Simplex Representation for Subspace Clustering

Jun Xu[1], Mengyang Yu[1], Ling Shao[1], *Senior Member, IEEE*, Wangmeng Zuo[2], *Senior Member, IEEE*, Deyu Meng[3], Lei Zhang[4], *Fellow, IEEE*, David Zhang[4,5], *Fellow, IEEE*
[1]Inception Institute of Artificial Intelligence, Abu Dhabi, UAE
[2]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[3]School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, China
[4]Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR, China
[5]School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen), Shenzhen, China

The self-expressive property of data points, i.e., each data point can be linearly represented by the other data points in the same subspace, has proven effective in leading subspace clustering methods. Mainstream self-expressive based methods follow a three-stage framework: 1) a coefficient matrix is computed from the data by solving an optimization problem; 2) an affinity matrix is constructed from the coefficient matrix; and 3) the final segmentation is obtained by applying spectral clustering to the affinity matrix. To construct a feasible affinity matrix, most of these methods employ exponentiation, absolute symmetrization, or squaring operations. However, all these operations force negative entries in the coefficient matrix (which cannot be explicitly avoided) to be positive when computing the affinity matrix, and consequently damage the inherent correlations among the data. To overcome this problem, in this paper, we introduce a simplex representation (SR) scheme as an alternative to previous sparse or low-rank representations. Specifically, we introduce non-negativity into the representation to make it more physically meaningful, and constrain the sum of the representational vector to maintain its discrimination of the original data. The proposed SR based subspace clustering (SRSC) model is reformulated as a linear equality-constrained problem, which is solved efficiently under the alternating direction method of multipliers framework. Experiments on benchmark datasets demonstrate that the proposed SRSC algorithm is very efficient and outperforms state-of-the-art subspace clustering methods on accuracy.

***Index Terms*—Subspace clustering, simplex representation, self-expressiveness.**

I. Introduction

IMAGE processing problems often contain high-dimensional data, whose structure typically lie in a union of low-dimensional subspaces [1]. Recovering these low-dimensional subspaces from the high-dimensional data can reduce the computational and memory cost of subsequent algorithms. To this end, many image processing tasks require that the high-dimensional data is clustered in such a way that each cluster can be fitted by a low-dimensional subspace. This is known as the subspace clustering (SC) problem [1].

SC has been extensively studied over the past few decades [2]–[48]. Most existing SC methods fall into one of four categories: iterative methods [2], [3], algebraic methods [4], [5], [14], statistical methods [6]–[9], and self-expressive methods [10]–[13], [15]–[48]. Among these methods, self-expressive ones are most commonly studied due to their theoretical soundness and promising performance in real-world applications, such as motion segmentation [16], face clustering [18], and digit clustering [45]. Self-expressive methods usually follow a three-step framework. **Step 1**: a coefficient matrix is obtained for the data points by solving an optimization problem, which usually incorporates sparse or low-rank regularizations because of their good mathematical properties. **Step 2**: an affinity matrix is constructed from the coefficient matrix by employing exponentiation [13], absolute symmetrization [15], [16], [29], [32]–[40], [45], and squaring operations [17]–[24], [26]–[28], [41], [43], etc. **Step 3**: spectral techniques [49] are applied to the affinity matrix and the final clusters are obtained for the data points.

Self-expressive methods [13], [15]–[24], [26]–[35], [37]–[48] obtain the expected coefficient matrix under the famous *self-expressiveness* property [15], which states that each data point in a union of multiple subspaces can be *linearly* represented by the other data points in the same subspace. In some real-world applications, the data points often lie in a union of multiple affine rather than linear subspaces [16]. To cluster data points lying in affine subspaces, the affine constraint is introduced [15], [16]. In this case, each data point can be written as an affine combination of the other points, with the sum of the coefficients being one. However, most self-expressive methods [13], [15]–[18], [28]–[35], [37]–[45], [47], [48] suffer from three major drawbacks. First, negative coefficients cannot be fully avoided since the existing methods do not explicitly consider non-negative constraint in **Step 1**. Though after symmetrizing the data, i.e., adding the negative of data to original data, is reasonable from the mathematic viewpoint. However, from the physical viewpoint, using symmetrizing data is physically problematic since these synthetic data lie in a very different space when compared to that of real-world data. The problem is, constructing the affinity matrix will make these negative coefficients positive by absolute operation. This is not reasonable, since after absolute operation, negative and positive coefficients all have the same positive affinity weights among data points. Hence, we argue that, in real-world applications, it is physically problematic to reconstruct a data point by allowing the others to "cancel each other out" with complex additions and subtractions [50]. Second, under the affine constraint, the coefficient vector is not flexible enough to handle real-world cases where the data

points are corrupted by noise or outliers. Third, the exponentiation, absolute symmetrization, and squaring operations in **Step 2** force negative coefficients to become positive, thus damaging the inherent correlations among the data points.

To solve the three drawbacks mentioned above, we introduce a Simplex Representation (SR) for self-expressive based subspace clustering. Specifically, the SR is introduced from two interdependent reasons. First, to make it more adaptable to real scenarios, we extend the affine constraint to the scaled affine constraint, in which the coefficient vector in the optimization model will sum to a scalar s ($s > 0$) instead of 1. By changing the value of s, we can alter the generative and discriminative properties of the optimization model. Note that the affine constraint is only a special case of the introduced scaled affine constraint when $s = 1$. Second, to make the overall optimization model more physically meaningful, we introduce a non-negative constraint to its coefficients. Introducing the non-negative constraint has three primary benefits: 1) in **Step 1**, it can eliminate the need for complex additions and subtractions among the data points in the optimization model. 2) Since the coefficient matrix is naturally non-negative in **Step 1**, the non-negative constraint maintains the inherent correlations among the data points by avoiding exponentiation, absolute symmetrization, and squaring when building the affinity matrix in **Step 2**. 3) With the scaled affine constraint, the non-negativity could potentially enhance the discriminability of the coefficients [51], so that an input data point will more likely be reconstructed by data points from the same subspace. With the introduced SR, we propose a novel SR-based Subspace Clustering (SRSC) model. Experiments on commonly used benchmark datasets show that the proposed SRSC is very efficient and achieves lower clustering errors than state-of-the-art SC algorithms. The major contributions of this article are summarized as follows:

- We introduce a new representation framework, i.e., the Simplex Representation (SR), as an alternative to the popular sparse or low-rank representation for solving the SC problem. The proposed SR can maintain the inherent correlations among the data points, and thus avoid damages on the affinity matrix brought by previous spectral clustering based SC methods.
- We propose an SRSC model to solve the SC problem. We reformulate the proposed SRSC model into a linear equality-constrained problem with two variables, and solve the relaxed problem with an alternating direction method of multipliers (ADMM) [52] algorithm. Each variable can be updated efficiently, and the convergence of the algorithm can be guaranteed.
- Experiments on three benchmark datasets, i.e., Hopkins 155 [53], Extended Yale B [54], and MNIST [55], demonstrate that the proposed SRSC algorithm is very efficient, and achieves much better performance than state-of-the-art SC algorithms on motion segmentation, face clustering, and hand-written digits/letters clustering.

The remainder of this paper is organized as follows. In §II, we provide a brief survey of related work. In §III, we first present the proposed SRSC model, then provide its optimization, and, finally, give an SRSC based SC algorithm. Extensive experiments are conducted in In §IV to evaluate the SRSC algorithm and compare it with state-of-the-art SC algorithms. Finally, conclusion is given in §V.

II. Related Work

A. Prior Work on Subspace Clustering

According to the mathematical frameworks they employ, most existing SC algorithms [2]–[18], [28]–[35], [37]–[45], [47], [48] can be divided into four main categories: iterative methods, algebraic methods, statistical methods, and self-expressive methods.

Iterative methods such as K-subspaces [2] and algebraic subspace clustering (ASC) [3] solve the SC problem by alternately between assigning points to subspaces and fitting subspaces to corresponding clusters. Such approaches are limited by their requirement to know the number and dimensions of the subspaces, and they are sensitive to the initialization.

Algebraic methods solve the SC problem by modeling each subspace as a gradient of a polynomial [4], [5], [14]. These methods do not require prior information for each subspace. Two representative methods in this category are the Shape Interaction Matrix (SIM) [4] and the Generalized Principal Component Analysis (GPCA) [5]. Algebraic methods suffer from exponentially expensive computations due to the polynomial fitting. Besides, these methods are difficult to handle points near the intersection of subspaces.

Statistical methods model each subspace as a Gaussian distribution, and solve the SC problem as estimating a Gaussian Mixture Model. Specific algorithms include Agglomerative Lossy Compression (ALC) [9], Mixture of Probabilistic Principle Component Analysis (MPPCA) [6], Multi-Stage Learning (MSL) [7], and the well-known RANSAC [8]. These methods typically require prior information of the latent subspaces, such as their number and dimensions. The computational complexity of these methods is also exponential with respect to the number of subspaces and their dimensions.

Self-expressive methods use local information around each data point to compare the similarity of two points. Inspired by the work of compressed sensing [56], [57], the Sparse Subspace Clustering (SSC) algorithm [15] solves the clustering problem by seeking a sparse representation of data points over themselves. By resolving the sparse representations for all data points and constructing an affinity graph, SSC automatically finds different subspaces as well as their dimensions from a union of subspaces. A robust version of SSC that deals with noise and corruptions or missing observations is also given in [16]. Instead of finding a sparse representation, the Low-Rank Representation (LRR) method [17], [18], [28] poses the SC problem as finding a low-rank representation of the data over themselves. Lu *et al.* proposed a clustering method based on Least Squares Regression (LSR) [32] to take advantage of data correlations and group highly correlated data together. The grouping information can be used to construct an affinity matrix that is block diagonal and can be used for SC through spectral clustering algorithms. Recently, Lin *et al.* analyzed

the grouping effect in depth and proposed a SMooth Representation (SMR) framework [33] which also achieves a state-of-the-art performance for the subspace clustering problem. Different from SSC, the LRR, LSR, and SMR algorithms all use Normalized Cuts [58] in the spectral clustering step. You *et al.* proposed a scalable Orthogonal Matching Pursuit (OMP) method [45] to solve the SSC model [16]. You *et al.* also developed an elastic-net subspace clustering (EnSC) model [46] which correctly predicted relevant connections among different clusters. Ji *et al.* [42] developed the first unsupervised network for subspace clustering by learning the *self-expressiveness* property [15].

