



# Scalable and robust sparse subspace clustering using randomized clustering and multilayer graphs



Maryam Abdolali<sup>a</sup>, Nicolas Gillis<sup>b,1,\*</sup>, Mohammad Rahmati<sup>a</sup>

<sup>a</sup> Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran

<sup>b</sup> Department of Mathematics and Operational Research, Faculté Polytechnique, Université de Mons, Rue de Houdain 9, Mons 7000, Belgium

## ARTICLE INFO

### Article history:

Received 18 December 2018

Revised 13 May 2019

Accepted 15 May 2019

Available online 16 May 2019

### Keywords:

Sparse subspace clustering

Randomized clustering

Hierarchical clustering

Multilayer graph

Spectral clustering

## ABSTRACT

Sparse subspace clustering (SSC) is a state-of-the-art method for partitioning data points into the union of subspaces. However, it is not practical for large datasets as it requires solving a LASSO problem for each data point, where the number of variables in each LASSO problem is the number of data points. To improve the scalability of SSC, we propose to select a few sets of anchor points using a randomized hierarchical clustering method, and, for each set of anchor points, solve the LASSO problems for each data point allowing only anchor points to have a non-zero weight. This generates a multilayer graph where each layer corresponds to a set of anchor points. Using the Grassmann manifold of orthogonal matrices, the shared connectivity among the layers is summarized within a single subspace. Finally, we use  $k$ -means clustering within that subspace to cluster the data points, as done by SSC. We show on both synthetic and real-world datasets that the proposed method not only allows SSC to scale to large-scale datasets, but that it is also much more robust as it performs significantly better on noisy data and on data with close subspaces and outliers, while it is not prone to oversegmentation.

© 2019 Elsevier B.V. All rights reserved.

## 1. Introduction

Finding a low-dimensional subspace which best represents a set of high-dimensional data points is a fundamental problem in many fields such as machine learning and signal processing. In fact, dimensionality reduction is an essential tool for understanding and preprocessing data sets. Using the assumption that the data points have fewer degrees of freedom than the ambient high dimension, several methods were developed to discover the underlying low-dimensional structure. Principal component analysis (PCA) is the most popular method for this matter [1]. However, the classical methods neglect the fact that the data set often contains data of different intrinsic structures. For example, facial images of multiple individuals under varying illumination conditions belong to multiple manifolds/subspaces and should be approximated by multiple subspaces instead of one [2]. This leads to a more general problem often referred to as *subspace clustering* that has attracted much attention recently. It generalizes the classical PCA in order to model data belonging to multiple subspaces; see, e.g., the survey [3] and the references therein.

Formally the subspace clustering problem can be defined as follows: given a set of  $N$  data points  $X = \{x_i \in \mathbb{R}^d\}_{i=1}^N$  from the union of  $n$  unknown subspaces  $S_1, S_2, \dots, S_n$  with unknown intrinsic dimensions  $d_1, d_2, \dots, d_n$  ( $d_i < d$  for all  $i$ ), the goal is to partition/cluster the data points according to their underlying subspaces and estimate the subspace parameters corresponding to each cluster.

Over the past two decades, many methods have been proposed to deal with this problem. They are usually classified in four categories [3]: iterative, statistical, algebraic and spectral-based methods. Iterative methods [4–6] formulate the problem as a non-convex optimization problem and optimize it using a two-step iterative approach, similarly as  $k$ -means: Given an initial clustering, alternatively (i) calculate a basis for each subspace, and (ii) assign each point to the closest subspace. Despite being simple and intuitive, the convergence is only guaranteed to a local minimum and it is highly sensitive to noise and outliers. Statistical approaches [7,8] treat the problem as modeling the data with mixture of Gaussian distributions. Similar to iterative methods, statistical methods are sensitive to noise and outliers. The algebraic methods [9,10] fit a set of polynomials to the data points using an algebraic and geometric reformulation of the problem. However, not only the methods in this category are sensitive to noise and outliers but also they do not scale well with the increase of the dimension of the data points. Spectral-based methods are inspired

\* Corresponding author.

E-mail addresses: [mabdolali@aut.ac.ir](mailto:mabdolali@aut.ac.ir) (M. Abdolali), [nicolas.gillis@umons.ac.be](mailto:nicolas.gillis@umons.ac.be) (N. Gillis), [rahmati@aut.ac.ir](mailto:rahmati@aut.ac.ir) (M. Rahmati).

<sup>1</sup> EURASIP member.

by classical spectral clustering techniques [11,12] based on using the spectrum of a specially constructed similarity matrix from the data points. The main difference between methods in this category is in the way the similarity matrix is constructed. Global spectral based approaches such as spectral curvature clustering (SCC) [13] tend to construct better similarity matrices (at the expense of a higher computational cost and more sensitivity to noise) compared to local based alternatives [14,15]. However, with advances in sparse representations (see, e.g., [16] and the references therein), a new set of methods has attracted a lot of attention within the spectral-based approaches. The key idea is that each data point can be expressed as a sparse linear combination of other data points within the same subspace. A property which is often referred to as *self-expressiveness*. Three main representative methods in this category are sparse subspace clustering (SSC) [17,18], low-rank representation based clustering (LRR) [19,20], and least square regression (LSR) [21]. All three approaches are based on the following model:

$$\begin{aligned} \min_{C \in \mathbb{R}^{N \times N}, E \in \mathbb{R}^{d \times N}} & f(C) + \lambda g(E) \\ \text{such that } & X = XC + E \text{ and } C_{i,i} = 0 \text{ for all } i, \end{aligned} \quad (1)$$

where  $X \in \mathbb{R}^{d \times N}$  is the input matrix whose columns are the data points,  $E \in \mathbb{R}^{d \times N}$  is the noise, and  $C \in \mathbb{R}^{N \times N}$  is the coefficient matrix. The function  $f(\cdot)$  is

- the component-wise  $\ell_1$  norm  $\|C\|_1 = \sum_{i,j} |C_{i,j}|$  for SSC which enhances the sparsity of  $C$  (a LASSO problem),
- the nuclear norm  $\|C\|_*$  for LRR, that is, the sum of the singular values of  $C$ , which enhances  $C$  to be of low-rank, and
- the Frobenius norm  $\|C\|_F = \sum_{i,j} C_{i,j}^2$  for LSR which enhances  $C$  to have low energy (this usually does not lead to sparse nor low-rank solutions, but it is computationally much cheaper as the solution can be written in closed form).

The function  $g$  is a regularization function that is used for modeling the noise (usually based on the  $\ell_1$  or the Frobenius norm). The solution  $C$  to (1) provides crucial information about the links between the data points:  $C_{ij} \neq 0$  means that the data points  $i$  and  $j$  share some information hence it is likely they belong to the same subspace. These methods then apply spectral clustering on the graph corresponding to the adjacency matrix  $|C| + |C|^T$ . Note that an important strength of these methods is that they do not need to know the dimensions  $d_i$ 's of the subspaces, and can estimate the number of clusters as done by spectral clustering (looking at the decay of the eigenvalues of the Laplacian of the adjacency matrix).

SSC has strong theoretical guarantees in noisy and noiseless cases for both independent and disjoint subspaces [18,22–24]. The behavior of LRR in the presence of noise and disjoint subspaces is still not well understood [25]. LSR is computationally less demanding compared to SSC and LRR, but it is guaranteed to preserve the subspaces only when they are independent. Despite theoretical guarantees and empirical success of SSC, optimizing the corresponding LASSO problems is not practical for large real-world data sets with more than 10,000 samples, the bottleneck being the resolution of the large-scale convex optimization problem (1) (although it can be decoupled into  $N$  independent LASSO problems), with a computational cost of  $\Omega(N^2)$  operations. To overcome this issue, we propose a new scalable approach, referred to as *scalable and robust SSC* (SR-SSC), based on solving several problems of the form (1) but where only a few rows of  $C$  are allowed to be non-zero (using a randomized subsampling of anchor points) in order to generate a multilayer graph that we merge using the technique from [26]. The computational cost is linear in  $N$  so that SR-SSC can scale to much larger data sets.

## 1.1. Outline and contribution

The main contribution of this paper is to propose a novel scalable approach that extends SSC for clustering large-scale data sets. The two key ingredients of this novel method are (i) a randomized hierarchical clustering algorithm that allows to identify different sets of anchor points that are good representatives of the data set, and (ii) a multilayer graph technique that summarizes the information across several graphs on the same vertices. As we will show, our method significantly improves the performance of SSC in the challenging cases of noisy and close subspaces, and overcomes the oversegmentation issue of SSC.

The paper is organized as follows. In Section 2, we briefly review and discuss the related works proposing scalable SSC-like methods. In Section 3, we present a new approach that makes SSC scalable to large data sets and is more robust, which we refer to as scalable and robust SSC (SR-SSC). We investigate the properties and performance of SR-SSC using synthetic and real-world data sets in Section 4. Section 5 concludes the paper.

## 2. Related works

Several methods in the literature have already addressed the scalability of SSC. One of the earliest attempts is the method referred to as scalable SSC (SSSC) [27] that applies SSC on a randomly selected subset of data points and assigns the rest of the data points based on the obtained clusters. This approach is very sensitive to the selection method, and suboptimal because the data points which were not selected are not taken into account for generating the clusters.

In [28], instead of solving the LASSO problems (1), authors use a greedy algorithm, namely orthogonal matching pursuit (OMP), to obtain the sparse representation for each data point. Even though OMP is faster than the original  $\ell_1$ -based SSC and is scalable for up to 100,000 samples, it is a greedy method with weaker theoretical guarantees [28,29]. Also, the computational cost is still high, requiring  $O(N^2)$  operations. In each iteration of OMP to approximate a given data point, the residual is orthogonal to all previously selected data points. This property enforces constraints on sampling distribution in each subspace [30]. Furthermore, OMP tends to partition subspaces into multiple components (oversegmentation issue) [31].

Mixture of  $\ell_1$  and  $\ell_2$  norms was used in [32] to take advantage of subspace preserving of the  $\ell_1$  norm and the dense connectivity of the  $\ell_2$  norm. Later in [33], an oracle-based algorithm, dubbed ORacle Guided Elastic Net solver (ORGEN), was proposed to identify a support set for each sample efficiently. However, in this approach, a convex optimization problem is solved several times for each sample which can limit the scalability of the algorithm. Besides, to the best of our knowledge, there is no principled method to choose the regularization parameters in ORGEN which has a significant role in the trade-off between  $\ell_1$  (subspace preserving) and  $\ell_2$  (connectivity).

More recently, a nearest neighbor filtering approach (KSSC) [34] was presented to choose the support set for each sample more efficiently compared to ORGEN. In this approach, the LASSO problem in SSC is restricted to the  $K$  nearest neighbors chosen as support set for each point. Even though authors provided some theoretical guarantees for correct connections in noisy and noiseless cases, this approach requires to find the nearest neighbors for each point which requires  $O(N^2)$  operations. Additionally this approach is very likely to oversegment the subspaces. Let us illustrate this with a simple example similar to that in [35] (we will use a similar example in Section 4). Suppose  $8m$  points are chosen around four circles in a 4-dimensional subspace as follows:

the columns of the input matrix  $X$  are of the form

$$[\cos \theta_k, \sin \theta_k, s\delta, s'\delta]^T,$$

where  $k = 0, 1, \dots, 2m - 1$ ,  $\theta_k = \frac{\pi k}{m}$ ,  $s, s' \in \{-1, 1\}$  and  $\delta \in \mathbb{R}$ . Let us denote  $x_i$  the  $i$ th column of  $X$  ( $i = 1, 2, \dots, 8m$ ). The symmetrized convex hull of the columns of  $X$  is defined as

$$P_X = \text{conv}(\pm x_1, \pm x_2, \dots, \pm x_{8m}).$$

Solving (1) is equivalent to finding the extreme points of the closest face of the polytope  $P_X$  for each (normalized) data point [36]. It can be proved that by choosing a sufficiently large value for  $m$ , the vertices corresponding to closest face to the point  $[\cos \theta_k, \sin \theta_k, s\delta, s'\delta]^T$  are [35]:

$$[\cos \theta_{k\pm 1}, \sin \theta_{k\pm 1}, s\delta, s'\delta]^T, [\cos \theta_k, \sin \theta_k, -s\delta, s'\delta]^T,$$

$$\text{and } [\cos \theta_k, \sin \theta_k, s\delta, -s'\delta]^T.$$

Depending of  $m$ , the value of  $\delta$  can be chosen large enough so that the  $K$  nearest neighbors are  $[\cos \theta_{k+\ell}, \sin \theta_{k+\ell}, s\delta, s'\delta]^T$  where  $\ell = 1, -1, 2, -2, \dots$ . This leads to the following four disjoint components for the subspaces (4 subspaces of dimension 2), even in the noiseless case:

$$\left\{ [\cos \theta_k, \sin \theta_k, s\delta, s'\delta]^T \right\}_{k=0}^{2m-1} \quad \text{with } s, s' \in \{-1, 1\}.$$

This is due to the fact that nearest points do not necessarily contain all the vertices of the closest face in the polytope. This observation is well-supported by sparse representation literature as well, where several *screening rules* based algorithms [37,38] were proposed to predict the *active* support set for each data point in the LASSO optimization problem. More recently, [39] used sketching to speed up SSC, LRR and LSR, but their approach sacrifices accuracy for efficiency.

