

Greedy orthogonal matching pursuit for subspace clustering to improve graph connectivity



Changchen Zhao^{a,b}, Wen-Liang Hwang^{a,c}, Chun-Liang Lin^{a,*}, Weihai Chen^{b,*}

^a National Chung Hsing University, 250 Kuo Kuang Rd., Taichung 402, Taiwan

^b Beihang University, 37 Xueyuan Rd., Beijing 100191, China

^c Academia Sinica, 128 Academia Rd., Taipei 11529, Taiwan

ARTICLE INFO

Article history:

Received 23 May 2017

Revised 21 February 2018

Accepted 13 May 2018

Available online 16 May 2018

Keywords:

Subspace clustering

Graph connectivity

Greedy OMP

Noisy analysis

Exact clustering guarantee

ABSTRACT

Subspace clustering methods based on self-expressiveness model have recently attracted much attention. However, there exists a gap between subspace-preserving coefficient and the final clustering result due to the lack of graph connectivity. The problem has not been well tackled in the published literature. This paper significantly improves the graph connectivity by adding an effective projection step to the recently proposed method SSC-OMP. With this projection step, it is possible to establish a theoretical guarantee that the subspace-preserving condition leads directly to the exact clustering result, which bridges the gap. Moreover, the potential advantage of the proposed algorithm over prior methods is its robustness to noise. Experimental results demonstrate that the proposed approach enjoys a high clustering accuracy and a fast processing speed in comparison with state-of-the-art algorithms.

© 2018 Elsevier Inc. All rights reserved.

1. Introduction

Subspace clustering is the mathematical abstraction of many computer vision applications in which the data lie on a union of low-dimensional subspaces of a high-dimensional ambient space. The goal is to partition the data into their own subspaces. This problem has found numerous applications including motion/video segmentation [13,40,43,48], face clustering [19,24], image representation and compression [20], document summarization [4], etc. Numerous approaches have been proposed. Early methods include algebraic methods, iterative methods [2], statistical methods [30], and spectral clustering-base methods (please refer to [9,39] for a review of these methods). Among them, the sparse representation based model with spectral clustering [13,45] has become popular in recent years due to their simplicity and robust performance. These methods address the problem in two steps: (1) learning a coefficient matrix using the *self-expressiveness model* based on sparse representation techniques; and (2) constructing a graph and applying spectral clustering on the graph. The self-expressiveness means that each point is assumed to be represented as a linear combination of the points that come from the same subspace and the data from other subspaces are not involved, which is also called “subspace-preserving property” [45,46].

However, there exists a gap between a subspace-preserving coefficient and the exact clustering result, i.e., the correct (subspace-preserving) coefficient does not always lead to correct clustering, known as the graph connectivity problem. This

* Corresponding authors.

E-mail addresses: chunlin@dragon.nchu.edu.tw (C.-L. Lin), whchen@buaa.edu.cn (W. Chen).

is because the points in the same subspace might not be well connected so that multiple connected components might exist in this subspace. As a result, the data tend to be over-segmented. This paper seeks to address this problem by increasing the graph connectivity.

The graph connectivity problem was originally raised and investigated by Nasihatkon and Hartley [31]. They claimed that the points in a single subspace cannot be segmented into smaller subspaces holds for 2- and 3-dimensional subspaces. It is possible to over-segment the data for dimensions higher than 3. A major line of research addressed this problem by imposing regularization term with various norms to the original self-expressiveness model. For example, Lu et al. [29] preferred a least square solution (l_2 norm regularization) by means of least squares regression (LSR). Low rank representation (LRR) [27,28] proposed nuclear norm regularization to encourage the coefficient to be low rank. Regularizations that are a mixture of l_1 and l_2 [14,32,45] or l_1 and nuclear norm [42,49] have also been proposed. The l_2 and nuclear norm regularized algorithms usually have closed form solutions, thus they are more computationally efficient. However, they have strict subspace-preserving condition, i.e., the solution is subspace-preserving only when subspaces are independent and the data is uncorrupted.

Another group of methods tried to improve the clustering accuracy by incorporating knowledge from other domain. For example, Deng et al. [8] proposed a clustering technique called enhanced soft subspace clustering (ESSC) by integrating within-class compactness and between-class separation in the subspace. Deng et al. [10] applied the concept of transfer learning to prototype-based fuzzy clustering, i.e., the knowledge induced in the source domain, such as the cluster centers and the subspace of the clustering results, has been appropriately used to boost the clustering performance in the target domain. In the Structured Sparse Subspace Clustering (S3C) model [25], both the affinity and the segmentation are learned in a joint optimization framework to exploit the dependency of them. [1] showed that the performance can be boosted by fusing multi-modality data instead of taking just one modality into consideration. These methods can improve the clustering accuracy. However, they leave the question how the affinity leads to exact clustering undiscussed.

Park et al. [33] addressed the gap problem by proposing greedy subspace clustering, which consists of two steps: the first step, nearest subspace neighbor (NSN), determines a neighborhood set for each point; and the second step, greedy subspace recovery (GSR), recovers subspaces from the given neighborhoods. They established statistical guarantees for exact clustering with general subspace conditions. Wang et al. [41] addressed this problem in both noiseless and noisy cases. They improved the graph connectivity of the original l_1 norm regularized optimization by adding a simple post-processing step. They also provided the theoretical guarantee that, under mild conditions, with high probability, no two points from different subspaces are clustered together. These methods have the theoretical results that the solution leads directly to exact clustering, and the conditions are milder than those using l_1 norm regularization. However, the computational load is heavy because the conventional optimization approach to the l_1 norm regularization problem employs LASSO methods which need a great deal of computation. The computational load becomes significant when dealing with large scale dataset.

Recently, deep learning has been widely used in this field. Different from the methods based on learning an affinity matrix, deep learning-based methods use deep networks to learn a low-dimensional representation such that simple clustering such as K-means can be directly applied for the final result. Hershey et al. [18] proposed a deep clustering framework that trains a deep network to form a low-rank pairwise affinity matrix that approximates the ideal affinity matrix. Peng et al. [35] proposed the PARTY algorithm that introduces sparsity prior to the hidden layers such that the reconstruction relation over the whole data set is preserved into the hidden representation. These methods avoid learning an affinity matrix. However, the graph connectivity problem still exists in the hidden layers of the deep network and has not been fully discussed. For example, incorporating regularization on the intermediate representation that maximizes within-class coherence and minimizes between-class coherence is a promising solution to improve the clustering accuracy.

In this paper, we propose a projection step which guarantees an exact clustering result and is computationally efficient. We start by investigating the recently proposed method: sparse subspace clustering by orthogonal matching pursuit (SSC-OMP) [46], which enjoys mild subspace-preserving conditions and fast processing speed. SSC-OMP suffers from the graph connectivity problem, which is not addressed in [11,46,47]. We show that the graph connectivity can be improved significantly by adding a projection step after OMP picks up one point. It is also found that the projection step can be moved out of the iteration and put at the end of the algorithm. More importantly, the projection step makes it possible to guarantee that the subspace-preserving condition leads directly to an exact clustering result.

Paper contributions. 1. A theoretical result is established that addresses and bridges the gap between a subspace-preserving coefficient and the exact clustering result.

2. A novel post-processing step is proposed to address the graph connectivity problem of sparse subspace clustering. The proposed algorithm is computationally efficient and with better graph connectivity.

3. The proposed algorithms address the case when the data are contaminated by noise, which is absent in the previous researches [11,46,47].

1.1. Problem statement and notations

Definition 1. (Subspace clustering). Given a union of K subspaces $S = \bigcup_{l=1}^K S_l$, each S_l of dimension $d_l < n$, $N_l > d_l + 1$ points are generated i.i.d. from arbitrary unknown continuous distribution supported on S_l for $l = 1, \dots, K$, and normalized to unit

l_2 norm. Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{n \times N}$ be the matrix containing all the N points, and $N = \sum_{l=1}^K N_l$. Subspace clustering is to partition the data $\mathbf{X}$ into their respective clusters.

In the sparse representation step, the self-expressiveness model solves the following sparse representation problem:

$$\arg \min_{\mathbf{c}_i \in \mathbb{R}^N} \|\mathbf{c}_i\|_0, \quad \text{s.t. } \mathbf{x}_i = \mathbf{X}\mathbf{c}_i, \mathbf{c}_{ii} = 0 \quad (1)$$

for $i = 1, 2, \dots, N$, where $\mathbf{c}_i \in \mathbb{R}^N$ is the sparse representation of $\mathbf{x}_i$ with respect to $\mathbf{X}$. A graph $G = (V, E)$ is constructed based on the coefficient matrix $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N]$ for $i = 1, 2, \dots, N$ learned by Eq. (1), where the vertices V correspond to the points and the edge between vertices i and j is non-zero only if $\mathbf{c}_{ij} \neq 0$ and $\mathbf{c}_{ji} \neq 0$. An affinity matrix $\mathbf{W} = \{w_{ij}\}$ is constructed with every entry $w_{ij} = (|\mathbf{c}_{ij}| + |\mathbf{c}_{ji}|)/2$, $1 \leq i, j \leq N$. The clustering result is obtained by applying spectral clustering on this graph.

Definition 2. (Subspace-preserving). A coefficient $\mathbf{c}_i \in \mathbb{R}^N$ of a point $\mathbf{x}_i \in S_l$ with respect to the dictionary $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$ is call subspace-preserving if

$$\forall j = 1, \dots, N, \quad \mathbf{c}_{ij} \neq 0 \implies \mathbf{x}_j \in S_l \quad (2)$$

A coefficient matrix $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N]$ is call subspace-preserving if $\forall i = 1, \dots, N$, $\mathbf{c}_i$ is subspace-preserving.