B. The Self-expressiveness Based Framework

Most state-of-the-art subspace clustering (SC) methods are designed under the self-expressive framework. Mathematically, denoting the data matrix as $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_N] \in \mathbb{R}^{D\times N}$, each data point $\boldsymbol{x}_i \in \mathbb{R}^D$, $i \in \{1, ..., N\}$ in $\boldsymbol{X}$ can be expressed as

$$\boldsymbol{x}_i = \boldsymbol{X}\boldsymbol{c}_i, \tag{1}$$

where $\boldsymbol{c}_i \in \mathbb{R}^N$ is the coefficient vector. If the data points are listed column by column, (1) can be rewritten as

$$\boldsymbol{X} = \boldsymbol{X}\boldsymbol{C}, \tag{2}$$

where $\boldsymbol{C} \in \mathbb{R}^{N\times N}$ is the coefficient matrix. To find the desired $\boldsymbol{C}$, existing SC methods [15]–[18], [29], [32]–[35], [37]–[41], [45] impose various regularizations, such as sparsity and low rankness. Below, $\|\cdot\|_F$, $\|\cdot\|_1$, $\|\cdot\|_2$, $\|\cdot\|_{2,1}$, $\|\cdot\|_*$, λ, and p denote the Frobenius norm, the ℓ_1 norm, the ℓ_2 norm, the $\ell_{2,1}$ norm, the nuclear norm, the regularization parameter, and a positive integer, respectively. The optimization models of several representative works are summarized as follows:

Sparse Subspace Clustering (SSC) [16]:

$$\min_{\boldsymbol{C}} \|\boldsymbol{C}\|_1 \quad \text{s.t.} \quad \boldsymbol{X} = \boldsymbol{X}\boldsymbol{C}, \boldsymbol{1}^\top\boldsymbol{C} = \boldsymbol{1}^\top, \text{diag}(\boldsymbol{C}) = \boldsymbol{0}. \tag{3}$$

Low-Rank Representation (LRR) [18]:

$$\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_{2,1} + \lambda\|\boldsymbol{C}\|_*. \tag{4}$$

Least Squares Regression (LSR) [32]:

$$\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \quad \text{s.t.} \quad \text{diag}(\boldsymbol{C}) = \boldsymbol{0}. \tag{5}$$

SSC by Orthogonal Matching Pursuit (SSCOMP) [45]:

$$\min_{\boldsymbol{c}_i} \|\boldsymbol{x}_i - \boldsymbol{X}\boldsymbol{c}_i\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{c}_i\|_0 \le p, \boldsymbol{c}_{ii} = 0. \tag{6}$$

Once the coefficient matrix $\boldsymbol{C}$ is computed, the affinity matrix $\boldsymbol{A}$ is usually constructed by exponentiation [13], absolute symmetrization [15], [16], [29], [32]–[35], [37]–[40], [45], and squaring operations [17], [18], [41], etc. For example, the widely used absolute symmetrization operation in [15], [16], [29], [32]–[35], [37]–[40], [45] is defined by

$$\boldsymbol{A} = \frac{|\boldsymbol{C}| + |\boldsymbol{C}^\top|}{2}. \tag{7}$$

After the affinity matrix $\boldsymbol{A}$ is obtained, spectral clustering techniques [58] are applied to obtain the final segmentation of the subspaces.

However, the self-expressiveness based framework suffers from one major drawback: the exponentiation, absolute symmetrization, and squaring operations will force the negative entries in $\boldsymbol{C}$ to be positive in $\boldsymbol{A}$, and hence damage the inherent correlations among the data points in $\boldsymbol{X}$. Besides, the affine constraint in SSC limits the model flexibility, making it difficult for SSC to deal with complex real world applications. In order to remedy these drawbacks, in this paper, we introduce the Simplex Representation into the SC problem.

III. Simplex Representation Based Subspace Clustering

In this section, we propose a Simplex Representation (SR) based subspace clustering (SRSC) model, develop an optimization algorithm to solve it, and present a novel SRSC based algorithm for subspace clustering (SC).

A. Proposed SRSC Model

$$\min_{\boldsymbol{c}_i} \|\boldsymbol{x}_i - \boldsymbol{X}\boldsymbol{c}_i\|_2^2 + \lambda\|\boldsymbol{c}_i\|_2^2 \text{ s.t. } \boldsymbol{c}_i \ge 0, \boldsymbol{1}^\top\boldsymbol{c}_i = s, \tag{8}$$

where $\boldsymbol{c}_i$ is the coefficient vector of $\boldsymbol{x}_i$ over the data matrix $\boldsymbol{X}$, and $\boldsymbol{c}_i \ge 0$ means that each entry of $\boldsymbol{c}_i$ is non-negative. $\boldsymbol{1}$ is the vector of all ones and $s > 0$ is a scalar denoting the sum of entries in the coefficient vector $\boldsymbol{c}_i$. We use the term "simplex" here because the entries in the coefficient vector $\boldsymbol{c}_i$ are constrained by a positive simplex, i.e., they are non-negative and sum up to a scalar s.

In real-world applications, data are often corrupted by outliers due to ad-hoc data collection techniques. Existing SC methods deal with outliers by explicitly modeling them as an additional variable, and updating this variable using the ADMM algorithm. For example, in the seminal work of SSC [15], [16] and its successors [32], [34], [38], [40], c_{ii} is set as 0 for $\boldsymbol{x}_i$, indicating that each data point cannot be represented by itself, thus avoiding trivial solution of identity matrix. However, this brings additional computational costs and prevent the whole algorithm from converging [16], [18].

Different from these existing methods [16], [32], [34], [38], [40], we do not consider the constraint of $c_{ii} = 0$, for three major reasons. First, the positive λ in the regularization term can naturally prevent trivial solution of identity matrix $\boldsymbol{C} = \boldsymbol{I}$. Second, $c_{ii} \neq 0$ has a clear physical meaning, allowing a sample $\boldsymbol{x}_i$ in the subspace to be *partially* represented by itself. This is particularly useful when $\boldsymbol{x}_i$ is corrupted by noise. Third, when the data are corrupted by outliers in the same subspace, $c_{ii} = 1$ can help each data point $\boldsymbol{x}_i$ *fully* represents itself. By removing the constraint of $c_{ii} = 0$, our proposed SRSC model is very robust to noise and outliers, as will be demonstrated in the experimental section (§IV).

We can also rewrite the SR-based model (8) for all N data points in the matrix form:

$$\begin{aligned} \min_{\boldsymbol{C}} &\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \\ &\text{s.t.} \quad \boldsymbol{C} \ge 0, \boldsymbol{1}^\top\boldsymbol{C} = s\boldsymbol{1}^\top, \end{aligned} \tag{9}$$

where $\boldsymbol{C} \in \mathbb{R}^{N\times N}$ is the coefficient matrix. Here, the constraint $\boldsymbol{C} \ge 0$ favors positive values for entries corresponding

Algorithm 1: Projection of vector $\boldsymbol{u}_{k+1}$ onto a simplex

Input: Data point $\boldsymbol{u}_{k+1} \in \mathbb{R}^N$, scalar s;
1. Sort $\boldsymbol{u}_{k+1}$ into $\boldsymbol{w}$: $w_1 \geq w_2 \geq ... \geq w_N$;
2. Find $\alpha = \max\{1 \leq j \leq N : w_j + \frac{1}{j}(s - \sum_{i=1}^{j} w_i) > 0\}$;
3. Define $\beta = \frac{1}{\alpha}(s - \sum_{i=1}^{\alpha} w_i)$;
Output: $\boldsymbol{z}_{k+1}$: $z_{k+1}^i = \max\{u_{k+1}^i + \beta, 0\}$, $i = 1, ..., N$.

to data points from the same subspace, while suppressing entries corresponding to data points from different subspaces, thus making the coefficient matrix $\boldsymbol{C}$ discriminative. The constraint $\mathbf{1}^\top \boldsymbol{C} = s\mathbf{1}^\top$ limits the sum of each coefficient vector $\boldsymbol{c}_i$ to be s, thus making the representation more discriminative since each entry should be non-negative.

B. Model Optimization

The proposed SRSC model (9) cannot be solved analytically. In this section, we solve it by employing variable splitting methods [59], [60]. Specifically, we introduce an auxiliary variable $\boldsymbol{Z}$ into the SRSC model (9), which can then be equivalently reformulated as a linear equality-constrained problem:

$$\begin{aligned} &\min_{\boldsymbol{C},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{Z}\|_F^2 \\ &\text{s.t.} \quad \boldsymbol{Z} \geq 0, \mathbf{1}^\top \boldsymbol{Z} = s\mathbf{1}^\top, \boldsymbol{Z} = \boldsymbol{C}, \end{aligned} \tag{10}$$

whose solution w.r.t. $\boldsymbol{C}$ coincides with the solution of (9). Since the objective function in Eqn. (10) is separable w.r.t. the variables $\boldsymbol{C}$ and $\boldsymbol{Z}$, it can be solved using the ADMM [52]. The corresponding augmented Lagrangian function is

$$\begin{aligned} &\mathcal{L}(\boldsymbol{C}, \boldsymbol{Z}, \boldsymbol{\Delta}, \rho) \\ =&\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{Z}\|_F^2 + \langle \boldsymbol{\Delta}, \boldsymbol{Z} - \boldsymbol{C}\rangle + \frac{\rho}{2}\|\boldsymbol{Z} - \boldsymbol{C}\|_F^2 \\ =&\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{Z}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{Z} - \boldsymbol{C} + \frac{1}{\rho}\boldsymbol{\Delta}\|_F^2 \\ =&\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \frac{2\lambda+\rho}{2}\|\boldsymbol{Z} - \frac{\rho}{2\lambda+\rho}(\boldsymbol{C} - \frac{1}{\rho}\boldsymbol{\Delta})\|_F^2 \\ &+ \frac{\lambda\rho}{2\lambda+\rho}\|\boldsymbol{C} - \frac{1}{\rho}\boldsymbol{\Delta}\|_F^2, \end{aligned} \tag{11}$$

where $\boldsymbol{\Delta}$ is the augmented Lagrangian multiplier and $\rho > 0$ is the penalty parameter. Denote by $(\boldsymbol{C}_k, \boldsymbol{Z}_k)$ and $\boldsymbol{\Delta}_k$ the optimization variables and the Lagrange multiplier at iteration k ($k = 0, 1, 2, ...$), respectively. We initialize the variables $\boldsymbol{C}_0$, $\boldsymbol{Z}_0$, and $\boldsymbol{\Delta}_0$ to be conformable zero matrices. By taking the derivatives of the Lagrangian function $\mathcal{L}$ w.r.t. $\boldsymbol{C}$ and $\boldsymbol{Z}$, and setting them to be zeros, we can alternatively update the variables as follows:

(1) **Updating $\boldsymbol{C}$ while fixing $\boldsymbol{Z}_k$ and $\boldsymbol{\Delta}_k$:**
$\boldsymbol{C}$ is updated by solving the following problem:

$$\boldsymbol{C}_{k+1} = \arg\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{C} - (\boldsymbol{Z}_k + \frac{1}{\rho}\boldsymbol{\Delta}_k)\|_F^2. \tag{12}$$

This is a standard least square regression problem and has a closed-from solution given by

$$\boldsymbol{C}_{k+1} = (\boldsymbol{X}^\top\boldsymbol{X} + \frac{\rho}{2}\boldsymbol{I})^{-1}(\boldsymbol{X}^\top\boldsymbol{X} + \frac{\rho}{2}\boldsymbol{Z}_k + \frac{1}{2}\boldsymbol{\Delta}_k). \tag{13}$$

Algorithm 2: Solve the SRSC model (10) via ADMM

Input: Data matrix $\boldsymbol{X}$, Tol > 0, $\rho > 0$, K;
Initialization: $\boldsymbol{C}_0 = \boldsymbol{Z}_0 = \boldsymbol{\Delta}_0 = \mathbf{0}$, T = False, $k = 0$;
While (T == False) **do**
1. Update $\boldsymbol{C}_{k+1}$ by Eqn. (13);
2. Update $\boldsymbol{Z}_{k+1}$ by Eqn. (14);
3. Update $\boldsymbol{\Delta}_{k+1}$ by Eqn. (15);
4. **if** (Convergence condition is satisfied) or ($k \geq K$)
5. T $\leftarrow$ True;
end if
end while
Output: Matrices $\boldsymbol{C}$ and $\boldsymbol{Z}$.

