Efficient subspace clustering of hyperspectral images using similarity-constrained sampling

Jhon Lopez,* **Carlos Hinojosa**, **and Henry Arguello**
Universidad Industrial de Santander, Department of Computer Science, Bucaramanga, Colombia

Abstract. The unsupervised classification of hyperspectral images (HSIs) draws attention in the remote sensing community due to its inherent complexity and the lack of labeled data. Among unsupervised methods, sparse subspace clustering (SSC) achieves high clustering accuracy by constructing a sparse affinity matrix. However, SSC has limitations when clustering HSI images due to the number of spectral pixels. Specifically, the temporal complexity grows at a cubic ratio of the size of the data, making it inefficient for addressing HSI subspace clustering. We propose an efficient SSC-based method that significantly reduces the temporal and spatial computational complexity by splitting the HSI clustering task using similarity-constrained sampling. Our similarity-constrained sampling strategy considers both edge and superpixel information of the HSI to boost the clustering performance. This sampling strategy enables an intelligent selection of spectral signatures, and then, we split the clustering problem into multiples threads. Experimental results on widely used HSI datasets show that the efficiency of the proposed method outperforms baseline methods by up to 30% in overall accuracy and up to six times in computing time. © *2021 Society of Photo-Optical Instrumentation Engineers (SPIE)* [DOI: 10.1117/1.JRS.15.036507]

Keywords: spectral imaging; sampling; unsupervised learning; clustering; sparse subspace clustering.

Paper 210113 received Feb. 24, 2021; accepted for publication Jul. 13, 2021; published online Jul. 27, 2021.

1 Introduction

Hyperspectral imaging devices capture spectral information from a scene by obtaining a large amount of spatial information at different electromagnetic radiation frequencies. Hyperspectral images (HSIs) are considered three-dimensional (3D) datasets or data cubes with two-dimensions (2D) in the space domain (x, y) and one in the wavelength domain λ. Since different materials reflect electromagnetic energy differently at specific wavelengths,[1] the knowledge of spectral content in various spatial locations from one scene can be a valuable tool for detecting, identifying, and classifying of materials and objects with complex compositions.[2] In particular, each pixel in HSI is represented by a vector with values that correspond to the intensity in different spectral bands and is known as a spectral signature or spectral pixel, which can be used to distinguish materials in a scene. The classification of spectral images is an essential task for many practical applications, such as precision in agriculture,[3] monitoring, and managing the environment,[4] security, and defense.[5]

Clustering is a common unsupervised classification task in which a set of objects is clustered, such that objects in the same group (also known as cluster) are more similar to each other than those in other groups. For years, many clustering algorithms have been proposed, and they mainly differ in how they define clusters and how they find them efficiently. For instance, spectral clustering (SC)[6] is a popular and highly effective algorithm that finds the membership of the data points to a cluster using the spectrum of a symmetric non-negative affinity matrix with inputs that measure the similarities between the connected points. Therefore, the essential step in SC-based methods is to construct a similarity graph.[6] The sparse subspace clustering (SSC)[7] is a popular and accurate algorithm that proposes capturing the general geometric relationship between all data points by expressing each data point as a linear combination of the other points, from all other points, and

*Address all corresondence to Jhon Lopez, jhon2208456@correo.uis.edu.co

then the solution set is restricted to being scattered or sparse, minimizing the ℓ_1 norm of the matrix of representation coefficients. Using the sparse representation matrix, the similarity graph is constructed, from which the segmentation of the data is obtained using SC.[7,8]

Considering that spectral pixels with similar spectra have a high probability of belonging to the same low-dimensional structure, some SSC-based algorithms have been successfully used for HSI clustering.[9–14] In general, such approaches take advantage of the spatial-contextual information of the HSI and incorporate a regularization constraint in the SSC optimization model. However, their main limitation is the overall computational burden. For instance, given an HSI with M rows, N columns, and L spectral bands, SSC needs to compute the $n \times n$ sparse coefficient matrix, where $n = NM$ spectral pixels, with a computational complexity of $O(Ln^3)$.[10] Recently, different works have proposed more efficient approaches to addressing the scalability issue of SSC.[15–20] For instance, the authors in Ref. 15 address the subspace clustering problem by randomly dividing the dataset into two subsets to alleviate the SSC computational burden. Although such methods achieve a good performance when clustering large datasets, they do not fully exploit the information and spectral–spatial dependence of HSIs.

