SIDE-INFORMATION-INDUCED REWEIGHTED SPARSE
SUBSPACE CLUSTERING

HUA HUANG
School of Mathematics and Statistics, Xidian University, Xi'an 710171, Shaanxi, China
Key Laboratory of Group & Graph Theories and Applications
Chongqing University of Arts and Sciences, Chongqing 402160, China

WEIWEI WANG*
School of Mathematics and Statistics, Xidian University, Xi'an 710171, Shaanxi, China

CHENGWU LU
Key Laboratory of Group & Graph Theories and Applications
Chongqing University of Arts and Sciences, Chongqing 402160, China

XIANGCHU FENG AND RUIQIANG HE
School of Mathematics and Statistics, Xidian University, Xi'an 710171, Shaanxi, China

(Communicated by Changzhi Wu)

ABSTRACT. Subspace clustering segments a collection of data from a union of several subspaces into clusters with each cluster corresponding to one subspace. The geometric information of the dataset reflects its intrinsic structure and can be utilized to assist the segmentation. In this paper, we propose side-information-induced reweighted sparse subspace clustering (SRSSC) for high-dimensional data clustering. In our method, the geometric information of the high-dimensional data points in a target space is utilized to induce subspace clustering as side-information. We solve the method by iterating the reweighted $l_1$-norm minimization to obtain the self-representation coefficients of the data and segment the data using the spectral clustering framework. We compare the performance of our proposed algorithm with some state-of-the-art algorithms using synthetic data and three famous real datasets. Our proposed SRSSC algorithm is the simplest but the most effective. In the experiments, the results of these clustering algorithms verify the effectiveness of our proposed algorithm.

1. Introduction. With the advent of the era of big data, there are large amounts of data in which the number of items and the number of features per item have substantially increased. This kind of high-dimensional data is ubiquitous in many fields such as computer vision, signal and image processing. With carrying more information, such data also challenge the storage requirements, computing efficiency and the performance of the algorithms. However, high-dimensional data usually come from a union of several low-dimensional subspaces of the high-dimensional

2010 Mathematics Subject Classification. Primary: 62H30, 68T10; Secondary: 90C26.
Key words and phrases. Subspace clustering, geometric information, high-dimensional data, $l_1$-minimization, spectral clustering, self-representation.
The first author is supported by National Natural Science Foundation of China (Grants 61472303,61772389).
* Corresponding author: Weiwei Wang.
ambient space. Each subspace has the corresponding low-dimensional structure or cluster. In many real problems, the low-dimensional structure of the data in each class or category can be well represented by a low-dimensional subspace of the high-dimensional ambient space. Finding such structures or subspaces can greatly help the processing of the high-dimensional data. A useful method for detecting the subspace is subspace clustering, which separates the high-dimensional data into multiple low-dimensional subspaces according to their underlying relationships or intrinsic structures. In the past few years, the subspace clustering problem has caused widespread concern because subspace clustering is a very effective means of processing high-dimensional data. It has numerous applications in computer vision, such as in motion/video/image segmentation [28, 29, 24], pattern recognition, face clustering [16, 8] and handwritten digit recognition [19]. Figure 1 gives a simple illustration of subspace clustering. Many detailed definitions of subspace clustering can be found in reference [30].

Due to its simplicity and outstanding performance, spectral clustering-based methods [22] have become extremely popular among subspace clustering methods. These methods employ the spectral clustering algorithm as the framework. These methods can be divided into two stages. In the first stage, they construct an affinity matrix whose entries measure the similarity between pairs of points. In the second stage, the segmentation results of the data are obtained by applying spectral clustering algorithms, such as normalized cuts [26]. The main challenge of this kind of method is how to define the informative affinity matrix. The Local Subspace Affinity (LSA) [36] and Spectral Local Best-fit Flats (SLBF) [38] use the local information around each point to build an affinity matrix. The main difficulty of these methods is how to deal with points near the intersection of two subspaces because the neighbors of a point may belong to different subspaces. To solve these issues, global spectral clustering-based methods utilize the global information to construct the similarity matrix, and they include Sparse Subspace Clustering (SSC) [8], Low-Rank Recovery (LRR) [20], Structured Sparse Subspace Clustering (StrSSC or $S^3C$) [18], etc. The key idea of SSC is the self-expressiveness property of the data, i.e., each data point can be efficiently represented by a linear or an affine combination of other points. Among the infinite possible representations of a data point, a sparse representation of a data ideally corresponds to a combination of a few points in the same subspace. The solution of a global sparse optimization program, such as the affinity matrix, is used in the spectral clustering framework to obtain the segmentation of the data. Similar to SSC, LRR assumes that the data have the self-expressiveness property, and then it seeks the lowest rank representation. The coefficient matrix of the lowest rank representation is used to obtain the clustering
of the data based on the spectral clustering framework. StrSSC utilizes the fact that the affinity and the segmentation depend on each other to define the subspace structured measure that is used instead of the sparsity constraint of SSC. StrSSC is an iterative procedure that alternates between the structured sparse representation and data segmentation. Currently, many methods have been extended from the methods that were mentioned above in reference [33]-[25]. The effectiveness of the clustering results has been demonstrated using synthetic data and benchmark datasets such as the Extended Yale B dataset [17], the COIL 20 database [23], and the USPS dataset [14].

Although these methods lead to state-of-the-art results in many applications, the computational costs of these methods are too expensive. Meanwhile, these methods focus on the sparse or low-rank self-representation, and do not explore the geometrical relationships between the data points that are drawn from a union of subspaces. As we know, the geometrical information, e.g., the angle between each pair of points in the linear space or affine space, has a direct effect on data segmentation. The angle between data points in the same subspace is much smaller than the angle between data points in different subspaces. In this paper, we utilize the geometric information to construct the affine matrix, i.e., the sparse coefficient matrix is weighted using the information of the angle as a penalty. We propose a geometrical information-induced weighted \( l_1 \)-minimization problem to construct affinity matrix, and then efficiently obtain the segmentation of the data using spectral clustering. This method fully considers the influence of the geometric relationship between data points on data clustering. We call it side-information-induced reweighted sparse subspace clustering (SRSSC). Experiments that were conducted using synthetic data and several real datasets demonstrate that our method outperforms the state-of-the-art subspace clustering methods.