As a conclusion, we see that there is a need for an efficient algorithm that is computationally cheaper than SSC, ideally running in  $O(N)$  where  $N$  is the number of data points, while not worsening its weaknesses. In the next section, we present an approach that selects anchor points randomly and a framework that can compensate for possible inaccurate selection. As we will see, the proposed approach is not only scalable but more robust than SSC and not prone to oversegmentation.

### 3. Robust and scalable sparse subspace clustering algorithm using randomized clustering and multilayer graphs

In order to keep the good performances of SSC while reducing the computational cost, we propose an approach in which a small number of samples is chosen as anchor points so that only a few rows of  $C$  are allowed to be non-zero in (1), similarly as for the methods presented in Section 2. Clearly, the choice of the anchor points plays a critical role. Moreover, the computational cost of the method for choosing the anchor points should be low otherwise there would be no gain in throwing away the rest of the points (in particular, picking the  $K$  nearest neighbors is rather expensive; see the discussion in the previous section). However, choosing the anchor points with no prior knowledge is a difficult task. To reduce the role of the anchor points, we propose a new framework which constructs a multilayer graph based on several sets of well-chosen anchor points using randomization. The details of the two key steps are discussed in the following two sections, namely the selection of the anchor points and the summarization of the multilayer graph.

#### 3.1. Selection of the sets of anchor points via randomized hierarchical clustering

In this section, we describe the proposed method for selecting the sets of anchor points in the data set. Classical ap-

proaches such as  $k$ -medoids [40] and affinity propagation-based algorithms [41,42] have quadratic time complexity which limits their applications for large-scale problems. Another group of algorithms select the anchor points using the low-dimensional structure of the data. These methods assume the data lies on one or multiple low-dimensional subspaces. The rank revealing QR and similar methods [43,44] select the samples such that the resulting submatrix is well-conditioned. There are also randomized methods that select candidates using a concept known as leverage scores. These scores are proportional to the  $\ell_2$  norm of the rows of the right singular vectors of the data matrix. Some approaches select the columns with highest scores [45] but it is more common to use these scores to form a biased probability distribution [46] (that is, sampling the columns using a nonuniform probability distribution that depends on the calculated leverage scores). Apart from the time consuming SVD factorization (to obtain the right singular vectors), it is possible to have a considerable amount of redundancy in the final set. Moreover, to reach the theoretical guarantees of these approaches, it is recommended to repeat the algorithm around 40 times which makes it impractical for large-scale problems [44]. Another class of method uses convex relaxations [47,48] but these methods requires the introduction of  $N^2$  variables and are impractical for our purpose. Hence to the best of our knowledge, there is currently no approach that can select samples from a data set that lies on a union of subspaces in a reasonable amount of time. Instead of selecting the anchor points fully randomly as done in [27], we select them in a way so that they are well-spread in the data set, that is, they are good representatives of the data points. To do so, a simple randomized top-down hierarchical clustering technique is used. It works as follows. We construct a tree structure and start with a single root node which contains all the data points (with no clusters/partitions). Each node in this tree structure is a collection of data points. Let the data point that is closest to the average of data within node be the centroid of that node. Centroid can be considered as representative data point of each node. Then at each step, we

1. Select the node whose sum of the squared Euclidean distances between the data points it contains and their centroid is the largest. Let us use  $I$  to denote the set of indices of the data points corresponding to this node.
2. Split this node as two child nodes as follows:
  - (a) Generate randomly a vector  $v \in \mathbb{R}^d$  (we used the Gaussian distribution  $N(0, 1)$  for each entry of  $v$ ),
  - (b) Project each data point  $x_i$  within the current node onto the one-dimensional subspace spanned by  $v$ , using the inner product with  $v$ , that is, compute,  $v^T x_i$  for  $i \in I$ .
  - (c) Choose a threshold  $\delta$  so that the child nodes

$$I_1(\delta) = \{i \mid v^T x_i > \delta, i \in I\} \quad \text{and}$$

$$I_2(\delta) = \{i \mid v^T x_i \leq \delta, i \in I\}$$

satisfy two properties: they are well-balanced (that is, they contain roughly the same number of data points) and stable (that is, modifying  $\delta$  slightly does not modify the two child nodes significantly). To do so, we use the technique proposed in [49] using a simple way to quantify each property: the fact that the nodes are well-balanced is measured using  $F(\delta) = \frac{|I_1(\delta)|}{|I|}$  which should be close to  $\frac{1}{2}$ , where  $|\cdot|$  indicates the cardinality of a set, and the stability is measured using

$$G(\delta) = \frac{1}{|I|(\bar{\delta} - \underline{\delta})} \left| \{x_i \mid \underline{\delta} \leq v^T x_i \leq \bar{\delta}, i \in I\} \right|$$

where  $\bar{\delta} = \min(1, \delta + \hat{\delta})$ ,  $\underline{\delta} = \max(0, \delta - \hat{\delta})$ , and  $\hat{\delta}$  is a small parameter (we used 0.01 as suggested in [49]). To have

stable clusters,  $G(\delta)$  should be close to zero, that is, modifying  $\delta$  should not transfer many data points from one node to the other.<sup>2</sup> Finally,  $\delta$  is chosen such that  $H(\delta) = -\log(F(\delta)(1-F(\delta))) + G(\delta)^2$  is minimized. Of course, many other choices are possible for  $H(\delta)$ , but the overall algorithm is not too sensitive to this choice as long as both  $F$  and  $G$  are taken into account. In fact, we have observed for example that choosing  $H(\delta) = F(\delta)$  which simply makes the child nodes have the same number of data points (up to one if  $|I|$  is odd) does not perform as well.

The construction of this tree is continued until the number of leaf nodes in the tree reaches the number  $k$  of anchor points needed. In each node, we select an anchor point which is the data point that is the closest to the average of the node, that is, to the centroid of the node. The randomized algorithm for selecting the anchor points is summarized in Algorithm 1.

---

**Algorithm 1:** Randomized algorithm for selecting anchor points.

---

- Input:** Data matrix  $X \in \mathbb{R}^{d \times N}$ , and the desired number of anchor points  $k$ .
- Output:**  $k$  anchor points as the columns of the matrix  $D \in \mathbb{R}^{d \times k}$ , and the set  $\Omega$  containing their corresponding indices in  $X$ .
- 1: Initialization: Construct a tree structure with a single (root) node and set the data points contained this node to  $I = \{1, \dots, N\}$ .
  - 2: **while** the number of leaf nodes is less than  $k$  **do**
  - 3:   Select the node to split: the one with the maximum sum of squared Euclidean distances between its data points and its centroid.
  - 4:   Generate a random vector  $v \in \mathbb{R}^d$  and project the data points within the selected node onto vector  $v$ .
  - 5:   Select the threshold  $\delta$  such that  $H(\delta) = -\log(F(\delta)(1-F(\delta))) + G(\delta)^2$  is minimized.
  - 6:   Create two child nodes with  $I_1(\delta) = \{i \mid v^T x_i > \delta, i \in I\}$  and  $I_2(\delta) = \{i \mid v^T x_i \leq \delta, i \in I\}$ .
  - 7:   Select the closest data points to the average of the data points in  $I_1$  and  $I_2$  as the centroids of the corresponding nodes respectively and store their values and indices.
  - 8: **end while**
  - 9: Set the centroids of the final  $k$  leaves as the columns of the matrix  $D$ , and their corresponding indices as the set  $\Omega$ .
- 

The computational cost to split a node with  $N$  points in dimension  $d$  is  $O(Nd)$  operations (the most expensive step is to compute the inner products  $v^T x_i$  for all  $i$ ). To extract  $k$  anchors points, the overall procedure hence require at most  $O(kNd)$  operations. However, unless there are very unbalanced clusters in the data set, each child node will contain roughly half the data points of its parent node so that the expected computational cost is actually rather  $O(\log(k)Nd)$  operations in practice (splitting all the nodes of a single level of the tree requires  $O(Nd)$  operations since each level contains all the data points, and there will be  $O(\log(k))$  levels in most cases).

The idea of random projection is, we believe, a key step of SRSSC because it has the following three key properties that will make it successful:

<sup>2</sup> For stable clustering, ideally, no data point should be in the small interval defined by  $[\delta, \delta]$  so that  $|\{x_i \mid \delta \leq v^T x_i \leq \delta, i \in I\}| = 0$  hence  $G(\delta) = 0$ . In fact,  $G(\delta)$  counts (and normalizes) the number of points that belong to this interval. The closer this value is to zero, the less data points are in the interval and hence, the more stable the clustering is.

1. It selects anchor points that are good representatives of the data set since they are located nearby centroids of clusters that were generated in such a way that distant points are more likely to be in different clusters than nearby points.
2. It is computationally efficient, requiring in general  $O(\log(k)Nd)$  operations to identify  $k$  anchor points.
3. It generates different sets of anchor points, which is essential for our multilayer graph strategy described in the next section.

For example, using random sampling performs worse because it does not satisfy the first property above; see Section 4 where we also compare our approach to existing algorithms from the literature. Further research includes the design of other approaches to perform the selection of anchor points retaining these three properties.

### 3.2. Construction and summarization of the multilayer graph

Choosing the number of anchor points depends on many factors: the number of subspaces and their dimensions, the affinity between subspaces, the distribution of data points in each subspace, and the level of noise. Hence choosing a single set of ‘good’ anchor points, as done for example in [27], is highly non-trivial. To leverage this difficulty, we propose to choose several sets of different anchor points, as described in the previous section. Let us select  $L$  sets of  $k$  anchor points in the data sets, with indices  $\Omega^{(j)}$  ( $j = 1, 2, \dots, L$ ), and construct  $L$  dictionaries  $\{D^{(i)} = X(:, \Omega^{(i)}) \in \mathbb{R}^{d \times k}\}_{i=1}^L$  (note that we could select a different number of anchor points in each set, but we do not consider this case for simplicity). Similarly to SSC but only allowing anchor points to have a non-zero weight, we compute the set of sparse representation coefficients  $\{C^{(i)} \in \mathbb{R}^{k \times N}\}_{i=1}^L$  by solving

$$\min_{C^{(i)} \in \mathbb{R}^{k \times N}} \|C^{(i)}\|_1 + \frac{\mu}{2} \|X - D^{(i)}C^{(i)}\|_F^2$$

such that  $C_{j, \Omega^{(i)}(j)}^{(i)} = 0$  for  $j = 1, 2, \dots, k$ . (2)

The constraint ensures that no anchor point uses itself for self representation as in SSC. The optimization problem (2) can be solved for example using alternating direction method of multipliers (ADMM) [50]; see Algorithm 2 which we provide for completeness (the superscripts are dropped for convenience). ADMM is a good choice for our purpose: it has a low computational cost per iteration (linear in the number of variables), while its slow convergence (linear at best) is not a bottleneck since a high precision is not necessary as we only need to know the order of magnitude of

---

**Algorithm 2:** ADMM for (2) for dictionary-based SSC.

---

- Input:**  $X \in \mathbb{R}^{d \times N}$ ,  $\Omega$  as the indices of the  $k$  column of the dictionary, parameters  $\lambda$
- Output:** Approximate solution to the problem  $\min_{C \in \mathbb{R}^{k \times N}} \|C\|_1 + \frac{\mu}{2} \|X - X(:, \Omega)C\|_F^2$ .
- 1: Initialization:  $C = A = \Delta = 0$ ,  
 $D = X(:, \Omega)$ ,  $\mu = \frac{\lambda}{\max_{j, i \in \Omega(j)} |d_j^T x_i|}$ ,  $\rho = \lambda$
  - 2: **while** some convergence criterion is not met **do**
  - 3:    $A \leftarrow (\mu D^T D + \rho I_k)^{-1} (\mu D^T X + \rho C - \Delta)$
  - 4:    $C \leftarrow T_{\frac{\lambda}{\rho}}(A + \Delta/\rho)$   
      where  $T_\gamma(y) = \max(0, |y| - \gamma) \text{sign}(y)$  is the soft-thresholding operator
  - 5:    $C_{i, \Omega^{(i)}} = 0$  for  $i = 1, 2, \dots, k$
  - 6:    $\Delta = \Delta + \rho(A - C)$
  - 7: **end while**
-

the entries of  $C^{(i)}$  –also, the data is usually rather noisy hence it does not make much sense to solve (2) to high precision.

