Towards Robust Arbitrarily Oriented Subspace Clustering

Zhong Zhang, Chongming Gao, Chongzhi Liu, Qinli Yang, and Junming Shao(✉)

School of Computer Science and Engineering,
University of Electronic Science and Technology of China, Chengdu 611731, China
{zhongzhang,chongming.gao,liuchongzhi}@std.uestc.edu.cn
{qinli.yang,junmshao}@uestc.edu.cn

Abstract. Clustering high-dimensional data is challenging since meaningful clusters usually hide in the arbitrarily oriented subspaces, and classical clustering algorithms like k-means tend to fail in such case. Subspace clustering has thus attracted growing attention in the last decade and many algorithms have been proposed such as ORCLUS and 4C. However, existing approaches are usually sensitive to global and/or local noisy points, and the overlapping subspace clusters are little explored. Beyond, these approaches usually involve the exhaustive local search for correlated points or subspaces, which is infeasible in some cases. To deal with these problems, in this paper, we introduce a new subspace clustering algorithm called RAOSC, which formulates the Robust Arbitrarily Oriented Subspace Clustering as a group structure low-rank optimization problem. RAOSC is able to recover subspace clusters from a sea of noise while noise and overlapping points can be naturally identified during the optimization process. Unlike existing low-rank based subspace clustering methods, RAOSC can explicitly produce the subspaces of clusters without any prior knowledge of subspace dimensionality. Furthermore, RAOSC does not need a post-processing procedure to obtain the clustering result. Extensive experiments on both synthetic and real-world data sets have demonstrated that RAOSC allows yielding high-quality clusterings and outperforms many state-of-the-art algorithms.

Keywords: Subspace clustering · Correlation clustering

1 Introduction

In high-dimensional data set, meaningful clusters usually hide in the arbitrarily oriented subspaces, i.e., subsets of points showing linear correlations among subsets of dimensions [11]. To explain this idea, consider a real-world example illustrated in Fig. 1. The 2D plot represents a Height/Weight Standard[1], which consists of two subspace clusters of male and female, respectively. In contrast to

[1] http://www.angelo.edu/dept/rotc/height_weight_chart.php.

G. Li et al. (Eds.): DASFAA 2019, LNCS 11446, pp. 276–291, 2019.
https://doi.org/10.1007/978-3-030-18576-3_17

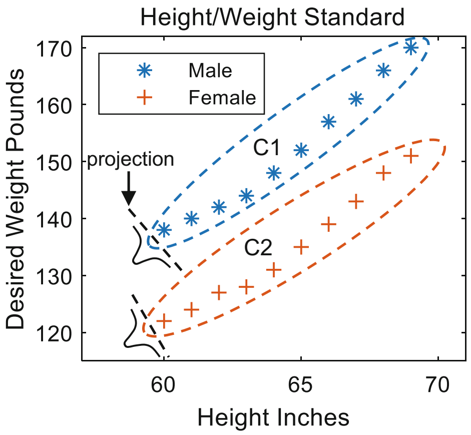


Fig. 1. A real-world example of subspace clustering. C1 and C2 are two subspace clusters.

the classical clustering, points are grouped into the same cluster because they exhibit high correlation (i.e., they locate near the same line or plane), rather than closeness. Only when projecting the points into the subspace orthogonal to the plane where they are lying in, they will exhibit high density. Since each cluster lies in an arbitrarily oriented subspace, it is referred to as arbitrarily oriented subspace clustering or correlation clustering.

Clustering a 2D data is just a piece of cake, but how about a high-dimensional data having dozens or hundreds of attributes? In such case, detecting subspace clusters is a challenging task since many dimensions are irrelevant and only a few of dimensions truly contribute to the cluster structure. The word "Relevant" means that a cluster shows high correlation in and only in these relevant dimensions. More importantly, the relevant dimensions often differ largely for different clusters [11]. Therefore, global dimensionality reduction methods like Principal Component Analysis (PCA) cannot be used to preserve the subspace cluster structure. To tackle this problem, most approaches adopt certain assumptions/heuristics and start from a local search of subspaces and clusters.

During the past decade, many subspace clustering approaches have been proposed from various perspectives. The earliest attempt is to heuristically examine all possible axis-parallel subspaces and identify clusters, algorithms include CLIQUE [4], ENCLU [7], PROCLUS [2], SUBCLU [10], DUSC [5], to name a few. However, these algorithms can only find axis-parallel subspace clusters. Afterwards, arbitrarily oriented subspace clustering emerges. Most of these algorithms rely on the search of local correlated points to identify clusters and subspaces. Algorithms include, for examples, ORCLUS [3], 4C [6], CURLER [21], SSCC [9], FOSSCLU [8], ORSC [18] and CoSync [19]. However, most previous solutions of finding suitable subspaces work well if and only if subspace clusters are locally well separated and no noise/outlier points exist. In the presence of noise/outliers in the local neighbourhood of cluster points or cluster representatives in the entire feature space, most previous methods fail to detect meaningful subspace clusters. Besides, due to the (exhaustive) heuristic local search, they are generally time consuming especially when the dimensionality is high.

In contrast to previous methods that use certain heuristic ways to search subspace clusters, we turn to formulate the subspace clustering task as a group structure low-rank optimization problem. The key idea is to assign data points to clusters to meet the **correlation** and **closeness** criteria. Specifically, if we examine an arbitrarily oriented subspace cluster w.r.t. its relevant dimensions, we find these cluster objects exhibit a high correlation. Meanwhile, when projecting these cluster objects into the orthogonal complementary space spanned by the irrelevant dimensions, they exhibit a high closeness. We argue that taking both of the correlation and closeness into account improves the robustness. Motivated by the observations, in this paper, we propose a new subspace clustering algorithm called RAOSC, which formulates the Robust Arbitrarily Oriented Subspace Clustering as a group structure low-rank optimization problem. It has several attractive properties. Firstly, since the optimization does not rely on the local search, it is more efficient. Furthermore, the optimization problem well characterizes the two criteria of correlation and closeness simultaneously where most previous methods only consider the correlation criterion. This makes RAOSC more robust to noisy objects or outliers. Inspired by [17], we develop an effective and efficient optimization algorithm to solve RAOSC. During the optimization process, noise and overlapping points can be naturally identified. Last but not least, unlike the previous low-rank representation (LRR) based subspace clustering methods [12,13] that cannot give explicit subspaces of clusters and need a two-step algorithm to do clustering, RAOSC is able to find explicit subspace for each cluster without knowing the subspace's dimensionality. It directly obtains the discrete cluster membership indicators by the optimization, no further post-processing procedure is needed. In summary, the main contributions of our work are listed as follows.