1.1 *Paper Contribution*

Considering that an efficient and careful selection of data subsets could speed up the unsupervised learning process, and the spatial-contextual information of the HSI could increase the clustering accuracy, this paper proposes an efficient subspace clustering algorithm for HSI using similarity-constrained sampling. Specifically, the proposed approach first clusters the spectral pixels into high spatially correlated blocks using edge and superpixel information; then, it separates the data points within each segment into two subsets. Next, our proposed method employs SSC on the first subset to learn the underlying subspace structure. Finally, the spectral pixels belonging to the second subset are projected on the learned structure, and their cluster membership is computed considering the smallest projection error. The obtained results show that an efficient and careful selection of data subsets speeds up the unsupervised learning process while achieving high-clustering performance in terms of overall accuracy (OA).

We organize the paper as follows: in Sec. 2, we briefly review some related works that rely on SSC and focus on improving the scalability limitations of SSC or exploiting the spatial information to improve HSI clustering. In Sec. 3, we develop the proposed method and its complexity analysis. In Sec. 4, we present the experimental results, and Sec. 5 contains the conclusions of this work.

2 Related Work

In the literature, the scalability issue of SSC and its use in performing land cover segmentation on spectral images have been studied separately. In this section, we review some related works from these two points of view. The underlying idea behind the SSC algorithm is the self-expressiveness property of the data, which states that it is possible to efficiently represent each data point in a union of subspaces by a linear or affine combination of other points. In general, such a representation is not unique because there are infinitely many ways to express a data point as a combination of other points. The main observation is that a sparse representation of a data point ideally corresponds to a combination of a few points from its own subspace. This motivates solving a global sparse optimization problem, the solution of which is used in an SC[8,21] framework to infer the clustering of data.

Specifically, let us consider a given collection of n data points $\mathbf{D} = \{\mathbf{d}_1, \cdots, \mathbf{d}_n\}$ that lie in the union of k linear or affine subspaces. SSC expresses each data point $\mathbf{d}_j$ as a linear combination of all other points in $\mathbf{D}$, i.e., $\mathbf{d}_j = \sum_{i \neq j} \boldsymbol{\Gamma}_{ij} \mathbf{d}_i$, where $\boldsymbol{\Gamma}_{ij}$ is nonzero only if $\mathbf{d}_i$ and $\mathbf{d}_j$ are from the same subspace, for $(i, j) \in \{1, \cdots, n\}$. In this formulation, the matrix of the data $\mathbf{D}$ is a self-expressive dictionary in which each point can be written as a linear combination of other points. Further, the representations $\{\boldsymbol{\Gamma}_{ij}\}$ are called subspace-preserving; as shown in Ref. 8, if the subspaces are low-dimensional and independent, subspace-preserving representation can be obtained using ℓ_1 minimization. Then, assuming that $\boldsymbol{\Gamma}_j$ is sparse, SSC solves the following optimization problem:

$$\min_{\Gamma_j \in \mathbb{R}^n} \|\mathbf{\Gamma}_j\|_1 + \frac{\tau}{2}\left\|\mathbf{d}_j - \sum_{i \neq j} \Gamma_{ij}\mathbf{d}_i\right\|_2^2, \tag{1}$$

where $\tau > 0$ and $\mathbf{\Gamma}_j = [\Gamma_{1j}, \cdots, \Gamma_{nj}]^T$ encodes information about membership of $\mathbf{d}_j$ to the subspaces. Subsequently, an affinity matrix between any pair of points $\mathbf{d}_i$ and $\mathbf{d}_j$ is defined as $A_{ij} = |\Gamma_{ij}| + |\Gamma_{ji}|$. Finally, the clustering result is obtained by applying SC[8,21] to the Laplacian matrix $\mathbf{L}$ induced by the affinity matrix $\mathbf{A}$, where $\mathbf{L}$ is given as

$$\mathbf{L} = \mathbf{I} - \beta^{-1/2}\mathbf{A}\beta^{-1/2}, \tag{2}$$

where $\beta = \in \mathbb{R}^{n\times n}\text{diag}\{b_i\}$ with $b_i = \sum_{j=1}^{n} \mathbf{A}_{ij}$ and $\mathbf{I} \in \mathbb{R}^{n\times n}$ is an identity matrix. Although the representation produced by SSC is guaranteed to be subspace preserving, the affinity matrix may lack connectedness,[22] i.e., the data points from the same subspace may not form a connected component of the affinity graph due to the sparseness of the connections, causing oversegmentation.