The proposed side-information-induced reweighted algorithm in this paper both improves the accuracy of subspace clustering and greatly reduces the computational complexity. Compared with the state-of-the-art methods, the main advantages of our work are summarized as follows.

(1) We address the effect of the geometric information of the high-dimensional data on the data segmentation. By utilizing the side-information, which reveals the intrinsic structure of the dataset, the proposed SRSSC more efficiently and accurately segments the high-dimensional dataset.

(2) The solution of the iterative reweighted \( l_1 \)-minimization problem is much more approximate to the \( l_0 \)-minimization, which is NP-hard. The affinity matrix that is constructed by the SRSSC algorithm is more accurate and sparser. The results that are obtained by spectral clustering with this affinity matrix are more accurate than those of the state-of-the-art algorithms.

(3) Experiments on synthetic data and several commonly used real datasets demonstrate that our method outperforms state-of-the-art subspace clustering methods, which include LSR [21], SMR [13], LRR, SSC, StrSSC and RSSC [35]. Specially, the accuracy of SRSSC has been greatly improved when the angles of the pairs of data are greater than 10 degrees. Figure 2 illustrates the distributions of the angles of the pairs of data in the three real databases.

The outline of this paper is as follows. Section 2 briefly introduces the theory of the SSC model using the \( l_0 \)-minimization framework. Section 3 proposes the side-information-induced reweighted sparse subspace clustering (SRSSC) and the
optimization algorithm. Section 4 presents the experimental results from using several benchmark databases to demonstrate the superior performance of the proposed method. Finally, Section 5 concludes this paper.

2. Related work. Consider a given data matrix $X \in \mathbb{R}^{D \times N}$, where each column of $X$ represents a point $x_i \in \mathbb{R}^D (i = 1, 2, \ldots, N)$ that is drawn from an unknown union of $K \geq 1$ linear or affine subspaces $\{S_l\}_{l=1}^K$. The dimensions $d_l = \text{dim}(S_l)$, where $0 < d_l < D$, of these subspaces are unknown, and the points that belong to specific subspaces are also unknown. The goal of subspace clustering is to find the parameters of each subspace and separate the high-dimensional data into multiple low-dimensional subspaces according to their underlying relationship so that the points in the same cluster belong to the same subspace.

2.1. Sparse subspace clustering. We assume that the data points are self-expressive, i.e., each data point in the union of subspaces can be well represented by a linear combination of other points in the dataset. More precisely, $x_i \in S_l$ can be written as

$$x_i = Xz_i, z_{ii} = 0,$$

where $z_i \triangleq [z_{i1}, z_{i2}, \ldots, z_{iN}]^T$. The constraint $z_{ii} = 0$ is used to eliminate the trivial solution of representing a point as a linear combination of itself. This can be expressed in matrix form as

$$X = XZ, \text{diag}(Z) = 0. \quad (2)$$

However, the representation of $X$ is not unique. Since the number of data points is larger than the spatial dimension, i.e., $N > D$, this equation is underdetermined. There are infinitely many representations of each data point in $X$. The compressive sensing [1] field needs to find the sparse representation of a data point, which means that it needs to find as few data points as possible to represent a data point. In mathematics, it is equal to solving the $l_0$ quasi-norm minimization problem

$$\min_{z_i \in \mathbb{R}^N} \|z_i\|_0$$

s.t. $x_i = Xz_i, z_{ii} = 0, \quad (3)$

where $\|z_i\|_0$ is the member of nonzero components of $z_i$. This problem is nonconvex and NP-hard. We can rewrite the $l_0$ quasi-norm minimization (3) for all data points.
in the matrix form as follows:

\[
\min_{Z \in \mathbb{R}^{N \times N}} \|Z\|_0 \\
\text{s.t.} X = XZ, \text{diag}(Z) = 0.
\] (4)

Since we all know that the \(l_1\)-norm is the convex envelope of the \(l_0\) quasi-norm and the \(l_1\)-norm minimization has a closed form solution in the literature [3], SSC finds the approximate solutions by using the \(l_1\)-norm minimization instead of the \(l_0\) quasi-norm minimization. The \(l_1\)-norm minimization problem can be written as

\[
\min_{z_i \in \mathbb{R}^N} \|z_i\|_1 \\
\text{s.t.} x_i = Xz_i, z_{ii} = 0,
\] (5)

where \(\|z_i\|_1 = \sum_{j=1}^N |z_{ij}|\). This formulation corresponds to the matrix form as

\[
\min_{Z \in \mathbb{R}^{N \times N}} \|Z\|_1 \\
\text{s.t.} X = XZ, \text{diag}(Z) = 0.
\] (6)

The SSC uses the optimal solution \(Z^*\) of the problem (6) to construct the affinity matrix(similarity matrix) \(A = (|Z^*| + |Z^*|^\top)/2\) and then obtains the final clustering results by spectral clustering framework.

2.2. \(l_0\)-sparse subspace clustering. Most of subspace clustering methods , such as SSC and StrSSC, require independence or disjointness assumptions on the subspaces. In practice, these assumptions are not guaranteed to hold. Yang, Yingzhen, et al proposed \(l_0\)-induced sparse clustering (\(l_0\)-SSC) [37] and proved that the similarity matrix can be obtained by \(l_0\)-SSC for arbitrary distinct underlying subspace almost surely under the mild i.i.d assumption on the data. \(l_0\)-SSC directly solved the original \(l_0\) quasi-norm minimization problem (3). Allowing some tolerance for inexact representation, \(l_0\)-SSC turn to optimize the following problem:

\[
\min_{Z \in \mathbb{R}^{N \times N}} \|Z\|_0 \\
\text{s.t.} \|X - XZ\|_F \leq \delta, \text{diag}(Z) = 0.
\] (7)

which is equivalent to the following problem:

\[
\min_{Z \in \mathbb{R}^{N \times N}, \text{diag}(Z) = 0} L(Z) = \|X - XZ\|_F^2 + \lambda \|Z\|_0
\] (8)