- We formulate the identification of arbitrarily oriented subspace clustering as an optimization problem by exploiting two intrinsic properties of a subspace cluster: **correlation** and **closeness**. We integrate the two properties and formulate a group structure low-rank model for subspace clustering.
- We develop an effective and efficient optimization algorithm for RAOSC. During the optimization process, noise and overlapping points can be naturally identified. To the best of our knowledge, for arbitrarily oriented subspace clustering problem, we are the first to handle both noise and overlapping points in one unified framework.
- We perform extensive experiments on synthetic and real-world data sets and compare with the state-of-the-art algorithms to demonstrate the effectiveness of our approach.

2 The Proposed Method

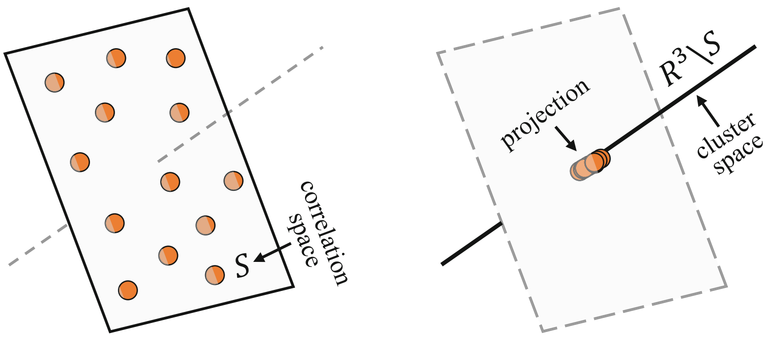
2.1 Problem Formulation

Formally, let $\mathbf{X} \in \mathbb{R}^{m \times n}$ be a data matrix of n instances with m dimensions, the objective of this study is to find k overlapping arbitrarily oriented subspace clusters $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k\}$, and a noise point set $\mathcal{C}_0$. $\mathbf{X}_i$ is a matrix containing data points in cluster $\mathcal{C}_i$. For each cluster, we use a diagonal indicator matrix $\mathbf{P}_i \in \{0,1\}^{m \times m}$ to indicate the relevant dimensions in the original space, and use $\bar{\mathbf{P}}_i = \mathbf{I} - \mathbf{P}_i$ to indicate the irrelevant dimensions, where $\mathbf{I}$ denotes an identity matrix. Specifically, $\mathbf{P}_i^T \mathbf{X}_i$ sets the rows of $\mathbf{X}_i$ corresponding to the irrelevant dimensions to zero, and leaves the rest of the rows corresponding to the relevant dimensions untouched, which extracts the axis-parallel relevant dimensions of $\mathbf{X}_i$. Since subspace clusters may accommodate in arbitrarily oriented subspaces, thereby, inspired by [14], an orthonormal rigid rotation matrix $\mathbf{S}_i \in \mathbb{R}^{m \times m}$ is further introduced. $\mathbf{S}_i$ rotates the i-th cluster so that its relevant dimensions align to the parallel axes. Combine these two matrices, $\mathbf{P}_i^T \mathbf{S}_i^T \mathbf{X}_i$ is thus used to characterize the relevant subspace for the cluster $\mathcal{C}_i$. Finally, we use a diagonal matrix $\mathbf{G}_i \in \{0,1\}^{n \times n}$ to indicate the corresponding cluster membership, where $\mathbf{G}_i(j,j) = 1$ if the j-th data point is grouped into the i-th subspace cluster, and 0 otherwise. Thus $\mathbf{X}\mathbf{G}_i$ leaves the columns corresponding to the i-th cluster points untouched and sets the others to zero. Therefore, for a given data set, the subspace clustering is conducted by learning the three matrices for each cluster.

2.2 Clustering via Correlation and Closeness

In this study, we consider that a subspace cluster should satisfy two criteria: **correlation** and **closeness**. Specifically, we first refer to the subspace spanned by the relevant dimensions as the correlation space, the subspace spanned by the irrelevant dimensions as the cluster space. Note the two subspaces are orthogonal complementary to each other w.r.t. the full space. Data points in a subspace cluster should show high correlation in the correlation space. Meanwhile, they should be as close as possible when projecting them into the cluster space. By contrast, most existing approaches only consider the correlation for subspace clustering. To illustrate this basic idea, Fig. 2 gives a toy example. Figure 2(a) shows that data points in a subspace cluster should locate near an arbitrarily oriented 2D plane in the full 3D space (i.e., strong correlation). In addition, data points are close to each other when projecting them into the cluster space (i.e., high closeness) (see Fig. 2(b)). In the following, we will formulate our objective function in terms of the two criteria.

For the i-th cluster, we assume that we can find an orthonormal rigid rotation matrix $\mathbf{S}_i$ [14], which rotates the original space and thus the first d_i dimensions span the correlation space accommodating all data points in the i-th cluster (e.g., S in Fig. 2). And the last $(m - d_i)$ dimensions span the cluster space (e.g., $R^3 \backslash S$ in Fig. 2). Note that the dimensionality d_i is not a parameter that needs to be manually set. It is automatically determined in the optimization process.



(a) High correlation. (b) High closeness.

Fig. 2. Illustration of subspace clustering with criteria of correlation and closeness in correlation space $\mathcal{S}$ and cluster space $R^3\backslash\mathcal{S}$.

Accordingly, we use two diagonal indicator matrices $\mathbf{P}_i$ and $\bar{\mathbf{P}}_i$ defined as follows to split the two subspaces.

$$\mathbf{P}_i = \begin{bmatrix} \mathbf{I}_{d_i} & \\ & \mathbf{0}_{m-d_i} \end{bmatrix}, \bar{\mathbf{P}}_i = \begin{bmatrix} \mathbf{0}_{d_i} & \\ & \mathbf{I}_{m-d_i} \end{bmatrix}. \tag{1}$$

A data point $\mathbf{x}$ can project into the correlation space and the cluster space by $\mathbf{P}_i^T\mathbf{S}_i^T\mathbf{x}$ and $\bar{\mathbf{P}}_i^T\mathbf{S}_i^T\mathbf{x}$, respectively.