2.1 *Fast and Scalable Subspace Clustering Methods*

In the state-of-art, different works have proposed efficient approaches to address the scalability issue of SSC. For instance, in Ref. 20, the authors proposed the scalable and robust SSC (SR-SSC) algorithm, which selects a few sets of anchor points using a randomized hierarchical clustering method. Then, within each set of anchor points, it solves the LASSO[18] problem for each data point, allowing only anchor points to have nonzero weights, thus reducing the number of variables drastically. Another scalable algorithm is the oracle guided elastic net (EnSC),[16] which studies the geometry of the elastic net regularizer (a mixture of the ℓ_1 and ℓ_2 norms) and uses it to derive a probably correct and scalable active set method for finding the optimal coefficients. In anther work entitled scalable exemplar-based subspace clustering for class-imbalanced data (ESC-FFS),[19] the authors looked for a subset of data that best represents all of the data points measured by the ℓ_1 norm of the representation coefficients. In addition, in Ref. 17, the authors proposed the orthogonal matching pursuit (OMP) algorithm to solve the SSC optimization problem, which we refer to as SSC-OMP throughout the paper. Finally, in Ref. 15, the authors proposed the scalable SSC (SSSC) to address the subspace clustering problem by randomly dividing the dataset into two subsets. Then, the SSC is applied to the first subset to learn the underlying subspace structure of the data. Such knowledge about the data subspaces is used to cluster the remaining data. In this regard, each point in the second subspace is projected onto each of the previously computed subspaces. Then, the difference of each point in a subspace is computed and assigned to the subspace in which it obtained the minimal residual. Despite SSSC achieving good performance when clustering large data, this approach relies on random initialization to separate the data into two subsets and sacrifices clustering accuracy for computational efficiency. In general, the aforementioned scalable methods do not fully exploit the structure of HSI or take advantage of spectral–spatial dependency of the data when clustering HSI imagery.

2.2 *SSC-Based Methods for HSI*

Some SSC-based methods that take advantage of the neighboring spatial information but still present the scalability issue of SSC have been proposed for HSI. Under the context of HSIs, the $M \times N \times L$ 3D image data cube can be rearranged into a 2D matrix $\mathbf{D} \in \mathbb{R}^{n\times L}$ to apply the SSC algorithm, where $n = MN$. Taking into account that the spectral pixels belonging to the same land cover material are arranged in common regions, different works[9–13,23] aim at obtaining a piecewise smooth sparse coefficient matrix to incorporate such contextual dependence. For instance, the authors in Ref. 9 proposed the 3DS-SSC method, which helps to guarantee spatial smoothness and reduce the representation bias by adding a 3D regularization term in the SSC optimization problem, which enforces a local averaging constraint on the sparse coefficient matrix. Another approach proposed in Ref. 12 takes into consideration the SSC model to obtain a more accurate coefficient matrix, which is used to build the adjacent matrix. The authors in

Ref. 12 also built a weight matrix $\mathbf{W}$ that takes into account spatial information to regularize connections in the $\mathbf{\Gamma}$ coefficient matrix, shown in Eq. (1). On the other hand, in Ref. 23, the authors proposed a similar approach to the previously mentioned work,[12] in which they built a more precise adjacent matrix using the max-pooling operation. In particular, the authors in Ref. 23 proposed mapping the feature points into a much higher dimensional kernel space to extend the linear SSC model to nonlinear manifolds, which can better fit the complex nonlinear structure of HSIs. With the help of the kernel sparse representation, a more accurate representation coefficient matrix can be obtained. A spatial max-pooling operation is utilized for the representation coefficients to generate more discriminant features by integrating the spatial-contextual information, which is essential for the accurate modeling of HSIs.

3 Fast Similarity-Constrained Sampling Sparse Subspace Clustering

Most subspace clustering methods have problems assigning a point to a subspace when they are close to the intersection of two or more subspaces.[8] Considering the spatial information, we propose using a sampling restricted by spatial similarity to separate data that are possibly close to the intersection of multiple subspaces. To do this, the HSI is initially divided into high spatially correlated segments containing points that possibly belong to the intersection of subspaces. Then, efficient sampling is applied to each segment to select in-sample and out-sample points. Therefore, SSC will be applied individually at in-sample points of each segment. The assignment of the out-sample points to the clusters is carried out using the minimum residual obtained by projecting each point on already clustered in-sample points. An overview of the proposed method, named fast similarity-constrained sampling sparse subspace clustering (F4SC), is shown in Fig. 1.

3.1 *Edge Detection and Superpixels*

Commonly, SSC is used to cluster HSI by only taking advantage of the spectral information. However, these algorithms do not take into account the spatial relationship among pixels of the HSI when deciding the class to which each point belongs. In contrast, in this work, we rely on edge detection and superpixels to consider the spatial information present in the HSI since, with the combination of these two methods, the generated segments are better adapted to the structure