\(l_0\)-SSC employed proximal gradient descent to solve the above problem and obtained a sub-optimal solution which is used to build a similarity matrix for clustering. In the literature [4], Baraniuk, R. G, et al proposed directly using \(S_0\) and \(l_0\) as constraints for low rank and sparsity (\(S_0/l_0\)) -LRSSC. They obtained the following nonconvex optimization problem:

\[
\min_{Z \in \mathbb{R}^{N \times N}, \text{diag}(Z) = 0} \frac{1}{2}\|X - XZ\|_F^2 + \lambda \|Z\|_{S_0} + \tau \|Z\|_0
\] (9)

\((S_0/l_0)\) -LRSSC used a proximal average method to approximate the proximal map of the joint solution. The above two methods obtained more accurate approximation of sparsity of the data representation matrix than it is the case with \(l_1\) -norms, but these methods have no advantage on computational complexity.
2.3. **Reweighted sparse subspace clustering.** Since the $l_1$-norm minimization problem can be efficiently solved via convex programming tools [12], the $l_1$ regularization has been widely used in signal analysis, data compression, image processing and so on. According to the definitions of the $l_0$ quasi-norm and the $l_1$-norm, while the $l_0$ quasi-norm treats the coefficients equally, the $l_1$-norm penalizes the larger coefficients more heavily than smaller ones. To solve this imbalance, E.J. Candès et al. proposed a weighted $l_1$ minimization to more democratically penalize nonzero coefficients [5].

Consider the weighted $l_1$ minimization problem

\[
\min_{z_i \in \mathbb{R}^N} \| Wz_i \|_1 \quad \text{s.t.} \quad x_i = Xz_i, z_{ii} = 0,
\]

(10)

where $W \in \mathbb{R}^{N \times N}$ is the diagonal matrix, and the diagonal elements of this matrix are positive weights. When

\[
w_{jj} = \begin{cases} \frac{1}{z_{jj}}, & z_{jj} \neq 0 \\ \infty, & z_{jj} = 0 \end{cases}, \quad j = 1, 2, \ldots, N.
\]

(11)

the weighted $l_1$ minimization problem is guaranteed to find the correct solution of the $l_0$-norm minimization problem (3). How to obtain the weight matrix without the correct solution $Z$ is a remaining question. In literature [11], FOCUSS was proposed by Gorodnitsky and Rao as an iterative method for finding the sparse solutions of underdetermined systems. At each iteration, FOCUSS finds the solution of a reweighted $l_2$ minimization problem. M. Fazel et al. proposed a similar idea in [9]. They locally minimized the logarithm of the determinant as a smooth approximation for minimizing the rank of matrices by an iterative $l_1$-norm minimization technique, and examined the vector case as a special case. E.J. Candès et al. used the same strategy to solve the remaining question. They proposed an iterative algorithm that solves a sequence of weighted $l_1$-minimization problems. The value of the current solution is used for the weighted matrix computation of the next iteration. The iterative weighting $l_1$ minimization framework largely improves the performance of the standard $l_1$ minimization framework.

Xu J et al. applied this algorithm to segment high-dimensional data and proposed reweighted sparse subspace clustering (RSSC) [35]. Specifically, the iterative weighting $l_1$ minimization framework is used to solve the weighted $l_1$ minimization problem

\[
\min_{Z \in \mathbb{R}^{N \times N}} \| W \odot Z \|_1 \quad \text{s.t.} \quad X = XZ, \text{diag}(Z) = 0,
\]

(12)

where $\odot$ denotes the elementwise product between two matrices. The optimal solution $Z^*$ of this problem is used to construct the affinity matrix, like with SSC. The spectral clustering framework is employed to obtain the final clustering results for the affinity matrix. Due to the outperformance of iterative weighting $l_1$ minimization, RSSC is more accurate when compared to other state-of-the-art subspace clustering algorithms such as SSC.

3. **Side-information-induced reweighted sparse subspace clustering.** In the linear space or affine space, the geometric information is important structural information for subspace clustering, especially the angle between each pair of points.
In general, the angle between two points in the same subspace is less than the angle between two points in different subspaces.

Definition 1: The angle between two data points $x_i$ and $x_j$, which is denoted as $\theta_{ij}$, is defined as the cluster selection using the improved Sparse Subspace Clustering:

$$\cos(\theta_{ij}) \triangleq \frac{x_i^T x_j}{\|x_i\|_2 \|x_j\|_2}$$

(13)

According to the formula above, the cosine of the angle between every two points is greater than 0 and less than 1, i.e., $\cos(\theta_{ij}) \in [0, 1]$.

3.1. Side-information-induced reweighted $l_1$-norm minimization program.

To make full use of geometric information between data points, we propose a side-information-induced $l_0$ minimization problem that heavily penalizes the representative coefficient $z_{ij}$ where the angle between points $x_i$ and $x_j$ is larger than others. This can be formulated as

$$\min_{Z \in \mathbb{R}^{N \times N}} \|D \odot Z\|_0$$

s.t. $X = XZ, \text{diag}(Z) = 0.$

(14)

The weighting matrix $D \in \mathbb{R}^{N \times N}$ is the side-information, which is also a penalty for the coefficient matrix $Z$. We take the element of $D$ as $d_{ij} = \rho \cdot e^{-\frac{x_i^T x_j}{\|x_i\|_2 \|x_j\|_2}}$. $\rho$ is positive, and it is taken as the scale parameter. In this work, we set $\rho = 1$.