Since we require closeness of a cluster in the cluster space, one intuitive option is to minimize the pairwise distance of all cluster points, which can be formulated as $\sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}_i} ||\bar{\mathbf{P}}_i^T\mathbf{S}_i^T\mathbf{x}-\bar{\mathbf{P}}_i^T\mathbf{S}_i^T\mathbf{y}||$. To characterize the correlation in the correlation space, we consider a group structure low-rank model, i.e., a group of data points in the same cluster are low-rank in the full-dimensional space. We can minimize the ranks of clusters to find subspace clusters. That is, $\min_{\mathcal{C}} \sum_{i=1}^k rank(\mathbf{X}_i)^2$. Since multiplying an orthogonal matrix, and discarding the irrelevant dimensions of a subspace cluster in the full-dimensional space do not change the rank, we can equivalently write it as $\min_{\mathbf{P},\mathbf{S},\mathcal{C}} \sum_{i=1}^k rank(\mathbf{P}_i^T\mathbf{S}_i^T\mathbf{X}_i)^2$. Note the square on the rank function is used to avoid trivial solution as discussed in [17]. We formulate the problem as the following objective function.

$$\begin{aligned} &\min_{\mathbf{P},\mathbf{S},\mathcal{C}} \sum_{i=1}^{k} \Big(rank(\mathbf{P}_i^T\mathbf{S}_i^T\mathbf{X}_i)^2 + \sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}_i} ||\bar{\mathbf{P}}_i^T\mathbf{S}_i^T(\mathbf{x}-\mathbf{y})|| \Big) \\ &s.t.\ \forall\, 1 \le i \le k,\ \ \mathbf{S}_i^T\mathbf{S}_i = \mathbf{I}. \end{aligned} \tag{2}$$

We use an alternative diagonal indicator matrix $\mathbf{G}_i$ to rewrite Eq. (2) as follows.

$$\begin{aligned}
&\min_{\mathbf{P},\mathbf{S},\mathbf{G}} \sum_{i=1}^{k} \Big(rank(\mathbf{P}_i^T\mathbf{S}_i^T\mathbf{X}\mathbf{G}_i)^2 + \sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}_i} ||\bar{\mathbf{P}}_i^T\mathbf{S}_i^T(\mathbf{x}-\mathbf{y})|| \Big) \\
&s.t. \sum_{i=1}^{k}\mathbf{G}_i \succeq \mathbf{I}, \sum_{i=1}^{k} Tr(\mathbf{G}_i) = kl, \\
&\forall\, 1 \le i \le k, \;\; \mathbf{G}_i \subseteq \{0,1\}^{n\times n}, \mathbf{S}_i^T\mathbf{S}_i = \mathbf{I},
\end{aligned} \tag{3}$$

where the term $\sum_{i=1}^{k}\mathbf{G}_i \succeq \mathbf{I}$ and $\sum_{i=1}^{k} Tr(\mathbf{G}_i) = kl$ ensure that each data point has to be assigned to at least one group, and some data points can be assigned to multiple groups to allow overlapping clustering. l is an integer parameter that controls the degree of overlapping, which is within the range of $[n/k, n]$. $l = n/k$ means no overlapping clusters, $l = n$ is complete overlapping (i.e., each data point belongs to all clusters). Since l depends on the number of data points so that it is not convenient to set, we use a variable substitution trick which replaces l with $\tilde{l} \in [0, 1]$. It stands for the proportion of the overlapping data points, where $l = \left\lfloor \frac{\tilde{l}n(k-1)+n}{k} \right\rfloor$.

Since the rank minimization problem is NP-hard, the common practice is to relax the rank function to the nuclear norm. However, inspired by [17], we use the Schatten-1 norm for relaxation. Schatten-1 norm is numerically equal to the nuclear norm, but it can be more efficiently optimized by adopting certain strategy. In addition, finding an optimal cluster assignment on the minimization of all pairwise distances (i.e., $\sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}_i} ||\bar{\mathbf{P}}_i^T\mathbf{S}_i^T\mathbf{x} - \bar{\mathbf{P}}_i^T\mathbf{S}_i^T\mathbf{y}||$) is also computationally infeasible. We relax it by minimizing distances between data points and cluster centroids. Thereby, the second term can be relaxed as a k-means style term. The overall objective function is given as follows.

$$\begin{aligned}
&\min_{\mathbf{P},\mathbf{S},\mathbf{G},\mathbf{M}} \sum_{i=1}^{k} \Big((||\mathbf{P}_i^T\mathbf{S}_i^T(\mathbf{X}-\mathbf{M}_i)\mathbf{G}_i||_{Sp}^{p})^2 \\
&+ \alpha||\bar{\mathbf{P}}_i^T\mathbf{S}_i^T(\mathbf{X}-\mathbf{M}_i)\mathbf{G}_i||_F^2 \Big) \\
&s.t. \sum_{i=1}^{k}\mathbf{G}_i \succeq \mathbf{I}, \sum_{i=1}^{k} Tr(\mathbf{G}_i) = kl, \\
&\forall\, 1 \le i \le k, \;\; \mathbf{G}_i \subseteq \{0,1\}^{n\times n}, \mathbf{S}_i^T\mathbf{S}_i = \mathbf{I},
\end{aligned} \tag{4}$$

where $||\mathbf{X}||_{Sp}^{p} = Tr((\mathbf{X}\mathbf{X}^T)^{\frac{p}{2}})$ is the Schatten-p norm of matrix $\mathbf{X}$, and we take $p = 1$ in this study. $\mathbf{M}_i = \boldsymbol{\mu}_i\mathbf{1}_n^T$, $\boldsymbol{\mu}_i$ is the mean vector of the i-the cluster. Using $\mathbf{X} - \mathbf{M}_i$ to replace $\mathbf{X}$ at the first term is to make the rank approximation more robust [17]. $\alpha > 0$ is a real number parameter to balance the dimensionality of the correlation space and the cluster space, which will be discussed later.

2.3 Optimization Algorithm

To solve the optimization problem, we use an iterative algorithm to update $\mathbf{P}_i$, $\mathbf{S}_i$, $\mathbf{G}_i$ and $\mathbf{M}_i$ one at a time while fixing the others to achieve a local optimum.