Throughout this paper, matrices and vectors are denoted, respectively, by bold uppercase and lowercase letters. Constants and variables are denoted, respectively, by non-bold uppercase and lowercase letters. $\|\cdot\|_p$ denotes the l_p norm. $|T|$ denotes the cardinality of set T . Let $\mathbf{X}_{-i} \in \mathbb{R}^{n \times (N-1)}$ be the matrix containing all the points with $\mathbf{x}_i$ excluded, $\mathcal{D}_T$ be the set of points indexed by T , and $S_T = \text{span}(\mathcal{D}_T)$ be the subspace spanned by $\mathcal{D}_T$.

The remainder of this paper is organized as follows. Section 2 introduces the OMP and SSC-OMP algorithms that are most related to the proposed algorithm. Section 3 presents in detail two versions of the proposed algorithms, GOMP^{pre} and GOMP. Section 4 theoretically analyses the proposed algorithms in comparison with OMP, based on which the condition for exact clustering is established. Experimental results on both synthetic and real world data are reported in Sections 5 and 6 concludes the whole article.

2. Previous works

2.1. Orthogonal Matching Pursuit (OMP)

Orthogonal Matching Pursuit [34] solves the following optimization problem [12]:

$$\arg \min_{\mathbf{c}} \|\mathbf{c}\|_0, \quad \text{s.t. } \mathbf{y} = \mathbf{D}\mathbf{c} \quad (3)$$

where $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N] \in \mathbb{R}^{n \times N}$ is the observed signal and dictionary, respectively, $\mathbf{c} \in \mathbb{R}^N$ is the coefficient, and $\|\cdot\|_0$ is the l_0 norm that counts the number of non-zero entries of a vector. The OMP algorithm is presented in Algorithm 1.

Algorithm 1 Orthogonal Matching Pursuit (OMP).

Input: $\mathbf{y} \in \mathbb{R}^n$: observed signal

$\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N] \in \mathbb{R}^{n \times N}$: dictionary

s : sparsity level

ϵ : residual norm threshold

Output: $\mathbf{c} \in \mathbb{R}^N$: sparse coefficient

1: Initialize $k = 0$, residual $\mathbf{r}_0 = \mathbf{y}$, support set $T_0 = \emptyset$

2: **while** $k < s$ and $\|\mathbf{r}_k\|_2 > \epsilon$ **do**

3: $i_k^* = \arg \max_{i=1, \dots, N} |\mathbf{d}_i^T \mathbf{r}_k|$, $T_{k+1} = T_k \cup \{i_k^*\}$

4: $\mathbf{r}_{k+1} = \mathbf{y} - \mathbf{P}_{T_{k+1}}(\mathbf{y})$, where $\mathbf{P}_{T_{k+1}}$ is the projection to the subspace spanned by $\{\mathbf{d}_j, j \in T_{k+1}\}$

5: $k \leftarrow k + 1$

6: **end while**

7: **return** $\mathbf{c} = \mathbf{D}_{T_k}^+(\mathbf{y})$

In the k th iteration, the algorithm finds the atom (column) in $\mathbf{D}$ that has the largest absolute of the inner product with the residual $\mathbf{r}_k$, and orthogonally projects the signal $\mathbf{y}$ onto the span of the vectors indexed by the support set. The dimension of the span increases and the l_2 norm of the residual decreases. As a result, the procedure is guaranteed to converge in a finite number of iteration. Researchers have established the conditions under which the OMP algorithm reliably recovers the correct support set [7,38].

2.2. Sparse Subspace Clustering by Orthogonal Matching Pursuit (SSC-OMP)

Dyer et al. [11] first introduced OMP to sparse subspace clustering as an approach to the l_1 regularized optimization (Eq. 1). A major issue lies in the sufficient condition under which OMP gives a subspace-preserving solution. Both the mutual coherence and the restricted isometry property [5,6,38] established in the compressed sensing community cannot be applied here because the dictionary atoms are highly correlated with each other. Dyer et al. provided a condition that highlights the tradeoff between the mutual coherence which characterizes the similarity between points living in different subspaces, and the covering radius which characterizes the similarity of the points within the same subspace. You and Vidal [47] provided a preliminary geometric interpretation for SSC-OMP. Recently, You et al. [46] established a much weaker yet more succinct and interpretable condition under three data generating models: independent deterministic subspace model, arbitrary deterministic subspace model, and arbitrary random subspace model. It is the mildest subspace-preserving condition on the generating subspaces for SSC-OMP.

3. Subspace clustering by greedy orthogonal matching pursuit

3.1. Greedy orthogonal matching pursuit, preliminary

The preliminary version of the greedy orthogonal matching pursuit, GOMP^{pre}, is summarized in Algorithm 2.

Algorithm 2 Greedy Orthogonal Matching Pursuit, preliminary (GOMP^{pre}).

Input: $\mathbf{y} \in \mathbb{R}^n$: observed signal
 $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N] \in \mathbb{R}^{n \times N}$: dictionary
 s : sparsity level
 ϵ_1, ϵ_2 : residual norm thresholds
Output: $\mathbf{c} \in \mathbb{R}^N$: sparse coefficient

- 1: Initialize $k = 0$, residual $\mathbf{r}_0 = \mathbf{y}$, support set $T_0 = \emptyset$
- 2: **while** $k < s$ and $\|\mathbf{r}_k\|_2 > \epsilon_1$ **do**
- 3: $i_k^* = \arg \max_{i=1, \dots, N} |\mathbf{d}_i^T \mathbf{r}_k|$, $\tilde{T}_{k+1} = T_k \cup \{i_k^*\}$
- 4: $T_{k+1} = \{j \mid \|(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}})\mathbf{d}_j\|_2 \leq \epsilon_2, j = 1, \dots, N\}$
- 5: $\mathbf{r}_{k+1} = \mathbf{y} - \mathbf{P}_{T_{k+1}}(\mathbf{y})$, where $\mathbf{P}_{T_{k+1}}$ is the projection to the subspace spanned by $\{\mathbf{d}_j, j \in T_{k+1}\}$
- 6: $k \leftarrow k + 1$
- 7: **end while**
- 8: **return** $\mathbf{c} = \mathbf{D}_{T_k}^+ \mathbf{y}$

The self-expressiveness model represents a given point as a linear combination of the least number of other points. However, in order to increase the graph connectivity, it is desirable to involve as many as possible the points in the same subspace. To this end, once a new point is selected, we can search for other points lying in this subspace spanned by the selected points and involve them in the support set. This motivates us to add Step 4 to the original OMP algorithm. After picking up a new point, the algorithm searches for the points lying in the current subspace by projecting all the points in $\mathbf{X}$ onto the current subspace. The points with projection error smaller than a threshold are regarded as the points lying in the same subspace, i.e., $T_{k+1} = \{j \mid \|(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}})\mathbf{d}_j\|_2 \leq \epsilon_2, j = 1, \dots, N\}$. Strictly speaking, ϵ_2 should be zero. In practice, we set ϵ_2 a small positive number near zero.

3.2. Greedy orthogonal matching pursuit

The formal version of the greedy orthogonal matching pursuit, GOMP, is summarized in Algorithm 3.

It can be seen that the first 6 steps are the OMP method. We put Step 4 in GOMP^{pre} out from each iteration and do it only once at the very end after OMP. As we will explain it in detail later, the advantages are: (1) under mild condition GOMP is equivalent to GOMP^{pre}; and (2) GOMP reduces the computational load of GOMP^{pre}.

3.3. Subspace clustering by greedy orthogonal matching pursuit

Let $\mathbf{X}_{-i} \in \mathbb{R}^{n \times (N-1)}$ be the matrix containing all the points with $\mathbf{x}_i$ excluded. For subspace clustering, we compute the coefficient $\mathbf{c}_i \in \mathbb{R}^{N-1}$ for $\mathbf{x}_i$ with respect to $\mathbf{X}_{-i}$ using either GOMP^{pre} or GOMP, e.g., $\mathbf{c}_i = \text{GOMP}(\mathbf{x}_i, \mathbf{X}_{-i})$. Insert a zero in the i th entry in $\mathbf{c}_i$ to get $\mathbf{c}_i^* \in \mathbb{R}^N$, and $\mathbf{C}^* = [\mathbf{c}_1^*, \dots, \mathbf{c}_N^*] \in \mathbb{R}^{N \times N}$. The affinity matrix can be constructed $\mathbf{W} = (|\mathbf{C}^*| + |\mathbf{C}^{*T}|)/2$. By applying spectral clustering on $\mathbf{W}$ we get the clustering results. This procedure is summarized in Algorithm 4.