According to the definition of the $l_0$ quasi-norm, the solutions of equation (14) and equation (4) are equivalent. As we know, the optimization of $l_0$ quasi-norm minimization is NP-hard. We use the $l_1$-norm to relax the $l_0$ quasi-norm, as is the usual practice. Then, equation (11) could be relaxed as follows:

$$\min_{Z \in \mathbb{R}^{N \times N}} \|D \odot Z\|_1$$

s.t. $X = XZ, \text{diag}(Z) = 0.$

(15)

When all the elements of $D$ are one, the above problem is a standard $l_1$ minimization problem. It is convex and can be efficiently solved via a convex programming algorithm [3][12]. Obviously, the solution of (15) is not the correct solution of (14). How does this happen? According to the definitions of the $l_0$ quasi-norm and the $l_1$-norm, while the $l_0$ quasi-norm treats the coefficients equally, the $l_1$ -norm penalizes the larger coefficients more heavily than smaller ones. To solve this imbalance, E.J. Candès et al. proposed a weighted $l_1$ minimization to more democratically penalize nonzero coefficients [5]. The performance of the weighted $l_1$ minimization is better than the standard $l_1$ minimization in many applications. Xu J et al. applied this algorithm to segment high-dimensional data and proposed reweighted sparse subspace clustering (RSSC) [35]. Based on this heuristic, we propose a side-information-induced reweighted $l_1$-norm minimization as follows:

$$\min_{Z \in \mathbb{R}^{N \times N}} \|W \odot D \odot Z\|_1$$

s.t. $X = XZ, \text{diag}(Z) = 0.$

(16)

where this weighting matrix $W = [w_1, w_2, \ldots, w_N] \top \in \mathbb{R}^{N \times N}$ is designed to solve equation (16), which is more approximate to the solution of the equation (14). We term this model as side-information-induced reweighted sparse subspace clustering (SRSSC).
3.2. Iterative reweighted algorithm for side-information-induced $l_1$-norm minimization. In the last few decades, the $l_1$-norm was widely used as a sparsity-promoting functional, and the sparsity-promoting nature of the $l_1$ norm minimization was empirically confirmed. With heuristic literature [35], we employ the same strategy to solve problem (16).

The SRSSC can be written as:

$$\min_{z_i \in \mathbb{R}^N} \sum_{j=1}^{N} |w_{ij}d_{ij}z_{ij}| \quad i = 1, 2, \ldots, N. \quad (17)$$

$$s.t. x_i = Xz_i, z_{ii} = 0$$

The above problem (17) equals problem (16), and can be viewed as a relaxation of the weighted $l_0$ minimization problem. Intuitively, the larger element of the weighting matrices $w_{ij}$ will penalize $d_{ij}z_{ij}$ more heavily, which will make it smaller or even zero. Inversely, a smaller $w_{ij}$ will encourage the magnitude of the penalty of $d_{ij}z_{ij}$ to be larger or even nonzero. Hence, as a rule of thumb, the weight $w_{ij}$ should be inversely related to the true absolute value of $d_{ij}z_{ij}$. However, if we do not know the true values of $d_{ij}z_{ij}$, how can we know the weighting matrix $W$? The iterative weighting $l_1$ minimization framework [5] which proposed by E.J. Candès et al., largely improves the performance of the standard $l_1$ minimization framework. Similar to this method, we can make full use of the log-sum heuristic method to solve problem (17).

Consider the following problem:

$$\min_{z_i \in \mathbb{R}^N} \sum_{j=1}^{N} \log(|d_{ij}z_{ij}| + \epsilon) \quad i = 1, 2, \ldots, N. \quad (18)$$

$$s.t. x_i = Xz_i, z_{ii} = 0,$$

where $\epsilon > 0$. Equivalently, this problem (18) can be written as follows:

$$\min_{z_i, u_i \in \mathbb{R}^N} \sum_{j=1}^{N} \log(u_{ij} + \epsilon) \quad i = 1, 2, \ldots, N. \quad (19)$$

$$s.t. x_i = Xz_i, z_{ii} = 0, |d_{ij}z_{ij}| \leq u_{ij}$$

Obviously, if $z_i^*$ is a solution of problem (18), then $(z_i^*, u_i^*)$ is a solution to (19) and vice versa. However, the surrogate objective function $\sum_{j=1}^{N} \log(u_{ij} + \epsilon)$ is concave and smooth. We can use an iterative linearization method to find the solution of the cardinality minimization [9]. Let $z_i^{(k)}, u_i^{(k)}$ and $d_i^{(k)}$ denote the $k$th iterations of the optimization variables $z_i, u_i$ and $d_i$, respectively. The first-order Taylor series expansion of the concave surrogate function $\sum_{j=1}^{N} \log(u_{ij} + \epsilon)$ about $z_{ij}$ is given by the following:

$$\sum_{j=1}^{N} \log(u_{ij} + \epsilon) \approx \sum_{j=1}^{N} \log(u_{ij}^{(k)} + \epsilon) + \sum_{j=1}^{N} \frac{d_{ij}^{(k)}}{u_{ij}^{(k)} + \epsilon}(z_{ij} - z_{ij}^{(k)}) \quad (20)$$

Since the gradient of $\log(u_{ij} + \epsilon)$ with respect to $z_{ij}$ is $\frac{d_{ij}^{(k)}}{u_{ij}^{(k)} + \epsilon}$, the iterative linearization of the concave function gives the heuristic method for the cardinality
minimization as follows:
\[
(z_i^{(k+1)}, u_i^{(k+1)}, d_i^{(k+1)}) = \arg\min_{u_i, z_i, d_i} \sum_{j=1}^{N} \frac{d_{ij}^{(k)}}{u_{ij}^{(k)} + \epsilon} z_{ij}
\]
\[
s.t. \, x_i = X z_i, \, |d_{ij} z_{ij}| \leq u_{ij}, \, i = 1, 2, \ldots, N,
\]
which is equivalent to
\[
(z_i^{(k+1)}, d_i^{(k+1)}) = \arg\min_{u_i, z_i, d_i} \sum_{j=1}^{N} \frac{1}{u_{ij}^{(k)} + \epsilon} |d_{ij}^{(k)} z_{ij}|
\]
\[
s.t. \, x_i = X z_i, \, i = 1, 2, \ldots, N,
\]