Updating G. Let $\mathcal{O}$ denote the objective function Eq. (4), we write the Lagrange function of Eq. (4) w.r.t. variables $\mathbf{G}$ as follows.

$$L(\mathbf{G}, \mathit{\Lambda}) = \mathcal{O}(\mathbf{G}) + g(\mathit{\Lambda}, \mathbf{G}), \tag{5}$$

where $\mathit{\Lambda}$ is the Lagrange dual variable, $g(\mathit{\Lambda}, \mathbf{G})$ encodes the constraints on $\mathbf{G}$ in problem Eq. (4).

By taking the derivative of Eq. (5) w.r.t. $\mathbf{G}$, and setting it to zero, we have:

$$\sum_{i=1}^{k} 2\mathbf{A}_i\mathbf{G}_i + \frac{\partial g(\mathit{\Lambda}, \mathbf{G}_i)}{\partial \mathbf{G}_i} = 0, \tag{6}$$

where $\mathbf{A}_i$ and $\mathbf{B}_i$ are:

$$\mathbf{A}_i = \tilde{\mathbf{X}}_i^T\mathbf{S}_i\mathbf{P}_i\mathbf{B}_i\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i + \alpha\tilde{\mathbf{X}}_i^T\mathbf{S}_i\bar{\mathbf{P}}_i^2\mathbf{S}_i^T\tilde{\mathbf{X}}_i, \tag{7}$$

$$\mathbf{B}_i = p\big|\big|\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i\mathbf{G}_i\big|\big|_{Sp}^{p}(\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i\mathbf{G}_i^2\tilde{\mathbf{X}}_i^T\mathbf{S}_i\mathbf{P}_i)^{\frac{p-2}{2}}. \tag{8}$$

$\tilde{\mathbf{X}}_i = \mathbf{X} - \mathbf{M}_i$. $\mathbf{A}_i$ and $\mathbf{B}_i$ depend on $\mathbf{G}_i$. It can be solved via an iteration based re-weighted algorithm [16,17]. We first calculate $\mathbf{A}_i$ and $\mathbf{B}_i$ based on the current $\mathbf{G}_i$. After $\mathbf{A}_i$ and $\mathbf{B}_i$ are fixed and treated as constants, we can solve the following problem which satisfies Eq. (6) to update $\mathbf{G}_i$.

$$\begin{aligned} &\min_{\mathbf{G}} \sum_{i=1}^{k} Tr(\mathbf{G}_i^T\mathbf{A}_i\mathbf{G}_i) \\ &s.t. \sum_{i=1}^{k}\mathbf{G}_i \succeq \mathbf{I}, \sum_{i=1}^{k} Tr(\mathbf{G}_i) = kl, \\ &\forall\, 1 \le i \le k,\ \ \mathbf{G}_i \subseteq \{0,1\}^{n\times n}. \end{aligned} \tag{9}$$

Due to the diagonality and the discrete constraints of $\mathbf{G}_i$, Eq. (9) can be equivalently written as:

$$\begin{aligned} &\min_{g} \sum_{i=1}^{k}\sum_{j=1}^{n} a_{ij}g_{ij} \\ &s.t. \sum_{i=1}^{k} g_{ij} \ge 1, \sum_{i=1}^{k}\sum_{j=1}^{n} g_{ij} = kl, \\ &\forall\, 1 \le i \le k,\ 1 \le j \le n,\ g_{ij} \in \{0,1\}, \end{aligned} \tag{10}$$

where g_{ij} is the j-th diagonal element of matrix $\mathbf{G}_i$, a_{ij} is the j-th diagonal element of matrix $\mathbf{A}_i$. The objective function above derives a 0-1 integer programming problem, which is usually NP-hard. Fortunately, the constraints imposed on g_{ij} significantly reduce the searching space, we can still obtain an efficient algorithm to solve this problem. Firstly, for every data point, we assign cluster i which has the minimum a_{ij} value to the j-th data point. This ensures every data point has been assigned to one cluster. Afterwards, we deal with the remaining $(kl - n)$ overlapping data points. We sort the remaining a_{ij} in an ascending order, then record the first $(kl - n)$ subscripts (ij) of a_{ij} and set the corresponding $g_{ij} = 1$.

Meanwhile, noise can be naturally identified during the cluster membership assigning process. We additionally let $\mathbf{G}_0$ denote the noise indicator matrix, $\mathbf{G}_0(j, j) = 1$ if data point j is a noise, otherwise 0. Given the current cluster indicator vector $\mathbf{g}_i$ for the i-th cluster, we first collect all a_{ij} which satisfy $g_{ij} = 1$, then find their median m_i. A data point j in the i-the cluster is a noise, if $a_{ij} > \lambda m_i$, where $\lambda > 0$ is a real number parameter. The rationale is, if we closely look into a_{ij}, it consists of two parts: the first part can be regarded as a weighted Mahalanobis-like distance between $\mathbf{x}_j$ and the group mean of the i-th cluster in the correlation space. If data point j has been assigned to the i-the subspace cluster but deviates from the principal component directions of the i-th subspace cluster (i.e., deviates from the plane where the subspace cluster is lying in), it tends to produce a large value of a_{ij}. The second part represents the Euclidean distance between $\mathbf{x}_j$ and the group mean in the cluster space. Noise points are far away from the cluster center in the cluster space, which also produce large a_{ij}. In summary, a noise point can be pinpointed by checking the anomaly large a_{ij}. Note that the overlapping points are identified by fulfilling the constraint of $\sum_{i,j} g_{ij} = kl$. We need to recalculate $l = \left\lfloor \frac{\tilde{l}(n - Tr(\mathbf{G}_0))(k-1) + (n - Tr(\mathbf{G}_0))}{k} \right\rfloor$ at each iteration otherwise noise points will be assigned to the overlapping clusters. We summarize the algorithm to solve problem Eq. (10) in Algorithm 1.