Once the coefficient matrices  $\{C^{(i)}\}_{i=1}^L$  are computed, a multilayer graph  $G$  with  $L$  layers  $G^{(i)} = (V, W^{(i)})$  is constructed: The set  $V$  contains the vertices, one for each data point, and  $W^{(i)}$  is the weighted adjacency matrix of the  $i$ th layer. Denoting  $E^{(i)}(\Omega(j), :) = C^{(i)}(j, :)$  for  $j = 1, 2, \dots, k$  while the other rows of  $E^{(i)}$  are equal to zero,  $W^{(i)}$  is obtained by symmetrizing  $E^{(i)}$ , that is,

$$W^{(i)} = |E^{(i)}| + |E^{(i)T}|, \quad (3)$$

similarly as for SSC. Note that  $W^{(i)}$  is a sparse matrix with less than  $2kN$  non-zero coefficients. Now, we need to combine the information from the individual graphs in each layer. To do so, we adopt the method presented in [26] to merge the information of different layers into a proper representation such that it strengthens the connectivity/information that majority of graphs tend to agree on. Let us briefly describe this technique. The problem of merging a multilayer graph is combined with the problem of merging different subspaces on a Grassmann manifold. For each individual graph  $G^{(i)}$ , we compute its  $p$ -dimensional subspace representation as the matrix  $U^{(i)} \in \mathbb{R}^{N \times p}$  whose columns are the eigenvectors corresponding to the  $p$  smallest eigenvalues of the normalized Laplacian matrix of the corresponding graph, that is,

$$\mathcal{L}^{(i)} = I - D_g^{(i)-1/2} W^{(i)} D_g^{(i)-1/2} \quad (4)$$

where  $D_g^{(i)}$  is the diagonal matrix where each diagonal element is the degree of the corresponding vertex in  $G^{(i)}$ . The problem of combining the multilayer graph is then formulated as follows [26]:

$$\min_{U \in \mathbb{R}^{N \times p}} \sum_{i=1}^L \text{trace}(U^T \mathcal{L}^{(i)} U) - \alpha \sum_{i=1}^L \text{trace}(U U^T U^{(i)} U^{(i)T})$$

such that  $U^T U = I$ . (5)

The above optimization problem finds a subspace representation that satisfies two goals. The first term makes sure the connectivity information of each individual graph is preserved within the subspace  $U$  (since this information is contained in their Laplacian matrices  $\mathcal{L}^{(i)}$ ). The second term is a distance metric that incites the subspace  $U$  to be close to the subspaces  $U^{(i)}$  corresponding to each graph (on the Grassmannian manifold  $U^T U = I$ ). The parameter  $\alpha$  balances the two terms. The solution  $U$  to problem (5) is the final subspace representation and can be obtained by finding the  $p$  eigenvectors corresponding to the smallest eigenvalues of

$$L_f = \sum_{i=1}^L \mathcal{L}^{(i)} - \alpha \sum_{i=1}^L U^{(i)} U^{(i)T}. \quad (6)$$

The final clusters are obtained by clustering rows of these eigenvectors using  $k$ -means, as for spectral clustering. As we will see in Section 4.1.1, the choice of the parameter  $\alpha$  will not influence the overall procedure as long as it is chosen in a reasonable range. Note that choosing  $\alpha = 0$  amounts to the naive strategy of summing up the adjacency matrices of the different layers, which will turn out to perform rather poorly.

**SR-SSC.** Finally, our proposed algorithm to perform scalable and robust sparse subspace clustering (SR-SSC) is summarized in Algorithm 3.

**Remark 1.** The choice of  $p$ , that is, of the number of clusters, can also be done automatically, as for SSC, for example by looking at the drop in the eigenvalues of the  $\mathcal{L}^{(i)}$ 's and use a majority voting.

---

**Algorithm 3:** Scalable and robust sparse subspace clustering (SR-SSC) using randomized hierarchical clustering and multilayer graphs.

---

**Input:** Data matrix  $X \in \mathbb{R}^{d \times N}$ , the number  $L$  of sets of anchor points (that is, the number of graphs), the number  $k$  of anchor points per set, the parameter  $\alpha$ , the number of clusters  $p$ .

**Output:** A clustering of the columns of  $X$  in subspaces

- 1: **for**  $i = 1, 2, \dots, L$  **do**
  - 2: Choose  $k$  anchor points which form the dictionary  $D^{(i)}$  using the randomized hierarchical procedure described in Section 3.1.
  - 3: Solve (2) to obtain  $C^{(i)}$  (we use Algorithm 2).
  - 4: Construct the symmetrized adjacency matrix  $W^{(i)}$  as in (3).
  - 5: Compute the normalized Laplacian matrix  $\mathcal{L}^{(i)}$  as in (4).
  - 6: Compute the  $p$  eigenvectors  $U^{(i)} \in \mathbb{R}^{N \times p}$  corresponding to the smallest eigenvalues of  $\mathcal{L}^{(i)}$ .
  - 7: **end for**
  - 8: Compute the final Laplacian matrix  $L_f$  as in (6).
  - 9: Compute the  $p$  eigenvectors  $U \in \mathbb{R}^{N \times p}$  corresponding to the smallest eigenvalues of  $L_f$ .
  - 10: The final clustering of the columns of  $X$  are obtained by clustering the rows of  $U$  using  $k$ -means with  $p$  clusters.
- 

### 3.3. Computational cost of SR-SSC

Let us analyze the computational cost of Algorithm 3, that is, of SR-SSC. Let us first analyze the for loop which is performed  $L$  times. Identifying the anchor points requires at most  $O(kNd)$  operations; see Section 3.1. The main computational cost for solving (2) using Algorithm 2 is performed at step 3 with a cost of  $O(k^2N)$  operations to compute  $D^T D$ ,  $O(k^3)$  operations to compute the inverse, and  $O(kdN)$  operations to compute  $D^T X$ . Assuming we fix the number of iterations of Algorithm 2 and since  $k \ll N$ , the total computational cost for solving (2) using Algorithm 2 is  $O(k^2N + kNd)$ .

Constructing  $\mathcal{L}^{(i)}$  can be done directly from  $C^{(i)}$  and requires  $O(kN)$  operations. To compute the eigenvectors of  $\mathcal{L}^{(i)}$ , we use ARPACK which is a sparse eigenvalue solver (since  $\mathcal{L}^{(i)}$  has a most  $2kN$  non-zero entries) which requires  $O(pkN)$  operations [51].

It remains to form  $L_f$  defined in (6), and compute the eigenvectors  $U$  corresponding to the  $p$  smallest eigenvalues. Note there is no need to form matrix  $L_f$  explicitly. The first term  $\sum_{i=1}^L \mathcal{L}^{(i)}$  of the global Laplacian matrix  $L_f$  contains at most  $O(LkN)$  non-zero elements (since each  $\mathcal{L}^{(i)}$  has at most  $O(kN)$  non-zero elements), and the second term has rank at most  $O(Lp)$  since each  $U^{(i)}$  has  $p$  columns. Computing the eigenvalue decomposition of a sparse (with  $O(LkN)$  non-zero entries) plus low-rank matrix (of rank  $Lp$ ) of dimension  $N \times N$  using ARPACK requires  $O(LNp(k+p))$  operations. In fact, ARPACK is based on implicitly restarted Arnoldi method which reduces to implicitly restarted Lanczos method when the matrix is symmetric (as in our case). Arnoldi/Lanczos are adaptations of the power method for finding a few eigenvalues and the corresponding eigenvectors of a large-scale structured or sparse matrix based on simple iterations using matrix-vector multiplications. Hence it is capable of obtaining the eigenvectors of a matrix for which the explicit stored form is not available as long as the matrix-vector multiplication can be done efficiently.

The total computational cost of SR-SSC is  $O(LN(k^2 + kd + p(k+p)))$  operations, which is linear in  $N$  hence scalable for large data sets as long as  $L$  and  $k$  remain small. As we will see, a value

of  $L$  below 10 is usually enough while  $k$  should not be significantly larger than the lower bound  $\sum_{i=1}^n (d_i + 1)$  (since we need to select at least  $d_i + 1$  points in the  $i$ th subspace for SSC to work properly). Since the  $d_i$ 's are usually unknown, a good choice is to pick  $k$  as a multiple of  $pd'$  where  $d' \leq d$  is a guess for the average dimension of the subspaces, e.g.,  $k = 10pd'$ .

Note that the original SSC algorithm corresponds to tacking  $L = 1$  and  $k = N$ , with a computational cost of  $O(N^3 + N^2d)$  operations (for our particular implementation using ADMM).

**Remark 2.** If the input matrix  $X$  is sparse, the computational cost is reduced further as the term  $Nd$  is replaced with the number of non-zero entries of  $X$ .

**Remark 3.** Our algorithm could be made faster by replacing SSC with LRR or LSR, or use OMP instead of LASSO. However, this will in most cases lead to a decrease in accuracy since the original SSC algorithm performs better than these approaches in general (cf. the discussion in Sections 1 and 2) while the main goal of this paper is to make SSC scalable.

**Remark 4.** In the recent paper [52], authors proposed a way to select anchor points sequentially that allows to provably identify  $d_\ell$  data points in each subspace, assuming these subspaces are independent. The idea is to extract anchors as long as the approximation error of the data points by these anchors is above a given threshold. As far as we know and as stated in [52], this is the only work that provides such a theoretical result for the selection of anchor points. It appears we can use exactly the same trick for the selection of our anchor points: Under the independent subspace assumption, as long as a less than  $d_\ell$  anchors have been extracted from the  $\ell$ th subspace, the reconstruction error of at least one data point in that subspace will be lower bounded (this depends on the angles between subspaces, and how the data points are spread within the  $\ell$ th subspace). Of course, in practice, we do not know the values of this lower bound, but we can stop extracting anchors when the reconstruction error for each data point is below some chosen threshold; as done in [52]. For this work, we have not used such a strategy; this is a topic for further investigation.

*Choice of the parameters  $\mu$  and  $\lambda$ .* The regularization parameter  $\mu$  in (2), which depends directly on  $\lambda$  in Algorithm 3, balances the importance between the data fitting term and the sparsity of the coefficient matrix. A smaller  $\mu$  leads to fewer wrong connections (enforcing sparsity) whereas a larger value for  $\mu$  increases the number of true connections (denser connectivity), which imposes a natural trade-off. In other words, we want to set  $\mu$  as small as possible while making sure it leads to sufficient connectivity. It was suggested in [18] (and previously in sparse representation literature) to set  $\mu$  as  $\mu = \lambda\mu_0$  with  $\lambda > 1$  and  $\mu_0 = \frac{1}{\max_{j \neq i} |x_j^T x_i|}$  where  $\mu_0$  is the smallest value for which the coefficient vector for some sample would be zero. The value for  $\mu$  (and hence  $\lambda$ ) that leads to the best results for SSC is data dependent. For data from subspaces with small intrinsic dimension, the regularization parameter should be small to enforce sparsity (as each data point should ideally use  $d$  samples to represent itself) and for subspaces with high intrinsic dimension, it should be chosen larger to relax sparsity. Hence, [53] suggested the regularization parameter should be in the order of  $\sqrt{d}$ . In this paper, we will fix the value of  $\lambda$  to some prescribed value that leads to reasonable results, without giving particular attention for fine tuning this parameter.

#### 4. Numerical experiments

In this section, we evaluate the performance of SR-SSC on both synthetic and real-world data sets. All experiments are imple-

mented in Matlab, and run on a computer with Intel(R) Core(TM) i7-3770 CPU, 3.40GHz, 16GB. The code is available from <https://sites.google.com/site/nicolasgillis/code>.