(2) **Updating $\boldsymbol{Z}$ while fixing $\boldsymbol{C}_k$ and $\boldsymbol{\Delta}_k$:**
$\boldsymbol{Z}$ is updated by solving the following problem:

$$\begin{aligned} \boldsymbol{Z}_{k+1} = \arg\min_{\boldsymbol{Z}} &\|\boldsymbol{Z} - \frac{\rho}{2\lambda+\rho}(\boldsymbol{C}_{k+1} - \rho^{-1}\boldsymbol{\Delta}_k)\|_F^2 \\ \text{s.t.} \quad &\boldsymbol{Z} \geq 0, \mathbf{1}^\top\boldsymbol{Z} = s\mathbf{1}^\top. \end{aligned} \tag{14}$$

This is a quadratic programming problem and the objective function is strictly convex, with a close and convex constraint, so there is a unique solution. As such, problem (14) can be solved using, for example, active set methods [61], [62] or projection based methods [63]–[65]. Here, we employ a projection based method [64], whose computational complexity is $\mathcal{O}(N \log N)$ to project a vector of length N onto a simplex. Denoting by $\boldsymbol{u}_{k+1}$ an arbitrary column of $\frac{\rho}{2\lambda+\rho}(\boldsymbol{C}_{k+1} - \rho^{-1}\boldsymbol{\Delta}_k)$, the solution of $\boldsymbol{z}_{k+1}$ (the corresponding column in $\boldsymbol{Z}_{k+1}$) can be solved by projecting $\boldsymbol{u}_{k+1}$ onto a simplex [64]. The solution of problem (14) is summarized in Algorithm 1.

(3) **Updating $\boldsymbol{\Delta}$ while fixing $\boldsymbol{C}_k$ and $\boldsymbol{Z}_k$:**

$$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho(\boldsymbol{Z}_{k+1} - \boldsymbol{C}_{k+1}). \tag{15}$$

We repeat the above alternative updates until a certain convergence condition is satisfied or the number of iterations reaches a preset threshold K. Under the convergence condition of the ADMM algorithm, $\|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F \leq$ Tol, $\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F \leq$ Tol, and $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F \leq$ Tol must be simultaneously satisfied, where Tol > 0 is a small tolerance value. Since the objective function and constraints are convex, problem (10) solved by the ADMM algorithm, is guaranteed to converge to a global optimal solution. We summarize the updating procedures in Algorithm 2.

C. Theoretical Analysis

Our optimization problem (9) is a convex optimization problem, which means $f(\boldsymbol{C}) := \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2$ is a convex function and the set $\mathcal{S} := \{\boldsymbol{C}|\boldsymbol{C} \geq 0, \mathbf{1}^\top\boldsymbol{C} = s\mathbf{1}^\top\}$ is a convex set. Then, the function f has a minimum on the hyper-plane $\overline{\mathcal{S}} := \{\boldsymbol{C}|\mathbf{1}^\top\boldsymbol{C} = s\mathbf{1}^\top\}$, which is the linear span of $\mathcal{S}$. Below, we discuss the solution space of f over $\mathcal{S}$.

Theorem 1. *Suppose $\boldsymbol{C}^*$ is the minimum of the convex function f over the convex set $\mathcal{S}$. If $\boldsymbol{C}^*$ is not the minimum of f on $\overline{\mathcal{S}}$, then $\boldsymbol{C}^*$ is on the boundary of $\mathcal{S}$: $\partial\mathcal{S}$.*

Proof. We first make an assumption that $\boldsymbol{C}^* \notin \partial\mathcal{S}$, then we will derive a contradiction. If our assumption holds, there

exists a high-dimensional open sphere $B_r(\boldsymbol{C}^*) = \{\boldsymbol{C} | \|\boldsymbol{C} - \boldsymbol{C}^*\| < r\} \subset \mathbb{R}^{N\times N}$ centered at $\boldsymbol{C}^* \cap \mathcal{S}$ with radius $r > 0$, such that for all $\boldsymbol{C} \in B_r$, $f(\boldsymbol{C}) \geq f(\boldsymbol{C}^*)$ holds.

We consider $\forall \boldsymbol{D} \in \overline{\mathcal{S}}$, $\exists \lambda \in (0, 1)$ such that $\boldsymbol{C}^* + \lambda(\boldsymbol{D} - \boldsymbol{C}^*) \in B_r(\boldsymbol{C}^*)$. In fact, we can set $\lambda < \min(\frac{r}{\|\boldsymbol{D}-\boldsymbol{C}^*\|}, 1)$. Then we have $\|\boldsymbol{C}^* + \lambda(\boldsymbol{D} - \boldsymbol{C}^*) - \boldsymbol{C}^*\| = \|\lambda(\boldsymbol{D} - \boldsymbol{C}^*)\| < r$, which means $\boldsymbol{C}^* + \lambda(\boldsymbol{D} - \boldsymbol{C}^*) \in B_r(\boldsymbol{C}^*)$. In this way, we have

$$f(\boldsymbol{C}^* + \lambda(\boldsymbol{D} - \boldsymbol{C}^*)) \geq f(\boldsymbol{C}^*). \tag{16}$$

However, due to the convexity of f, we have the following Jensen's inequality:

$$\begin{aligned} f(\boldsymbol{C}^* + \lambda(\boldsymbol{D} - \boldsymbol{C}^*)) &= f((1-\lambda)\boldsymbol{C}^* + \lambda\boldsymbol{D}) \\ &\leq (1-\lambda)f(\boldsymbol{C}^*) + \lambda f(\boldsymbol{D}). \end{aligned} \tag{17}$$

Combining this with Eqn. (16), we obtain $\lambda f(\boldsymbol{D}) \geq \lambda f(\boldsymbol{C}^*)$ and hence $f(\boldsymbol{D}) \geq f(\boldsymbol{C}^*)$, $\forall \boldsymbol{D} \in \mathbb{R}^{N\times N}$. Then, $\boldsymbol{C}^*$ is the minimum of f on hyper-plane $\overline{\mathcal{S}}$, resulting a contradiction. □

D. Discussion

The constraint "$\mathbf{1}^\top \boldsymbol{c} = 1$" has already been used in Sparse Subspace Clustering (SSC) [15], [16] to deal with the presence of affine, rather than linear, subspaces. However, limiting the sum of the coefficient vector $\boldsymbol{c}$ to be 1 is not flexible enough for real-world clustering problems. What's more, suppressing the sum of the entries in the coefficient vector $\boldsymbol{c}$ can make it more discriminative, since these entries should be non-negative and sum up to a scalar s. Considering the extreme case where s is nearly zero, each data point must be represented by its most similar data points in the homogeneous subspace. To this end, in our proposed SRSC model, we extend the affine constraint of "summing up to 1" to a scaled affine constraint of "summing up to a scalar s". According to our experiments (please refer to §IV), this extension can indeed improve the performance of SC.

E. Subspace Clustering via Simplex Representation

Subspace Clustering Algorithm. Denote by $\mathcal{X} = \{\boldsymbol{x}_i \in \mathbb{R}^D\}_{i=1}^N$ a set of data points drawn from a union of n subspaces $\{\mathcal{S}_j\}_{j=1}^n$. Most existing spectral clustering based SC algorithms [16], [18], [29], [33], [38], [45] first compute the coefficient matrix $\boldsymbol{C}$, and then construct a non-negative affinity matrix $\boldsymbol{A}$ from $\boldsymbol{C}$ by exponentiation [13], absolute symmetrization [15], [16], [29], [32]–[35], [37], [38], [40], [45], or squaring operations [17], [18], [41], etc. In contrast, in our proposed SRSC model, the coefficient matrix is guaranteed to be non-negative by the introduction of a simplex constraint. Hence, we can remove the absolute operation and construct the affinity matrix by

$$\boldsymbol{A} = \frac{\boldsymbol{C} + \boldsymbol{C}^\top}{2}. \tag{18}$$

As a common post-processing step in [15]–[18], [29], [32]–[35], [37], [38], [40], [41], [45], we apply the spectral clustering technique [66] to the affinity matrix $\boldsymbol{A}$, and obtain the final segmentation of data points. Specifically, we employ the widely used Normalized Cut algorithm [58] to segment the affinity matrix. The proposed SRSC based subspace clustering algorithm is summarized in Algorithm 3.

Algorithm 3: Subspace Clustering by SRSC

Input: A set of data points $\mathcal{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ lying in a union of n subspaces $\{\mathcal{S}_j\}_{j=1}^n$;

1. Obtain the coefficient matrix $\boldsymbol{C}$ by solving the SRSC model:
$$\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \text{ s.t. } \boldsymbol{C} \geq 0, \mathbf{1}^\top\boldsymbol{C} = s\mathbf{1}^\top ;$$
2. Construct the affinity matrix by
$$\boldsymbol{A} = \tfrac{\boldsymbol{C}+\boldsymbol{C}^\top}{2} ;$$
3. Apply spectral clustering [66] to the affinity matrix;

Output: Segmentation of data: $\boldsymbol{X}_1, ..., \boldsymbol{X}_n$.

Complexity Analysis. Assume that there are N data points in the data matrix $\boldsymbol{X}$. In Algorithm 2, the costs for updating $\boldsymbol{C}$ and $\boldsymbol{Z}$ are $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2 \log N)$, respectively. The costs for updating $\boldsymbol{\Delta}$ and ρ are negligible compared to the updating costs of $\boldsymbol{C}$ and $\boldsymbol{Z}$. As such, the overall complexity of Algorithm 2 is $\mathcal{O}(N^3K)$, where K is the number of iterations. The costs for affinity matrix construction and spectral clustering in Algorithm 3 can be ignored. Hence, the overall cost of the proposed SRSC is $\mathcal{O}(N^3K)$ for data matrix $\boldsymbol{X} \in \mathbb{R}^{D\times N}$.