Algorithm 3 Greedy Orthogonal Matching Pursuit (GOMP).**Input:** $\mathbf{y} \in \mathbb{R}^n$: observed signal $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N] \in \mathbb{R}^{n \times N}$: dictionary s : sparsity level ϵ_1, ϵ_2 : residual norm thresholds**Output:** $\mathbf{c} \in \mathbb{R}^N$: sparse coefficient1: Initialize $k = 0$, residual $\mathbf{r}_0 = \mathbf{y}$, support set $T_0 = \emptyset$ 2: **while** $k < s$ and $\|\mathbf{r}_k\|_2 > \epsilon_1$ **do**3: $i_k^* = \arg \max_{i=1, \dots, N} |\mathbf{d}_i^T \mathbf{r}_k|$, $T_{k+1} = T_k \cup \{i_k^*\}$ 4: $\mathbf{r}_{k+1} = \mathbf{y} - \mathbf{P}_{T_{k+1}}(\mathbf{y})$, where $\mathbf{P}_{T_{k+1}}$ is the projection to the subspace spanned by $\{\mathbf{d}_j, j \in T_{k+1}\}$ 5: $k \leftarrow k + 1$ 6: **end while**7: $T_{k+1} = \{j \mid \|(\mathbf{I} - \mathbf{P}_{T_k})\mathbf{d}_j\|_2 \leq \epsilon_2, j = 1, \dots, N\}$ 8: **return** $\mathbf{c} = \mathbf{D}_{T_k}^+ \mathbf{y}$ **Algorithm 4** Subspace Clustering by Greedy Orthogonal Matching Pursuit (SC-GOMP).**Input:** $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{n \times N}$: data points s : sparsity level ϵ_1, ϵ_2 : residual norm thresholds**Output:** segmentation of data $\mathbf{X}$ 1: compute $\mathbf{c}_i^*$ from GOMP($\mathbf{x}_i, \mathbf{X}_{-i}$) for $i = 1, \dots, N$ 2: compute $\mathbf{C}^* = [\mathbf{c}_1^*, \dots, \mathbf{c}_N^*]$ and $\mathbf{W} = (|\mathbf{C}^*| + |\mathbf{C}^{*T}|)/2$

3: apply spectral clustering to get the result.

4. Theoretical analysis

Proposition 1. Under the data generation model of Definition 1, with probability one, any $p < d_l$ points in S_l are linearly independent.

This is a natural result under the given data generation model, which is widely used in theoretical analysis of subspace clustering [41,44] (please refer to [44] for the proof). Proposition 1 says that for any data point $\mathbf{x} \in S_l$ that is generated according to a continuous distribution supported on S_l , it is almost surely that at least d_l points in S_l are required to linearly represent $\mathbf{x}$. This property plays an important role in the following analysis.

In the analysis we first assume the noiseless case $\epsilon_2 = 0$. Recall the two-step update of the support set in GOMP^{pre}, i.e., $T_k \rightarrow \tilde{T}_{k+1} \rightarrow T_{k+1}$. The first step selects one point and adds it to the support set:

$$i^* = \arg \max_{i=1, \dots, N} |\mathbf{d}_i^T \mathbf{r}_k|, \quad \tilde{T}_{k+1} = T_k \cup \{i^*\}. \quad (4)$$

The second step searches for the points that lie in the current subspace

$$T_{k+1} = \{j \mid \|(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}})\mathbf{d}_j\|_2 = 0, j = 1, \dots, N\}. \quad (5)$$

For the first step, due to the fact that the residual $\mathbf{r}_k$ is always orthogonal to S_{T_k} , so $\dim(S_{T_k}) < \dim(S_{\tilde{T}_{k+1}})$. We call this the *subspace increase step*. For the second step, the dimension of $S_{T_{k+1}}$ remains the same as $S_{\tilde{T}_{k+1}}$ but the number of points increases (it will be proved later). We call this the *subspace inflation step*¹.

Theorem 1. Let $i_{k,OMP}^*$, $S_{T_{k+1},OMP}$, and $\mathbf{r}_{k,OMP}$ be the index, the subspace spanned by $\mathcal{D}_{T_{k+1}}$, and the residual in the k th iteration in the OMP algorithm, respectively. Let $i_{k,GOMP^{pre}}^*$, $S_{T_{k+1},GOMP^{pre}}$, and $\mathbf{r}_{k,GOMP^{pre}}$ be the corresponding variables in the GOMP^{pre} algorithm. We have:

$$i_{k,OMP}^* = i_{k,GOMP^{pre}}^* \quad (6)$$

$$S_{T_{k+1},OMP} = S_{T_{k+1},GOMP^{pre}} \quad (7)$$

$$\mathbf{r}_{k+1,OMP} = \mathbf{r}_{k+1,GOMP^{pre}} \quad (8)$$

for $k = 0, \dots, s$.

¹ Note that the names of these two steps are only for the convenience of presentation, but has nothing to do with the real world physical phenomenon.

The proof of [Theorem 1](#) can be found in Appendix.

[Theorem 1](#) indicates that in each iteration, both the OMP and GOMP^{pre} algorithms select the same point, form the same subspace, and produce the same residual. The difference lies in the support set, where OMP contains only s indices while GOMP^{pre} contains all the points lying in the current subspace.

[Theorem 1](#) allows one to propose GOMP. The subtle difference lies in how the support set is updated. In GOMP^{pre}, the increase of the number of points in the support set, $|T_k| \rightarrow |T_{k+1}|$, follows the “one, many, one, many, ...” manner, while in GOMP, it follows “one, ..., one, many”. However, according to [Proposition 1](#) that, with probability 1, any $p < d_l$ points in the data $\mathbf{X}^{(l)} \in \mathbb{R}^{n \times N_l}$ lying in S_l are linearly independent. That is to say, the probability of $|\tilde{T}_{k+1}| > |T_k|$ is almost zero for $k < \min_l(d_l)$. Hence, there is no need to do the subspace inflation step in every iteration, and the equivalent results are guaranteed.

Another advantage is that GOMP is generally faster than GOMP^{pre}. The reason is quite simple because one does not need to compute the projection operation (Step 4, [Algorithm 2](#)), which is the most time consuming step, in every iteration.

Theorem 2. Given a union of K subspaces $S = \bigcup_{l=1}^K S_l$, each S_l of dimension $d_l < n$, $N_l > d_l + 1$ points are generated i.i.d. from arbitrary unknown continuous distribution supported on S_l for $l = 1, \dots, K$, and normalized to unit l_2 norm, denoted as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{n \times N}$, exact clustering result is guaranteed if the solution is subspace-preserving.

Proof. To show the subspace-preserving solution leads to exact clustering result, it is equivalent to show that, with probability 1, the number of connected components in any subspace is 1. Assuming that there are more than 1 connected components in the subspace, for example, let $\mathbf{x}_i \in S_l$ be any point in one of the connected component, say C_1 . Let $\mathbf{X}[l] \in \mathbb{R}^{n \times N_l}$ be all the points sampled from subspace S_l , and $\mathbf{c}_i^*$ be the optimal solution of $\mathbf{x}_i$, only the entries corresponding to $\mathbf{X}[l]$ are considered, denoted by $\mathbf{c}^*[l] \in \mathbb{R}^{N_l}$, because the entries outside $\mathbf{c}^*[l]$ are all zeros (subspace-preserving property). Hence, $\mathbf{x}_i$ can be represented as follows:

$$\mathbf{x}_i = [\mathbf{X}[l]_{C_1} \quad \mathbf{X}[l]_{C_1^c}] \mathbf{c}^*[l] \quad (9)$$

where $\mathbf{X}[l]_{C_1}$ and $\mathbf{X}[l]_{C_1^c}$ denote, respectively, the points in C_1 and other connected components. $\mathbf{c}^*[l]$ can be divided accordingly, $\mathbf{c}^*[l] = [\mathbf{c}^*[l]_{C_1} \quad \mathbf{c}^*[l]_{C_1^c}]^T$, and we know that $\mathbf{c}^*[l]_{C_1^c} = \mathbf{0}$. According to [Proposition 1](#), we have $|C_1| \geq d_l$, i.e., there are at least d_l linearly independent points included in the support set T . In such a case, the span of the points indexed by T is equivalent to S_l , i.e., $S_T = S_l$. Recall that the projection step in both GOMP^{pre} and GOMP,

$$T_{k+1} = \{j \mid \|(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}}) \mathbf{d}_j\|_2 = 0, j = 1, \dots, N\}. \quad (10)$$

It is easy to show that $\mathbf{X}[l]_{C_1^c}$ satisfies the above equation if $S_{\tilde{T}_{k+1}} = S_l$. Therefore $\mathbf{c}^*[l]_{C_1^c} \neq \mathbf{0}$, which contradicts with the assumption that $\mathbf{c}^*[l]_{C_1^c} = \mathbf{0}$. $\square$

[Theorem 2](#) is based on the subspace-preserving condition, which has been thoroughly discussed in prior papers [[11,46,47](#)]. The advantage of this theorem is that subspace-preserving condition leads directly to the correct clustering result, which bridges the gap.

5. Experimental results

We conduct experiments using the proposed algorithms on both of synthetic and real world data in comparison with representative state-of-the-art algorithms.

Evaluation metrics. We employ the following metrics to evaluate the algorithms.

- **clustering accuracy:** This is the commonly used metric employed by all the other subspace clustering methods. It gives the percentage of the correctly classified points by comparing the label given by the spectral clustering algorithms with the true data label.

- **running time:** We calculate the time of building an affinity matrix for the whole set of points. We use a computer with Intel i7-4770 CPU and 32 GB RAM in Matlab environment.

- **connectivity** [[46](#)]: Let the normalized Laplacian matrix of an undirected graph be $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$, where $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the affinity and $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. The second smallest eigenvalue of $\mathbf{L}$ is used to measure the connectivity of the graph. In practice, the connectivity is computed as $c = \min_i \lambda_i^2$, where λ_i^2 is the algebraic connectivity for each cluster.

- **Within-class recovery rate:** suppose $\mathbf{x}_i \in S_l$, for each $\mathbf{c}_i$, we compute the percentage of recovered points in S_l by examining the l_0 norm and then average over all i , i.e., $r = \frac{100}{T} \sum_i \|\mathbf{w}_{ij} \mathbf{c}_{ij}\|_0 / (N_l - 1)$, where $\mathbf{w}_{ij} \in \{0, 1\}$ is true weight between point $\mathbf{x}_i$ and $\mathbf{x}_j$.