##### 4.1. Synthetic data sets

In this section, we investigate the performance of SR-SSC under different conditions and depending on the choice of the parameters: the number of graphs  $L$ , the number of anchor points  $k$ , and the value of  $\alpha$ . To do so, we use synthetic data sets that consist of three 10-dimensional spaces ( $d_1 = d_2 = d_3 = 10$ ) within a 20-dimensional space ( $d = 20$ ) with the following bases:

$$U_1 = \begin{pmatrix} \cos(\theta)I_{10} \\ \sin(\theta)I_{10} \end{pmatrix}, U_2 = \begin{pmatrix} \cos(\theta)I_{10} \\ -\sin(\theta)I_{10} \end{pmatrix}, U_3 = \begin{pmatrix} I_{10} \\ I_{10} \end{pmatrix},$$

where  $I_q$  is the identity matrix of dimension  $q$  and  $\theta \in [0, 2\pi]$ . For each subspace, we pick  $N/3$  random samples (we will choose  $N$  as a multiple of 3) generated as linear combinations of  $U_i$  where the weights in the linear combinations are chosen at random using the Gaussian distribution of mean 0 and variance 1 (in Matlab,  $U_1 * \text{randn}(10, N/3)$ ). Finally, an i.i.d. random Gaussian noise with zero mean and standard deviation  $\sigma$  is added to the data matrix (in Matlab,  $\sigma * \text{randn}(d, N)$ ). Similar to [22], the data points are normalized such that their  $\ell_2$  norm is equal to one.

The affinity between subspaces can be defined as an average of the cosine of the angles between subspaces; see [22]. It was shown in [22] that the larger the affinity, the more difficult it is for SSC to identify the right subspaces. For our particular synthetic data sets, decreasing the value of  $\theta$  from  $\frac{\pi}{2}$  to 0 increases the affinity and hence makes the subspace clustering task more challenging. For evaluating the performance of SR-SSC on this synthetic data set, the value for regularization parameter  $\mu$  in (2) is chosen as  $\mu = \lambda\mu_0$  where  $\mu_0$  is the minimum value which avoids a zero coefficient matrix  $C$ , and  $\lambda$  is set as 40.

*Budget.* To have a fair comparison between SR-SSC with different number of graphs and anchor points, we define the budget of SR-SSC as

$$\text{budget} = L \times k.$$

Hence for a fixed budget, a larger value of  $L$  will imply a smaller value of the number of anchor points  $k$ . Note that this choice of the budget favors SR-SSC with fewer graphs since the computational cost of SR-SSC is not linear in  $k$  (it is in  $L$ ); see Section 3.3. We made this choice for simplicity and to be conservative in the sense that using more graphs will not increase the computational cost for a fixed budget.

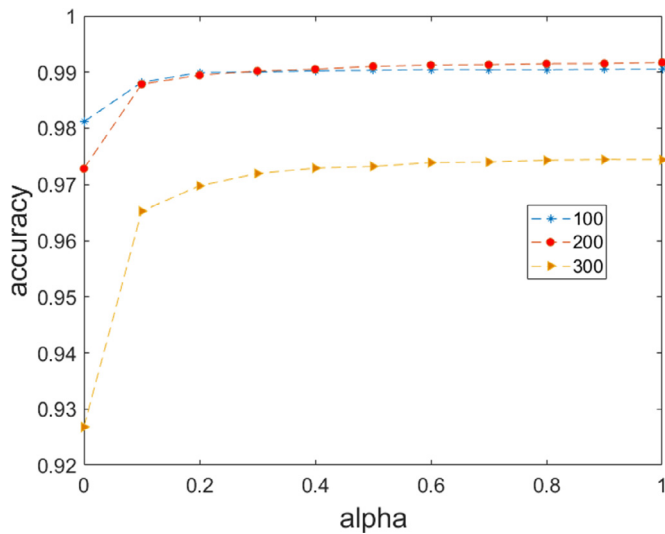
*Accuracy.* To assess the quality of a clustering, we will use the accuracy, defined as

$$\text{accuracy} = \frac{\text{\#of correctly classified points}}{\text{total \# of points}}.$$

##### 4.1.1. Effect of the parameter $\alpha$

The multilayered graph technique from [26] has a parameter  $\alpha$  that balances the connectivity of each graph and the distances between their subspace representations; see Section 3.2. The authors in [26] recommended to use  $\alpha = 0.5$  (and observed that their approach is not too sensitive to this parameter).

To see the effect of  $\alpha$  on SR-SSC, we use the synthetic data set described above with  $N = 3000$ ,  $\sigma = 0.2$  and  $\theta = 20^\circ$ . Fig. 1 displays the average accuracy of SR-SSC with  $L = 5$  and  $k = 100, 200, 300$  for different values of  $\alpha$  over 10 trials. We observe that the performance is quite stable for values of  $\alpha$  around 0.5. (We have made the same observation in our experiments with real



**Fig. 1.** The average accuracy of SR-SSC for different values for the parameter  $\alpha$ , with 100, 200, 300 anchor points. Note that the accuracy decreases for 300 anchor points, this behavior is explained in Section 4.1.2.

data sets although we do not report the results here to limit the size of the paper.) It is rather interesting to note that, for  $\alpha = 0$ , there is a clear drop in the performance of SR-SSC. In other words, merging naively the information about the Laplacians  $\mathcal{L}^{(i)}$ 's of each layer by summing them together does not work well. Finally, we will use the value of  $\alpha = 0.5$  for two main reasons: (1) it was the value recommended in [26], and (2) using higher values, e.g.,  $\alpha = 1$  leads to similar results for real data sets as well (for example, for the MNIST data set, it allows an increase of accuracy of 0.5% for the whole data set, but when analyzing a subset clusters, it reduces accuracy by about 0.5%). The slight decrease in the general performance for  $k = 300$  is addressed in the next section.

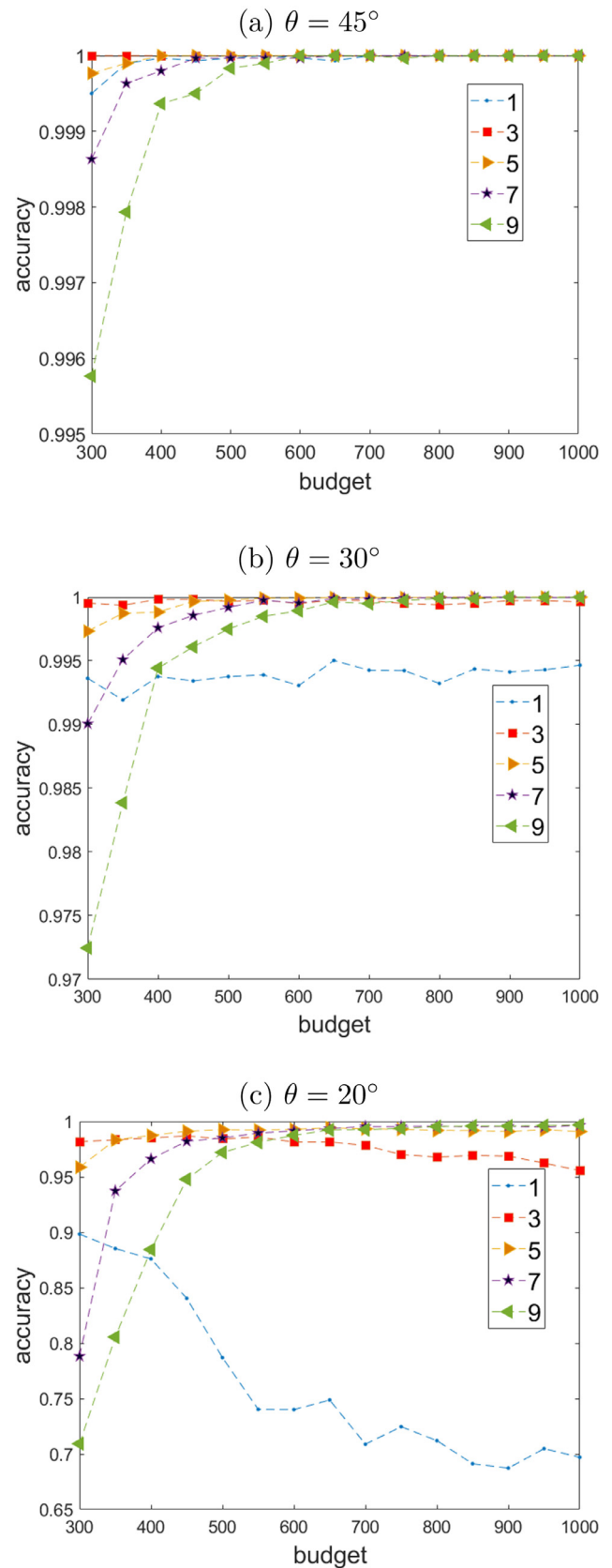
#### 4.1.2. Role of the number of graphs $L$ and anchor points $k$

In this section, we investigate the performance of SR-SSC depending on the number  $L$  of graphs (that is, the number of sets of anchor points) and the number  $k$  of anchor points per graph for different affinities between subspaces and noise levels. We will show that selecting multiple sets of anchor points can significantly improve the performance of SR-SSC when the subspace clustering problem gets more challenging (that is, larger affinity and higher noise level).

Let us use the synthetic data sets described in the previous section with  $N = 3000$  and  $\sigma = 0.2$ . Fig. 2 shows the average accuracy of SR-SSC over 10 generated synthetic data sets for different values of the budget, for different numbers of graphs and for 3 values of  $\theta$  (45, 30, 20 degrees).

We observe that if the affinity between subspaces is large enough (namely, for  $\theta = 20^\circ$ ), the performance of SSR-SSC with one graph decreases as the budget increases; a similar observation was already reported in [18]. This is due to the fact that as the affinity increases, the chances of choosing wrong connections increases as well. However, the multilayered graph structure reduces this effect as it seeks for connections that are agreed on by a majority of individual graphs [26]. In fact, we see that for a budget sufficiently large, using more graphs improves the performance of SR-SSC significantly. For example, for a budget of 1000, SR-SSC with 1 graph (this means that 1/3 of the data points are used as anchors) has average accuracy below 70% while with 9 graphs, it has average accuracy above 99%.

Fig. 3 displays the box plots for the accuracy of SR-SSC over 10 trails for  $\theta = 20^\circ$ . We observe that increasing the number of



**Fig. 2.** Average accuracy of SR-SSC for different number of graphs and for different budgets.

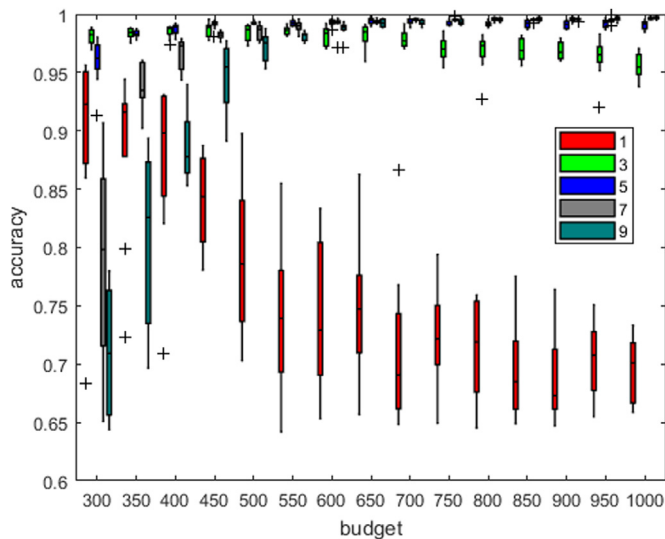


Fig. 3. The boxplot of the accuracy over 10 trials for different number of graphs.