IV. Experiments

In this section, we evaluate the proposed SRSC method for the subspace clustering (SC) problem. The evaluations are presented using two protocols. First, we validate the effectiveness of the non-negativity and scaled affinity in the simplex constraint. Then, we compare the proposed SRSC with state-of-the-art SC methods. Both the validation and comparison are performed on three real-world SC problems: motion segmentation for video analysis, human faces clustering, and hand-written digits/letters clustering.

A. Implementation Details

The proposed SRSC model (9) is solved under the ADMM [52] framework. There are four parameters to be determined in the ADMM algorithm: the regularization parameter λ, the sum s of the entries in the coefficient vector, the penalty parameter ρ and the iteration number K. In all the experiments, we fix $s = 0.5$, $K = 5$ and $\rho = 0.5$. As in most competing methods [13]–[18], [29], [32]–[35], [37]–[41], [45], the parameter λ is tuned on each dataset to achieve the best performance of SRSC on that dataset. The influence of λ on each dataset will be introduced in §IV-C. All experiments are run under the Matlab2014b environment on a machine with a CPU of 3.50GHz and a 12GB RAM.

B. Datasets

We evaluate the proposed SRSC method on the Hopkins-155 dataset [53] for motion segmentation, the Extended Yale B [54] and ORL [67] datasets for human face clustering, and the MNIST [55] and EMNIST [68] for hand-written digits/letters clustering.

Hopkins-155 dataset [53] contains 155 video sequences, 120 of which contain two moving objects and 35 of which contain three moving objects, corresponding to 2 or 3 low-dimensional subspaces of the ambient space. On average, each

Fig. 1: Exemplar motion segmentations of one frame from different sequences in the Hopkins-155 dataset [53].

two-motion sequence has 30 frames and each frame contains $N = 266$ data points, while each three-motion sequence has 29 frames and each frame contains $N = 393$ data points. Similar to the experimental settings in previous methods, such as SSC [16], on this dataset we employ principal component analysis (PCA) [69] to project the original trajectories of different objects into a 12-dimensional subspace, in which we evaluate the comparison methods. This dataset [53] has been widely used as a benchmark to evaluate SC methods for motion segmentation. Figure 5 presents some segmentation examples from the Hopkins-155 dataset [53], where different colors indicate different moving objects.

Extended Yale B dataset [54] contains face images of 38 human subjects, and each subject has 64 near-frontal images (gray-scale) taken under different illumination conditions. The original images are of size 192×168 pixels and we resize them to 48×42 pixels in our experiments. For dimension reduction purposes, the resized images are further projected onto a $6n$-dimensional subspace using PCA, where n is the number of subjects (or subspaces) selected in our experiments. Following the experimental settings in SSC [16], we divide the 38 subjects into 4 groups, consisting of subjects 1 to 10, subjects 11 to 20, subjects 21 to 30, and subjects 31 to 38. For each of the first three groups we select $n \in \{2, 3, 5, 8, 10\}$ subjects, while for the last group we choose $n \in \{2, 3, 5, 8\}$. Finally, we apply SC algorithms for each set of n subjects. Figure 2 (top) shows some face images from Extended Yale B, captured under different lighting conditions.

ORL dataset [67] contains overall 400 human face images of 40 subjects, each having 10 samples. Similar to [42], we resize the original face images from 112×92 to 32×32. For each human subject, the face images were taken under varying lighting conditions, with different facial expressions (e.g., open eyes or closed eyes, smiling or not smiling, etc.), as well as different facial details (e.g., w/ glasses or w/o glasses). The **ORL** dataset is more difficult to tackle than Extended Yale B [54] for two reasons: 1) its varying face images varies much more complex; 2) it only contains 400 images, much smaller than Extended Yale B (2432 images). Figure 2 (down) shows some face images from ORL.

Fig. 2: Face images from the Extended Yale B dataset [54] (top) and ORL dataset [67] (down).

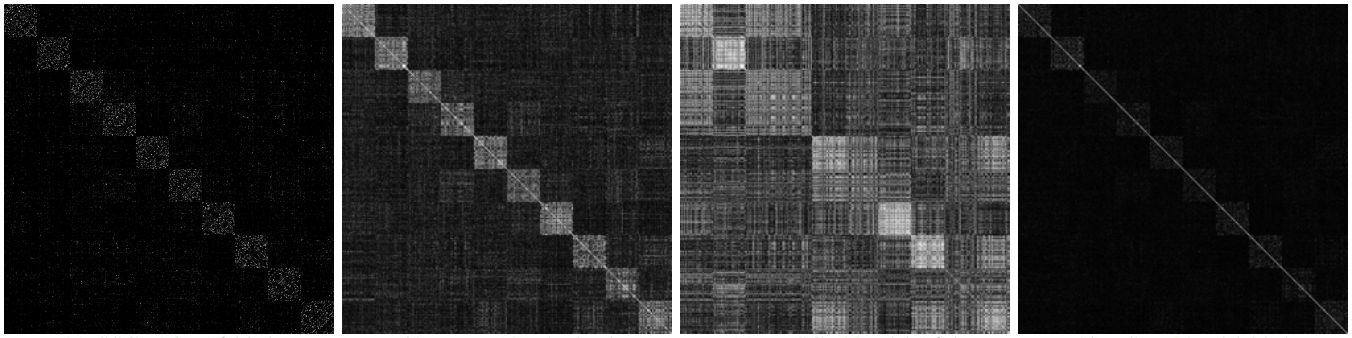
Fig. 3: Digit images from the MNIST dataset [55].

MNIST dataset [55] contains 60,000 gray-scale images of 10 digits (i.e., $\{0, ..., 9\}$) in the training set and $10,000$ images in the testing set. The images are of size 28×28 pixels. In our experiments, we randomly select $N_i \in \{50, 100, 200, 400, 600\}$ images for each of the 10 digits. Following [45], for each image, a set of feature vectors is computed using a scattering convolution network (SCN) [70]. The final feature vector is a concatenation of the coefficients in each layer of the network, and is translation invariant and deformation stable. Each feature vector is of $3,472$-dimension. The feature vectors for all images are then projected onto a 500-dimensional subspace using PCA. Figure 3 shows some examples of the handwritten digit images in this dataset.

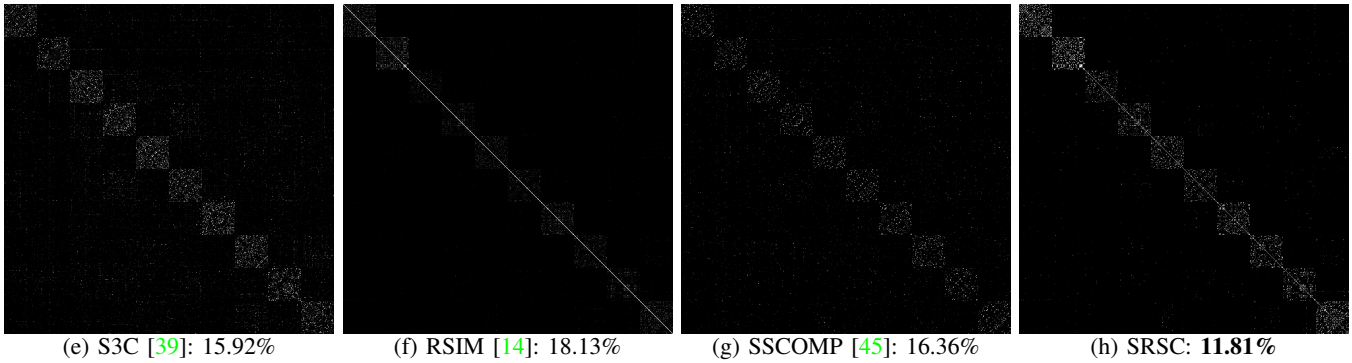
EMNIST dataset [68] is an extension of the MNIST dataset that contains gray-scale handwritten digits and letters. This dataset contains overall $190,998$ images corresponding to 26 lower case letters. We use them as the data for a 26-class clustering problem. The images are of size 28×28. In our experiments, we randomly select $N_i = 500$ images for each of the 26 digits/letters. Following [48], for each image, a set of feature vectors is computed using a scattering convolution network (SCN) [70], which is translational invariant and deformation stable. The feature vectors are of $3,472$-dimension, the feature vectors for all images are projected onto a 500-dimensional subspace using PCA.

C. Comparison with State-of-the-art Methods

Comparison Methods. We compare the proposed SRSC with several state-of-the-art SC methods, including SSC [15], [16], LRR [17], [18], LRSC [29], LSR [32], SMR [33], S3C [38], [39], RSIM [14], SSCOMP [45], EnSC [46], DSC [42], and ESC [48]. For SRSC, we fix $s = 0.5$, or as reported in ablation study, i.e., $s = 0.9$ on Hopkins-155, $s = 0.25$ on Extended YaleB, $s = 0.4$ on ORL, and $s = 0.15$ on MNIST. For the SMR method, we use the J_1 affinity matrix (i.e., (7) in §II as described in [33], for fair comparison. For the other methods, we tune their corresponding parameters on each of the three datasets, i.e., the Hopkins-155 dataset [53] for motion segmentation, the Extended Yale B dataset [54] for



(a) SSC [16]: 16.99% (b) LRR [18]: 17.97% (c) LRSC [29]: 24.16% (d) LSR [32]: 24.98%



(e) S3C [39]: 15.92% (f) RSIM [14]: 18.13% (g) SSCOMP [45]: 16.36% (h) SRSC: **11.81%**

Fig. 4: Affinity matrices and corresponding average clustering errors by different methods on the handwritten digit images from MNIST [55]. 50 images for each digit of $\{0, 1, ..., 9\}$ are used to compute the affinity matrix by different methods. All images are normalized to $[0, 1]$ by dividing by the maximal entries in the corresponding affinity matrices.

face clustering, and the MNIST dataset [55] for handwritten digit clustering, to achieve their best clustering results.

Comparison on Affinity Matrix. The affinity matrix plays a key role in the success of SC methods. Here, we visualize the affinity matrix of the proposed SRSC and the comparison methods, on the SC problem. We run the proposed SRSC algorithm and the competing methods [14], [16], [18], [29], [32], [38], [39], [45], [46] on the MNIST dataset [55]. The training set contains $6,000$ images for each digit in $\{0, 1, ..., 9\}$. We randomly select 50 images for each digit, and use the 500 total images to construct the affinity matrices using these competing SC methods. The results are visualized in Figure 4. As can be seen, the affinity matrix of SRSC shows better connections within each subspace, and generates less noise than most of the other methods. Though LSR [32] and RSIM [14] have less noise than our proposed SRSC, their affinity matrices suffer from strong diagonal entries, indicating that, for these two methods, the data points are mainly reconstructed by themselves. As will be shown as follows, with the simplex constraint, the proposed SRSC algorithm can better exploit the inherent correlations among the data points and achieve better clustering performance than the other competing SC methods.