Each iteration is the solution of a convex optimization problem as a weighted \(l_1\)-norm minimization problem. Therefore, we can employ the algorithm to solve problem (16). At the beginning of the iteration, we use the data points \(x_i\) and \(x_j\) to obtain the \(d_{ij}^{(0)} = e^{-\frac{x_i^T x_j}{2 \sigma_i^2 \sigma_j^2}}\), and we choose \(w_{ij}^{(0)} = [1, 1, \ldots, 1] \in R^{N}\) when the solution of problem (13) is the solution of the standard \(l_1\)-norm minimization. Based on (19), the \((i, j)\) entry of the weighting matrix is \(w_{ij}^{(k)} = \frac{1}{|d_{ij}^{(k)} z_{ij}| + \epsilon}\) in the \(k\)-th iteration, and \(d_{ij}^{(k+1)} = e^{-\frac{(x_i^{(k)} x_j^{(k)})^T (x_i^{(k)} x_j^{(k)})}{2 \sigma_i^2 \sigma_j^2}}\). This also reflects that the weights are inversely proportional to the solution.

3.3. **ADMM algorithm for solving SRSSC.** Problem (16) can be solved using the well-known lasso problem [27]. Based on the Alternating Direction Method of Multipliers (ADMM) [2] framework, we consider the following equivalent optimization program:

\[
Z = \arg\min \|W \odot D \odot Z\|_1 + \lambda_e \|E\|_1 + \frac{\lambda_x}{2} \|X - XZ - E\|_F^2
\]
\[
s.t. A^\top 1 = 1, A = Z - \text{diag}(Z).
\]

Here, matrix \(A\) helps to obtain faster updates as an auxiliary matrix and parameters \(\lambda_e\) and \(\lambda_x\) balance the three terms in the objective function. Two Lagrangian multipliers \(\delta \in R^{N}\) and \(\Delta \in R^{N \times N}\) augment the penalty terms to obtain the Lagrange function of the optimization program from (23) as

\[
L(A, Z, W, D, W, \delta, \Delta) = \|W \odot D \odot Z\|_1 + \lambda_e \|E\|_1
\]
\[
+ \frac{\lambda_x}{2} \|X - XZ - E\|_F^2
\]
\[
+ \delta^\top (A^\top 1 - 1) + tr(\Delta^\top (A - (Z - \text{diag}(Z))))
\]
\[
+ \frac{\mu}{2} (\|A^\top 1 - 1\|^2_2 + \|A - (Z - \text{diag}(Z))\|^2_2)
\]

where \(tr(\cdot)\) is the trace operator. We update \(A, Z, W, D, E, \delta\) and \(\Delta\) alternatively while keeping the other variables fixed. We use \(A^{(k)}, Z^{(k)}\) and \(E^{(k)}\) to denote the optimization variables at the \(k\)-th iteration \(\delta^{(k)}\) and \(\Delta^{(k)}\) as the Lagrange multipliers at the \(k\)-th iteration.

**Obtain for \(A^{(k+1)}\)** By fixing \(A^{(k+1)}, W^{(k)}, D^{(k)}, \delta^{(k)}\) and \(\Delta^{(k)}\) and minimizing the Lagrange function (24) with respect to \(Z\), we obtain \(Z^{(k+1)}\) using a closed-form
Algorithm 1: ADMM Algorithm for solving problem (23)

**Input:** Data matrix and the maximal iteration number K

**Initialize:** set $k = 0$. Initialize $A^{(0)}, Z^{(0)}, E^{(0)}, \delta^{(0)}$, and $\Delta^{(0)}$, to zero and initialize $W^{(0)} = \frac{1}{\|x\|^2},$ where $D^{(0)} = e^{-\left(\frac{1}{\|x_i\|^2}\right)}$, and $i, j = 1, 2, \ldots, N$.

**while** $k < K$ **or** not converge **do**

- Update $A^{(k+1)}, Z^{(k+1)}, E^{(k+1)}$ according to formulas (25), (26), and (28), respectively.
- Update $W^{(k+1)} = \frac{\epsilon_2}{D^{(k+1)} \circ Z^{(k+1)} + \epsilon_1}$ where $D(k+1) = e^{-\left(\frac{1}{\|x_i(k)\|^2}\right)}$, $i, j = 1, 2, \ldots, N$.
- Update the Lagrange multipliers $\delta^{(k+1)}$ and $\Delta^{(k+1)}$ according to formulas (31) and (32).
- $k = k + 1$.

**End while**

**Output:** $Z^* = Z^{(k+1)}$.

solution as

$$Z^{(k+1)} = \hat{Z}^{(k+1)} - \text{diag}(\hat{Z}^{(k+1)})$$

$$\hat{Z}^{(k+1)} = \tau_\eta(W^{(k)} \circ D^{(k)}) \left( A^{(k+1)} + \frac{\Delta^{(k)}}{\mu} \right)$$

(25)

where $\tau_\eta(W(\cdot))$ is the shrinkage-thresholding operator that is defined by

$$\tau_\eta(W(X)) = \max(|X| - \eta W) \odot \text{sgn}(X).$$

(26)

**Update for $E^{(k+1)}$** By fixing $A^{(k+1)}$ and minimizing the Lagrange function (24) with respect to $\epsilon$, we obtain

$$E^{(k+1)} = \tau_{\lambda_2}(X A^{(k+1)} - X).$$

(27)

**Update for $D^{(k+1)}$ and $W^{(k+1)}$** By fixing $Z^{(k+1)}$, we obtain $D^{(k+1)}$ and $W^{(k+1)}$ as follows:

$$D^{(k+1)} = \{d_{ij}^{(k+1)} = e^{-\left(\frac{x_i^{(k)} \cdot x_j^{(k)}}{\|x_i^{(k)}\|^2 + \|x_j^{(k)}\|^2}\right)} \},$$

(28)

$$W^{(k+1)} = \frac{\epsilon_2}{D^{(k+1)} \circ \hat{Z}^{(k+1)} + \epsilon_1}.$$

(29)

**Update for $\delta^{(k+1)}$ and $\Delta^{(k+1)}$** By fixing $A^{(k+1)}$ and $Z^{(k+1)}$, we obtain the Lagrange multipliers $\delta^{(k+1)}$ and $\Delta^{(k+1)}$ with the step size of $\mu$ as

$$\delta^{(k+1)} = \delta^{(k)} + \mu(A^{(k+1)} - 1),$$

(30)

$$\Delta^{(k+1)} = \Delta^{(k)} + \mu(A^{(k+1)} - Z^{(k+1)}).$$

(31)

In brief, the ADMM algorithm for solving the optimization program (23) proceeds as shown in Algorithm 1. For more details on the ADMM algorithm, the readers can refer to [2].