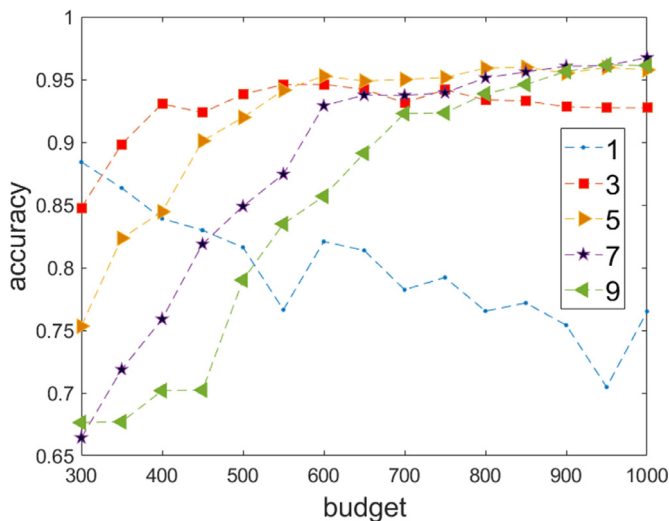


Fig. 4. Effect of increasing the noise on the performance of SR-SSC ( $\theta = 30^\circ$ ,  $\sigma = 0.4$ ). This figure can be compared with Fig. 2(b) which corresponds to  $\theta = 30^\circ$ ,  $\sigma = 0.2$ .

layers and the number of anchor points not only increases the average accuracy but also reduces the variance significantly. Clearly by increasing the number of layers, the chance of choosing good anchor points and hence constructing consistent graphs increases. This indicates a natural trade-off between computational cost and increase in performance of SR-SSC.

Noise is the other main factor that makes subspace clustering more challenging. To study the effect of the noise, we use the previous experiment setting with  $\theta = 30^\circ$  but increase the standard deviation  $\sigma$  of the additive Gaussian noise from 0.2 to 0.4. The result is plotted in Fig. 4. Comparing Fig. 2(b) and Fig. 4, it is clear that adding more layers leads to a more stable performance in the presence of a higher noise level. In particular, the average accuracy of SR-SSC with 1 graph drops from over 99% to below 75%, while with 5 to 9 graphs, it drops only from 100% to above 95%.

It is interesting to compare the performance of individual graphs with the merged one. Let us use the synthetic data sets with  $\theta = 20^\circ$  and use  $L = 5$  for SR-SSC. The average accuracy with respect to the budget is plotted in Fig. 5. We observe that even though the accuracy of individual graphs vary between 70% to 95%,

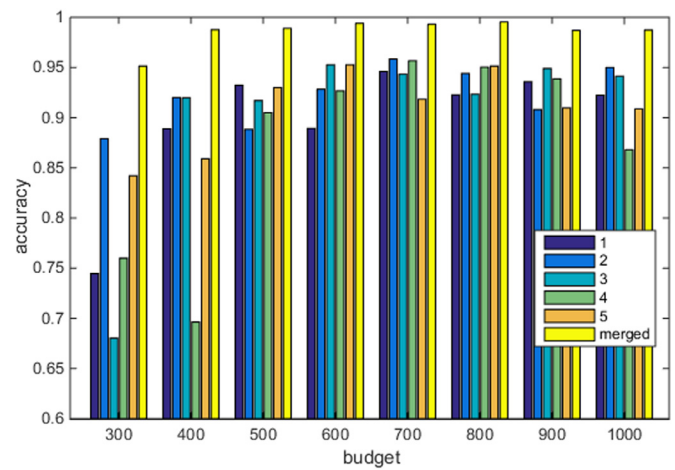


Fig. 5. Influence of merging individual graphs on accuracy. The individual performance of 5 layers of graph is compared with the performance of the final merged representation (yellow). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the performance of the merged graph is stable and around 98% (for a budget larger than 300). This is due to the fact that the multi-layer graph framework is able to highlight the information shared by the majority of graphs.

#### 4.1.3. The role of randomized hierarchical clustering for selecting the anchor points

It is interesting to compare the performance of the randomized hierarchical clustering with other subset selection methods. Fig. 6 illustrate the performance on the synthetic data set with  $\theta = 20^\circ$  of the following standard methods: QR [43], leverage score sampling [46], hybrid sampling [44] and uniform random sampling. It shows the average accuracy over 5 trials for 1, 3 and 5 layers of graphs. The QR method is deterministic and hence is shown only for 1 layer. It can be seen that increasing the number of layers can generally improve the performance of all randomized methods. The hybrid method performs similarly as the randomized hierarchical clustering, but with much higher computational cost due to SVD decomposition (cf. the discussion in Section 3.1). Note that the naive approach of using uniform sampling does not perform too poorly, the reason being that the samples from the synthetic data set are evenly and well-spread among the different subspaces.

#### 4.1.4. Effect of number of subspaces

In order to investigate the effect of the number of subspaces on the performance of SR-SSC, several 5-dimensional subspaces with ambient dimension of 50 are randomly generated (in Matlab,  $\text{randn}(50, 5) * \text{randn}(5, p)$  where  $p = 100$  is the number of samples in each subspace). The number of subspaces vary from 2 to 20. Fig. 7 shows the ratio of accuracy between SR-SSC with one layer and SSC, for different number of selected anchor points. Clearly selecting too few samples from each subspace affects the performance of SR-SSC (from the theory point of SSC, at least 6 points from each subspace are needed for the representation). When the number of subspaces increases, SR-SSC requires slightly more samples to match SSC accuracy. In any case, this example illustrates that SR-SSC is able to select good sets of anchor points since its performance compared to SSC is the same as long a sufficiently many anchor points are selected (10 anchor points in average per subspace is enough –even when there are many highly dependent subspaces; e.g., for the case of 20 subspaces of dimension 5 in ambient dimension 50).



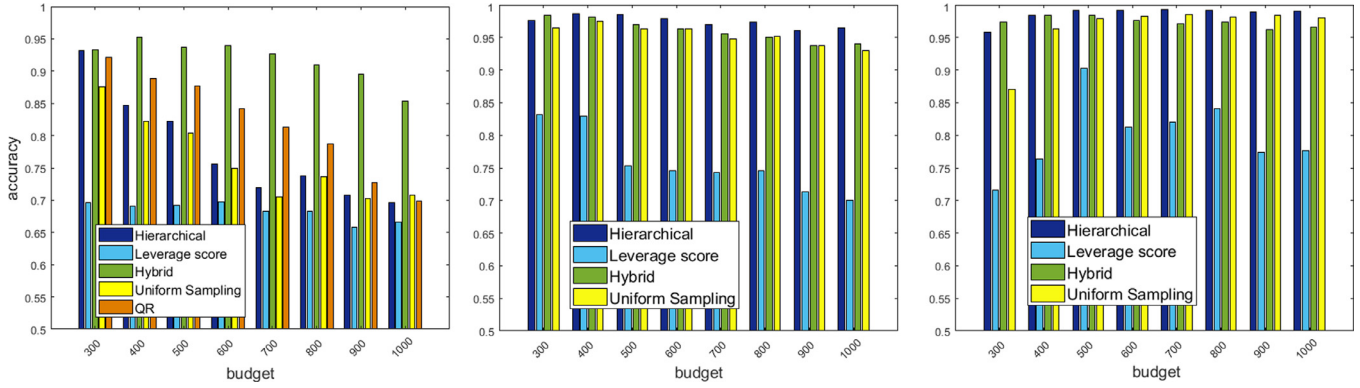


Fig. 6. Comparison of performance of SR-SSC with other anchor selection methods. From left to right: one layer, three layers and five layers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

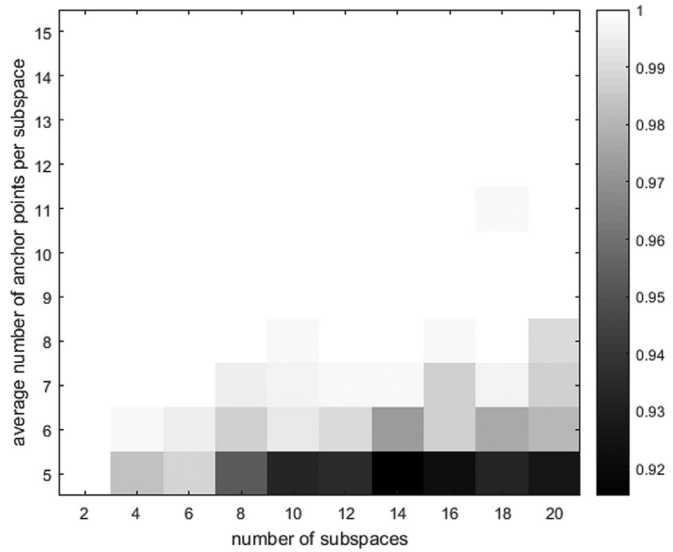


Fig. 7. Effect of the number of subspaces on the ratio of performance of SR-SSC (1 layer) and SSC.

4.1.5. Effect of adding outliers

Eventhough there exists several methods in the literature [22,54] to deal with outliers, these methods are often based on assumptions that might not be true in all data sets (such as dense representation of outliers or strong connectivity of the graph). Hence, it might not be always possible to correctly separate inliers from outliers and this can deteriorate significantly the performance of the clustering algorithm. To analyze the behavior of SR-SSC in the presence of outliers, the same synthetic data set with  $N = 3000$  under 3 different affinities ( $\theta = 45^\circ$ ,  $\theta = 30^\circ$  and  $\theta = 20^\circ$ ) is used. We add different numbers of outliers (expressed as a percentage of the 3000 data points) to the data set. Each outlier is generated as follows: each entry is generated using the Gaussian distribution of mean 0 and standard deviation 1, and then the outlier is normalized to have unit  $\ell_2$  norm as done in [22]: ‘This guarantees that outlier detection is not trivially accomplished by exploiting differences in the norms between inliers and outliers’. We set the value of the budget to 1000 and evaluate the average performance of SR-SSC in correctly clustering the inliers for different layers of graphs over 20 trials (we apply SR-SSC with 3 clusters on the data set with the outliers, and then compute the accuracy taking into account only the inliers).

Surprisingly, for  $\theta = 45^\circ$ , the accuracy remains 100% for any number of graphs when adding as many outliers as data points. For the two other cases ( $\theta = 20^\circ, 30^\circ$ ), the percentage of outliers

Table 1

Percentage of outliers corresponding to the break-down point of SR-SSC (that is, the accuracy goes below 95% beyond that point) for different budget affinities and various number of graph layers.

$\theta$	1 Graph	3 Graphs	5 Graphs	7 Graphs	9 Graphs
$30^\circ$	42.5	47.5	57.5	70	77.5
$20^\circ$	0	2.5	17.5	20	22.5

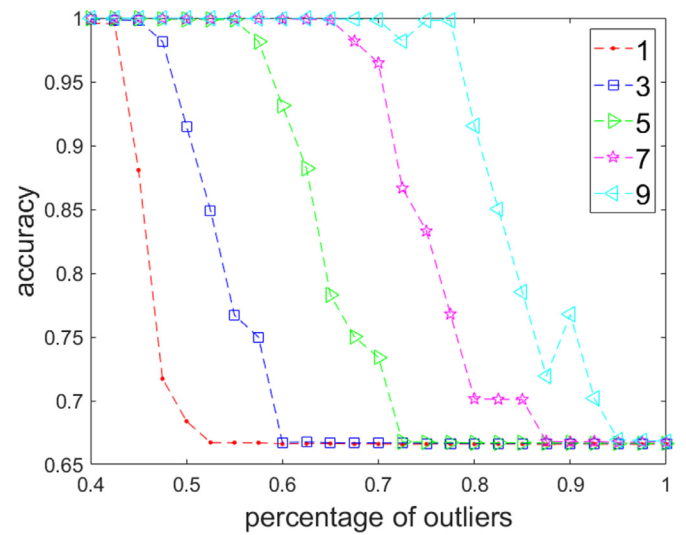
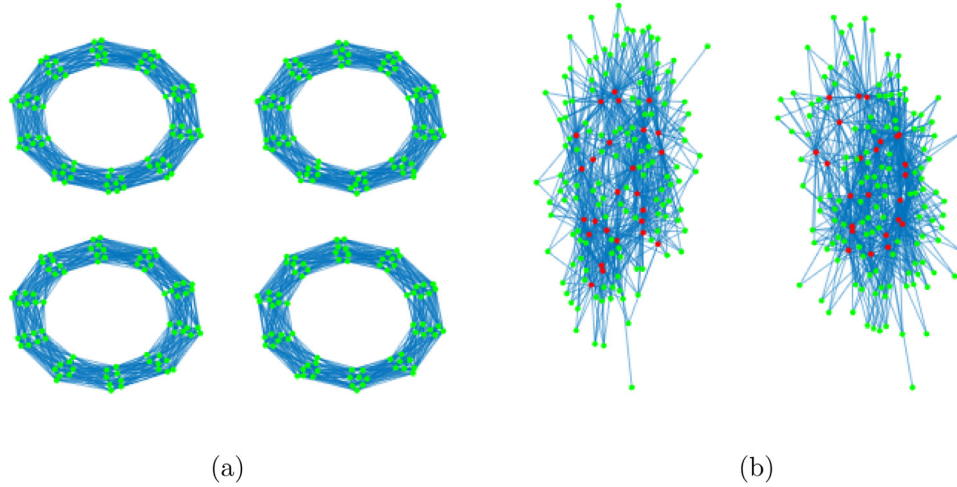


Fig. 8. The effect of outliers on performance of SR-SSC for  $\theta = 30^\circ$ , and a fixed budget of 1000. The figure reports the average accuracy over 20 runs of SR-SSC for different percentages of added outliers.

which breaks down the accuracy of SR-SSC (with no outliers) is reported in Table 1 and the average performance of SR-SSC for  $\theta = 30^\circ$  over 20 trials is illustrated in Fig. 8. Not surprisingly, we observe that as the affinity increases, the percentage of allowed outliers to maintain the accuracy decreases. Moreover, we observe that adding more layers to the graph (for a fixed budget) leads to a more robust and stable performance as the percentage of allowed outliers to maintain the accuracy increases significantly. (Recall that that for  $\theta = 20^\circ$ , the one-layered graph performs poorly even without outliers; see Section 4.1.2.)