Updating S and P. Adopting the similar derivation of updating $\mathbf{G}_i$, we can use the iteration based re-weighted method to solve $\mathbf{S}_i$ by optimizing the following objective function.

$$\begin{aligned} &\min_{\mathbf{S}} \sum_{i=1}^{k} Tr(\mathbf{S}_i^T \mathbf{C}_i \mathbf{S}_i \mathbf{P}_i \mathbf{P}_i^T) + Tr(\mathbf{S}_i^T \mathbf{D}_i \mathbf{S}_i \bar{\mathbf{P}}_i \bar{\mathbf{P}}_i^T) \\ &s.t. \ \forall\, 1 \le i \le k, \ \ \mathbf{S}_i^T \mathbf{S}_i = \mathbf{I}, \end{aligned} \tag{11}$$

where $\mathbf{C}_i$, $\mathbf{D}_i$ and $\mathbf{E}_i$ are:

$$\mathbf{C}_i = \tilde{\mathbf{X}}_i \mathbf{G}_i \mathbf{E}_i \mathbf{G}_i^T \tilde{\mathbf{X}}_i^T, \tag{12}$$

$$\mathbf{D}_i = \alpha \tilde{\mathbf{X}}_i \mathbf{G}_i^2 \tilde{\mathbf{X}}_i^T, \tag{13}$$

$$\mathbf{E}_i = p \left|\left| \mathbf{G}_i^T \tilde{\mathbf{X}}_i^T \mathbf{S}_i \mathbf{P}_i \right|\right|_{Sp}^p (\mathbf{G}_i^T \tilde{\mathbf{X}}_i^T \mathbf{S}_i \mathbf{P}_i^2 \mathbf{S}_i^T \tilde{\mathbf{X}}_i \mathbf{G}_i)^{\frac{p-2}{2}}. \tag{14}$$

Note that $\mathbf{P}_i \mathbf{P}_i^T$ leaves the upper left $d_i \times d_i$ matrix untouched and sets the other elements to zero, thus the value of $Tr(\mathbf{S}_i^T \mathbf{C}_i \mathbf{S}_i \mathbf{P}_i \mathbf{P}_i^T)$ is equal to the summation of the eigenvalues of that upper left matrix. It is similar for $\bar{\mathbf{P}}_i \bar{\mathbf{P}}_i^T$.

To solve problem Eq. (11), we first calculate the eigenvalues of $\mathbf{C}_i$ and $\mathbf{D}_i$ and put them in an array as $\boldsymbol{\delta}_i = \{\delta_{Ci}^{(1)}, \ldots, \delta_{Ci}^{(m)}, \delta_{Di}^{(1)}, \ldots, \delta_{Di}^{(m)}\}$, then sort $\boldsymbol{\delta}_i$ in

Algorithm 1. Algorithm to solve problem Eq. (10).

```
Input:
    Variable $\mathbf{A}_i (1 \leq i \leq k)$, number of clusters $k$, parameters $\tilde{l}$, $\lambda$.
Output:
    Cluster and noise indicators $\mathbf{G}_i (0 \leq i \leq k)$.
1: Initialize $\mathbf{G}_0 = \mathbf{0}_{n \times n}$.
2: for $i = 1$ to $k$ do
3:     // Make sure every point has been assign to one cluster.
4:     for $j = 1$ to $n$ do
5:         $\mathbf{G}_i(j,j) = \begin{cases} 1, & i = \underset{i}{\operatorname{argmin}}\, \mathbf{A}_i(j,j) \\ 0, & \text{otherwise} \end{cases}$
6:     end for
7:     // Handle the noise points.
8:     $\mathcal{J} = \{j \mid \mathbf{G}_i(j,j) = 1\}$.
9:     $m_i$ = median of $\mathbf{A}_i(\mathcal{J}, \mathcal{J})$.
10:    for $j$ in $\mathcal{J}$ do
11:        if $\mathbf{A}_i(j,j) > \lambda \times m_i$ then
12:            $\mathbf{G}_0(j,j) = 1$.
13:            $\mathbf{G}_i(j,j) = 0$.
14:        end if
15:        $\mathbf{A}_i(j,j) = \text{Inf}$.
16:    end for
17: end for
18: // Handle the overlapping points.
19: $l = \left\lfloor \frac{\tilde{l}(n - Tr(\mathbf{G}_0))(k-1) + (n - Tr(\mathbf{G}_0))}{k} \right\rfloor$.
20: $\mathcal{I} = \{(i,j) \mid$ sort $\mathbf{A}_i(j,j)$ in an ascending order then leave the first $kl - (n - Tr(\mathbf{G}_0))$
    entries$\}$.
21: $\mathbf{G}_i(j,j) = 1$, if $(i,j) \in \mathcal{I}$.
```

an ascending order. Without loss of generality, we assume that d_i eigenvalues of $\mathbf{C}_i$ and $m - d_i$ eigenvalues of $\mathbf{D}_i$ are in the m-smallest set of $\boldsymbol{\delta}_i$. Then we permute the m-smallest set so that the first d_i and the last $m - d_i$ entries are the eigenvalues of $\mathbf{C}_i$ and $\mathbf{D}_i$, respectively. The optimal solution of problem Eq. (11) can be obtained by putting d_i eigenvectors of $\mathbf{C}_i$ and $m - d_i$ eigenvectors of $\mathbf{D}_i$ corresponding to their smallest eigenvalues into $\mathbf{S}_i$'s columns. Note that d_i is automatically determined by the sorting rather than a parameter. Then we can update $\mathbf{P}_i$ and $\bar{\mathbf{P}}_i$ by using Eq. (1).

Updating M. $\mathbf{M}_i$ is a matrix whose columns are identical, every column is the mean vector of the i-th cluster. Thus the actual variable that needs to be solved is $\boldsymbol{\mu}_i$. We solve the following problem to update $\mathbf{M}_i$.

$$\begin{aligned} &\min_{\boldsymbol{\mu}} \sum_{i=1}^{k} Tr(\mathbf{G}_i^T (\mathbf{X} - \mathbf{M}_i)^T \mathbf{F}_i (\mathbf{X} - \mathbf{M}_i) \mathbf{G}_i) \\ &s.t.\ \forall\, 1 \leq i \leq k,\ \ \mathbf{M}_i = \boldsymbol{\mu}_i \mathbf{1}_n^T, \end{aligned} \tag{15}$$