Results on Motion Segmentation. We first study how the parameter s influences the average clustering errors of the proposed SRSC algorithm. The clustering errors with respect to the value of s are plotted in Figure 5, As can be seen, the proposed SRSC obtains an average clustering error of 1.53% when $s = 0.5$. Note that SRSC achieves its lowest average clustering error of 1.04% when $s = 0.9$. The parameter λ is set as $\lambda = 0.001$. We then compare the proposed SRSC with the other competing SC algorithms [14], [16], [18], [29], [32], [33], [38], [39], [45], [46]. The results on the average clustering errors are listed in Table I, from which we can see that the proposed SRSC achieves the lowest clustering error. Besides, the speed of the proposed SRSC approach is only slightly slower than LRSC, LSR, SSCOMP and EnSC, and much faster than the other competing methods. Note that the fast speed of SRSC owes to both efficient solution in each iteration and fewer iterations in the ADMM algorithm.

Results on Human Face Clustering. Here, we compare the proposed SRSC algorithm with the competing methods, on the commonly used Extended Yale B dataset [54] and ORL dataset [67] for human face clustering. On ORL [67], we also compared with the Deep Subspace Clustering (DSC) method, which achieves state-of-the-art performance on this dataset.

We study how the scalar s influences the clustering errors (%) of the proposed SRSC algorithm, and take the Extended Yale B dataset [54] for an example. The average clustering errors with respect to the value of s are plotted in Figure 6. As can be seen, SRSC achieves an average clustering error of 3.26% when $s = 0.5$, and achieves lowest average clustering error of 2.16% when $s = 0.25$. We set $\lambda = 0.005$.

The comparison results of different algorithms are listed in Table II and Table III. From Table II, we observe that, for different numbers ($\{2, 3, 5, 8, 10\}$) of clustering subjects, the average clustering errors of SRSC are always significantly lower than the other competing methods. For example, when clustering 10 subjects, the average error of SRSC (when fixing

Method	SSC [16]	LRR [18]	LRSC [29]	LSR [32]	SMR [33]	S3C [39]	RSIM [14]	SSCOMP [45]	EnSC [46]	SRSC
Error (%)	2.18	3.28	5.41	2.33	2.27	2.61	1.76	5.35	1.81	**1.53**
Time (s)	0.49	0.28	**0.04**	0.05	0.35	2.86	0.18	0.06	0.05	0.07

TABLE I: Average clustering errors (%) and speed (in seconds) of different algorithms on the Hopkins-155 dataset [53] with the 12-dimensional data points obtained using PCA. SRSC can achieve clustering error of 1.04% when $s = 0.9$.

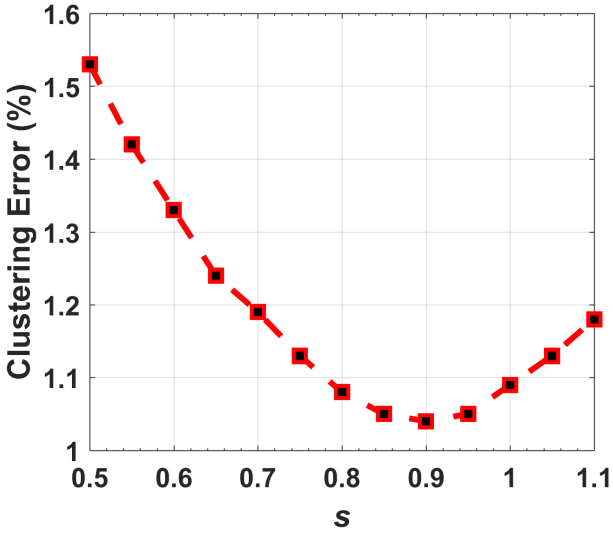


Fig. 5: Average clustering errors (%) of the proposed SRSC algorithm with different scalar s, on the Hopkins-155 dataset [53].

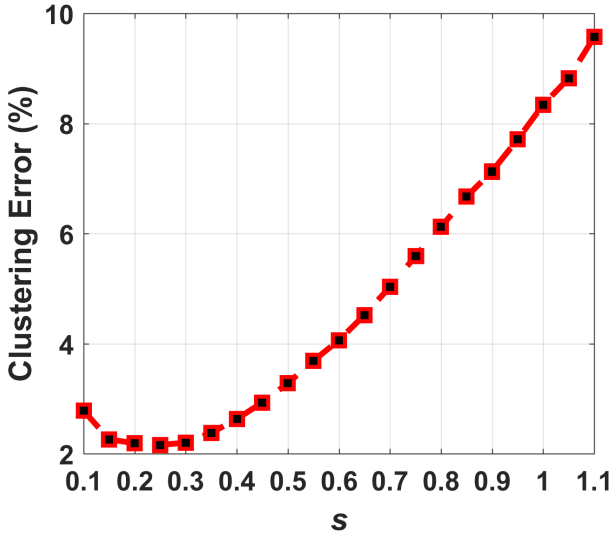


Fig. 6: Average clustering errors (%) of the proposed SRSC algorithm with different scalar s, on the Extended Yale B dataset [54].

$s = 0.5$) is 5.10%, while the errors of the other methods are 10.94% for SSC, 28.54% for LRR, 30.05% for LRSC, 28.59% for LSR, 28.18% for SMR, 5.16% for S3C, 6.56% for RSIM, 14.80% for SSCOMP, and 5.96% for EnSC, respectively. The performance gain of SRSC (5.10%) over its baseline LSR (28.59%) demonstrates the power of simplex representation on human face clustering. In terms of running time, the proposed SRSC algorithm is shown to be comparable to SSCOMP, which is the most efficient algorithm among all competing methods. In Table III, we can observe that, the proposed SRSC method achieves lower clustering error than previous methods except the deep learning based DSC. This shows that simplex representation performs better than previous sparse or low rank representation frameworks, while learning based method performs better with end-to-end training.

Results on Handwritten Digit Clustering. For the digit clustering problem, we evaluated the proposed SRSC with the competing methods on the MNIST dataset [55] and the EMNIST dataset [68]. We follow the experimental settings in SSCOMP [45]. On the MNIST [55], the testing data consists of a randomly chosen number of $N_i \in \{50, 100, 200, 400, 600\}$ images for each of the 10 digits, while on the EMNIST [68], the testing data consists of $N_i = 500$ images for each of the 26 digits or letters. The feature vectors are extracted from the scattering convolution network [70]. The features extracted from the digit/letter images are originally $3,472$-dimensional, and are projected onto a 500-dimensional subspace using PCA.

We evaluate the influence of the scalar s on the average clustering error (%) of SRSC on the MNIST dataset. The curve of clustering errors w.r.t. s is plotted in Figure 7. Average clustering errors are computed over 20 trials sampled from

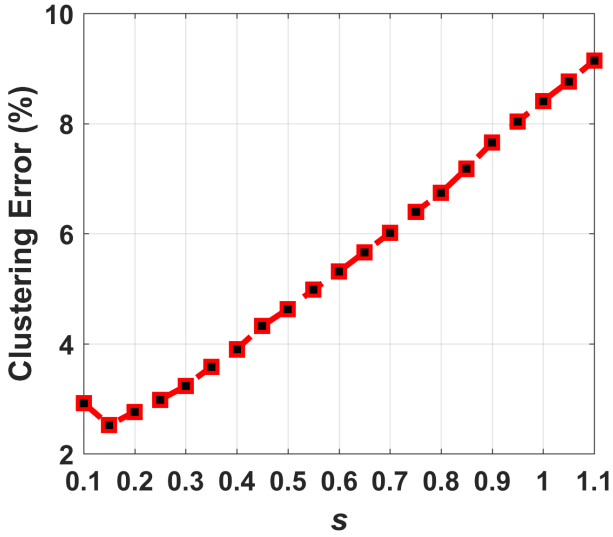


Fig. 7: Average clustering errors (%) of the proposed SRSC algorithm with different scalar s, on the MNIST dataset [55], with 600 images for each digit.

$6,000$ randomly selected images ($N_i = 600$). As can be seen, the proposed SRSC algorithm achieves an average clustering error of 4.53%, and obtains its lowest average clustering error (2.52%) when $s = 0.15$, with 600 images for each digit from the MNIST dataset. Similar results can be observed on the EMNIST dataset [68]. The parameter λ is set as $\lambda = 0.01$.

We list the average clustering errors (%) of the comparison methods in Tables IV and V. It can be seen that the proposed SRSC algorithm performs better than all other competing methods. For example, on MNIST dataset, SRSC achieves

# of Subjects	2 Subjects		3 Subjects		5 Subjects		8 Subjects		10 Subjects	
	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)
SSC [16]	1.86	0.17	3.10	0.31	4.31	0.74	5.85	2.72	10.94	4.27
LRR [18]	4.99	0.15	6.86	0.29	14.04	0.65	24.18	1.30	28.54	1.81
LRSC [29]	4.42	**0.03**	6.14	0.06	13.06	0.13	24.12	0.28	30.05	0.45
LSR [32]	4.98	**0.03**	6.87	0.05	14.14	0.12	24.39	0.25	28.59	0.36
SMR [33]	4.68	0.21	6.56	0.30	13.78	0.76	24.25	2.00	28.18	2.59
S3C [39]	1.27	1.31	2.71	2.58	3.41	3.52	4.13	7.14	5.16	13.98
RSIM [14]	2.17	0.09	2.96	0.17	4.13	0.47	5.82	1.77	6.56	3.19
SSCOMP [45]	2.76	**0.03**	5.35	**0.04**	7.89	**0.08**	11.40	**0.18**	14.80	**0.27**
EnSC [46]	1.64	**0.03**	2.91	0.05	3.84	0.11	5.84	0.20	5.96	0.29
SRSC	1.26	**0.03**	1.58	**0.04**	2.70	**0.08**	5.66	**0.18**	5.10	0.28
SRSC ($s = 0.25$)	**0.96**	**0.03**	**1.32**	**0.04**	**2.22**	**0.08**	**3.00**	**0.18**	**3.18**	0.28

TABLE II: Average clustering errors (%) and speed (in seconds) of different algorithms on the Extended Yale B dataset [54] with the $6n$-dimensional (n is the number of subjects) data points obtained using PCA.