4.1.6. Overshadowing the oversegmentation issue

An important issue in SSC is the connectivity of the graphs corresponding to the data points within each subspace. In particular, there is no guarantee that for dimensions higher than four, the points from the same subspace form a single connected



**Fig. 9.** Illustration of the oversegmentation of SSC for two subspaces. The green dots correspond to the samples while the blue edges between the samples correspond to non-zero weights in the coefficient matrix. (a) The graph corresponding to the coefficient matrix of standard SSC with four connected components. (b) The graph corresponding to the coefficient matrix of SR-SSC with two connected components. The red dots correspond to the anchor points.

component [35]. In this section, we show that by choosing anchor points as representative samples, the oversegmentation issue of SSC is alleviated. For this purpose, the points from two subspaces in a 8-dimensional space are created similarly as in [35]. More precisely, we consider 160 points from each subspace chosen around two orthogonal circles. Half of the points on the first subspace are given by

$$x_1(k, s, s') = [\cos \theta_k, \sin \theta_k, s\delta, s'\delta, 0, 0, 0, 0]^T, \quad (7)$$

and of the other half by

$$x_2(k, s, s') = [s\delta, s'\delta, \cos \theta_k, \sin \theta_k, 0, 0, 0, 0]^T, \quad (8)$$

where  $\theta = \frac{\pi k}{10}$  ( $k = 0, 1, \dots, 19$ ),  $s, s' \in \{-1, 1\}$  and  $\delta = 0.1$ . Similarly, the second subspace contains the points

$$y_1(k, s, s') = [0, 0, 0, 0, \cos \theta_k, \sin \theta_k, s\delta, s'\delta]^T,$$

and

$$y_2(k, s, s') = [0, 0, 0, 0, s\delta, s'\delta, \cos \theta_k, \sin \theta_k]^T.$$

For a point  $x_1(k, s, s')$  defined in (7), the non-zero weights in SSC will correspond to the four points  $x_1(k \pm 1, s, s')$ ,  $x_1(k \pm 1, -s, s')$  and  $x_1(k \pm 1, s, -s')$ . This implies that there will be no connections between the points  $x_1(k, s, s')$  and  $x_2(k, s, s')$  within the same subspace, and similarly for  $y_1(k, s, s')$  and  $y_2(k, s, s')$ . The graph corresponding to the SSC coefficient matrix  $C$  is shown in Fig. 9(a). Clearly, the coefficient matrix of SSC corresponds to a graph with 4 connected components. For this reason, SSC has accuracy of 75%.

However, selecting a few anchor points strengthens the connectivity within each subspace. The intuition is that if the points use only few anchor points that are well spread within the data set, then the chance of them forming a single connected component in the similarity graph increases. This is what we observe when choosing 50 samples (out of 320 data points) on the above data set (using our proposed randomized hierarchical clustering). The graph corresponding to coefficient matrix is shown in Fig. 9(b), and SR-SSC accuracy is 100%. Note that the same observation holds for this example as long as the number of anchor points is less than around 200.

#### 4.1.7. How selecting few anchor points can help subspace clustering

In order to get a better intuition about how SR-SSC can improve the performance of clustering, we reuse the synthetic data set from Section 4.1 and generate 3000 samples from the 3 aforementioned subspaces with two different values of  $\theta$ , namely, 30

and 20. We set the budget to 1000 and  $\sigma = 0.2$  for both cases and consider 1, 3 and 5 graphs. We calculate the top 3 eigenvectors of the matrix  $L_f$  corresponding to each case and plot the second and third eigenvectors in Figs. 10 and 11 for  $\theta = 30^\circ$  and  $\theta = 20^\circ$ , respectively. The eigenvectors corresponding to each subspace are plotted in the same color. This presentation provides a 2-dimensional representation for the coefficient matrices [11]. We observe that by selecting few anchor points, especially in the more challenging case ( $\theta = 20^\circ$ ), we are strengthening the within subspace connectivity (the samples within each subspace are gathered together more closely) and at the same time, they are more separated from the other subspaces. This illustrates the fact that by selecting few *well-spread* anchor points, the samples from the same subspace tend to form a strong single connected component (hence, avoiding the oversegmentation issue). Moreover, this also prevents undesirable (strong) connections between samples from different subspaces that is typical in the presence of noise and close subspaces.

## 4.2. Real-world data sets

In this section, the performance of SR-SSC is evaluated using three large-scale and challenging data sets: handwritten digits (MNIST), object images (CIFAR10) and forest data set (Coverttype). We compare the accuracy and running times of SR-SSC with standard SSC based on ADMM (this is SR-SSC with  $L = 1$  and  $k = N$ ) and four state-of-the-art methods for sparse subspace clustering, namely OMP [28], SSSC [27] (which is closely related to SR-SSC with  $L = 1$ ), KSSC [34] and ORGEN [33]; see Section 2 for a brief description of these methods. For OMP and ORGEN, we used the code available from <http://vision.jhu.edu/code/>.

### 4.2.1. MNIST data set of handwritten digits

The MNIST database contains 70,000 gray scale images of 10 handwritten digits, each of size 28-by-28 pixels. For each image, using a scattering convolution network [55], a feature vector of dimension 3472 is extracted and then projected to dimension 500 by PCA as done in [28,33]. Nine data sets corresponding to different number of digits are formed (namely digits from 0 to  $i$  for  $i = 1, 2, \dots, 9$ ), with around 5000 to 7000 data points of each digit. We also consider data sets with similar digits (2 and 3, 3 and 5, and 1 and 7 are similar), namely  $\{1,2,3\}$ ,  $\{1,3,5\}$ ,  $\{2,3,5\}$  and  $\{1,2,3,5,7\}$ . For SR-SSC, we use  $L = 5$ , and we select  $100n$  anchor

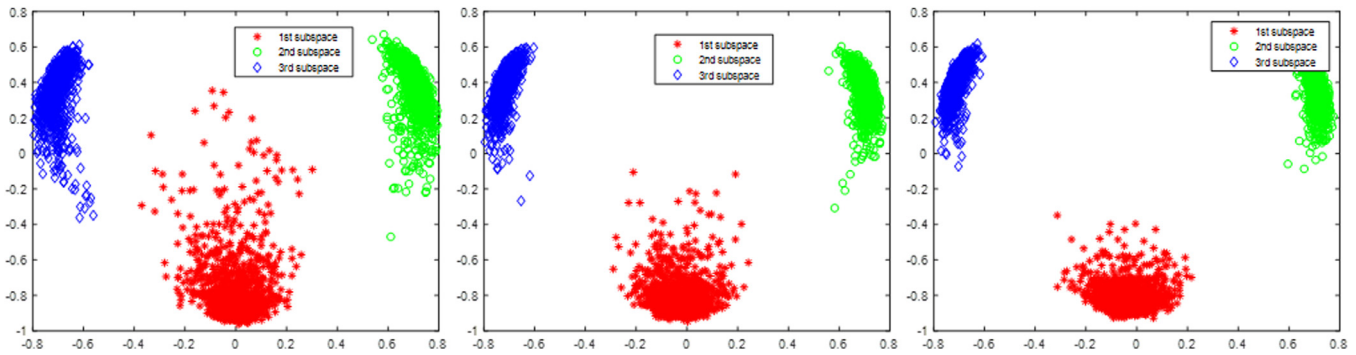


Fig. 10. Two dimensional embedded representation of SR-SSC for  $\theta = 30^\circ$  for synthetic data with three clusters. From left to right: one layer, three layers and five layers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

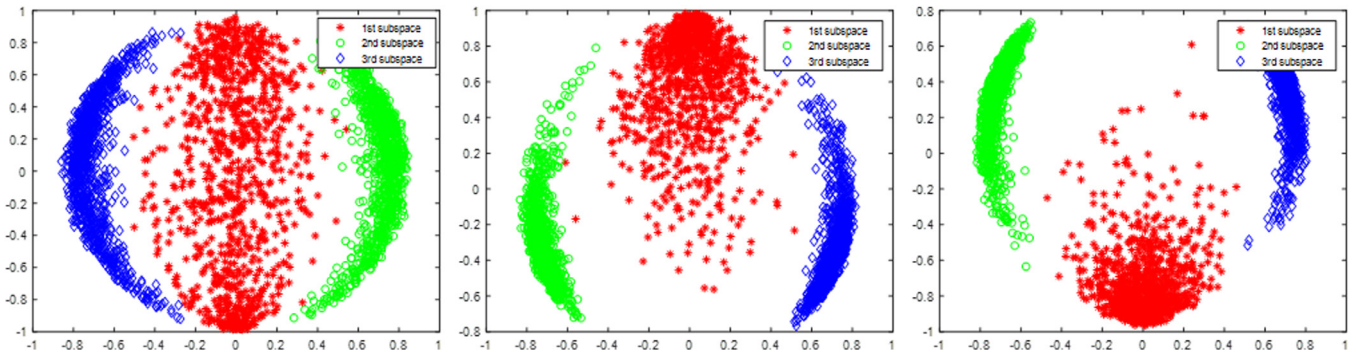


Fig. 11. Two dimensional embedded representation of SR-SSC for  $\theta = 20^\circ$  for synthetic data with three clusters. From left to right: one layer, three layers and five layers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

points where  $n$  is the number of clusters. For a fair comparison, a dictionary with size corresponding to the whole budget for each case is randomly selected for SSSC. The regularization parameter  $\mu$  in (2) is chosen as  $\mu = \lambda\mu_0$  where  $\mu_0$  is the minimum value which avoids a zero coefficient matrix  $C$ , and  $\lambda$  is chosen 120 for our approach, ORGEN and SSSC. The sparsity parameter of OMP and regularization parameter of ORGEN are set to 10 and 0.95 according to the corresponding papers [28,33]. The number of nearest neighbors for KSSC is chosen to be 30.

The average clustering accuracy and running time for each approach for over 10 trials is reported in Tables 2 and 3. We observe that

- SSC runs out of memory for data sets with more than 10,000 points. Since the data set [0:1] has low affinity, it performs well in this situation.

- SR-SSC has the most stable clustering accuracy with a minimum of 89.89% while being computationally efficient. This validates the fact that the use of a multilayer graph in SR-SSC makes it more robust (given that the number of graphs  $L$  and the number of anchor points  $k$  is large enough).
- OMP performs poorly as soon as the affinity starts to increase.
- There is a drop of performance of ORGEN, KSSC and OMP by adding the digit 3 to the data set. This is due to the similarity between the digits 2 and 3: these methods oversegment the cluster containing the digit 1 while failing to distinguish between digits 2 and 3 (for example, ORGEN puts 51.8% of digits 1 in one cluster, 45.6% of digits 1 in another while digits 2 and 3 are put together in a single cluster with 2.6% of digits 1). This is confirmed by the low performance of OMP, ORGEN, KSSC and SSSC in clustering the digits {1,2,3} and {1,3,5} in Table 2. However, the performance of the aforementioned methods is

Table 2

Accuracy (in %) of the different clustering methods over different sets of digits of the MNIST data set. The value M indicates that SSC ran out of 16GB memory. The best accuracy is indicated in bold, the second best is underlined.