- **Precision:** for the support set of $\mathbf{c}_i$, we count the number of indices corresponding to the points in the same subspace and divide it by the total number of non-zero elements in $\mathbf{c}_i$. The final quantity is obtained by taking average over all the points, i.e., $r = \frac{100}{T} \sum_i \|\mathbf{w}_{ij} \mathbf{c}_{ij}\|_0 / \|\mathbf{c}_i\|_0$.

- **Friedman test:** to evaluate the significant improvement of the proposed algorithms over the compared methods, we apply the Friedman test to analyze the results obtained by each evaluation metric. We report the p -value for interpretation and the improvement is significant if the p -value is less the significance level $\alpha = 0.05$.

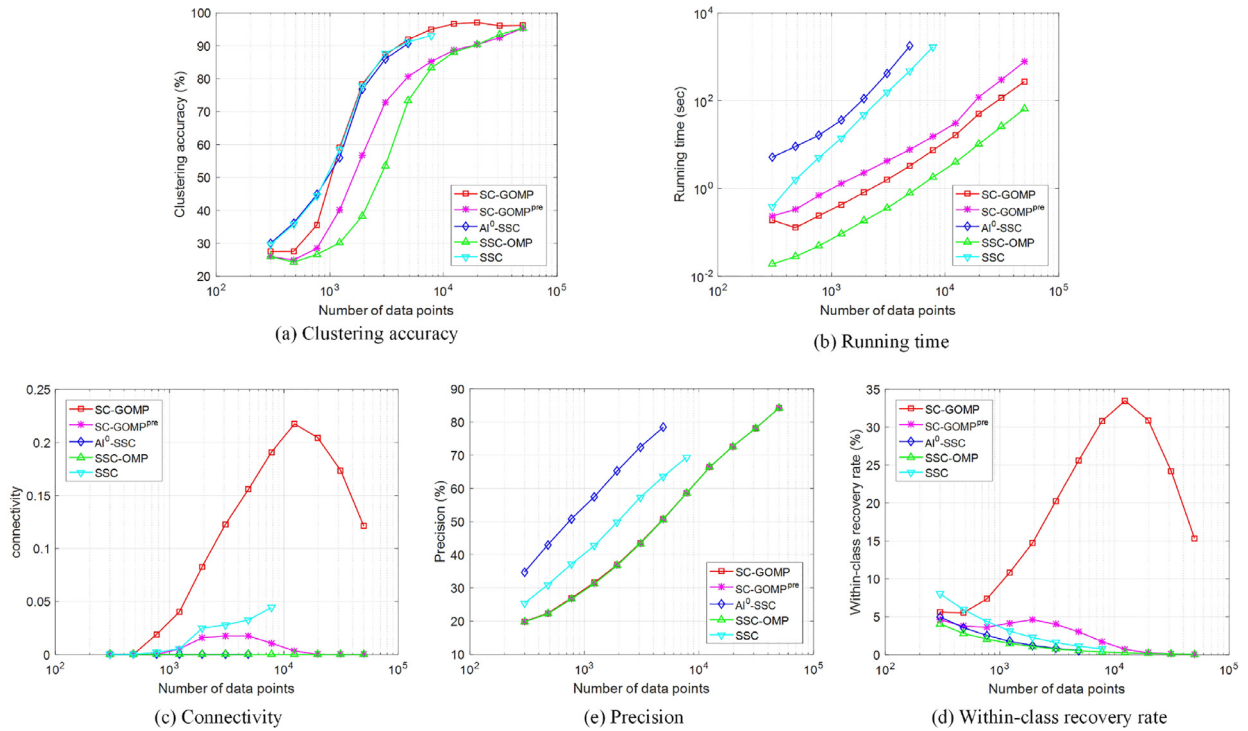


Fig. 1. Experimental results on synthetic data. Five methods are compared, i.e., SC-GOMP, SC-GOMP^{pre}, SSC-OMP, Al⁰-SSC and SSC. Five evaluations are considered, i.e., clustering accuracy, running time, connectivity, precision, and within-class recovery rate. The x-axis denotes the total number of points and is plotted in log scale. The y-axis of (d) is also in log scale.

Compared Methods. The compared methods include Sparse Subspace Clustering (SSC) [13], Al⁰-SSC [44], and SSC-OMP [46]. We use the codes provided by the authors on their websites². The algorithms proposed in this paper are denoted by SC-GOMP^{pre} and SC-GOMP, respectively.

5.1. Synthetic data

We test the proposed algorithms using synthetic data. $K = 10$ subspaces of dimension $d_l = 6$ are randomly generated in an ambient space of dimension $n = 9$. In each subspace we sample N_l points and normalize them to the l_2 unit sphere, where N_l ranges from 30 to 5000, resulting in that the total number of points varies from 300 to 50,000. Once the points are generated, we run all the above-mentioned algorithms and calculate the metrics. This procedure is repeated for 10 trials and the averaged results are reported. The Al⁰-SSC and SSC are run for $N_l \leq 500$ due to the computational complexity reason. The parameters for the compared methods are set according to the original papers respectively. For SC-GOMP^{pre} and SC-GOMP, the parameters are set $\epsilon_1 = 10^{-3}$, $\epsilon_2 = 10^{-9}$, and $s = 6$, respectively.

For clustering accuracy, SC-GOMP^{pre} and SC-GOMP achieve better results than the original algorithm SSC-OMP by a large margin. SC-GOMP is even better than SSC and Al⁰-SSC when the number of points is large, which demonstrates the advantage of SC-GOMP in dealing with large scale dataset. For running time, SSC and Al⁰-SSC are the most time consuming while SSC-OMP is the fastest. The processing speed of SC-GOMP^{pre} and SC-GOMP lies between them and is closer to SSC-OMP. Note that the running time is in log-scale so that one can see a slight increase to the fastest algorithm SSC-OMP and a significantly faster speed than the most time-consuming algorithms SSC and Al⁰-SSC. It can be concluded from Fig. 1(a,b) that SC-GOMP^{pre} and SC-GOMP achieve the best tradeoff between clustering accuracy and processing time and SC-GOMP is the best choice when dealing with large scale dataset.

The connectivity results can be seen in Fig. 1(c). SC-GOMP outperforms all the other algorithms by a large margin. SC-GOMP achieves the highest score 0.2175 at $N = 12,390$, which is 4.91 times greater than the highest score of SSC (0.0443 at $N = 7790$).

Secondly, to illustrate the difference between the coefficient matrices produced by different algorithms, we randomly generated another dataset with 3 subspaces of dimension 6 in an ambient space of dimension 9. Each subspace contains $N_l = 500$ points. We run each algorithm on the dataset, and the coefficient matrices are plotted in Fig. 2. The precision and

² SSC: <http://www.vision.jhu.edu/code/>, Al⁰-SSC: <https://github.com/yingshenyang/L0-SSC>, SSCOMP: <http://vision.jhu.edu/code/fetchcode.php?id=16>.

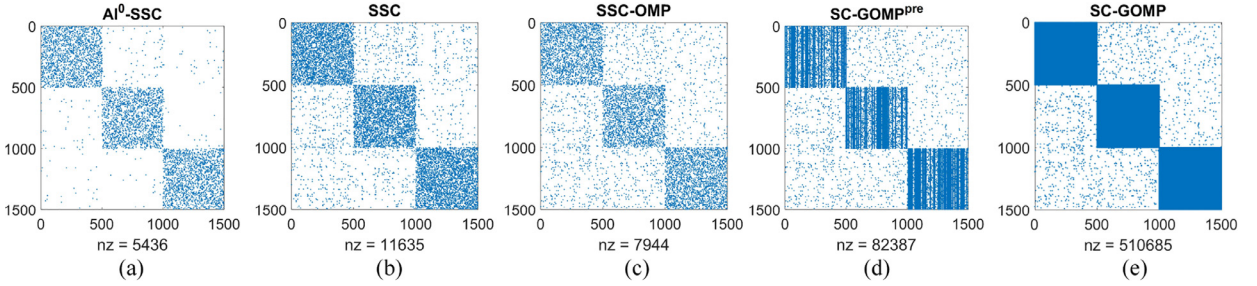


Fig. 2. Coefficient matrices produced by different algorithms. Blue dot in the matrix represents an non-zero element in that position. ‘nz’ denotes the number of non-zero elements in the matrix. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

within-class recovery rate (Figs. 1(d,e)) can be explained together with these plots. For each column, SC-GOMP^{pre} recovers more points in the same subspace than previous methods. Some columns recover even the whole set of points in that subspace. SC-GOMP recovers the most points in the same subspace.

The recovery rate of SC-GOMP in Fig. 1 starts to decrease when the number of points exceeds a certain threshold. This is partially due to fact that, as data are getting dense in the spaces, the Euclidean distance between a pair of points gets smaller. Using the a constant threshold ϵ_2 in the subspace inflation step will incorporate more points, thus, decrease the within-class recovery rate. An adaptive ϵ_2 proportional to reciprocal of the number of points will help fix this phenomenon.

5.2. Clustering on MNIST dataset

The MNIST database of handwritten digits [23] contains a training set of 60,000 examples, and a test set of 10,000 examples. Each image consists of one of the handwritten digits 0–9. We randomly select $N_l \in \{100, 200, 300, 400, 500\}$ examples from each digit in the training set, and follow the practice of [46] by extracting ScatNet features [3] on the digit images. Each image is converted to a feature vector of dimension 3472, projected to an 80 dimension vector using PCA, and normalized to unit l_2 norm. In each trial we run all the compared methods and obtain the results. The experiments are conducted for 10 trials and the averaged results are reported in mean and standard deviation (std) form as is shown in Table 1.

