory, Algorithms, Software and Applications*, International Series in Operational Research and Management Science, 166, Springer, 2012.
6. Bagirov, A.M., *Modified global k-means algorithm for minimum sum-of-squares clustering problems*, Pattern Recognition, 41, pp. 3192–3199, 2008.
7. Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N.V. and Yearwood, J., *Unsupervised and Supervised Data Classification via Nonsmooth and Global Optimization*, TOP, 11, pp. 1–93, 2003.
8. Bronson, R. and Costa, G.B., *Matrix Methods: Applied Linear Algebra*, 3rd Ed., Academic Press, Elsevier, 2009.
9. d'Aspremont, A., El Ghaoui, L., Jordan, M.I. and Lanckriet, G.R.G., *A Direct Formulation for Sparse PCA Using Semidefinite Programming*, SIAM Rev., 49(3), pp. 434–448, 2007.

10. Enki, D.G., Trendafilov, N.T. and Jolliffe, I.T., *A clustering approach to interpretable principal components*, Journal of Applied Statistics, 40(3), pp. 583–599, 2013.
11. Fiala, J., Kočvara, M. and Stingl, M., *PENLAB: A MATLAB solver for nonlinear semidefinite optimization*, Nov 2013, http://arxiv.org/pdf/1311.5240v1.pdf,
12. Gärtner, B. and Matoušek, J., *Approximation Algorithms and Semidefinite Programming*, Springer-Verlag, 2012.
13. Goemans, M.X. and Williamson, D.P., *Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming*, J. ACM, 42(6), pp. 1115–1145, 1995.
14. Hastie, T., Tibshirani, R. and Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer Series in Statistics, 2009.
15. Jain, A.K., *Data Clustering: 50 Years Beyond K-means*, Elsevier Science Inc., Pattern Recogn. Lett., 31(8), pp. 651–666, 2010.
16. Jolliffe, I.T., *Principal Component Analysis* Second edition, Springer-Verlag, New York, 2002.
17. Jolliffe, I.T., Trendafilov, N.T. and Uddin, M., *A modified principal component technique based on the lasso*, J. of Computational and Graphical Statistics, 12(3), pp. 531–547, 2003.
18. Kogan, J., Nicholas, C. and Teboulle, M. (Eds.), *Grouping Multidimensional Data: Recent Advances in Clustering*, Springer, 12, 2006.
19. Kulis, B., Surendran, A.C. and Platt, J.C., *Fast low-rank semidefinite programming for embedding and clustering*, in Proceedings of the International Workshop on Artificial Intelligence and Statistics, San Juan, Puerto Rico, 2007.
20. Laurent, M. and Rendl, F., *Semidefinite programming and integer programming*, In Nemhauser, G., Aardal, K. and Weismantel, R. (eds), Handbook on Discrete Optimization, Elsevier, pp. 393–514, 2005.
21. Ma, Z., *Sparse principal component analysis and iterative thresholding*, The Annals of Statistics, 41(2), pp. 772–801, 2013.
22. Macedo, E. and Freitas, A., *Statistical Methods and Optimization in Data Mining*, In: III International Conference of Optimization and Applications OPTIMA2012, pp. 164–169, 2012.
23. Macedo, E. and Freitas, A., *The Alternating Least-Squares Algorithm for CDPCA*, In: Plakhov, A., Tchemisova, T. and Freitas, A. (eds.) Optimization in the Natural Sciences, Communications in Computer and Information Science (CCIS), Springer Verlag, 499, pp. 173–191, 2015.
24. Overton, M.L. and Womersley, R.S., *Optimality Conditions and Duality Theory for Minimizing Sums of the Largest Eigenvalues of Symmetric Matrices*, Mathematical Programming, 62, pp. 321–357, 1993.
25. Peng, J. and Wei, Y., *Approximating k-means-type clustering via semidefinite programming*, SIAM J. OPTIM., 18(1), pp. 186–205, 2007.
26. Peng, J. and Xia, Y., *A New Theoretical Framework for K-Means-Type Clustering*, Foundations and Advances in Data Mining Studies in Fuzziness and Soft Computing, 180, pp. 79–96, 2005.
27. Pinto Da Costa, J.F., Alonso, H. and Roque, L., *A weighted principal component analysis and its application to gene expression data*, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 8(1), pp. 246–252, 2011.
28. R Development Core Team, R*: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org, 2015.
29. UCI Repository: http://archive.ics.uci.edu/ml/
30. Vandenberghe, L. and Boyd, S., *Semidefinite Programming*, SIAM Review, 38(1), pp. 49–95, 1996.
31. Vichi, M. and Kiers, H.A.L., *Factorial k-means analysis for two-way data*, Computational Statistics & Data Analysis, 37(1), pp. 49–64, 2001.
32. Vichi, M. and Saporta, G., *Clustering and Disjoint Principal Component Analysis*, Computational Statistics & Data Analysis, 53, pp. 3194–3208, 2009.
33. Weber, G.-W., Taylan, P., Ozogur, S. and Akteke-Ozturk, B., *Statistical Learning and Optimization Methods in Data Mining*, in Ayhan, H.O. and Batmaz, I. (eds.): Recent Advances in Statistics, Turkish Statistical Institute Press, Ankara, pp. 181–195, 2007.
34. Xu, R. and Wunsch, D., *Survey of Clustering Algorithms*, IEEE Transactions on Neural Networks, 16(3), pp. 645–648, 2005.
35. Yanai, H., Takeuchi, K. and Takane, Y., *Projection Matrices, Generalized Inverse Matrices, and Singular Value Decomposition*, Statistics for Social and Behavioral Sciences, Springer, 2011.
36. Zhao, D. and Zhang, M., *A primal-dual large-update interior-point algorithm for semi-definite optimization based on a new kernel function*, Statistics, Optimization & Information Computing, 1(1), pp. 41–61, 2013.
37. Zou, H., Hastie, T., Tibshirani, R., *Sparse principal component analysis*, J. of Computational and Graphical Statistics, 15(2), pp. 262–286, 2006.