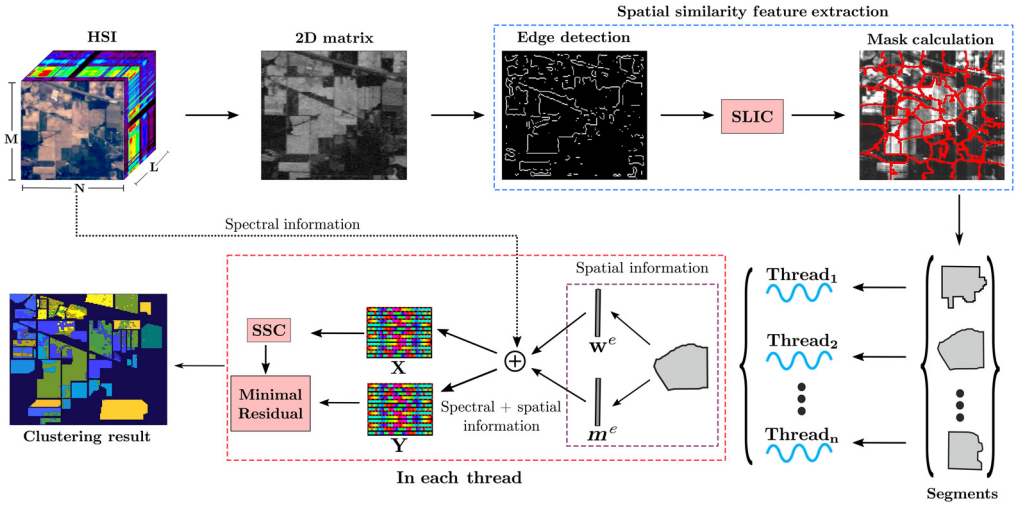

Fig. 1 F4SC. Initially, we obtain a 2D matrix from the HSI by averaging all of the spectral bands. Then, we extract spatial similarities by fusing the edge information and superpixel. Next, using the spectral information of the HSI and the extracted spatial information, we perform similarity-constrained sampling to obtain segments and obtain two subsets from each of them: **X**, **Y**. Finally, we perform simultaneously subspace clustering on **X** and use the learned subspaces structure to clustering **Y** in the threads configured in the proposed method execution environment.

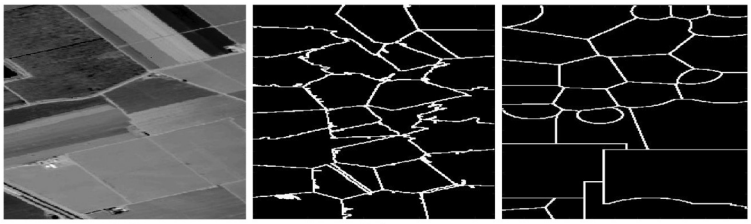

Fig. 2 (a) 2D matrix obtained from the HSI. (b) Result of edge detection by averaging all HSI spectral bands and then applying superpixels. (c) Averaging all HSI spectral bands and calculate superpixels.

of the possible subspaces present in the HSI, as shown in Fig. 2. Note that, in Fig. 2(c), the structure of the segments tends to be regular geometric figures, and a possible structure of the possible clusters present in the scene is not obtained. In contrast, in Fig. 2(b), it is achieved.

Edge detection in grayscale images has been thoroughly studied, and it is well established[24] due to its use in different areas, such as computer vision. However, for multichannel images, such as HSI, this topic is less developed since defining borders for these images is a challenge due to the high dimensionality of the data. Therefore, this work uses a practical approach that consists of adding all of the spectral bands of an HSI and obtaining a 2D image as

$$\mathbf{H}'(i,j) = \frac{1}{L}\sum_{k=1}^{L}\mathbf{H}(i,j,k) \quad \forall\ (i,j) \in (M,N). \tag{3}$$

It is worth mentioning that there are other methods of obtaining a 2D matrix from the HSI, such as principal component analysis[25] (PCA) or random projection.[26] Once the 2D image is obtained, the Sobel edge detection operator $\mathbf{\Omega}$[27] is applied to obtain the edge image as

$$\mathbf{B} = \mathbf{\Omega}(\mathbf{H}') \in \{\mathbf{0},\mathbf{1}\}^{M\times N}. \tag{4}$$

Considering the edge information, we also apply the SLIC[28] algorithm to obtain a segmentation of neighboring pixels. In the first row of Fig. 1, we represent the aforementioned procedure, which allows us to distinguish the points present at the intersection of subspaces. It is important to note that, in this step, we only extract the spatial information from the HSI using edge detection and a superpixel algorithm, as shown in Fig. 3, where the extraction of the spatial information of the points inside and outside the mask of a certain segment is shown. Specifically, we form the mask for each segment by taking pixels around its edge. In this sense, the mask allows us to define a region limit of each segment. We then refer to "points outside the mask" to those points within the segment edge and to "points inside the mask" to those points belonging to the segment but outside its edge. In the following section, we use the information given by these masks to perform a similarity-constrained sampling of the spectral signatures before subspace clustering.