Digits	SSC-ADMM	OMP	ORGEN	SSSC (500n)	KSSC	SR-SSC (5,100n)
[0: 1]	99.26	99.11	99.29	99.21	<b>99.97</b>	<u>99.36</u>
[0: 2]	M	<u>98.67</u>	98.29	50.81	<b>99.22</b>	98.32
[0: 3]	M	62.91	62.67	62.59	<u>63.19</u>	<b>98.06</b>
[0: 4]	M	69.63	69.67	64.34	<u>70.16</u>	<b>98.23</b>
[0: 5]	M	48.87	<u>75.42</u>	65.80	75.19	<b>93.41</b>
[0: 6]	M	76.18	78.53	68.41	<u>79.00</u>	<b>93.98</b>
[0: 7]	M	69.39	80.47	76.49	<u>81.43</u>	<b>94.26</b>
[0: 8]	M	59.79	<u>81.62</u>	75.77	70.23	<b>90.53</b>
[0: 9]	M	48.44	<b>93.85</b>	68.07	84.01	<u>89.89</u>
[1 2 3]	M	51.29	51.10	51.37	<u>52.34</u>	<b>98.39</b>
[1 3 5]	M	52.56	52.74	<u>55.44</u>	54.31	<b>89.20</b>
[2 3 5]	M	95.83	<u>96.85</u>	93.53	<b>97.88</b>	87.96
[1 2 3 5 7]	M	64.65	70.32	61.21	<u>71.76</u>	<b>92.58</b>

**Table 3**

Running times in minutes of the different clustering methods over different sets of digits on the MNIST data set. The value M indicates that SSC ran out of 16GB memory.

Digits	SSC-ADMM	OMP	ORGEN	SSSC (500n)	KSSC	SR-SSC (5,100n)
[0: 1]	239.07	0.69	2.12	0.47	1.35	0.64
[0: 2]	M	1.45	3.65	1.08	1.44	1.37
[0: 3]	M	2.36	5.27	2.01	1.80	2.48
[0: 4]	M	3.51	6.96	3.16	2.25	3.53
[0: 5]	M	5.00	8.81	4.66	2.68	5.36
[0: 6]	M	6.37	10.92	6.68	3.10	7.56
[0: 7]	M	8.25	13.45	9.03	3.59	11.30
[0: 8]	M	10.85	15.97	11.61	4.27	13.73
[0: 9]	M	13.02	18.28	14.95	4.79	17.71

**Table 4**

Accuracy (in %) of SR-SSC for different set of parameters over different sets of digits of the MNIST data set.

Digits	SR-SSC (3,30n)	SR-SSC (5,30n)	SR-SSC (5,50n)	SR-SSC (5,100n)	SR-SSC (3,200n)
[0: 1]	97.96	97.93	98.81	99.36	99.46
[0: 2]	96.86	97.07	97.66	98.32	98.49
[0: 3]	96.47	96.81	97.73	98.06	98.54
[0: 4]	96.72	96.61	97.74	98.23	98.51
[0: 5]	91.68	91.54	92.46	93.41	94.02
[0: 6]	92.12	92.90	93.37	93.98	94.02
[0: 7]	92.96	92.39	93.36	94.26	94.87
[0: 8]	83.36	84.01	86.23	90.53	93.95
[0: 9]	83.95	82.92	87.84	89.89	91.09

high in clustering digits {2,3,5} which means that unlike digit 1, the digit 5 cluster forms a well connected component so that digits 2 and 3 are also well separated. This confirms our previous results on synthetic data sets in Section 4.1 that showed that our proposed method performs well in the challenging case of close subspaces, and overshadows the oversegmentation effect.

- The running time of SR-SSC is similar to the other scalable SSC variants except KSSC. KSSC uses LADMPSPAP (linearized alternating direction method with parallel splitting and adaptive penalty) [56] which benefits from parallel computing. Moreover, KSSC uses mex implementations of vlfeat library [57] to select nearest neighbors efficiently. Hence, the reported computational time of KSSC is much lower than other methods and it is not fair to compare them.

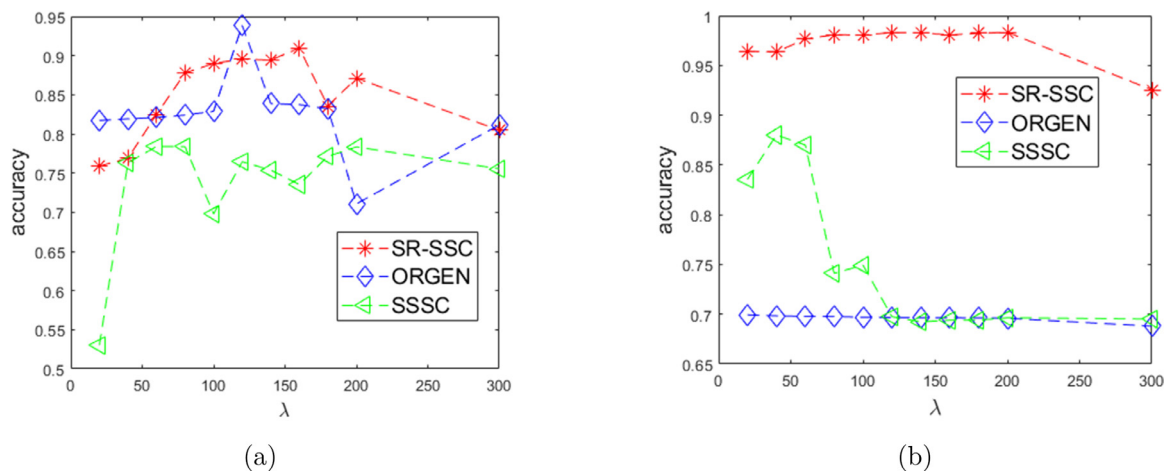
The performance of SR-SSC for different parameters (namely the number of graphs and anchor points) is shown in Table 4 for five

sets of parameters: (3,30n), (5,30n), (5,50n), (5,100n) and (3,200n). We observe that adding more anchor points increases the accuracy in almost all cases. However, the performance is quite stable among different settings before adding digit 8 and SR-SSC is still able to perform well even with only 30 anchor points per cluster. The very similar performances for the (3,30n) and (5,30n) cases highlight the fact that multilayer framework can improve the performance as long as enough anchor points are chosen for each individual layer. Hence the performance of SR-SSC depends on two main factors: (i) to have well-spread and sufficiently many anchor points and (ii) sufficiently many number of layers to best summarize informative shared connectivity among different layers.

Let us analyze the sensitivity of SR-SSC, ORGEN and SSSC to the regularization parameter of  $\lambda$  for two cases, namely for the digits [0: 9] (the whole data set) and [0: 3] (where an oversegmentation issue was observed for the other approaches; see Table 2). The accuracy of the different algorithms is measured for different value of  $\lambda$  and is plotted in Fig. 12. For SR-SSC and SSSC, the accuracy is averaged over three random trials. We observe that the performance of SR-SSC is quite stable (as long as the regularization parameter is not too high or too low) even when the chance of oversegmentation is high for the case of [0: 3] digits. SSSC and ORGEN are more sensitive to the choice of  $\lambda$ ; in particular, the highest accuracy of ORGEN for the whole data set is achieved for only a very limited range of this parameter.

#### 4.2.2. CIFAR10 data set of images

The performance of the same algorithms is compared on the challenging CIFAR10 data set. CIFAR10 consists of 32-by-32 colored images of 10 objects. For each image, we converted it to grayscale and then scattering convolution network is used again to extract



**Fig. 12.** The effect of regularization parameter  $\lambda$  on the performance of SR-SSC, ORGEN and SSSC for the digits (a) [0: 9] and (b) [0: 3].

**Table 5**

Accuracy of different clustering methods over different sets of clusters on the CIFAR10 data set. The value M indicates that SSC ran out of 16GB memory. The best accuracy is indicated in bold, the second best is underlined.

# clusters	SSC-ADMM	OMP	ORGEN	SSSC (700n)	KSSC	SR-SSC (3,100n)	SR-SSC (5,100n)	SR-SSC (7,100n)
2	50.68	50.02	50.82	50.01	74.87	80.33	<u>82.44</u>	<b>83.15</b>
3	33.72	33.85	33.89	33.45	50.27	<u>52.01</u>	51.79	<b>52.86</b>
4	M	25.62	25.36	25.06	39.75	51.40	<b>52.29</b>	<u>51.76</u>
5	M	20.56	20.51	20.07	31.79	39.68	<b>41.71</b>	<u>40.97</u>
6	M	17.09	17.16	16.70	28.53	<u>36.58</u>	36.56	<b>37.93</b>
7	M	14.59	14.73	14.33	23.59	29.83	<b>31.35</b>	<u>30.79</u>
8	M	12.79	12.89	12.55	22.27	27.23	<u>30.58</u>	<b>31.11</b>
9	M	11.36	11.51	11.16	21.96	26.96	<u>27.74</u>	<b>28.07</b>
10	M	10.23	10.37	10.07	19.22	<u>24.36</u>	23.88	<b>26.44</b>
[1 2 4]	33.75	33.36	33.90	33.42	39.00	60.69	<u>61.85</u>	<b>63.51</b>
[1 3 4]	45.87	33.39	33.57	33.41	39.28	<u>58.46</u>	58.25	<b>59.43</b>
[1 2 6]	33.76	33.93	33.90	33.41	41.67	<u>61.63</u>	61.20	<b>62.90</b>
[1 3 6]	57.49	33.35	33.61	33.43	40.37	56.08	<u>58.39</u>	<b>58.43</b>

**Table 6**

Computational time in minutes of different clustering methods over different sets of clusters on the CIFAR10 data set. The value M indicates that SSC ran out of 16GB memory.

# clusters	SSC-ADMM	OMP	ORGEN	SSSC (700n)	KSSC	SR-SSC (3,100n)	SR-SSC (5,100n)	SR-SSC (7,100n)
2	57.69	0.81	6.19	0.16	0.60	0.18	0.29	0.40
3	162.70	1.60	10.01	0.40	0.88	0.38	0.63	0.89
4	M	2.59	15.37	0.80	1.22	0.65	1.08	1.58
5	M	3.84	20.52	1.35	1.54	1.03	1.70	2.38
6	M	5.23	25.36	2.06	1.91	1.50	2.59	3.49
7	M	7.08	29.53	2.71	2.17	2.22	3.63	4.84
8	M	8.83	35.21	3.85	2.67	3.08	4.78	6.54
9	M	11.15	40.27	5.05	3.02	3.88	6.07	8.52
10	M	13.40	47.60	6.12	3.57	4.99	7.94	10.74

**Table 7**

Accuracy of different clustering methods on Covertypes Dataset.

SSC-ADMM	OMP	ORGEN	SSSC (4000)	KSSC	SR-SSC (5,700)	SR-SSC (3,1400)	SR-SSC (7,700)	SR-SSC (5,1400)
M	48.76	<b>53.52</b>	36.50	33.40	48.35	48.75	<u>48.87</u>	48.58

feature vectors which are projected to dimension 500 using PCA. Ten different sets of data corresponding to different numbers of clusters are formed. For each cluster, 5000 samples of the corresponding class are selected randomly. We consider 3, 5 and 7 layers of graphs for our method and 100n anchor points are selected for each graph.

The sparsity parameter of OMP algorithm is set to 10 and for the regularization parameter  $\mu$  is chosen as for the MNIST data set, with  $\lambda = 120$  for all different sets of experiments for ORGEN, SSSC and SR-SSC. The results are reported in Tables 5 and 6.

SR-SSC has the highest accuracy in all cases, while it takes less than a minute to cluster 10,000 of data points and less than 15 minutes to cluster 50,000 points. The performance of SR-SSC is slightly higher for 7 graphs compared to 3 and 5 graphs, but the difference is not significant.

CIFAR is a challenging data set as the linearity assumption of subspaces is violated, which explains the relative low accuracy of SSC and its variants. However, SR-SSC still manages to outperform all the SSC variants. Furthermore, we compared the performances over four extra combinations of clusters at the bottom of Table 5. There is a clear drop in performance of SSC in clustering [1 2 4] compared to [1 3 4], and [1 2 6] compared to [1 3 6]. This can be explained by the close affinity between clusters 1 and 2 compared to 1 and 3.

In terms of computational time, SR-SSC compares favorably with OMP and SSSC, as for MNIST. The computational time of ORGEN is significantly higher compared to other scalable approaches

for this data set. This is due to the adaptive support set selection method of ORGEN which depends on the correlation between the data points and the *oracle point*. High correlation among data points in CIFAR decreases the amount of discarded data points for each iteration which plays a crucial role for the computational cost of the algorithm.