Method	**SSC** [16]	**LRR** [18]	**LRSC** [29]	**LSR** [32]	**SMR** [33]	**RSIM** [14]	**SSCOMP** [45]	**EnSC** [46]	**DSC** [42]	**SRSC**
Error (%)	32.50	38.25	32.50	27.25	25.75	26.25	36.00	23.25	**14.00**	21.75
Time (s)	12.32	6.01	1.43	1.21	7.97	9.81	0.77	0.82	**0.23**	0.93

TABLE III: Average clustering errors (%) and speed (in seconds) of different algorithms on the ORL dataset [67] with the 32×32 data points obtained by resizing. SRSC can achieve clustering error of 21.25% when $s = 0.4$.

# of Images	500		1,000		2,000		4,000		6,000	
	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)
SSC [16]	16.99	27.36	15.95	49.25	14.42	94.84	14.00	239.95	14.40	423.47
LRR [18]	17.97	15.64	16.88	28.14	16.63	54.19	15.42	137.11	15.45	241.98
LRSC [29]	24.16	0.34	21.58	0.96	21.91	4.74	20.94	27.40	20.09	78.21
LSR [32]	24.98	**0.29**	20.24	0.91	20.56	4.74	22.24	28.86	20.12	83.14
SMR [33]	18.45	19.56	13.97	35.18	9.46	67.75	9.14	171.39	7.36	302.48
S3C [39]	15.92	135.20	12.23	263.33	10.54	521.93	9.46	1308.50	9.34	2544.41
RSIM [14]	18.13	10.06	15.70	18.09	11.48	34.84	10.53	88.14	10.21	155.56
SSCOMP [45]	16.36	0.32	13.33	**0.88**	9.40	**4.50**	8.78	**26.94**	8.75	**76.84**
EnSC [46]	13.45	0.37	9.31	0.98	7.69	4.67	6.71	27.14	5.86	77.24
SRSC	11.81	0.35	6.90	1.02	5.65	5.23	5.31	31.42	4.53	89.62
SRSC ($s = 0.15$)	**10.29**	0.37	**5.40**	1.03	**4.36**	5.25	**3.09**	31.43	**2.52**	89.65

TABLE IV: Average clustering errors (%) and speed (in seconds) of different algorithms on the MNIST dataset [55]. The features are extracted from a scattering network and projected onto a 500-dimensional subspace using PCA. The experiments are repeated for 20 times and the average results are reported.

Method	**SSC** [16]	**LRR** [18]	**LRSC** [29]	**LSR** [32]	**SMR** [33]	**RSIM** [14]	**SSCOMP** [45]	**EnSC** [46]	**ESC** [48]	**SRSC**
Error (%)	30.11	33.89	31.51	32.21	29.58	30.15	35.66	26.23	25.42	**23.45**
Time (s)	863.29	495.13	162.32	172.28	621.89	320.14	159.22	160.98	**157.67**	193.56

TABLE V: Average clustering errors (%) and speed (in seconds) of different algorithms on the EMNIST dataset [68]. The features are extracted from a scattering network and projected onto a 500-dimensional subspace using PCA. The experiments are repeated for 10 times and the average results are reported. SRSC can achieve clustering error of 22.76% when $s = 0.35$.

clustering errors of 11.81%, 6.90%, 5.65%, 5.31%, and 4.53% when the number of digit images are 500, 1000, 2000, 4000, and 6000, respectively. On EMNIST dataset, SRSC achieves clustering error of 23.45%, when the number of images for each digit/letter are 500. The results of the proposed SRSC on the two datasets are much better than the other competing algorithms, including the recently proposed EnSC and ESC, respectively. This validates the advantages of the proposed simplex representation over previous sparse or low-rank representation, for hand-written digit/letter clustering.

D. Ablation Study: Effectiveness of Simplex Representation

We perform comprehensive ablation studies on the Hopkins-155 [53], Extended Yale B [54], and MNIST [55] datasets to validate the effectiveness of the proposed SRSC model. SRSC includes two constraints, i.e., $\boldsymbol{c} \geq 0$ and $\mathbf{1}^\top \boldsymbol{c} = s$. To analyze the effectiveness of each constraint, we compare the clustering errors of the proposed SRSC model with several baseline methods.

The first baseline is the trivial Least Square Regression (LSR) model as follows:

$$\textbf{LSR:} \ \min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda \|\boldsymbol{C}\|_F^2. \tag{19}$$

By removing each individual constraint in the simplex constraint of the proposed SRSC model (9), we have the second baseline method, called the Non-negative Least Square Regression (NLSR) model:

$$\textbf{NLSR:} \ \min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda \|\boldsymbol{C}\|_F^2 \quad \text{s.t.} \quad \boldsymbol{C} \geq 0. \tag{20}$$

The NLSR model can be formulated by either removing the scaled affinity constraint $\mathbf{1}^\top \boldsymbol{c} = s$ from SRSC or adding a non-negative constraint to the LSR model (19). For structural clearance of this manuscript, we put the solution of the NLSR model (20) in Appendix A. We can also remove the non-negative constraint $\boldsymbol{C} \geq 0$ in the scaled simplex set

Method	LSR	NLSR	SLSR	SRSC
Error (%)	3.67	1.75	3.16	**1.04**

TABLE VI: Average clustering errors (%) of LSR, NLSR, SLSR ($s = 0.9$), and SRSC ($s = 0.9$) on Hopkins-155 dataset [53], with the 12-dimensional data points obtained using PCA.

n	LSR	NLSR	SLSR	SRSC
2	4.98	3.46	4.35	**0.96**
3	6.87	4.95	6.56	**1.32**
5	14.14	10.03	13.77	**2.22**
8	24.39	17.74	22.14	**3.00**
10	28.59	20.29	26.31	**3.18**

TABLE VII: Average clustering errors (%) of LSR, NLSR, SLSR (s=0.22), and SRSC ($s = 0.25$) on the Extended Yale B dataset [54], with the $6n$-dimensional (n is the number of subjects) data points obtained using PCA.

$\{\boldsymbol{C} \in \mathbb{R}^{N\times N} | \boldsymbol{C} \geq 0, \mathbf{1}^\top \boldsymbol{C} = s\mathbf{1}^\top\}$, and obtain a Scaled-affine Least Square Regression (SLSR) model:

$$\textbf{SLSR:} \min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \quad \text{s.t.} \quad \mathbf{1}^\top\boldsymbol{C} = s\mathbf{1}^\top. \tag{21}$$

The solution of the SLSR model (21) is given in Appendix B.

The proposed SRSC method, as well as the three baseline methods, can be expressed in a standard form:

$$\min_{\boldsymbol{C}} \textit{Data Term} + \textit{Regularization Term} \text{ s.t. } \textit{Constraints}. \tag{22}$$

For the NLSR, SLSR, and SRSC methods, the data and regularization terms are the same as that of the basic LSR model (9), i.e., $\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2$ and $\lambda\|\boldsymbol{C}\|_F^2$, respectively. The difference among these comparison methods lies in the *Constraints*. In Table IX, we summarize the proposed SRSC, and the three baseline methods LSR, NLSR, and SLSR.

In Tables VI-VIII, we list the average clustering errors (%) of the proposed SRSC and the three baseline methods, on the Hopkins-155 [53], the Extended Yale B [54], and the MNIST [55] datasets. Note that here we tune the parameter s to achieve the best performance of SRSC on different datasets.

Effectiveness of Non-negative Constraint. The effectiveness of non-negative constraint $\boldsymbol{C} \geq 0$ in the proposed SRSC model can be validated from two complementary aspects. First, it can be validated by evaluating the performance improvement of the baseline methods NLSR (20) over LSR [32]. Since the only difference between them is the additional non-negative constraint in NLSR (20), the performance gain of NSLR over LSR can directly reflect the effectiveness of the non-negative constraint. Second, the effectiveness of non-negative constraint can also be validated by comparing the performance of the proposed SRSC (9) and the baseline method SLSR (21). Since the only difference between the two methods is that SLSR (21) lacks a non-negative constraint, the performance gain of SRSC over SLSR should be due to the introduced non-negativity.

The results listed in Table VI, show that, on the Hopkins-155 dataset, the baseline methods LSR and NLSR achieve average clustering errors of 3.67% and 1.75%, respectively. This demonstrates that, by adding the non-negative constraint to LSR, the resulting baseline method NLSR has a significantly reduced clustering error. Besides, the average clustering errors of SRSC ($s = 0.9$) and SLSR ($s = 0.9$) are 1.04% and 3.16%, respectively. This shows that, if we remove the non-negative constraint from the SRSC model, the resulting SLSR model will have a significantly lower clustering performance. Similar trends can be found from Tables VII and VIII on the Extended Yale B dataset [54] and the MNIST dataset [55], respectively. These results validate the contribution of the non-negative constraint to the success of the proposed SRSC method.

# of Images	LSR	NLSR	SLSR	SRSC
500	24.98	15.29	22.00	**10.29**
1,000	20.24	12.66	18.59	**5.40**
2,000	20.56	10.13	16.87	**4.36**
4,000	22.24	9.07	15.48	**3.09**
6,000	20.12	8.34	14.23	**2.52**

TABLE VIII: Average clustering errors (%) of LSR, NLSR, SLSR ($s = 0.24$), and SRSC ($s = 0.15$) on the MNIST dataset [55]. The data point features are extracted from a scattering network and projected onto a 500-dimensional subspace obtained using PCA. The experiments are independently repeated 20 times.

Effectiveness of Scaled-Affine Constraint. The effectiveness of scaled-affine constraint can be validated from two aspects. First, it can be validated by comparing the baseline methods LSR (19) and SLSR (21), which are summarized in Table IX. Since the only difference between them is that SLSR contains an additional scaled affine constraint over LSR, the performance gain of SLSR over LSR can directly validate the scaled affine constraint's effectiveness. Second, the effectiveness can also be validated by comparing the proposed SRSC (9) and the baseline NLSR (20). This is because NLSR can be formulated by removing the scaled affine constraint from SRSC.

From the results listed in Tables VI, we observe that, on the Hopkins-155 dataset [53], the proposed SRSC ($s = 0.9$) can achieve lower clustering error (1.04%) than the baseline method NLSR (1.75%). Similar conclusions can be drawn from Tables VII and VIII for experiments on the Extended Yale B dataset [54] and the MNIST dataset [55], respectively. All these comparisons demonstrate that the scaled affine constraint is another essential factor for the success of SRSC.