Algorithm 2. Algorithm to solve RAOSC

```
Input:
    Data matrix $\mathbf{X}$, the number of clusters $k$, parameters $\tilde{l}$, $\lambda$, $\alpha$.
Output:
    Cluster and noise indicators $\mathbf{G}_i (0 \leq i \leq k)$, subspace indicators $\mathbf{P}_i (1 \leq i \leq k)$ and
    the orthonormal rigid rotation matrix $\mathbf{S}_i (1 \leq i \leq k)$.
 1: Initialize all $\mathbf{G}_i$ such the constraints in Eq. (10) are satisfied. $\mathbf{S}_i$, $\mathbf{P}_i$, $\bar{\mathbf{P}}_i = \mathbf{I}$.
    $\mathbf{M}_i = \mathbf{0}$.
 2: repeat
 3:     Calculate $\tilde{\mathbf{X}}_i|_{i=1}^k = \mathbf{X} - \mathbf{M}_i$.
 4:     Calculate $\mathbf{A}_i|_{i=1}^k$ and $\mathbf{B}_i|_{i=1}^k$ using Eq. (7-8).
 5:     Update $\mathbf{G}_i|_{i=0}^k$ using Algorithm 1.
 6:     Calculate $\mathbf{C}_i|_{i=1}^k$, $\mathbf{D}_i|_{i=1}^k$, $\mathbf{E}_i|_{i=1}^k$ using Eq. (12-14).
 7:     Update $\mathbf{S}_i|_{i=1}^k$ by putting $m$ eigenvectors of $\mathbf{C}_i|_{i=1}^k$ or $\mathbf{D}_i|_{i=1}^k$ corresponding to the
        $m$-smallest eigenvalues in $\boldsymbol{\delta}_i|_{i=1}^k = \{\delta_{Ci}^{(1)}, ..., \delta_{Ci}^{(m)}, \delta_{Di}^{(1)}, ..., \delta_{Di}^{(m)}\}$ into its columns.
 8:     Permute columns of $\mathbf{S}_i|_{i=1}^k$ so that the first $d_i$ and the last $m - d_i$ columns are
        the eigenvectors of $\mathbf{C}_i|_{i=1}^k$ and $\mathbf{D}_i|_{i=1}^k$, respectively.
 9:     Update $\mathbf{P}_i|_{i=1}^k$ and $\bar{\mathbf{P}}_i|_{i=1}^k$ using Eq. (1).
10:     Update $\mathbf{M}_i|_{i=1}^k$ using Eq. (16).
11: until convergence or max no. iterations reached.
```

where $\mathbf{F}_i = \mathbf{S}_i\mathbf{P}_i\mathbf{B}_i\mathbf{P}_i^T\mathbf{S}_i^T + \alpha\mathbf{S}_i\bar{\mathbf{P}}_i^2\mathbf{S}_i^T$, though it is irrelevant for updating $\mathbf{M}_i$. By substituting the variable and calculating the derivative w.r.t. $\boldsymbol{\mu}_i$ and setting it to zero, it is easy to obtain the update rule of $\mathbf{M}_i$ as follows.

$$\mathbf{M}_i = \frac{1}{Tr(\mathbf{G}_i)}\mathbf{X}\mathbf{G}_i\mathbf{1}_n\mathbf{1}_n^T. \tag{16}$$

It can be seen that updating $\mathbf{M}_i$ is just simply calculating the mean of a cluster in the original space.

Finally, we summarize the overall optimization procedure in Algorithm 2.

2.4 Relationship to Existing Clustering Paradigms

For α, when setting it to a relative small value, step 7 in Algorithm 2 will put all $\mathbf{D}_i$'s eigenvectors into $\mathbf{S}_i$'s columns, so that the correlation space is vanished ($\mathbf{P}_i = \mathbf{0}$) and the cluster space gains full dimensionality ($\bar{\mathbf{P}}_i = \mathbf{I}$). This yields the problem Eq. (4) without the first term, which is the ordinary k-means algorithm (suppose no overlapping or noise). Similarly, if we set α to a large value, it yields the problem Eq. (4) without the second term, which degenerates to a generalized version of the LRS model [17]. When setting α to a medium value, it balances the dimensionality between the correlation space and the cluster space, thus the correlation and closeness of a cluster are both taken into consideration.

2.5 Time Complexity

Without loss of generality, we assume $m < n$ in the following analysis. The computational bottleneck of RAOSC lies in the SVD decomposition at step 4, 6 and 7 of Algorithm 2. In step 4, computing $\mathbf{B}_i$ needs to compute $||\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i\mathbf{G}_i||_{Sp}^p$ and $(\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i\mathbf{G}_i^2\tilde{\mathbf{X}}_i^T\mathbf{S}_i\mathbf{P}_i)^{\frac{p-2}{2}}$, both rely on the SVD of $\mathbf{P}_i^T\mathbf{S}_i^T\tilde{\mathbf{X}}_i\mathbf{G}_i$, which costs $O(m^2n)$. Thus computing all $\mathbf{B}_i$ costs $O(m^2nk)$. Computing $\mathbf{A}_i$ costs $O(m^2nk)$. Thus step 4 costs $O(m^2nk)$. Similar to step 4, step 6 costs $O(mn^2k)$. Step 7 needs to compute SVD of $\mathbf{C}_i$ and $\mathbf{D}_i$, which costs $O(m^3k)$. In summary, the time complexity of Algorithm 2 is $O((mn^2 + m^2n + m^3)kt)$, where k is usually a small constant that can be ignored, t is the number of iterations. Usually, the algorithm converges in a few iterations, e.g., 50 iterations.

3 Experiments

In this section, we evaluate our method with respect to its clustering results on both synthetic data and real-world data. We start with the synthetic data to show a proof-of-concept of finding arbitrarily oriented subspace clusters in the presence of noise and overlapping points. Afterwards, we compare our method with six state-of-the-art algorithms on nine real-world data sets obtained from the UCI and UCR repositories. For real-world data, we have no prior knowledge about the noise nor overlapping, and most of the typical comparison algorithms cannot handle such case. So we only perform clustering on real-world data sets with an assumption that there are no noise or overlapping points, since we have demonstrated it on the synthetic data.

3.1 Evaluation on Synthetic Data

We start with the synthetic data to demonstrate the effectiveness of finding arbitrarily oriented subspace clusters in the presence of noise and overlapping points. Here three synthetic data sets are generated. In details, synthetic data 1 consists of three subspace clusters in 3D space, where one of the subspace cluster forms a 2D plane and the other two subspace clusters form two cross lines passing through the plane's origin. Each subspace cluster contains 500 points with 5% level perturbation added to deviate from the subspace. Synthetic data 2 consists of two subspace clusters forming two perpendicular 2D planes in 3D space, each subspace cluster contains 500 uniformly distributed points with 5% level perturbation added. Synthetic data 3 consists of two perpendicular subspace clusters in 2D space. Each subspace cluster contains 500 uniformly distributed points forming a long and narrow rectangle shape. Note that synthetic data 2 and 3 have approximate 10% overlapping points. Finally, we add uniform noise points into all synthetic data set with noise level ranging from 0.1 to 0.8.