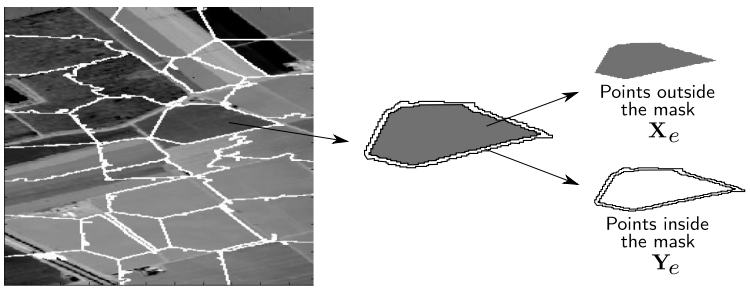

Fig. 3 Point selection inside and outside the mask.

3.2 *Similarity-Constrained Sampling*

The superpixel map is represented as an image $\mathbf{S} \in \mathbb{R}^{M \times N}$, which is calculated on the image $\mathbf{B}$. Using the information contained in $\mathbf{S}$, the proposed method performs a similarity-constrained sampling within each segment considering that the pixels of the superpixels maintain a high spatial correlation, i.e., the correlation is greater between the points near the centroid and decreases at the boundaries of the segment (superpixel). Specifically, our method first calculates a mask for each segment $e = 1, \cdots, N_v$ by taking pixels around the border to improve the separation, where N_v is the total amount of segments. We denote $\mathbf{w}^e \in \mathbb{R}^{u_e}$ as the vector of size u_e containing the indices of all pixels belonging to the superpixel e but lying outside the mask, see Fig. 3. Similarly, we denote $\mathbf{m}^e \in \mathbb{R}^{n_e}$ as the vector of size n_e containing the indices of all pixels belonging to the superpixel e that lie within the mask. Then, we perform the sampling on the HSI data for each segment e as follows:

$$\begin{aligned} \mathbf{X}_e &= \{\mathbf{H}_p : p \in \mathbf{w}^e\} \in \mathbb{R}^{L \times u_e} \\ \mathbf{Y}_e &= \{\mathbf{H}_p : p \in \mathbf{m}^e\} \in \mathbb{R}^{L \times n_e}, \end{aligned} \tag{5}$$

where $\mathbf{H}_p \in \mathbb{R}^L$ denotes the spectral pixel of $\mathbf{H}$ given by the spatial position index p. Through the paper, we refer to in-sample points ($\mathbf{X}_e$) as those spectral pixels from e that lie outside the mask and out-of-sample points ($\mathbf{Y}_e$) as those spectral pixels from e that only lie within the border mask. For ease of notation, we drop the subscript e in the $\mathbf{X}_e$ and $\mathbf{Y}_e$ matrices; nevertheless, note that all of the following operations are performed on each segment e.

3.3 *Subspace Clustering Approach*

Once the masks are defined, our method learns the underlying subspace structure given by the in-sample points $\mathbf{X}$. In particular, SSC solves the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{\Gamma},\mathbf{Z}} \|\mathbf{\Gamma}\|_1 + \frac{\lambda_Z}{2}\|\mathbf{Z}\|_F^2 \\ &\text{s.t. } \mathbf{X} = \mathbf{X}\mathbf{\Gamma} + \mathbf{Z}, \quad \operatorname{diag}(\mathbf{\Gamma}) = 0, \end{aligned} \tag{6}$$

where $\mathbf{\Gamma} \in \mathbb{R}^{L \times u_e}$ is the sparse representation of the data, $\mathbf{X} \in \mathbb{R}^{L \times u_e}$ are the spectral signatures of the HSI, $\mathbf{Z}$ denotes reconstruction errors for limited rendering ability, and λ_Z corresponds to a regularization parameter. Once the sparse representation matrix $\mathbf{\Gamma}$ is obtained, SSC constructs the similarity graph by solving $\mathbf{A} = |\mathbf{\Gamma}|^T + |\mathbf{\Gamma}|$ and applies SC.[6] We solve the optimization problem in Eq. (6) using the alternating direction method of multipliers described in Ref. 29.