Effectiveness of the Simplex Constraint. We find that the proposed simplex constraint, i.e., the integration of the non-negative and scaled affine constraints, can further boost the performance of SC. This can be validated by comparing the performance of the proposed SRSC (9) and the baseline method LSR (19). The results listed in Tables VI-VIII show that, on all three commonly used datasets, the proposed SRSC can achieve much lower clustering errors than the baseline method LSR. For example, on the MNIST dataset, the clustering errors of SRSC ($s = 0.15$) are 10.29%, 5.40%, 4.36%, 3.09%, and 2.52% when we randomly select $50, 100, 200, 400$, and 600 images for each of the 10 digits, respectively. Meanwhile, the corresponding clustering errors of the baseline method LSR are 24.98%, 20.24%, 20.56%, 22.24%, 20.12%, respectively. SRSC performs much better

Model	Data Term	Regularization Term	Constraint		Solution
			$\boldsymbol{C} \geq 0$	$\mathbf{1}^\top \boldsymbol{C} = s\mathbf{1}^\top$	
LSR (19)	$\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2$	$\|\boldsymbol{C}\|_F^2$	✗	✗	$\boldsymbol{C} = (\boldsymbol{X}^\top\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^\top\boldsymbol{X}$
NLSR (20)			✓	✗	See Appendix A
SLSR (21)			✗	✓	See Appendix B
SRSC (9)			✓	✓	See §III-B

TABLE IX: Summary of the proposed SRSC and three baseline methods LSR, NLSR, and SLSR.

than LSR in all cases. Similar trends can also be found in Table VI for the Hopkins-155 dataset and in Table VII for the Extended Yale B dataset.

V. Conclusion

In this paper, we proposed a simplex representation (SR) based model for spectral clustering based subspace clustering (SC). Specifically, we introduced the non-negative and scaled affine constraints into a simple least square regression model. The proposed SRSC model can reveal the inherent correlations among data points in a highly discriminative manner. Based on the SRSC model, a novel spectral clustering based algorithm was developed for subspace clustering. Extensive experiments on three benchmark clustering datasets demonstrated that the proposed SRSC algorithm is very efficient and achieves better clustering performance than state-of-the-art subspace clustering algorithms. The significant improvements of SRSC over the baseline models demonstrated the effectiveness of the proposed simplex representation.

Appendix A
Solution of the NLSR Model (20)

The NLSR model (20) does not have an analytical solution. We employ a variable splitting method [59], [60] to solve it. By introducing an auxiliary variable $\boldsymbol{Z}$, we can reformulate the NLSR model (20) into a linear equality-constraint problem with two variables $\boldsymbol{C}$ and $\boldsymbol{Z}$:

$$\min_{\boldsymbol{C},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \quad \text{s.t.} \quad \boldsymbol{Z} = \boldsymbol{C}, \boldsymbol{Z} \geq 0. \tag{23}$$

Since the objective function is separable w.r.t. the variables $\boldsymbol{C}$ and $\boldsymbol{Z}$, problem (23) can be solved under the alternating direction method of multipliers (ADMM) [52] framework. The Lagrangian function of the problem (23) is

$$\begin{aligned}\mathcal{L}(\boldsymbol{C}, \boldsymbol{Z}, \boldsymbol{\Delta}, \lambda, \rho) =& \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 \\ &+ \langle \boldsymbol{\Delta}, \boldsymbol{Z} - \boldsymbol{C}\rangle + \frac{\rho}{2}\|\boldsymbol{Z} - \boldsymbol{C}\|_F^2,\end{aligned} \tag{24}$$

where $\boldsymbol{\Delta}$ is the augmented Lagrangian multiplier and $\rho > 0$ is the penalty parameter. We initialize the vector variables $\boldsymbol{C}_0$, $\boldsymbol{Z}_0$, and $\boldsymbol{\Delta}_0$ to be conformable zero matrices and set $\rho > 0$ with a suitable value. Denote by $(\boldsymbol{C}_k, \boldsymbol{Z}_k)$ and $\boldsymbol{\delta}_k$ the optimization variables and the Lagrange multiplier at iteration k ($k = 0, 1, 2, ..., K$), respectively. The variables can be updated by taking derivatives of the Lagrangian function (24) w.r.t. the variables $\boldsymbol{C}$ and $\boldsymbol{Z}$ and setting them to be zero.

(1) **Updating $\boldsymbol{C}$ while fixing $\boldsymbol{Z}$ and $\boldsymbol{\Delta}$**:

$$\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{C}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{C} - (\boldsymbol{Z}_k + \rho^{-1}\boldsymbol{\Delta}_k)\|_F^2. \tag{25}$$

This is a standard least squares regression problem with closed form solution:

$$\boldsymbol{C}_{k+1} = (\boldsymbol{X}^\top\boldsymbol{X} + \frac{2\lambda + \rho}{2}\boldsymbol{I})^{-1}(\boldsymbol{X}^\top\boldsymbol{X} + \frac{\rho}{2}\boldsymbol{Z}_k + \frac{1}{2}\boldsymbol{\Delta}_k) \tag{26}$$

(2) **Updating $\boldsymbol{Z}$ while fixing $\boldsymbol{C}$ and $\boldsymbol{\Delta}$**:

$$\min_{\boldsymbol{Z}} \|\boldsymbol{Z} - (\boldsymbol{C}_{k+1} - \rho^{-1}\boldsymbol{\Delta}_k)\|_F^2 \quad \text{s.t.} \quad \boldsymbol{Z} \geq 0. \tag{27}$$

The solution of $\boldsymbol{Z}$ is

$$\boldsymbol{Z}_{k+1} = \max(0, \boldsymbol{C}_{k+1} - \rho^{-1}\boldsymbol{\Delta}_k), \tag{28}$$

where the "$\max(\cdot)$" operator outputs element-wisely the maximal value of the inputs.

(3) **Updating the Lagrangian multiplier $\boldsymbol{\Delta}$**:

$$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho(\boldsymbol{Z}_{k+1} - \boldsymbol{C}_{k+1}). \tag{29}$$

The above alternative updating steps are repeated until the convergence condition is satisfied or the number of iterations exceeds a preset threshold K. The convergence condition of the ADMM algorithm is: $\|\boldsymbol{Z}_{k+1} - \boldsymbol{C}_{k+1}\|_F \leq$ Tol, $\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F \leq$ Tol, and $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F \leq$ Tol are simultaneously satisfied, where Tol > 0 is a small tolerance value. Since the objective function and constraints are all strictly convex, problem (20) solved by the ADMM algorithm [52] is guaranteed to converge to a global optimal solution.

Appendix B
Solution of the SLSR Model (21)

We solve the SLSR model (21) by employing variable splitting methods [59], [60]. Specifically, we introduce an auxiliary variable $\boldsymbol{Z}$ into the SLSR model (21), which can then be equivalently reformulated as a linear equality-constrained problem:

$$\begin{aligned}&\min_{\boldsymbol{C},\boldsymbol{Z}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\|\boldsymbol{Z}\|_F^2 \\ &\text{s.t.} \quad \mathbf{1}^\top\boldsymbol{Z} = s\mathbf{1}^\top, \boldsymbol{Z} = \boldsymbol{C},\end{aligned} \tag{30}$$

whose solution for $\boldsymbol{C}$ coincides with the solution of (21). Since its objective function is separable w.r.t. the variables $\boldsymbol{C}$ and $\boldsymbol{Z}$, problem (30) can also be solved via the ADMM method [52]. The corresponding augmented Lagrangian function is the same as in Eqn. (11). Denote by $(\boldsymbol{C}_k, \boldsymbol{Z}_k)$ and $\boldsymbol{\Delta}_k$ the optimization variables and Lagrange multiplier at iteration k ($k = 0, 1, 2, ...$), respectively. We initialize the variables $\boldsymbol{C}_0$, $\boldsymbol{Z}_0$, and $\boldsymbol{\Delta}_0$ to be conformable zero matrices. By taking derivatives of the Lagrangian function $\mathcal{L}$ (11) w.r.t. $\boldsymbol{C}$ and $\boldsymbol{Z}$, and setting them to be zeros, we can alternatively update the variables as follows:

Algorithm 4: Projection of the vector $\boldsymbol{v}_{k+1}$ onto a scaled affine space

Input: Data point $\boldsymbol{v}_{k+1} \in \mathbb{R}^N$, scalar s.
1. Sort $\boldsymbol{v}_{k+1}$ into $\boldsymbol{w}$: $w_1 \geq w_2 \geq ... \geq w_N$;
2. Find $\alpha = \max\{1 \leq j \leq N : w_j + \frac{1}{j}(s - \sum_{i=1}^{j} w_i) > 0\}$;
3. Define $\beta = \frac{1}{\alpha}(s - \sum_{i=1}^{\alpha} w_i)$;
Output: $\boldsymbol{z}_{k+1}$: $z_{k+1}^i = v_{k+1}^i + \beta$, $i = 1, ..., N$.

(1) **Updating $\boldsymbol{C}$ while fixing $\boldsymbol{Z}_k$ and $\boldsymbol{\Delta}_k$**:

$$\boldsymbol{C}_{k+1} = \arg\min_{\boldsymbol{C}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{C} - (\boldsymbol{Z}_k + \frac{1}{\rho}\boldsymbol{\Delta}_k)\|_F^2. \tag{31}$$

This is a standard least square regression problem and has a closed-from solution given by

$$\boldsymbol{C}_{k+1} = (\boldsymbol{X}^\top\boldsymbol{X} + \frac{\rho}{2}\boldsymbol{I})^{-1}(\boldsymbol{X}^\top\boldsymbol{X} + \frac{\rho}{2}\boldsymbol{Z}_k + \frac{1}{2}\boldsymbol{\Delta}_k). \tag{32}$$

(2) **Updating $\boldsymbol{Z}$ while fixing $\boldsymbol{C}_k$ and $\boldsymbol{\Delta}_k$**:

$$\begin{aligned} \boldsymbol{Z}_{k+1} = \arg\min_{\boldsymbol{Z}} \|\boldsymbol{Z} - \frac{\rho}{2\lambda+\rho}(\boldsymbol{C}_{k+1} - \rho^{-1}\boldsymbol{\Delta}_k)\|_F^2 \\ \text{s.t.} \quad \mathbf{1}^\top\boldsymbol{Z} = s\mathbf{1}^\top. \end{aligned} \tag{33}$$

This is a quadratic programming problem and the objective function is strictly convex, with a close and convex constraint, so there is a unique solution. Here, we employ the projection based method [64], whose computational complexity is $\mathcal{O}(N \log N)$ to process a vector of length N. Denote by $\boldsymbol{v}_{k+1}$ an arbitrary column of $\frac{\rho}{2\lambda+\rho}(\boldsymbol{C}_{k+1} - \rho^{-1}\Delta_k)$, the solution of $\boldsymbol{z}_{k+1}$ (the corresponding column in $\boldsymbol{Z}_{k+1}$) can be solved by projecting $\boldsymbol{v}_{k+1}$ onto a scaled affine space [64]. The solution of problem (33) is summarized in Algorithm 4.