After solving the proposed SRSSC optimization program in (23), we obtain the sparse representation coefficient matrix of all the data points. Now, we use this
Algorithm 2 Side-Information-Induced Reweighted Sparse Subspace Clustering

Input: Data matrix \( X \in \mathbb{R}^{D \times N} \) and the number of desired clusters \( K \)

Solve problem (23) via Algorithm 1.

Form a similarity matrix \( C = \frac{1}{2}(|Z^*| + |Z^*^\top|) \).

Apply spectral clustering to the similarity matrix \( C \).

Output: The clustering result of the data points \( X \).

representation matrix to build a symmetric nonnegative similarity matrix

\[
C = \frac{1}{2}(|Z^*| + |Z^*^\top|). \tag{32}
\]

Then, we obtain the clustering results of the data points by applying spectral clustering to the similarity matrix \( C \).

In summary, the SRSSC algorithm can be outlined as in Algorithm 2.

3.4. Convergence analysis. Similar to RSSC, we update the weighting matrix \( W \) by approximating the \( l_0 \) quasi-norm using the log-sum surrogate function. Since the log-sum function is concave, the solution of its minimization problem can be found using an iterative linearization method. As shown in [5], the solution of the reweighted \( l_1 \)-norm minimization with the log-sum surrogate function is a local minimizer. When Algorithm 1 converges to a local minimum, the method that is proposed in this paper may not converge. In practice, if the optimal parameters are set wisely, the convergence of our method can be guaranteed. The experimental results in the next section also show the convergence of our method. Figure 3 illustrates the convergence behavior of SRSSC on the Extended YaleB dataset.

![Figure 3. Convergence of SRSSC on the Extended YaleB dataset](image)

4. Numerical experiments. In this section, we implement synthetic data and three famous real datasets, including the Extended Yale B, COIL 20 and USPS datasets, to test the clustering performance of our proposed SRSSC method when applied to subspace clustering. We compare the performance of SRSSC with several state-of-the-art subspace clustering algorithms, such as LSR, SMR, LRR, SSC, StrSSC, RSSC and our proposed SRSSC algorithm. The codes of the algorithms that are mentioned above were released by the authors. All the parameters that
were mentioned in their papers are kept the same, and we obtain the same results as they did in their papers.

The performance of the subspace clustering algorithms is measured using the clustering error, which is defined as follows:

$$ Error = \frac{N_{\text{error}}}{N_{\text{total}}} $$ \hspace{1cm} (33)

where $N_{\text{error}}$ denotes the number of misclassified points and $N_{\text{total}}$ represents the total number of data points. A smaller clustering error means better clustering performance.

All the experiments are performed in MATLAB 2014a on a PC with a 2.40 GHz Intel(R) Xeon(R) CPU and 8.00 GB RAM.

4.1. Experiments with synthetic data sets. In this experiment, we generate $K$ subspaces $\{s_i\}_{i=1}^{K}$ that are embedded in the $D$-dimensional ambient space. We set $K = 5$ and $D = 30$. The dimensions of these subspaces are 3, 5, 8, 11, and 15, respectively. Specifically, for each subspace, we obtain the orthogonal basis using the Singular Value Decomposition (SVD) [15] of a random matrix with $d_i \times n_i$ rows. Now, we generate the data vectors with $X_i = U_i R_i$, $1 \leq i \leq K$, where $R_i$ is a $d_i \times n_i$ i.i.d.$N(0, 1)$ random matrix. In our experiment, we set $n_i = 300$. We sample 100 data vectors from each subspace $X_i$ to construct the synthetic data set $X \in R^{D \times N}$. Here, $D = 30$ and $N = 500$. Finally, white Gaussian noise with a signal-to-noise ratio (snr) [34] of 15 is added to the data matrix (in MATLAB using awgn(X, snr)). We compare the performances of LSR, SMR, LRR, SSC, StrSSC, RSSC and our proposed SRSSC algorithm. All the parameters that were mentioned in their paper are kept the same. We set $\lambda = 753$ for both the RSSC and our SRSSC algorithm as in SSC. In RSSC, $\epsilon_1 = 9 \times 10^{-3}$ and $\epsilon_2 = 2.7 \times 10^{-4}$. In SRSSC, parameters $\epsilon_1$ and $\epsilon_2$ are kept the same as with RSSC. We repeated this experiment 10 times to make the results more general. The similarity matrices

![Figure 4](image-url)
Table 1. Performance comparison of the different algorithms using the synthetic data

| Algorithms | LSR | SMR | LRR | SSC | StrSSC | RSSC | Ours |
|------------|-----|-----|-----|-----|--------|------|------|
| Accuracy   |     |     |     |     |        |      |      |
| Mean       | 93.57% | 94.35% | 92.46% | 94.04% | 94.08% | 96.60% | 97.00% |
| Median     | 92.85% | 94.07% | 92.13% | 93.60% | 93.70% | 96.10% | 96.90% |

that are obtained by SSC, StrSSC, RSSC and SRSSC are shown in Figure 4. All of these similarity matrices in Figure 4 have obvious block diagonal structures. The similarity matrix in Figure 4(d) that was obtained by our SRSSC algorithm is sparser than that of Figure 4(a),(b) and (c). The accuracies of the state-of-the-art subspace clustering algorithms are reported in Table 1.