3.4 *Minimal Residual*

After learning the underlying subspace structure of $\mathbf{X}$, we use such knowledge to cluster the out-of-sample points $\mathbf{Y}$. Specifically, each point belonging to $\mathbf{Y}$ is projected on each of the computed subspaces from $\mathbf{X}$. In this work, the sparse representation of $\mathbf{Y}$ over $\mathbf{X}$ is computed and each $\mathbf{y}_i$ is assigned to the nearest subspace on sparse representation-based classification.[30] Then, for each out-of-sample data point $\mathbf{y}_i$, the following optimization problem is solved:

$$\min_{\mathbf{c}_i} \|\mathbf{y}_i - \mathbf{X}\mathbf{c}_i\|_2^2 + \gamma\|\mathbf{c}_i\|_2^2, \tag{7}$$

where $\gamma > 0$ is the tolerance error and $\mathbf{y}_i$ denotes an out-of-sample point. Once the optimal $\mathbf{c}_i$ is obtained, $\mathbf{y}_i$ is assigned to the nearest subspace by solving

$$f(\mathbf{y}_i) = \underset{j}{\operatorname{argmin}}\{\|\mathbf{y}_i - \mathbf{X}\delta_j(\mathbf{c}_i)\|_2\}, \tag{8}$$

where $f(\mathbf{y}_i)$ denotes the assignment of $\mathbf{y}_i$ and the nonzero entries of $\delta_j(c_i)$ are the elements in c_i associated with the j'th subspace.[15,30] The term δ is used to avoid overfitting.

Algorithm 1 F4SC algorithm with input parameters → $\mathbf{H}$, N_v, $\mathbf{k}$

Result: Assigning clusters of $\mathbf{H}$

1 $\mathbf{H}^* \leftarrow \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{L}} \mathbf{H}[:,:,\mathbf{i}]$

2 $\mathbf{B} \leftarrow \Omega(\mathbf{H}^*)$

3 $\mathbf{S_p} \leftarrow$ *algorithmSlic*($\mathbf{B}$, $\mathbf{N_v}$)

4 $\mathbf{Bw} \leftarrow$ *calculateMask*($\mathbf{S_p}$)

5 [**IDX**, **IDY**] $\leftarrow$ *getIndex*(**Bw**)

6 **for** $i \leftarrow 1$ **to** N_v **do**

7 [**idxx**, **idxy**] $\leftarrow$ **IDX**[i]

8 [**idyx**, **idyy**] $\leftarrow$ **IDY**[i]

9 $\mathbf{X} \leftarrow \mathbf{H}[\mathbf{idxx}, \mathbf{idxy}, :]$

10 $\mathbf{Y} \leftarrow \mathbf{H}[\mathbf{idyx}, \mathbf{idyy}, :]$

11 $k_i \leftarrow \mathbf{k}[i]$

12 **if** $k_i = 0$ **then**

13 $k_i \leftarrow$ *estimateNumberClusters*(X)

14 $\mathbf{G}^{(i)} \leftarrow \begin{cases} \mathbf{R} \leftarrow SSC(\mathbf{X}, k_i) \\ minimalResidual(\mathbf{X}, \mathbf{Y}, \mathbf{R}) \end{cases}$

In addition to Fig. 1, all of the steps of F4SC are also shown in Algorithm 1. In lines 1 to 5, we show how we extract the spatial information of the HSI and thus do the similarity-restricted sampling. Later, in lines 7 to 10, we obtain the spectral signatures for the in-sample and out-sample points for each segment. If the number of clusters present in a certain segment is unknown, these are estimated in line 13. Finally, in line 14, we perform the allocation of clusters for each segment.

3.5 *Complexity Analysis*

The computational complexity of the proposed method Algorithm 1 is $O(N_v t_1 L \psi^3) + O(N_v t_2 \psi k_i^2) + O(N_v n \psi^2) + O(2n)$, where ψ is the number of in-sample points that belong to a segment. Notice that, to compute the computational complexity, we assume that the value of ψ is the same for all segments. However, this value could change in practice and depends on the size of the segments. Table 1 details the complexity of each step on the proposed algorithm. Using the big O notation properties, the computational complexity of our method can be

Table 1 Complexity of the proposed algorithm, where t_1 and t_2 are the iterations required to solve the optimization problem shown in Eq. (6).

Term	Description
$O(N_v t_1 L \psi^3)$	Construction of the similarity graph
$O(N_v t_2 \psi k_i^2)$	SC[6]
$O(N_v n \psi^2)$	Projection of clustered points within the sample
$O(N_t)$	SLIC
$O(N_t)$	Edge detection

summarized as $O(N_v t_1 L\psi^3)$. Since our method uses SSC for assigning clusters at points in-sample of each segment, our computational complexity remains cubic, but as $\psi \ll n$ our method is faster than other subspace clustering-based approaches; further, it achieves high clustering accuracy due to the proposed similarity-constrained sampling technique.

4 Experimental Results

In this section, we compare our method (F4SC) against the following subspace clustering-based algorithms: SSSC,[15] SR-SSC,[20] SSC-OMP,[17] EnSC,[16] ESC-FFS,[19] and 3DS-SSC.[9] All of our experiments were conducted using MATLAB vR2020a with 16 workers in a parallel pool configured by default to execute the proposed method on a computer with an Intel Xeon Processor E5-2697 v3 and 180 GB RAM.