For comparison, we select three state-of-the-art subspace clustering algorithms and compare the clustering performance while varying the noise level. The comparison algorithms are NrKmeans [15], ISAAC [22] and ORSC [19]. The reason we select these algorithms is that they can handle noise and/or overlapping to a certain extent. We select the parameters according to the true statistics of the data and from a wide tuning range to obtain the best result. For numerical evaluation, we use the pair-counting F1-measure [1], which is commonly used when encountering overlapping clustering.

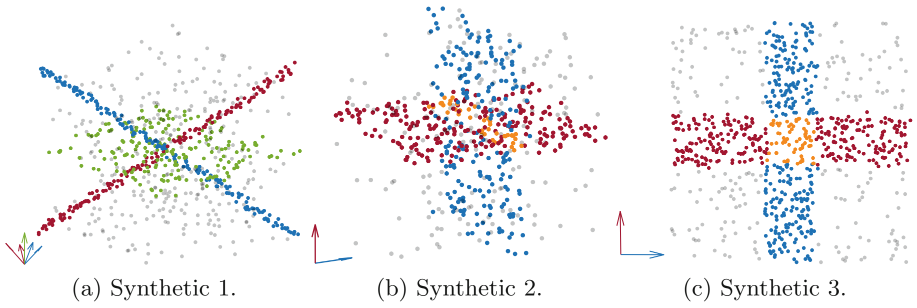


Fig. 3. Visualization of the clustering results of RAOSC on synthetic data sets. Colored points are the found subspace clusters. The orange points are the found overlapping points. Noise are plotted with gray points. Arrows at the bottom-left represent the found subspaces of the corresponding clusters. (Color figure online)

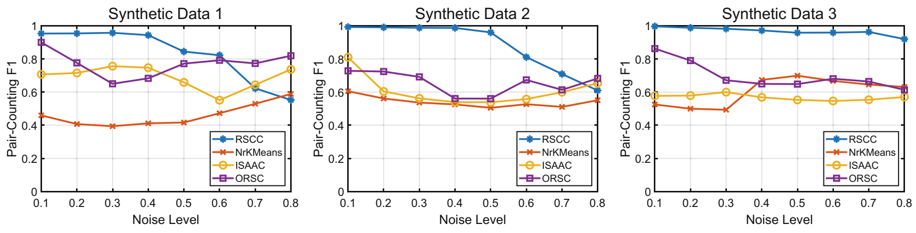


Fig. 4. Clustering results while varying noise levels on synthetic data sets in the presence of noise and overlapping points.

Figure 3 visualizes the clustering results of RAOSC. Note that we only plot the low noise level (about 0.3) results for legibility reason, though it can correctly find the subspace clusters and identify noise points at a higher level of noise. As we can see, RAOSC successfully assigns data points into the correct clusters in a sea of noise, and the noise and overlapping points are all correctly identified. Besides, it obtains the corresponding subspaces as well (indicated by the arrows at the bottom left corner). Figure 4 shows the numerical evaluation of the comparison algorithms. RAOSC achieves promising results and outperforms other algorithms in most cases. When the noise level is extremely high, the performance drops sharply, because the extreme noise points have covered the cluster structure and meaningful clusters no longer exist. We observe that the performance of comparison algorithms drop down at first and then go up. This might be because they are mild arbitrators. With the increase of noise level, they tend to assign noise into multiple overlapping clusters. This can still increase the score w.r.t. the pair-counting F1-measure though they actually produce the wrong assignment.

3.2 Evaluation on Real-World Data

Next we compare with extensive clustering algorithms on the real-world data sets. Nine real-world data sets are used in this study, which include Pendigits, Seeds, Soybean, Spam, Wine and Zoo from the UCI repository, OliveOil, Plane and Symbols from the UCR repository. The statistics of these data sets are given in Table 1.

Table 1. Statistics of the real-world data sets.

Name	#Classes	#Dim.	#Inst.	Name	#Classes	#Dim.	#Inst.
Wine	3	13	178	Seeds	3	7	210
Pendigits	10	16	10092	Soybean	4	35	47
Zoo	7	16	101	Spam	2	57	4601
Plane	7	144	210	Symbol	6	398	1020
Olive	4	570	60				

Here, six arbitrarily oriented subspace clustering algorithms are selected. ORCLUS [3] and 4C [6] are the two most typical arbitrarily oriented subspace clustering algorithms. SubKmeans [14] and FOSSCLU [8] are two recent subspace clustering algorithms. Different from ORCLUS and 4C, they only find one optimal subspace for clustering. LRR [12] and LRS [17] are two low-rank based subspace clustering algorithm. Though they are not considered as the classical subspace clustering algorithm in the data mining community, they are still closely related to our method. Source codes of all algorithms are downloaded from the authors' websites. The source code of RAOSC can be downloaded from Dropbox[2].

For a comprehensive evaluation, we tune all the algorithms' parameters from wide ranges while being compatible with the original papers. We search LRR's parameter λ within the set of $\{10^{-5}, 10^{-4}, \dots, 10^{5}\}$. For LRS, we select its parameter p from $\{0.1, 0.2, \dots, 1\}$, and set $K = 2$. We use PCA and k-means to do initialization as described in its paper. We search ORCLUS's parameter l in the range of $[2; \min(20, m)]$, and run 4C for $\epsilon \in [2; 20]$, $minPts \in [1; 15]$ and $\lambda \in [2; \min(20, m)]$. SubKmeans has no additional parameters except for k. FOSSCLU determines parameters automatically. For RAOSC, we search parameter α from $\{10^{-5}, 10^{-4}, \dots, 10^{5}\}$. In addition, we set parameter l to zero, λ to a large number, which gives no overlapping nor noise result. Similar to LRS, we use PCA and k-means to initialize the cluster indicator matrices and the cluster centroids. For all experiments, we standardize all data so that all features have zero mean and unit variance. Since some algorithms may run into a local optimum and produce insufficient outcomes, we run all algorithms for 10 times and

[2] https://www.dropbox.com/s/7csm3itojmb5glh/RAOSC_code.rar?dl=0.

Table 2. Clustering results in terms of NMI (%) on the real-world data sets. Results marked with † were aborted due to memory limit or convergence issues, or cannot obtain results in reasonable running time. In either case, we report the best results that have been achieved.