For clustering accuracy, AI⁰-SSC achieves best scores when $N_l = 100, 200$, and SC-GOMP outperforms all the other algorithms when $N_l = 300, 400, 500$. For running time, SSC-OMP needs the least processing time while AI⁰-SSC is the most time-consuming algorithm. SC-GOMP and SC-GOMP^{pre} turn out to be the second and third fastest algorithms. For connectivity and within class recovery rate, SC-GOMP and SC-GOMP^{pre} obtain much higher scores than the other three algorithms, while AI⁰-SSC and SSC perform better for precision evaluation.

The major problem with SSC and AI⁰-SSC is that the processing time grows extremely high as the number of data points increases, because they have to solve the l_1 regularization by LASSO (or basis pursuit) -based optimization approaches, which is known to be computationally inefficient when dealing with a large number of data points. On the contrary, SSC-OMP turns out to be the fastest algorithm to obtain the results but the clustering accuracy is not satisfying. The proposed algorithms overcome both limitations, achieving the best clustering accuracy when dealing with a large number of points with processing speed significantly faster than AI⁰-SSC and SSC and near that of SSC-OMP.

5.3. Clustering on RGB-D object dataset

The RGB-D Object Dataset [22] is a large dataset of 300 common household objects arranged in 51 categories. This dataset were recorded using a Kinect style 3D camera that records synchronized and aligned 640x480 RGB and depth images at 30 Hz. Each object was placed on a turntable and video sequences were captured for one whole rotation. We choose the first 6 categories in our experiment. The categories and the number of data samples in each category are listed in Table 2. The raw data need to be preprocessed as is illustrated in Fig. 3. A bounding box is obtained based on the mask image. The color image is cropped according to the bounding box, resulting in the cropped image as shown in Fig. 3(c). All the cropped images are converted to grayscale and resized to 32x32, resulting in a 1024-d column vector. We randomly select $N_l \in \{100, 300, 500\}$ examples from each category and project the data points in 80 dimension using PCA. The experiments are conducted for 10 trials and the averaged results are reported in mean and standard deviation (std) form as is shown in Table 3.

For clustering accuracy, SC-GOMP^{pre} and SC-GOMP outperform other methods in all the settings, except when $N_l = 300$ where AI⁰-SSC performs the best. In general, AI⁰-SSC exhibits comparative clustering accuracy to the proposed algorithms. However, its running time becomes the longest as the number of points increases. For running time, similar conclusion can be drawn to that in the previous experiment. It is surprising to observe that SC-GOMP^{pre} is faster than SC-GOMP, which is in contrast to the notion in the theoretical analysis that SC-GOMP is expected to be faster than SC-GOMP^{pre} because SC-GOMP

Table 1

Results in mean (std) format on MNIST dataset. The best and second mean values in each column are highlighted in bold face and blue, respectively. The two numbers in the last column denote the Friedman test p -values between the method in the current row with SC-GOMP^{pre} and SC-GOMP, respectively. Values less than the significance level $\alpha = 0.05$ are highlighted in bold face.

# of points	1000 ($N_t = 100$)	2000 ($N_t = 200$)	3000 ($N_t = 300$)	4000 ($N_t = 400$)	5000 ($N_t = 500$)	p -value
clustering accuracy (%)						
SSC	79.78 (3.95)	79.94 (2.39)	79.62 (1.23)	80.87 (1.09)	80.78 (1.08)	0.0253 , 0.1797
SSC-OMP	50.33 (4.37)	27.56 (5.31)	17.84 (1.60)	14.39 (1.70)	12.45 (0.85)	0.0253 , 0.0253
AI ⁰ -SSC	81.06 (3.36)	81.07 (2.69)	81.99 (2.34)	80.98 (0.36)	81.07 (0.64)	0.0253 , 0.6547
SC-GOMP ^{pre}	79.60 (6.24)	66.76 (3.40)	70.07 (6.74)	74.73 (4.60)	75.74 (3.91)	
SC-GOMP	76.54 (4.27)	80.33 (3.09)	82.87 (1.32)	83.24 (0.63)	83.78 (1.77)	
running time (sec)						
SSC	11.59 (0.56)	58.27 (0.92)	196.41 (8.20)	430.63 (7.24)	797.20 (12.28)	0.0253 , 0.0253
SSC-OMP	0.27 (0.01)	0.61 (0.02)	1.04 (0.01)	1.58 (0.02)	2.19 (0.03)	0.0253 , 0.0253
AI ⁰ -SSC	12.62 (0.27)	88.24 (0.93)	300.54 (11.11)	715.59 (71.73)	1355.39 (44.10)	0.0253 , 0.0253
SC-GOMP ^{pre}	9.87 (0.61)	27.22 (0.64)	43.19 (1.19)	61.75 (1.32)	81.46 (0.83)	
SC-GOMP	2.06 (0.16)	8.12 (0.17)	16.99 (0.07)	27.79 (0.12)	41.06 (0.18)	
connectivity ($\times 10^{-2}$)						
SSC	5.36 (0.99)	3.77 (0.71)	3.11 (0.40)	2.71 (0.33)	2.08 (0.89)	0.0253 , 0.0253
SSC-OMP	3.57 (1.32)	2.20 (1.12)	1.79 (0.75)	0.95 (0.60)	1.34 (0.49)	0.0253 , 0.0253
AI ⁰ -SSC	4.52 (0.71)	3.79 (0.45)	3.34 (0.41)	2.93 (0.22)	2.81 (0.21)	0.0253 , 0.0253
SC-GOMP ^{pre}	51.46 (2.60)	39.19 (1.83)	26.27 (2.19)	21.14 (3.15)	19.80 (1.43)	
SC-GOMP	14.30 (5.69)	20.15 (6.11)	21.52 (5.78)	22.13 (3.44)	24.10 (0.82)	
within class recovery rate (%)						
SSC	7.03 (0.08)	3.77 (0.03)	2.41 (0.01)	1.73 (0.01)	1.33 (0.01)	0.0253 , 0.0253
SSC-OMP	2.59 (0.03)	1.27 (0.02)	0.83 (0.01)	0.62 (0.00 ^a)	0.49 (0.00)	0.0253 , 0.0253
AI ⁰ -SSC	3.48 (0.02)	1.84 (0.02)	1.27 (0.01)	0.96 (0.00)	0.78 (0.00)	0.0253 , 0.0253
SC-GOMP ^{pre}	92.54 (0.92)	70.95 (1.05)	56.98 (0.85)	48.66 (0.74)	43.28 (0.53)	
SC-GOMP	30.22 (1.08)	29.94 (0.54)	29.59 (0.31)	29.24 (0.32)	29.10 (0.48)	
precision (%)						
SSC	65.17 (0.45)	73.41 (0.47)	78.11 (0.41)	80.70 (0.34)	82.04 (0.27)	0.0253 , 0.0253
SSC-OMP	25.87 (0.31)	25.39 (0.30)	25.04 (0.28)	24.71 (0.18)	24.60 (0.14)	0.1797, 0.0253
AI ⁰ -SSC	81.86 (0.53)	84.87 (0.50)	86.96 (0.29)	87.83 (0.25)	88.48 (0.23)	0.0253 , 0.0253
SC-GOMP ^{pre}	14.78 (0.93)	38.37 (1.36)	52.79 (0.85)	60.41 (0.89)	64.97 (0.36)	
SC-GOMP	50.53 (0.90)	51.36 (0.75)	51.53 (0.72)	51.94 (0.57)	52.28 (0.44)	

^a '0.00' means the first significant digit appears after the given decimal place.

Table 2

RGB-D Object Dataset selected categories used in our experiment.

Category	Apple	Ball	Banana	Bell pepper	Binder	Bowl
# of samples	607	820	726	634	710	580

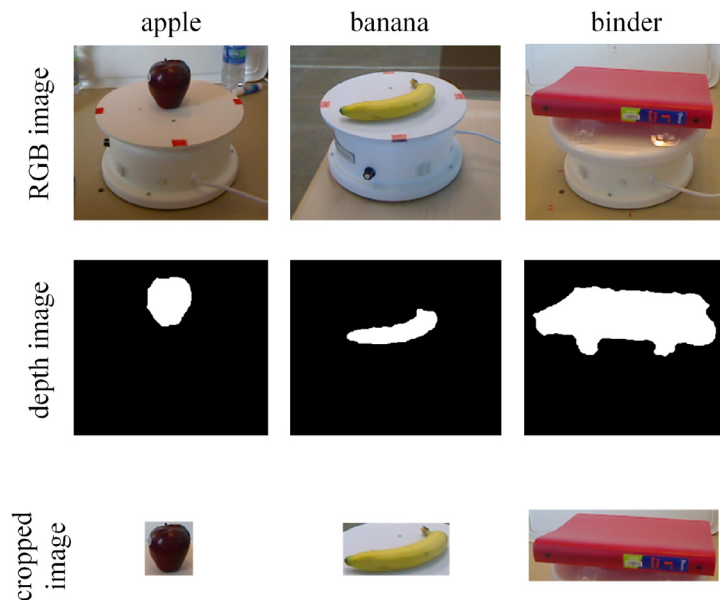
**Fig. 3.** Illustration of image cropping on RGB-D Object Dataset.

Table 3

Results in mean (std) format on RGB-D Object Dataset. The best and second best mean values in each column are highlighted in bold face and blue, respectively. The two numbers in the last column denote the Friedman test p -values between the method in the current row with SC-GOMP^{pre} and SC-GOMP, respectively. Values less than the significance level $\alpha = 0.05$ are highlighted in bold face.