4.1 *Hyperspectral Datasets and Used Metrics*

The proposed method was tested on three real hyperspectral datasets[31] with different imaging environments. These images were acquired by the airborne visible/infrared imaging spectrometer and the reflecting optics system imaging spectrometer. The first HSI is the Indian Pines with a spatial resolution of 145×145 and 203 spectral bands. The second image is Salinas with a spatial resolution of 512×217 and 204 spectral bands. The third image is Pavia University with a spatial resolution of 610×340 and 103 spectral bands. In the experiments, the number of k_i classes within a segment was set as manual input for the proposed subspace clustering Algorithm 1. The parameters shown in Eqs. (6) and (7) were fixed as $\lambda_Z = 2 \cdot 10^{-4}$ and $\gamma = 10^{-6}$, respectively. To evaluate the clustering precision, we used OA, average accuracy (AA), Cohen's kappa coefficient (kappa), and normalized mutual information (NMI).[32]

4.2 *Ablation Study*

We first conduct four ablation experiments and investigate different options in the proposed workflow shown in Fig. 1:

- Experiment 1: We only use superpixels for similarity-restricted sampling.
- Experiment 2: We use another superpixel segmentation method, known as linear spectral clustering (LSC),[33] instead of SLIC to segment the 2D image.
- Experiment 3: We use the Canny[34] algorithm for edge detection instead of Sobel.
- Experiment 4: We use a different method to get a 2D matrix from the HSI before applying edge detection. In particular, for these experiments, we use PCA to obtain the 2D matrix.

In Table 2, the results obtained by performing different experiments are shown, which led us to establish the workflow shown in Fig. 1 and Algorithm 1. From experiment 1, we can conclude that only using the superpixels technique is not enough to extract the spatial information from the HSI and the edge information helps to better discriminate among pixels within the intersection of the clusters. In the second experiment, we observe that we also achieve good performance using

Table 2 Results of ablation experiments using the Indian Pines HSI.

Experiment	N_v	OA	AA	Kappa	NMI	Time (s)
1	750	77.58	75.90	0.748	0.632	2.68
2	1050	81.78	79.25	0.795	0.683	6.96
3	400	75.07	68.73	0.719	0.591	2.83
4	1000	79.11	74.40	0.764	0.636	7.22
F4SC	1250	85.08	81.55	0.832	0.711	4.69

LSC, suggesting that any superpixels segmentation algorithm can be adopted in our proposed framework. However, observing the last row of Table 2, we conclude that our F4SC algorithm achieves the best clustering accuracy and computational time when using the SLIC algorithm to obtain the superpixels. In the third experiment, the Canny algorithm allows us to achieve a better clustering of the subspaces than the previous experiment using the default parameters of the method. However, the Canny algorithm requires tuning more parameters in comparison with the Sobel operator. Finally, in experiment 4, PCA was applied to obtain a 2D matrix from the HSI for edge detection using the Sobel operator. In this experiment, a better clustering performance was obtained compared with previous experiments; however, the PCA algorithm is slow, and its computational complexity grows exponentially with the dimensions of the HSI. It is worth mentioning that the number of segments shown in Table 2 leads to the best OA within each experiment configuration.

Through the development of all four experiments, we conclude that the best way to approach the cluster of subspaces in HSIs using the spatial information for our approach are the steps described in Fig. 1 and Algorithm 1. Once the reasons for how we define our workflow have been shown, the next step is to analyze the impact on precision and computational time in the number of segments obtained on image **B**. Then, in Figs. 4–6, the evolution of the OA and the execution time in terms of the number of segments (N_v) for each dataset previously mentioned are illustrated.

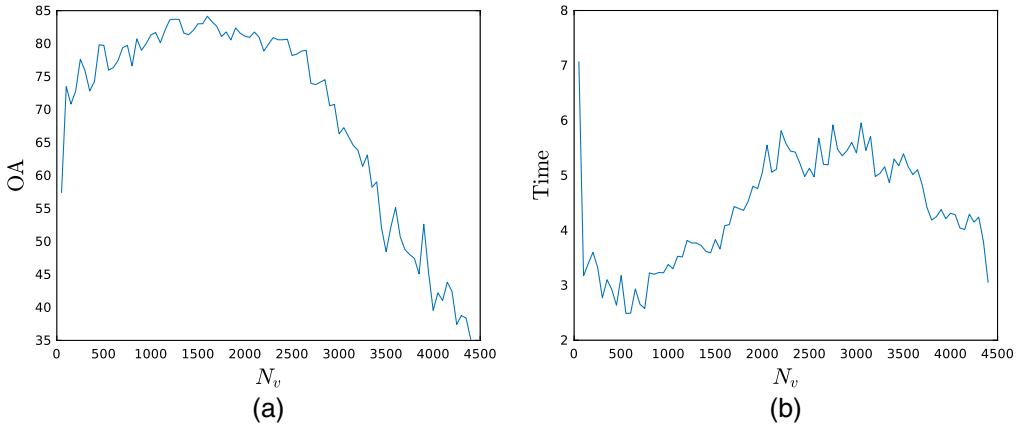

Fig. 4 Evolution of (a) OA and (b) execution time as a function of the number of segments (N_v) in the Indian Pines HSI.