(3) **Updating $\boldsymbol{\Delta}$ while fixing $\boldsymbol{C}_k$ and $\boldsymbol{Z}_k$**:

$$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k + \rho(\boldsymbol{Z}_{k+1} - \boldsymbol{C}_{k+1}). \tag{34}$$

We repeat the above alternative updates until a certain convergence condition is satisfied or the number of iterations reaches a preset threshold K. The convergence condition of the ADMM algorithm is met when $\|\boldsymbol{C}_{k+1} - \boldsymbol{Z}_{k+1}\|_F \leq \text{Tol}$, $\|\boldsymbol{C}_{k+1} - \boldsymbol{C}_k\|_F \leq \text{Tol}$, and $\|\boldsymbol{Z}_{k+1} - \boldsymbol{Z}_k\|_F \leq \text{Tol}$ are simultaneously satisfied, where $\text{Tol} > 0$ is a small tolerance value. Since the objective function and constraints are convex, the problem (21), solved by the ADMM algorithm, is guaranteed to converge to a global optimal solution.

REFERENCES

[1] R. Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2011. 1

[2] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering appearances of objects under varying illumination conditions. In *CVPR*, pages 11–18, 2003. 1, 2

[3] M. C. Tsakiris and R. Vidal. Algebraic clustering of affine subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):482–489, 2018. 1, 2

[4] J. Costeira and T. Kanade. A multibody factorization method for independent moving objects. *International Journal of Computer Vision*, 29(3), 1998. 1, 2

[5] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1–15, 2005. 1, 2

[6] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999. 1, 2

[7] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the em algorithm. In *CVPR*, 2004. 1, 2

[8] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 1, 2

[9] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1832–1845, 2009. 1, 2

[10] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. *ECCV*, 2006. 1, 2

[11] T. Zhang, A. Szlam, Y. Wang, and G. Lerman. Hybrid linear modeling via local best-fit flats. *CVPR*, 2010. 1, 2

[12] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. *CVPR*, 2007. 1, 2

[13] G. Chen and G. Lerman. Spectral curvature clustering. *International Journal of Computer Vision*, 81(3):317–330, 2009. 1, 2, 3, 5

[14] P. Ji, M. Salzmann, and H. Li. Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data. In *ICCV*, pages 4687–4695, 2015. 1, 2, 5, 6, 7, 8, 9

[15] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, pages 2790–2797. IEEE, 2009. 1, 2, 3, 5, 6

[16] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013. 1, 2, 3, 5, 6, 7, 8, 9

[17] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *ICML*, pages 663–670, 2010. 1, 2, 3, 5, 6

[18] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013. 1, 2, 3, 5, 6, 7, 8, 9

[19] J. Chen and J. Yang. Robust subspace segmentation via low-rank representation. *IEEE Transactions on Cybernetics*, 44(8):1432–1445, 2014. 1

[20] X. Fang, Y. Xu, X. Li, Z. Lai, and W. K. Wong. Robust semi-supervised subspace clustering via non-negative low-rank representation. *IEEE Transactions on Cybernetics*, 46(8):1828–1838, 2016. 1

[21] J. Wang, X. Wang, F. Tian, C. H. Liu, and H. Yu. Constrained low-rank representation for robust subspace clustering. *IEEE Transactions on Cybernetics*, 47(12):4534–4546, 2017. 1

[22] X. Peng, Z. Yu, Z. Yi, and H. Tang. Constructing the l2-graph for robust subspace learning and subspace clustering. *IEEE Transactions on Cybernetics*, 47(4):1053–1066, 2017. 1

[23] Z. Wen, B. Hou, Q. Wu, and L. Jiao. Discriminative transformation learning for fuzzy sparse subspace clustering. *IEEE Transactions on Cybernetics*, 48(8):2218–2231, 2018. 1

[24] J. Wen, Y. Xu, and H. Liu. Incomplete multiview spectral clustering with adaptive graph learning. *IEEE Transactions on Cybernetics*, pages 1–12, 2019. 1

[25] J. Lee, H. Lee, M. Lee, and N. Kwak. Nonparametric estimation of probabilistic membership for subspace clustering. *IEEE Transactions on Cybernetics*, pages 1–14, 2018. 1

[26] M. Brbi and I. Kopriva. ℓ_0-motivated low-rank sparse subspace clustering. *IEEE Transactions on Cybernetics*, pages 1–15, 2018. 1

[27] H. Zhang, J. Yang, F. Shang, C. Gong, and Z. Zhang. Lrr for subspace segmentation via tractable schatten- p norm minimization and factorization. *IEEE Transactions on Cybernetics*, 49(5):1722–1734, 2019. 1

[28] M. Yin, J. Gao, Z. Lin, Q. Shi, and Y. Guo. Dual graph regularized latent low-rank representation for subspace clustering. *IEEE Transactions on Image Processing*, 24(12):4918–4933, 2015. 1, 2

[29] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *CVPR*, 2011. 1, 2, 3, 5, 6, 7, 8, 9

[30] V. Karavasilis, K. Blekas, and C. Nikou. A novel framework for motion segmentation and tracking by clustering incomplete trajectories. *Computer Vision and Image Understanding*, 116(11):1135–1148, 2012. 1, 2

[31] A. Ellis and J. Ferryman. Biologically-inspired robust motion segmentation using mutual information. *Computer Vision and Image Understanding*, 122:47–64, 2014. 1, 2

[32] C. Lu, H. Min, Z. Zhao, L. Zhu, D. Huang, and S. Yan. Robust and efficient subspace segmentation via least squares regression. In *ECCV*, pages 347–360. Springer, 2012. 1, 2, 3, 5, 6, 7, 8, 9, 10

[33] H. Hu, Z. Lin, J. Feng, and J. Zhou. Smooth representation clustering. In *CVPR*, pages 3834–3841, 2014. 1, 2, 3, 5, 6, 7, 8, 9

[34] X. Peng, L. Zhang, and Z. Yi. Scalable sparse subspace clustering. In *CVPR*, June 2013. 1, 2, 3, 5

[35] J. Feng, Z. Lin, H. Xu, and S. Yan. Robust subspace segmentation with block-diagonal prior. In *CVPR*, pages 3818–3825, 2014. 1, 2, 3, 5

[36] F. Wu, Y. Hu, J. Gao, Y. Sun, and B. Yin. Ordered subspace clustering with block-diagonal priors. *IEEE Transactions on Cybernetics*, 46(12):3209–3219, 2016. 1

[37] B. Li, Y. Zhang, Z. Lin, and H. Lu. Subspace clustering by mixture of gaussian regression. In *CVPR*, pages 2094–2102, 2015. 1, 2, 3, 5

[38] C. Li and R. Vidal. Structured sparse subspace clustering: A unified optimization framework. In *CVPR*, pages 277–286, 2015. 1, 2, 3, 5, 6, 7

[39] C. Li, C. You, and R. Vidal. Structured sparse subspace clustering: A joint affinity learning and subspace clustering framework. *IEEE Transactions on Image Processing*, 26(6):2988–3001, June 2017. 1, 2, 3, 5, 6, 7, 8, 9

[40] J. Xu, K. Xu, K. Chen, and J. Ruan. Reweighted sparse subspace clustering. *Computer Vision and Image Understanding*, 138(0):25–37, 2015. 1, 2, 3, 5

[41] C. Peng, Z. Kang, and Q. Cheng. Subspace clustering via variance regularized ridge regression. In *CVPR*, 2017. 1, 2, 3, 5

[42] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid. Deep subspace clustering networks. In *NIPS*, pages 24–33, 2017. 1, 2, 3, 6, 9

[43] M. Yin, S. Xie, Z. Wu, Y. Zhang, and J. Gao. Subspace clustering via learning an adaptive low-rank graph. *IEEE Transactions on Image Processing*, 27(8):3716–3728, Aug 2018. 1, 2

[44] J. Yang, J. Liang, K. Wang, Y. L. Yang, and M. M. Cheng. Automatic model selection in subspace clustering via triplet relationships. In *AAAI*, 2018. 1, 2

[45] C. You, D. Robinson, and R. Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *CVPR*, pages 3918–3927, 2016. 1, 2, 3, 5, 6, 7, 8, 9

[46] C. You, C. Li, D. P. Robinson, and R. Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *CVPR*, pages 3928–3937, 2016. 1, 3, 6, 7, 8, 9

[47] C. You, D. P. Robinson, and R. Vidal. Provable selfrepresentation based outlier detection in a union of subspaces. In *CVPR*, pages 1–10, 2017. 1, 2

[48] C. You, C. Li, D. P. Robinson, and R. Vidal. Scalable exemplar-based subspace clustering on class-imbalanced data. In *ECCV*, September 2018. 1, 2, 6, 9

[49] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007. 1

[50] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. 1

[51] Liansheng Zhuang, Haoyuan Gao, Zhouchen Lin, Yi Ma, Xin Zhang, and Nenghai Yu. Non-negative low rank and sparse graph for semi-supervised learning. In *CVPR*, pages 2328–2335. IEEE, 2012. 2

[52] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, January 2011. 2, 4, 5, 11

[53] R. Tron and R. Vidal. A benchmark for the comparison of 3d motion segmentation algorithms. In *CVPR*, 2007. 2, 5, 6, 8, 9, 10

[54] 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(6):643–660, 2001. 2, 5, 6, 7, 8, 9, 10

[55] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2, 5, 6, 7, 8, 9, 10

[56] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transaction on Information Theory*, 52(2):489–509, 2006. 2

[57] D. Donoho. Compressed sensing. *IEEE Transaction on Information Theory*, 52(4):1289–1306, 2006. 2

[58] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 3, 5

[59] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49(1):1–23, 1943. 4, 11

[60] J. Eckstein and D. P. Bertsekas. On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992. 4, 11

[61] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006. 4

[62] E. Elhamifar, G. Sapiro, and S. Sastry. Dissimilarity-based sparse subset selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2182–2197, 2016. 4

[63] C. Michelot. A finite algorithm for finding the projection of a point onto the canonical simplex of rn. *Journal of Optimization Theory and Applications*, 50(1):195–200, July 1986. 4

[64] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *ICML*, pages 272–279. ACM, 2008. 4, 12

[65] L. Condat. Fast projection onto the simplex and the ℓ_1 ball. *Mathematical Programming*, 158(1):575–585, Jul 2016. 4

[66] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering analysis and an algorithm. In *NIPS*, volume 14, pages 849–856, 2001. 5

[67] F. S. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *IEEE Workshop on Applications of Computer Vision*, pages 138–142. IEEE, 1994. 5, 6, 7, 9

[68] G. Cohen, S. Afshar, J. Tapson, and A. Schaik. EMNIST: an extension of MNIST to handwritten letters. *CoRR*, abs/1702.05373, 2017. 5, 6, 8, 9

[69] I. Jolliffe. Principal component analysis and factor analysis. In *Principal Component Analysis*, pages 115–128. Springer, 1986. 6

[70] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, 2013. 6, 8