# of points	600 ($N_t = 100$)	1200 ($N_t = 200$)	1800 ($N_t = 300$)	2400 ($N_t = 400$)	3000 ($N_t = 500$)	p -value
clustering accuracy (%)						
SSC	59.87 (10.44)	52.56 (1.44)	51.74 (1.12)	50.96 (3.97)	51.75 (4.89)	0.0253,0.0253
SSC-OMP	21.22 (1.61)	19.38 (1.19)	17.93 (0.55)	17.52 (0.29)	17.46 (0.58)	0.0253,0.0253
AI ⁰ -SSC	65.05 (9.47)	70.63 (10.67)	73.75 (7.09)	61.68 (10.26)	61.84 (10.14)	0.1797,0.6547
SC-GOMP ^{pre}	73.08 (1.21)	74.92 (3.18)	73.11 (3.36)	65.07 (4.66)	64.90 (2.57)	
SC-GOMP	68.78 (0.68)	69.65 (2.15)	68.97 (0.52)	69.41 (0.41)	68.72 (0.42)	
running time (sec)						
SSC	3.65 (0.37)	18.28 (0.20)	48.83 (0.29)	107.06 (1.76)	205.24 (6.01)	0.0253,0.0253
SSC-OMP	0.15 (0.01)	0.32 (0.00)	0.54 (0.01)	0.79 (0.01)	1.09 (0.04)	0.0253,0.0253
AI ⁰ -SSC	3.46 (0.15)	17.50 (0.23)	60.78 (0.33)	150.51 (0.44)	305.95 (0.78)	0.0253,0.0253
SC-GOMP ^{pre}	3.26 (0.34)	7.48 (0.32)	13.26 (0.24)	19.94 (0.28)	27.03 (0.24)	
SC-GOMP	2.68 (0.03)	8.03 (0.05)	17.02 (0.19)	28.03 (0.24)	44.12 (0.21)	
connectivity ($\times 10^{-2}$)						
SSC	0.17 (0.16)	0.04 (0.05)	0.09 (0.02)	0.05 (0.04)	0.04 (0.03)	0.0253,0.0253
SSC-OMP	0.24 (0.14)	0.09 (0.04)	0.06 (0.03)	0.06 (0.03)	0.05 (0.03)	0.0253,0.0253
AI ⁰ -SSC	0.48 (0.26)	0.07 (0.02)	0.04 (0.02)	0.04 (0.01)	0.03 (0.02)	0.0253,0.0253
SC-GOMP ^{pre}	19.79 (2.20)	14.50 (5.42)	0.27 (0.27)	0.26 (0.18)	0.24 (0.10)	
SC-GOMP	22.74 (2.87)	21.28 (1.04)	21.30 (1.98)	20.12 (1.10)	19.53 (1.02)	
within class recovery rate (%)						
SSC	6.35 (0.16)	3.54 (0.05)	2.53 (0.02)	1.99 (0.01)	1.65 (0.01)	0.0253,0.0253
SSC-OMP	3.08 (0.04)	1.51 (0.02)	1.00 (0.01)	0.74 (0.01)	0.59 (0.00)	0.0253,0.0253
AI ⁰ -SSC	2.31 (0.05)	1.16 (0.01)	0.78 (0.01)	0.59 (0.00)	0.48 (0.00)	0.0253,0.0253
SC-GOMP ^{pre}	85.47 (0.94)	72.80 (1.43)	64.43 (0.87)	61.25 (0.58)	59.27(0.33)	
SC-GOMP	76.89 (0.82)	76.68 (0.41)	76.92 (0.56)	76.99 (0.45)	76.74 (0.35)	
precision (%)						
SSC	88.54 (0.51)	95.10 (0.34)	96.54 (0.16)	96.44 (0.10)	96.07 (0.06)	0.0253,0.0253
SSC-OMP	30.82 (0.40)	30.40 (0.39)	30.12 (0.35)	29.79 (0.20)	29.63 (0.26)	0.0253,0.0253
AI ⁰ -SSC	98.68 (0.26)	99.78 (0.05)	99.93 (0.03)	99.94 (0.03)	99.97 (0.02)	0.0253,0.0253
SC-GOMP ^{pre}	41.43 (2.03)	57.78 (2.34)	66.77 (0.66)	69.13 (0.32)	70.60 (0.29)	
SC-GOMP	29.57 (0.70)	29.77 (0.43)	29.33 (0.50)	29.59 (0.26)	29.59 (0.27)	

Table 4

UCI Dataset selected categories used in our experiment.

Category	# of samples	# of data dimensions	# of clusters
Anuran Calls (MFCCs)	7195	21	4
Chess (King-Rook vs. King-Pawn)	3196	36	2
SPECTF Heart [21]	267	44	2
Ionosphere [37]	351	34	2
Vehicle Silhouettes [36]	846	18	4
Wine [15]	178	13	3

needs only one step of projection while SC-GOMP^{pre} needs projection in each iteration. However, this notion is based on the assumption that each projection operation takes equivalent computational complexity. In this experiment, the projection in SC-GOMP takes much more time than all the projection operations in SC-GOMP^{pre} because SC-GOMP selects far more points in the support set than SC-GOMP^{pre}. The reason that SC-GOMP selects so many points is related to the parameter ϵ_2 . A larger ϵ_2 will involve more points in the support set.

The recovery rate of SC-GOMP^{pre} and SC-GOMP are much higher than all the other methods, whereas SSC and AI⁰-SSC outperform other methods in precision. This indicates that, for the coefficients obtained by different algorithms, almost all the non-zeros of SSC and AI⁰-SSC come from the same cluster (high precision) but only a few points in this cluster are recovered in the coefficient (low recovery), while although not all the non-zeros come from the same cluster (low precision), most points in this cluster are recovered in the coefficient (high recovery). This experiment demonstrates that, instead of pursuing the precision of the coefficient, to increase the recovery rate contributes to the final clustering accuracy.

5.4. Clustering on UCI dataset

The UC Irvine Machine Learning Repository [26] (UCI Dataset) holds 394 datasets from various sources as a service to the machine learning community. Each dataset has an independent goal such as classification, regression, clustering, etc. The number of data samples, dimensions, and clusters in each dataset differ from each other. We choose UCI dataset in order to examine the performance of the proposed algorithms not only on the image data but also on other types of data. We choose

Table 5

Results on the UCI Dataset. The best and second best values in each column are highlighted in bold face blue, respectively. The two numbers in the last column denote the Friedman test p -values between the method in the current row with SC-GOMP^{pre} and SC-GOMP, respectively. Values less than the significance level $\alpha = 0.05$ are highlighted in bold face.

category	Anuran	Chess	SPECTF	Ionosphere	Vehicle	Wine	p -value
clustering accuracy (%)							
SSC	67.41	50.75	50.56	50.71	42.55	57.30	0.1025, 0.1025
SSC-OMP	51.84	52.25	72.28	58.69	32.98	45.51	0.1025, 0.1025
AI ⁰ -SSC	70.87	51.44	54.31	63.53	40.43	88.76	0.4142, 0.4142
SC-GOMP ^{pre}	78.14	51.97	73.03	74.36	34.28	74.72	
SC-GOMP	68.41	51.13	73.03	64.67	34.04	74.72	
running time (sec)							
SSC	2069.87	193.91	0.36	0.65	7.14	0.13	0.0143, 0.0143
SSC-OMP	3.02	0.88	0.03	0.04	0.14	0.02	0.0143, 0.0143
AI ⁰ -SSC	3891.94	1546.97	230.03	38.42	253.80	8.88	0.0143, 0.0143
SC-GOMP ^{pre}	33.85	6.15	0.18	0.25	0.62	0.05	
SC-GOMP	28.69	2.36	0.10	0.13	0.32	0.04	
connectivity ($\times 10^{-2}$)							
SSC	0.37	2.90	17.37	5.18	2.04	0	0.414, 0.4142
SSC-OMP	1.57	9.97	19.69	6.99	11.94	19.36	0.4142, 0.1025
AI ⁰ -SSC	0.32	3.01	17.00	1.58	6.51	16.72	1.0000, 0.4142
SC-GOMP ^{pre}	14.82	0.23	15.11	4.93	0	37.52	
SC-GOMP	0.61	0.38	15.12	0.62	0	37.52	
within class recovery rate (%)							
SSC	0.53	0.49	6.29	9.64	3.58	1.10	0.4142, 0.4142
SSC-OMP	0.24	0.34	5.47	2.52	1.30	6.17	0.1025, 0.1025
AI ⁰ -SSC	0.16	0.33	11.43	3.72	1.44	6.44	0.1025, 0.1025
SC-GOMP ^{pre}	13.51	0.14	1.35	1.04	0.35	2.06	
SC-GOMP	29.53	0.18	1.35	1.82	0.35	2.06	
precision (%)							
SSC	64.10	80.93	65.14	55.01	37.21	33.33	1.0000, 1.0000
SSC-OMP	42.73	57.40	72.96	44.27	27.49	36.63	0.4142, 0.4142
AI ⁰ -SSC	82.64	79.44	74.18	84.36	48.73	67.11	0.0143 , 0.1025
SC-GOMP ^{pre}	73.17	74.40	59.93	59.97	24.67	40.82	
SC-GOMP	85.21	74.44	59.93	65.09	24.67	40.82	

Table 6

Results on the Extended Yale B Dataset. The best and second best values in each row are highlighted in bold face blue, respectively.

methods	SSC	SSC-OMP	AI ⁰ -SSC	SC-GOMP ^{pre}	SC-GOMP
accuracy (%)	54.43	67.73	76.35	72.25	77.71
time (s)	149.70	4.30	358.12	613.62	195.35
connectivity ($\times 10^{-2}$)	2.99	3.29	0.49	0.39	0.73
recovery rate (%)	6.47	5.89	4.19	3.28	3.84
precision (%)	55.90	37.39	89.17	69.41	60.95

6 datasets that are suitable for subspace clustering in this experiment. The number of data samples ranges from small-scale (178) to large-scale (7195), and the classification type include two-class classification and multi-class classification. The detailed information of the 6 datasets is listed in Table 4. We use all the data samples in each dataset so that the results are deterministic (no mean and std values are given). The experimental results are shown in Table 5.