4.2. Experiments using the Extended Yale B dataset. For the face clustering problem, multiple subjects are required to take facial images under different illumination conditions. Then, the images are clustered according to their subjects. As shown in [10], under the Lambertian assumption, the facial images of a subject under a wide variety of lighting conditions approximate a linear subspace of 9 dimensions. Therefore, the face clustering problem can be regarded as a subspace clustering problem.

Figure 5. Some sample images from the Extended Yale B database

The Extended Yale B dataset includes the facial images of 38 human subjects, and each subject contains 64 frontal face images under various illumination conditions. To reduce the computational and storage complexity, the authors of [17] downsampled every image from 192 × 168 pixels to 48 × 42 pixels and vectorized the image as a 2016-dimensional data point. Figure 5 shows some sample images from the Extended Yale B database. By adopting the same protocol that was proposed in [5], we distribute the 38 subjects into 4 groups: 1-10, 11-20, 21-30 and 31-38. We perform all choices of subjects for the first three groups. For the last group, we consider all choices of \( n \in \{2, 3, 4, 5, 6, 7, 8\} \) subjects. In the experiment of our proposed method, we set \( \epsilon_1 = 0.06, \epsilon_2 = 0.03 \) and \( \lambda_Z = 753 \). The clustering results
Table 2. The clustering errors (%) of some different algorithms using the Extended Yale B dataset

| No. of subject | Algorithms | LSR | SMR | LRR | SSC | StrSSC | RSSC | Ours |
|----------------|------------|-----|-----|-----|-----|--------|------|------|
| 2 Subjects     | Mean       | 7.35| 1.75| 2.13| 1.87| 1.23   | 0.57 | 0.48 |
|                | Median     | 7.03| 0.78| 0.78| 0   | 0      | 0    | 0    |
| 3 Subjects     | Mean       | 9.92| 3.03| 3.49| 3.29| 2.84   | 1.08 | 0.78 |
|                | Median     | 10.41| 2.08| 2.08| 0.52| 0.52   | 0    | 0    |
| 4 Subjects     | Mean       | 13.66| 3.25| 4.86| 4.86| 2.95   | 1.65 | 1.07 |
|                | Median     | 14.07| 2.34| 3.91| 1.95| 0.78   | 0.39 | 0.39 |
| 5 Subjects     | Mean       | 17.56| 3.91| 5.92| 4.33| 3.31   | 2.21 | 1.40 |
|                | Median     | 17.81| 2.50| 4.99| 2.50| 1.25   | 0.62 | 0.62 |
| 6 Subjects     | Mean       | 20.95| 5.28| 6.83| 4.87| 3.73   | 2.79 | 1.68 |
|                | Median     | 21.07| 2.86| 5.99| 3.39| 2.08   | 1.30 | 1.04 |
| 7 Subjects     | Mean       | 24.31| 6.38| 7.75| 5.40| 4.01   | 3.43 | 2.03 |
|                | Median     | 24.10| 3.13| 7.14| 4.46| 2.68   | 1.79 | 1.34 |
| 8 Subjects     | Mean       | 27.52| 6.83| 11.05| 5.92| 4.38   | 3.98 | 2.59 |
|                | Median     | 27.83| 3.71| 7.42| 4.69| 3.13   | 1.86 | 1.56 |
| 9 Subjects     | Mean       | 31.01| 7.14| 10.32| 6.46| 4.56   | 4.55 | 2.84 |
|                | Median     | 31.42| 4.51| 7.81| 4.77| 3.65   | 2.43 | 1.65 |
| 10 Subjects    | Mean       | 33.49| 7.81| 16.95| 7.40| 4.74   | 4.90 | 3.33 |
|                | Median     | 32.81| 7.03| 18.91| 5.63| 4.22   | 3.59 | 2.03 |

of the state-of-the-art algorithms that are mentioned above are presented in Table 2.

The average computation times of the different algorithms using the Yale B dataset are shown in Figure 6. LSR and SMR are faster than LRR, SSC, StrSSC, RSSC and SRSSC. Specially, the computation time of StrSSC is significantly longer than that of LRR, SSC, RSSC and SRSSC.

Figure 6. The average computation times of the different algorithms using the Yale B dataset

4.3. Experiment using the COIL 20 dataset. The COIL 20 database contains 20 objects. Figure 7 exhibits some representative examples from the COIL 20 dataset. The images of every object were taken 5 degrees apart when the object
Table 3. The clustering errors (%) of some comparative algorithms using the COIL 20 dataset

| No. of subject | Algorithms | LSR | SMR | LRR | SSC | StrSSC | RSSC | Ours |
|----------------|------------|-----|-----|-----|-----|--------|------|------|
| 2 Subjects     | Mean       | 15.05 | 13.75 | 14.86 | 8.13 | 0.76   | 1.35 | **0.43** |
|                | Median     | 13.91 | 10.42 | 13.07 | 0    | 0      | 0    | 0    |
| 3 Subjects     | Mean       | 22.16 | 21.97 | 21.81 | 10.87 | 1.86   | 1.37 | **0.72** |
|                | Median     | 20.22 | 19.44 | 19.65 | 1.85  | 0      | 0    | 0    |
| 20 Subjects    | Mean(Median)| 25.49 | 24.72 | 24.67 | 20.07 | 15.73  | 16.32 | **7.78** |

was rotated on a turntable. Therefore, each object has 72 images that are $32 \times 32$ pixels in size with 256 gray levels. Then, each image is transposed to a 1024-dimensional vector as a point. In our experiment, we consider $n \in \{2, 3, 20\}$ objects in the database. For two objects, there are 190 cases. When $n = 3$, the number of different cases is 1140. Therefore, this experiment is very representative of other cases that do not consider in this paper. In this experiment, all the parameters are the same as those in their papers. In our proposed SRSSC algorithm, we set $\epsilon_1 = 0.06, \epsilon_2 = 0.003$ and $\lambda_Z = 10$. The clustering results are exhibited in Table 3.