	Pendigits	OliveOil	Seeds	Soybean	Spam	Symbol	Wine	Zoo	Plane
RAOSC	**73.68**	**76.30**	**73.69**	**100.00**	**41.05**	**81.56**	**88.26**	**89.11**	91.23
SubKmeans	67.94	75.42	72.79	**100.00**	2.17	79.48	87.59	83.39	91.23
ORCLUS	68.32	75.86	72.69	96.89	36.13	65.60†	87.79	87.33	70.84
4C	69.99	63.96	16.50	**100.00**	11.78	81.41	48.11	85.48	88.08
FOSSCLU	70.20	0.00†	63.95	37.02	0.00†	0.00†	84.68	0.00†	0.00†
LRS	65.13	44.16	73.30	69.10	27.04	60.72	63.22	67.03	57.37
LRR	70.85	40.12	21.58	81.49	4.67	78.09	41.36	76.02	**93.29**

sort the results by their costs and remove the half with higher costs. Then we report the average NMI for evaluation.

Table 2 summarizes the clustering results. As shown in Table 2, SubKmeans, ORCLUS, and 4C achieve comparable results. However, SubKmeans only finds one optimal subspace for clustering rather than distinct subspaces for all clusters. ORCLUS and 4C are time consuming especially when the number of dimensions is high. Besides, it is hard to find the optimal parameters for 4C. FOSSCLU also only finds one optimal subspace for clustering, however, there seems to be convergence issue in the author provided implementation. In general, the two low-rank based algorithms LRR and LRS perform poorly on these data sets. Neither LRR nor LRS can give the explicit subspaces of clusters, they all find implicit subspace clusters in the full-dimensional space, where the low-rank structure is seem to be dim. By contrast, RAOSC outperforms other algorithms on eight data sets, only slightly lags behind on the Plane data set. RAOSC naturally characterizes the intrinsic correlation and closeness properties of subspace cluster, which accounts for the promising clustering results.

4 Conclusion

In this paper, towards the arbitrarily oriented subspace clustering problem, we propose a novel algorithm called RAOSC. RAOSC formulates the task as a group structure low-rank optimization problem, which well characterizes the intrinsic correlation and closeness properties of subspace cluster. RAOSC can not only recover the subspace clusters from a sea of noise points but also explicitly obtains the corresponding subspaces. It can naturally identify the noise and overlapping points during the optimization process. Empirical experiments on both synthetic data sets and real-world data sets have demonstrated its effectiveness. In future work, we would like to reduce the computational complexity. One potential route is to incorporate accelerated SVD, another is to develop data parallelism at the algorithmic level [20].

Acknowledgments. This work is supported by the National Natural Science Foundation of China (61403062, 61433014, 41601025), Science-Technology Foundation for Young Scientist of SiChuan Province (2016JQ0007), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (161062) and National key research and development program (2016YFB0502300).

References

1. Achtert, E., Goldhofer, S., Kriegel, H.P., Schubert, E., Zimek, A.: Evaluation of clusterings–metrics and visual support. In: Proceedings of the 28th IEEE International Conference on Data Engineering, pp. 1285–1288 (2012)
2. Aggarwal, C.C., Wolf, J.L., Yu, P.S., Procopiuc, C., Park, J.S.: Fast algorithms for projected clustering. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, vol. 28 (1999)
3. Aggarwal, C.C., Yu, P.S.: Finding generalized projected clusters in high dimensional spaces. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, vol. 29 (2000)
4. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, vol. 27 (1998)
5. Assent, I., Krieger, R., Emmanuel, M., Seidl, T.: DUSC: dimensionality unbiased subspace clustering. In: Proceedings of the 7th IEEE International Conference on Data Mining, pp. 409–414 (2008)
6. Böhm, C., Kailing, K., Kröger, P., Zimek, A.: Computing clusters of correlation connected objects. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, pp. 455–466 (2004)
7. Cheng, C.H., Fu, A.W., Zhang, Y.: Entropy-based subspace clustering for mining numerical data. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 84–93 (1999)
8. Goebl, S., He, X., Plant, C., Böhm, C.: Finding the optimal subspace for clustering. In: Proceedings of the 14th IEEE International Conference on Data Mining, pp. 130–139 (2014)
9. Günnemann, S., Färber, I., Virochsiri, K., Seidl, T.: Subspace correlation clustering: finding locally correlated dimensions in subspace projections of the data. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 352–360 (2012)
10. Kailing, K., Kriegel, H.P., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proceedings of the 2004 SIAM International Conference on Data Mining, pp. 246–256 (2004)
11. Kriegel, H.P., Kröger, P., Zimek, A.: Clustering high-dimensional data: a survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. Knowl. Discov. Data **3**(1), 1 (2009)
12. Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., Ma, Y.: Robust recovery of subspace structures by low-rank representation. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 171–184 (2013)
13. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: Proceedings of the 27th International Conference on Machine Learning, pp. 663–670 (2010)

14. Mautz, D., Ye, W., Plant, C., Böhm, C.: Towards an optimal subspace for k-means. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 365–373 (2017)
15. Mautz, D., Ye, W., Plant, C., Böhm, C.: Discovering non-redundant k-means clusterings in optimal subspaces. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1973–1982 (2018)
16. Nie, F., Yuan, J., Huang, H.: Optimal mean robust principal component analysis. In: Proceedings of the 31st International Conference on International Conference on Machine Learning, vol. 32, pp. 1062–1070 (2014)
17. Nie, F., Huang, H.: Subspace clustering via new low-rank model with discrete group structure constraint. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, pp. 1874–1880 (2016)
18. Shao, J., Gao, C., Zeng, W., Song, J., Yang, Q.: Synchronization-inspired co-clustering and its application to gene expression data. In: 2017 IEEE International Conference on Data Mining, pp. 1075–1080 (2017)
19. Shao, J., Wang, X., Yang, Q., Plant, C., Böhm, C.: Synchronization-based scalable subspace clustering of high-dimensional data. Knowl. Inf. Syst. **52**(1), 83–111 (2017)
20. Shao, J., Yang, Q., Dang, H.V., Schmidt, B., Kramer, S.: Scalable clustering by iterative partitioning and point attractor representation. ACM Trans. Knowl. Discov. Data **11**(1), 5 (2016)
21. Tung, A.K.H., Xu, X., Ooi, B.C.: CURLER: finding and visualizing nonlinear correlation clusters. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, pp. 467–478 (2005)
22. Ye, W., Maurus, S., Hubig, N., Plant, C.: Generalized independent subspace clustering. In: Proceedings of the 2016 IEEE International Conference on Data Mining, pp. 569–578 (2016)