The results show that, among the 6 datasets, SC-GOMP^{pre} achieves the best clustering accuracy in 3 datasets and the second best in 2 datasets, while SC-GOMP achieves the best clustering accuracy in 1 dataset and the second best in 2 datasets. For running time, SC-GOMP^{pre} and SC-GOMP are still the fastest algorithms except SSC-OMP. It is interesting to observe that there exist situations when SC-GOMP^{pre} and SC-GOMP give exactly the same results, which is not observed in other experiments. One possible reason is that the subspace assumption does not hold for these datasets, i.e., the data samples in the same class do not form a subspace that is separable to the other subspaces or even they lie in two clusters in the ambient space. One example is that there are two clusters of data of dimension 3, randomly sampled, respectively, from two separating cubic regions. None of these clusters lie in a subspace of dimension 2 and it is possible for SC-GOMP^{pre} and SC-GOMP to give the same results because the subspace inflation step would fail to incorporate any points in the support set. This experiment also indicates that the data from the UCI Dataset would not always satisfy the subspace assumption.

5.5. Clustering on extended yale b dataset

The Extended Yale Face Database B [16] contains 16,128 images of 28 human subjects under 9 poses and 64 illumination conditions. We used the subversion that contains 38 individuals and a total number of 2414 images of size 32*32 pixels provided by [17] in our experiment. Images were vectorized and normalized to have unit l_2 norm. Because no random selection scheme is employed, the results are deterministic, i.e., no standard deviation is given. The results are plotted in Table 6.

SC-GOMP achieves the best clustering accuracy, followed by Al^0 -SSC. It is interesting to find that the connectivity scores of SSC and SSC-OMP are much higher than those of the other methods but the accuracy scores are far less, which seems to contradict with the fact that graph connectivity contributes to the clustering accuracy. However, the fact is not wrong and the phenomenon is because that SSC and SSC-OMP ‘over-connect’ the data, i.e., the graph is well connected so that the whole graph turns out to be only one connected graph. The clustering accuracy naturally goes down under such a circumstance.

6. Conclusions

This paper addresses the graph connectivity problem of subspace clustering in order to bridge the gap between subspace-preserving solution and exact clustering results. We show that the graph connectivity of the sparse subspace clustering methods can be significantly improved by introducing an effective projection to the SSC-OMP, and thus, improving the clustering accuracy. Theoretical guarantee on exact clustering based on mild subspace-preserving condition is also provided. Extensive experiments on synthetic and real-world datasets are conducted. The results demonstrate that the proposed algorithms overcome the drawbacks of both LASSO-based and greedy-based algorithms, achieving a better clustering accuracy with a relatively faster processing speed. In addition, to improve the graph connectivity of the representation of the hidden layers of the deep learning-based subspace clustering methods is a promising future direction.

Acknowledgement

This research was sponsored by Ministry of Science and Technology, Taiwan under the grant 104-2622-E005-011, National Natural Science Foundation of China under grant No. 61573048, 61620106012, and International Scientific and Technological Cooperation Projects of China under grant No. 2015DFG12650.

A1. Proof of Theorem 1

Theorem 11. Let $i_{k,OMP}^*$, $S_{T_{k+1},OMP}$, and $\mathbf{r}_{k,OMP}$ be the index, the subspace spanned by $\mathcal{D}_{T_{k+1}}$, and the residual in the k th iteration in the OMP algorithm, respectively. Let $i_{k,GOMP^{pre}}^*$, $S_{T_{k+1},GOMP^{pre}}$, and $\mathbf{r}_{k,GOMP^{pre}}$ be the corresponding variables in the $GOMP^{pre}$ algorithm. We have:

$$i_{k,OMP}^* = i_{k,GOMP^{pre}}^* \quad (11)$$

$$S_{T_{k+1},OMP} = S_{T_{k+1},GOMP^{pre}} \quad (12)$$

$$\mathbf{r}_{k+1,OMP} = \mathbf{r}_{k+1,GOMP^{pre}} \quad (13)$$

for $k = 0, \dots, s$.

First we need the following Lemmas to prove Theorem 1.

Lemma 1. In the subspace inflation step where $\tilde{T}_{k+1}$ is updated to T_{k+1} according to Eq. (5), the support set will not shrink, i.e.,

$$\tilde{T}_{k+1} \subseteq T_{k+1} \quad (14)$$

Proof. It is equivalent to show that for $\forall \mathbf{d} \in \mathcal{D}_{\tilde{T}_{k+1}}$, it satisfies:

$$(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}})\mathbf{d} = \mathbf{0} \quad (15)$$

Let $\mathbf{D}_{\tilde{T}_{k+1}}$ be the matrix containing the points indexed by $\tilde{T}_{k+1}$ as its columns. The orthogonal projection can be written as $\mathbf{P}_{\tilde{T}_{k+1}} = \mathbf{D}_{\tilde{T}_{k+1}} \mathbf{D}_{\tilde{T}_{k+1}}^+$, where $\mathbf{D}_{\tilde{T}_{k+1}}^+$ is the Moore–Penrose pseudoinverse matrix of $\mathbf{D}_{\tilde{T}_{k+1}}$. By definition of the Moore–Penrose pseudoinverse, we have

$$\mathbf{D}_{\tilde{T}_{k+1}} \mathbf{D}_{\tilde{T}_{k+1}}^+ \mathbf{D}_{\tilde{T}_{k+1}} = \mathbf{D}_{\tilde{T}_{k+1}} \quad (16)$$

$$(\mathbf{I} - \mathbf{D}_{\tilde{T}_{k+1}} \mathbf{D}_{\tilde{T}_{k+1}}^+) \mathbf{D}_{\tilde{T}_{k+1}} = \mathbf{0} \quad (17)$$

This implies that Eq. (15) holds for $\forall \mathbf{d} \in \mathcal{D}_{\tilde{T}_{k+1}}$, which proofs the Lemma. $\square$

Lemma 2. In the subspace inflation step, we have:

$$\mathcal{S}_{\tilde{T}_{k+1}} = \mathcal{S}_{T_{k+1}} \quad (18)$$

Proof. According to Lemma 1, T_{k+1} can be divided into two parts, $T_{k+1} = \tilde{T}_{k+1} \cup \tilde{T}_{k+1}^C$, where $\tilde{T}_{k+1}^C = T_{k+1} \setminus \tilde{T}_{k+1}$. Note that $\tilde{T}_{k+1}^C$ can be empty if there is no more points satisfying Eq. (5) except those indexed by $\tilde{T}_{k+1}$. If $\tilde{T}_{k+1}^C = \emptyset$, then $\tilde{T}_{k+1} = T_{k+1}$, and the lemma holds. Thus, we focus on the case when $\tilde{T}_{k+1}^C \neq \emptyset$.

Let $\mathbf{d}_j$ be a point indexed by $\tilde{T}_{k+1}^C$, $\forall j \in \tilde{T}_{k+1}^C$. $\mathbf{d}_j$ satisfies

$$(\mathbf{I} - \mathbf{P}_{\tilde{T}_{k+1}})\mathbf{d}_j = 0 \quad (19)$$

It implies that $\mathbf{d}_j \in \mathcal{S}_{\tilde{T}_{k+1}}$, $\forall j \in \tilde{T}_{k+1}^C$. Therefore, $\mathcal{S}_{\tilde{T}_{k+1}^C}$ is a subspace of $\mathcal{S}_{\tilde{T}_{k+1}}$. $\square$

Lemmas 1 and 2 imply that for the subspace inflation step, all the points living in $\mathcal{S}_{\tilde{T}_{k+1}}$ will be incorporated in the support set, and the subspace $\mathcal{S}_{\tilde{T}_{k+1}}$ remains unchanged.

Lemma 3. Let $\mathcal{D}_1, \mathcal{D}_2$ be two sets of points with $\text{span}(\mathcal{D}_1) = \text{span}(\mathcal{D}_2)$. When added respectively to both sets by a same point $\mathbf{d}$, i.e., $\tilde{\mathcal{D}}_1 = \mathcal{D}_1 \cup \{\mathbf{d}\}$, $\tilde{\mathcal{D}}_2 = \mathcal{D}_2 \cup \{\mathbf{d}\}$, we have,

$$\text{span}(\tilde{\mathcal{D}}_1) = \text{span}(\tilde{\mathcal{D}}_2) \quad (20)$$

We skip the proof of this lemma because it can be proved by simple linear algebra. Now we start to prove Theorem 1 by mathematical induction.

Proof. It is easy to show that Eqs. 6–13 hold when $k = 0$. Suppose they hold when $k = k_0$ with $k_0 \geq 0$. When $k = k_0 + 1$, we have

$$\mathbf{r}_{k,\text{OMP}} = \mathbf{r}_{k,\text{GOMP}^{\text{pre}}} \quad (21)$$

$$\mathbf{i}_{k,\text{OMP}}^* = \mathbf{i}_{k,\text{GOMP}^{\text{pre}}}^* \quad (22)$$

this is true because the residuals from the last iteration are equal, and the indices are computed by the same equation.