Figure 7. Some sample images from the COIL 20 database (top) and the images that belong to the same object (bottom)

4.4. Experiment using the USPS dataset. The USPS digit database is a famous dataset for handwritten digit recognition. It is often used as a benchmark dataset for the subspace clustering algorithm. It consists of 11000 images of 10 subjects corresponding to handwritten digits from 0-9, respectively. Each image is $16 \times 16$ pixels. Some sample images from USPS are shown in Figure 8. As we see, some handwritten digits are scrawled, which makes it difficult to recognize handwritten digits. In the experiment, we consider 2, 3 and 10 objects in the USPS database. We set $\epsilon_1 = 0.07, \epsilon_2 = 0.0003$ and $\lambda_Z = 30$. in our proposed SRSSC algorithm. The results of our experiments on the USPS dataset are listed in Table 4. It shows that SRSSC achieves the lowest clustering error. It is well known that handwritten digits 1 and 7, 2 and 3, 3 and 5, 6 and 0, 7 and 9 are very similar, which makes it very difficult to distinguish them accurately when there are 10 objects.

5. Conclusion. In this work, we present side-information-induced reweighted sparse subspace clustering. The proposed algorithm makes the most of the geometric information between pairs of data. This information is directly related to the underlying structure of the data, so it is very useful for subspace clustering. Our proposed method is the simplest but the most effective. Extensive experiments demonstrate that SRSSC outperforms the existing state-of-the-art algorithms. In our future research, we will utilize more information about the data to design more effective models for subspace clustering and improve the accuracy of subspace clustering.
Figure 8. Some sample images from the USPS database

table 4. The clustering errors (%) of some comparative algorithms using the USPS dataset

| No. of subject | Algorithms | LSR | SMR | LRR | SSC | StrSSC | RSSC | Ours |
|----------------|-----------|-----|-----|-----|-----|--------|------|------|
| 2 Subjects     | Mean      | 15.49 | 15.25 | 14.76 | 11.80 | 11.48 | 10.90 | **10.80** |
|                | Median    | 13.91 | 13.42 | 13.07 | 9.00  | 8.50  | 9.50  | **7.00**  |
| 3 Subjects     | Mean      | 29.13 | 28.91 | 28.81 | 27.59 | 27.88 | 27.44 | **26.67** |
|                | Median    | 27.02 | 16.34 | 26.37 | 21.83 | 25.67 | 23.82 | **21.33** |
| 10 Subjects    | Mean(Median) | 46.25 | 45.97 | 45.53 | 44.89 | 44.25 | 44.03 | **43.95** |

Acknowledgments. The authors would like to thank the National Natural Science Foundation of China (Grants 61472303, 61772389) and the Fundamental Research Funds for the Central Universities (Grant NSIY21) for supporting this research work.

References

[1] R. G. Baraniuk and et al., Applications of sparse representation and compressive sensing [scanning the issue], Proceedings of the IEEE 98, 6 (2010), 906–909.
[2] S. Boyd and et al., Distributed optimization and statistical learning via the alternating direction method of multipliers, Foundations & Trends in Machine Learning, 3 (2010), 1–122.
[3] S. Boyd and L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.
[4] M. Brbić and I. Kopriva, \(l_0\)-Motivated Low-Rank Sparse Subspace Clustering, IEEE Transactions on Cybernetics, 2018.
[5] E. J. Candes, M. B. Wakin and S. P. Boyd, Enhancing sparsity by reweighted \(l_1\) minimization, Journal of Fourier Analysis & Applications, 14 (2008), 877–905.
[6] H. Z. Chen and et al., Discriminative and coherent subspace clustering, Neurocomputing, 284 (2018), 177–186.
[7] H. Z. Chen, W. Wang and X. Feng, Structured sparse subspace clustering with group-effect-within-cluster, Pattern Recognition, (2018), S0031320318301948.
[8] E. Elhamifar and R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), 2765–2781.
[9] M. Fazel, H. Hindi and S. P. Boyd, Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. American Control Conference, Proceedings of the 2003 IEEE, (2003).
[10] A. S. Georghiades, P. N. Belhumeur and D. J. Kriegman, From few to many: Illumination cone models for face recognition under variable lighting and pose, IEEE Transactions on Pattern Analysis and Machine Intelligence, 23 (2001), 643–660.
[11] I. F. Gorodnitsky and B. D. Rao, Sparse signal reconstruction from limited data using FO-CUSS: A re-weighted minimum norm algorithm, IEEE Transactions on Signal Processing, 45 (2002), 600–616.
[12] M. Grant, S. Boyd and Y. Ye, CVX: Matlab software for disciplined convex programming, (2008).
[13] H. Hu, Z. Lin, J. Feng and et al., Smooth representation clustering, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2014), 3834–3841.
[14] J. J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell., 16 (1994).
[15] V. C. Klema and A. J. Laub, The singular value decomposition: Its computation and some applications, IEEE Transactions on Automatic Control, 25 (1980), 164–176.
[16] V. P. Kshirsagar, M. R. Baviskar and M. E. Gaikwad, Face Recognition Using Eigenfaces, International Conference on Computer Research & Development, 2011.
[17] H. Hu, Z. Lin, J. Feng and et al., Smooth representation clustering, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, 3834–3841.
[18] J. J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell., 16 (1994).
[19] V. C. Klema and A. J. Laub, The singular value decomposition: Its computation and some applications, IEEE Transactions on Automatic Control, 25 (1980), 164–176.
[20] V. P. Kshirsagar, M. R. Baviskar and M. E. Gaikwad, Face Recognition Using Eigenfaces, International Conference on Computer Research & Development, 2011.
[21] H. Hu, Z. Lin, J. Feng and et al., Smooth representation clustering, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, 3834–3841.
[22] J. J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell., 16 (1994).
[23] V. C. Klema and A. J. Laub, The singular value decomposition: Its computation and some applications, IEEE Transactions on Automatic Control, 25 (1980), 164–176.
Received March 2019; revised July 2019.

E-mail address: hhuang@cqwu.edu.cn
E-mail address: wwwwang@mail.xidian.edu.cn
E-mail address: l.chengwu@163.com
E-mail address: xcfeng@mail.xidian.edu.cn
E-mail address: ruiqianghe@sina.com