#### 4.2.3. Covertypes data set

The Covertypes data set<sup>3</sup> contains 581,012 samples from 7 categories of tree types. Each sample has 54 categorical/integer attributes that were derived from the data originally obtained from the US Geological Survey (USGS) and USFS data. The sparsity parameter of OMP algorithm is set to 15 and  $\lambda$  is set to 50 for the regularization parameter  $\mu$  for both ORGEN and SR-SSC. The results are reported in Tables 7 and 8.

ORGEN achieves the best accuracy, but for a much higher computational cost (see the discussion in the previous section). SR-SSC offers the best trade-off between computational time and accuracy. For this data set, the accuracy is not too sensitive to the number of graphs and anchor points. Note that SSSC is much faster than SR-SSC in this case because (i) the data set is larger and (ii) SSSC uses SSC on only 4000 samples to learn the subspaces and then cluster the other data points (these are two independent steps).

<sup>3</sup> <http://archive.ics.uci.edu/ml/datasets/Covertypes>.

**Table 8**  
Computational time of different clustering methods on Coverttype Dataset (in minutes).

SSC-ADMM	OMP	ORGEN	SSSC (4000)	KSSC	SR-SSC (5,700)	SR-SSC (3,1400)	SR-SSC (7,700)	SR-SSC (5,1400)
M	783	1452	14.52	15.89	222.14	400.12	323.00	664.19

## 5. Conclusion

In this paper, we have proposed a new framework to overcome the scalability issue of SSC. Our proposed framework, referred to as scalable and robust SSC (SR-SSC), constructs a multilayer graph by solving LASSO problems using different sets of dictionaries. A fast hierarchical clustering method was used to select the anchor points within the dictionaries that are good representatives of the data set. Screening out large numbers of data points drastically reduces the computational cost and memory requirements of SSC. Moreover, the multilayer structure has the ability to obtain a summarized subspace representation from different layers of graphs such that it emphasizes shared information among different layers.

Our experimental results on synthetic and large-scale real-world data sets showed the efficiency of SR-SSC especially in challenging cases of noisy data and close subspaces. Moreover, by choosing few common representative points, the proposed framework has the ability to overshadow the oversegmentation problem of SSC. Of course, there is a trade-off between the computational cost (which is directly related to the number of graphs used and the number of anchor points selected for each graph) and the robustness of SR-SSC, which should be carefully balanced. We have observed in practice that using 5 to 10 graphs is a good choice, while the number of anchor point should be proportional to the number of clusters and their dimensions.

Further work include the improvement of the two key steps of SR-SSC, namely the selection of the sets of anchor points and the summarization of the multilayer graph. For example, not choosing the different sets of anchor points independently would be particularly interesting in order to make the different layers as complementary as possible. Also, studying the theoretical properties of SR-SSC would be a particular promising direction of further research. Another direction of research is to improve the implementation of SR-SSC; in particular SR-SSC is especially amenable to parallelization as the constructions of the different layers in the multilayer graph are independent.

## Conflict of interest

There is no conflict of interest.

## Acknowledgments

The authors thank the anonymous reviewers for their insightful comments which helped improve the paper. The authors would also like to thank Paul Van Dooren for helpful discussion and advice. MA acknowledges the support of the [Ministry of Science, Research and Technology](#), Iran. NG acknowledges the support of the [European Research Council](#) (ERC starting grant no. 679515) and of the [F.R.S.-FNRS](#) (incentive grant for scientific research no. F.4501.16).

## References

- [1] I.T. Jolliffe, *Principal component analysis and factor analysis*, in: *Principal Component Analysis*, Springer, 1986, pp. 115–128.
- [2] R. Basri, D.W. Jacobs, Lambertian reflectance and linear subspaces, *IEEE Trans. Pattern Anal. Mach. Intell.* 25 (2) (2003) 218–233.
- [3] R. Vidal, Subspace clustering, *IEEE Signal Process. Mag.* 28 (2) (2011) 52–68.
- [4] T. Zhang, A. Szlam, G. Lerman, Median k-flats for hybrid linear modeling with many outliers, in: *IEEE 12th Int. Conf. on Computer Vision Workshops*, IEEE, 2009, pp. 234–241.
- [5] P. Tseng, Nearest q-flat to m points, *J. Optim. Theory Appl.* 105 (1) (2000) 249–252.
- [6] J. Ho, M.H. Yang, J. Lim, K.C. Lee, D. Kriegman, Clustering appearances of objects under varying illumination conditions, in: *Proc. of the Conf. on Computer Vision and Pattern Recognition*, 1, IEEE, 2003.
- [7] M.E. Tipping, C.M. Bishop, Mixtures of probabilistic principal component analyzers, *Neural Comput.* 11 (2) (1999) 443–482.
- [8] Y. Sugaya, K.i. Kanatani, Geometric structure of degeneracy for multi-body motion segmentation, in: *ECCV Workshop SMVP*, Springer, 2004, pp. 13–25.
- [9] R. Vidal, Y. Ma, S. Sastry, Generalized principal component analysis (GPCA), *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (12) (2005) 1945–1959.
- [10] Y. Ma, A.Y. Yang, H. Derksen, R. Fossom, Estimation of subspace arrangements with applications in modeling and segmenting mixed data, *SIAM Rev.* 50 (3) (2008) 413–458.
- [11] U.V. Luxburg, A tutorial on spectral clustering, *Stat. Comput.* 17 (4) (2007) 395–416.
- [12] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: *Advances in Neural Information Processing Systems*, 2002, pp. 849–856.
- [13] G. Chen, G. Lerman, Spectral curvature clustering (SCC), *Int. J. Comput. Vis.* 81 (3) (2009) 317–330.
- [14] J. Yan, M. Pollefeys, A general framework for motion segmentation: independent, articulated, rigid, non-rigid, degenerate and non-degenerate, in: *European Conf. on Computer Vision*, Springer, 2006, pp. 94–106.
- [15] A. Goh, R. Vidal, Segmenting motions of different types by unsupervised manifold clustering, in: *Computer Vision and Pattern Recognition*, 2007. CVPR'07. IEEE Conference on, IEEE, 2007, pp. 1–6.
- [16] A.M. Bruckstein, D.L. Donoho, M. Elad, From sparse solutions of systems of equations to sparse modeling of signals and images, *SIAM Rev.* 51 (1) (2009) 34–81.
- [17] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: *Computer Vision and Pattern Recognition*, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 2790–2797.
- [18] E. Elhamifar, R. Vidal, Sparse subspace clustering: algorithm, theory, and applications, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (11) (2013) 2765–2781.
- [19] 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.
- [20] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 663–670.
- [21] 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: *Computer Vision–ECCV 2012*, 2012, pp. 347–360.
- [22] M. Soltanolkotabi, E.J. Candes, A geometric analysis of subspace clustering with outliers, *Ann. Stat.* 40 (4) (2012) 2195–2238.
- [23] M. Soltanolkotabi, E. Elhamifar, E.J. Candes, Robust subspace clustering, *Ann. Stat.* 42 (2) (2014) 669–699.
- [24] C. You, R. Vidal, Geometric conditions for subspace-sparse recovery, in: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 1585–1593.
- [25] Y.X. Wang, H. Xu, C. Leng, Provable subspace clustering: when LRR meets SSC, in: *Advances in Neural Information Processing Systems*, 2013, pp. 64–72.
- [26] X. Dong, P. Frossard, P. Vandergheynst, N. Nefedov, Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds, *IEEE Trans. Signal Process.* 62 (4) (2014) 905–918.
- [27] X. Peng, L. Zhang, Z. Yi, Scalable sparse subspace clustering, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 430–437.
- [28] C. You, D. Robinson, R. Vidal, Scalable sparse subspace clustering by orthogonal matching pursuit, in: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 3918–3927.
- [29] E.L. Dyer, A.C. Sankaranarayanan, R.G. Baraniuk, Greedy feature selection for subspace clustering, *J. Mach. Learn. Res.* 14 (1) (2013) 2487–2517.
- [30] E.L. Dyer, *Endogenous Sparse Recovery*, Ph.d. thesis, Rice University, 2012.
- [31] Y. Chen, G. Li, Y. Gu, Active orthogonal matching pursuit for sparse subspace clustering, *IEEE Signal Process. Lett.* 25 (2) (2018) 164–168.
- [32] Y. Panagakis, C. Kotropoulos, Elastic net subspace clustering applied to pop/rock music structure analysis, *Pattern Recognit. Lett.* 38 (2014) 46–53.
- [33] 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 Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3928–3937.
- [34] S. Tierney, Y. Guo, J. Gao, Efficient sparse subspace clustering by nearest neighbour filtering, *arXiv:1704.03958*.
- [35] B. Nasihatkon, R. Hartley, Graph connectivity in sparse subspace clustering, in: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 2137–2144.

- [36] D.L. Donoho, *Neighborly Polytopes and Sparse Solution of Underdetermined Linear Equations*, Tech. Rep, Stanford University, 2005.
- [37] J. Wang, J. Zhou, P. Wonka, J. Ye, Lasso screening rules via dual polytope projection, in: *Advances in Neural Information Processing Systems*, 2013, pp. 1070–1078.
- [38] R. Tibshirani, J. Bien, J. Friedman, T. Hastie, N. Simon, J. Taylor, R.J. Tibshirani, Strong rules for discarding predictors in lasso-type problems, *J. R. Stat. Soc.: Ser. B* 74 (2) (2012) 245–266.
- [39] P.A. Traganitis, G.B. Giannakis, Sketched subspace clustering, *IEEE Trans. Signal Process.* 66 (7) (2018) 1663–1675.
- [40] L. Kaufman, P. Rousseeuw, *Clustering by Means of Medoids*, North-Holland, 1987.
- [41] B.J. Frey, D. Dueck, Mixture modeling by affinity propagation, in: *Advances in Neural Information Processing Systems*, 2006, pp. 379–386.
- [42] B.J. Frey, D. Dueck, Clustering by passing messages between data points, *Science* 315 (5814) (2007) 972–976.
- [43] T.F. Chan, Rank revealing QR factorizations, *Linear Algebra Appl.* 88–89 (1987) 67–82.
- [44] C. Boutsidis, M.W. Mahoney, P. Drineas, An improved approximation algorithm for the column subset selection problem, in: *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, 2009, pp. 968–977.
- [45] D. Papailiopoulos, A. Kyriillidis, C. Boutsidis, Provable deterministic leverage score sampling, in: *Proceedings of the 20th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining - KDD 14*, 2014.
- [46] C. Boutsidis, M.W. Mahoney, P. Drineas, Unsupervised feature selection for principal components analysis, in: *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 08*, 2008.
- [47] J. Chen, X. Huo, Theoretical results on sparse representations of multiple-measurement vectors, *IEEE Trans. Signal Process.* 54 (12) (2006) 4634–4643.
- [48] E. Elhamifar, G. Sapiro, R. Vidal, See all by looking at a few: Sparse modeling for finding representative objects, in: *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 1600–1607.
- [49] N. Gillis, D. Kuang, H. Park, Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization, *IEEE Trans. Geosci. Remote Sens.* 53 (4) (2015) 2066–2078.
- [50] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, *Found. Trends® Mach. Learn.* 3 (1) (2011) 1–122.
- [51] R.B. Lehoucq, D.C. Sorensen, C. Yang, *ARPACK Users' Guide Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, Society for Industrial and Applied Mathematics, 1997.
- [52] C. You, C. Li, D.P. Robinson, R. Vidal, A scalable exemplar-based subspace clustering algorithm for class-imbalanced data, in: *European Conference on Computer Vision*, Springer, 2018, pp. 68–85.
- [53] E.J. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis? *J. ACM* 58 (3) (2011) 11.
- [54] C. You, D.P. Robinson, R. Vidal, Provable selfrepresentation based outlier detection in a union of subspaces, in: *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4323–4332.
- [55] J. Bruna, S. Mallat, Invariant scattering convolution networks, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1872–1886.
- [56] R. Liu, Z. Lin, Z. Su, Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning, in: *Asian Conference on Machine Learning*, 2013, pp. 116–132.
- [57] A. Vedaldi, B. Fulkerson, *Vlfeat: an open and portable library of computer vision algorithms*, in: *Proceedings of the 18th ACM International Conference on Multimedia*, ACM, 2010, pp. 1469–1472.