Note that $T_{k,\text{OMP}} \neq T_{k,\text{GOMP}^{\text{pre}}}$, but we have

$$\mathcal{S}_{T_{k,\text{OMP}}} = \mathcal{S}_{T_{k,\text{GOMP}^{\text{pre}}}}. \quad (23)$$

Consider the two point sets $\mathcal{D}_{T_{k,\text{OMP}}}$ and $\mathcal{D}_{T_{k,\text{GOMP}^{\text{pre}}}}$. They are, respectively, updated to $\mathcal{D}_{T_{k+1,\text{OMP}}}$ and $\mathcal{D}_{\tilde{T}_{k+1,\text{GOMP}^{\text{pre}}}}$ by adding the same point $\mathbf{d}_{i_{k+1,\text{OMP}}^*}$. According to Lemma 3 we get,

$$\mathcal{S}_{T_{k+1,\text{OMP}}} = \mathcal{S}_{\tilde{T}_{k+1,\text{GOMP}^{\text{pre}}}}. \quad (24)$$

According to Lemma 2 we have

$$\mathcal{S}_{T_{k+1,\text{OMP}}} = \mathcal{S}_{T_{k+1,\text{GOMP}^{\text{pre}}}} \quad (25)$$

because the residuals $\mathbf{r}_{k+1,\text{OMP}}$ and $\mathbf{r}_{k+1,\text{GOMP}^{\text{pre}}}$ are computed by $\mathbf{y}$ minus the projection of $\mathbf{y}$ onto the same subspace, they are equal. So far, all the three equations are verified for $k = k_0 + 1$. It follows that the claim of this theorem holds for all $k \geq 0$. $\square$

References

- [1] M. Abavisani, V.M. Patel, Multimodal sparse and low-rank subspace clustering, *Inf. Fus.* 39 (2018) 168–177.
- [2] T.E. Boult, L.G. Brown, Factorization-based segmentation of motions, in: *Proceedings of the IEEE Workshop on Visual Motion*, 1991, pp. 179–186.
- [3] J. Bruna, S. Mallat, Invariant scattering convolution networks, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1872–1886.
- [4] X. Cai, W. Li, R. Zhang, Enhancing diversity and coverage of document summaries through subspace clustering and clustering-based optimization, *Inf. Sci. (Ny)* 279 (2014) 764–775.
- [5] E.J. Candes, The restricted isometry property and its implications for compressed sensing, *C.R. Math.* 346 (9–10) (2008) 589–592.
- [6] E.J. Candès, J. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, *IEEE Trans. Inf. Theory* 52 (2) (2006) 489–509.
- [7] M.A. Davenport, M.B. Wakin, Analysis of orthogonal matching pursuit using the restricted isometry property, *IEEE Trans. Inf. Theory* 56 (9) (2010) 4395–4401.
- [8] Z. Deng, K.S. Choi, F.L. Chung, S. Wang, Enhanced soft subspace clustering integrating within-cluster and between-cluster information, *Pattern Recognit.* 43 (3) (2010) 767–781.
- [9] Z. Deng, K.-S. Choi, Y. Jiang, J. Wang, S. Wang, A survey on soft subspace clustering, *Inf. Sci. (Ny)* 348 (2016) 84–106.
- [10] Z. Deng, Y. Jiang, F.L. Chung, H. Ishibuchi, K.S. Choi, S. Wang, Transfer prototype-based fuzzy clustering, *IEEE Trans. Fuzzy Syst.* 24 (5) (2016) 1210–1232.
- [11] E.L. Dyer, A.C. Sankaranarayanan, R.G. Baraniuk, Greedy feature selection for subspace clustering, *J. Mach. Learn. Res.* 14 (1) (2013) 2487–2517.
- [12] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, Springer New York, 2010.
- [13] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2790–2797.

- [14] Y. Fang, R. Wang, B. Dai, X. Wu, Graph-based learning via auto-grouped sparse regularization and kernelized extension, *IEEE Trans. Knowl. Data Eng.* 27 (1) (2015) 142–154.
- [15] M. Forina, R. Leardi, C. Armanino, S. Lanteri, Parvus: an extendible package for data exploration, classification and correlation, *J. Chemom.* 4 (2) (1990) 191–193.
- [16] A.S. Georgiades, P.N. Belhumeur, D. Kriegman, From few to many: illumination cone models for face recognition under variable lighting and pose, *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (6) (2001) 643–660.
- [17] X. He, S. Yan, Y. Hu, P. Niyogi, H.-J. Zhang, Face recognition using Laplacian faces, *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (3) (2005) 328–340.
- [18] J.R. Hershey, Z. Chen, J.L. Roux, S. Watanabe, Deep clustering: Discriminative embeddings for segmentation and separation, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 31–35.
- [19] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, D. Kriegman, Clustering appearances of objects under varying illumination conditions, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2003.
- [20] W. Hong, J. Wright, K. Huang, Y. Ma, Multiscale hybrid linear models for lossy image representation, *IEEE Trans. Image Process.* 15 (12) (2006) 3655–3671.
- [21] L.A. Kurgan, K.J. Cios, R. Tadeusiewicz, M. Ogiela, L.S. Goodenday, Knowledge discovery approach to automated cardiac spect diagnosis, *Artif. Intell. Med.* 23 (2) (2001) 149–169.
- [22] K. Lai, L. Bo, X. Ren, D. Fox, A large-scale hierarchical multi-view RGB-D object dataset, in: *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011, pp. 1817–1824.
- [23] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
- [24] C.-G. Li, R. Vidal, Structured sparse subspace clustering: a unified optimization framework, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2015, pp. 277–286.
- [25] C.G. Li, C. You, R. Vidal, Structured sparse subspace clustering: a joint affinity learning and subspace clustering framework, *IEEE Trans. Image Process.* 26 (6) (2017) 2988–3001.
- [26] Lichman M., *UCI machine learning repository*, 2013. <http://archive.ics.uci.edu/ml>.
- [27] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (1) (2013) 171–184.
- [28] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: *Proceedings of the International Conference on Machine Learning*, 2010, pp. 663–670.
- [29] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, S. Yan, Robust and efficient subspace segmentation via least squares regression, in: *Proceedings of the European Conference on Computer Vision*, 2012, pp. 347–360.
- [30] Y. Ma, H. Derksen, W. Hong, J. Wright, Segmentation of multivariate mixed data via lossy data coding and compression, *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (9) (2007) 1546–1562.
- [31] B. Nasihatkon, R. Hartley, Graph connectivity in sparse subspace clustering, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2137–2144.
- [32] Y. Panagakis, C. Kotropoulos, Elastic net subspace clustering applied to pop/rock music structure analysis, *Pattern Recognit. Lett.* 38 (2014) 46–53.
- [33] D. Park, C. Caramanis, S. Sanghavi, Greedy subspace clustering, in: *Proceedings of the Advances in Neural Information Processing Systems*, 2014, pp. 2753–2761.
- [34] Y.C. Pati, R. Rezaifar, P. Krishnaprasad, Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, in: *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, vol. 1, 1993, pp. 40–44.
- [35] X. Peng, S. Xiao, J. Feng, W.Y. Yau, Z. Yi, Deep subspace clustering with sparsity prior, in: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2016, pp. 1925–1931.
- [36] J. Siebert, *Vehicle Recognition Using Rule - Based Methods*, Turing Institute, Glasgow, Scotland, 1987.
- [37] V.G. Sigillito, S.P. Wing, L.V. Hutton, K.B. Baker, Classification of radar returns from the ionosphere using neural networks, *Johns Hopkins APL Tech Dig* 10 (3) (1989) 262–266.
- [38] J.A. Tropp, Greed is good: algorithmic results for sparse approximation, *IEEE Trans. Inf. Theory* 50 (10) (2004) 2231–2242.
- [39] R. Vidal, Subspace clustering, *IEEE Signal Process Mag* 28 (2) (2011) 52–68.
- [40] Y. Wang, Y.-X. Wang, A. Singh, A deterministic analysis of noisy sparse subspace clustering for dimensionality-reduced data, in: *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1422–1431.
- [41] Y. Wang, Y.-X. Wang, A. Singh, Graph connectivity in noisy sparse subspace clustering, 2016.
- [42] Y.-X. Wang, H. Xu, C. Leng, Provable subspace clustering: When LRR meets SSC, in: *Proceedings of the Advances in Neural Information Processing Systems*, 2013, pp. 64–72.
- [43] J. Yan, M. Pollefeys, A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate, in: *Proceedings of the European Conference on Computer Vision*, vol. 2, 2006, pp. 94–106.
- [44] Y. Yang, J. Feng, N. Jojic, J. Yang, T.S. Huang, L0-sparse subspace clustering, in: *Proceedings of the European Conference on Computer Vision*, 2016, pp. 731–747.
- [45] C. You, C.-G. Li, D.P. Robinson, R. Vidal, Oracle based active set algorithm for scalable elastic net subspace clustering, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3928–3937.
- [46] C. You, D. Robinson, R. Vidal, Scalable sparse subspace clustering by orthogonal matching pursuit, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3918–3927.
- [47] C. You, R. Vidal, Geometric conditions for subspace-sparse recovery, in: *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1585–1593.
- [48] L. Zelnik-Manor, M. Irani, Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2003, pp. 287–293.
- [49] L. Zhuang, H. Gao, Z. Lin, Y. Ma, X. Zhang, N. Yu, Non-negative low rank and sparse graph for semi-supervised learning, in: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2328–2335.