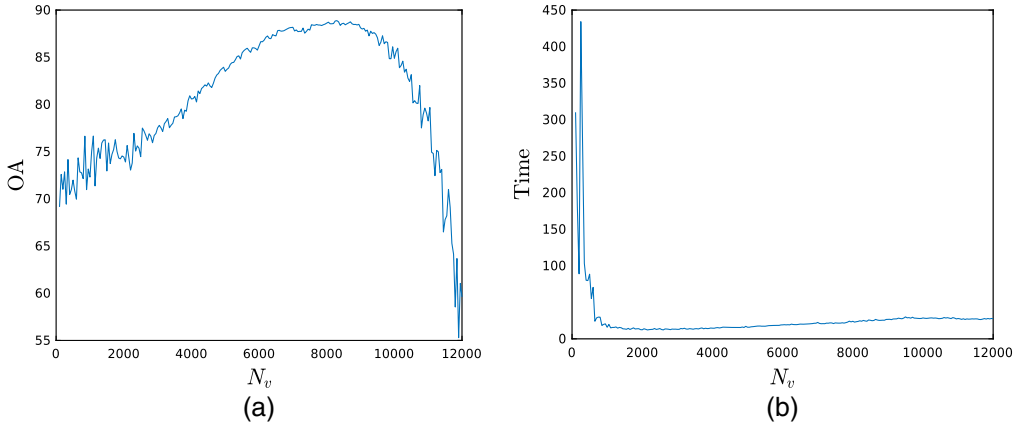

Fig. 5 Evolution of (a) OA and (b) execution time as a function of the number of segments (N_v) in the Salinas HSI.

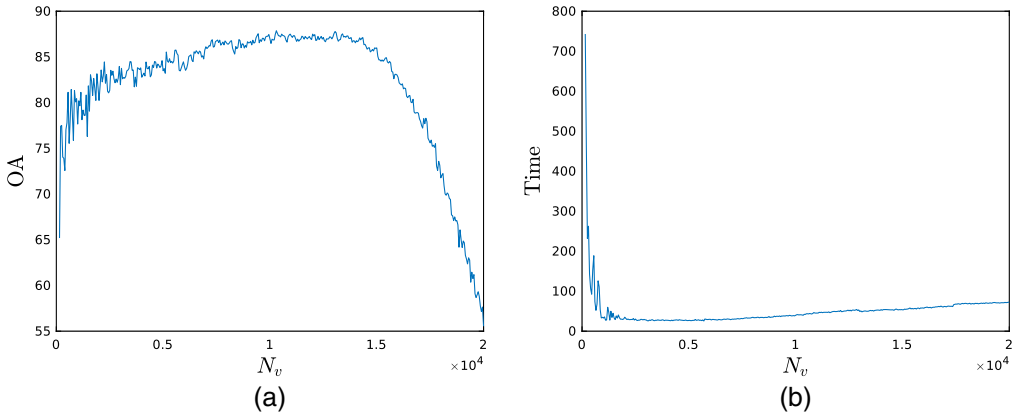

Fig. 6 Evolution of (a) OA and (b) execution time as a function of the number of segments (N_v) in the Pavia University HSI.

Table 3 Quantitative results for Indian Pines, where $N_v = 1250$ in the proposed method.

Class	SR-SSC	EnSC	SSC-OMP	ESC-FFS	3DS-SSC	SSSC	F4SC
Alfalfa	00.00	00.00	00.00	00.00	00.00	00.00	**89.10**
Corn-notill	53.20	59.77	16.70	44.53	26.82	*62.20*	**86.41**
Corn-mintill	14.00	19.68	04.00	*37.71*	36.87	19.00	**87.73**
Corn	07.00	0.880	00.00	08.86	*13.08*	04.40	**83.97**
Grass-pasture	22.30	34.10	**94.10**	52.58	57.14	35.30	*83.02*
Grass-trees	*82.90*	81.39	51.60	28.21	57.67	71.20	**88.36**
Grass-pasture-mowed	00.00	*4.73*	00.20	00.00	00.00	00.00	**60.71**
Hay-windrowed	85.10	85.67	27.60	85.98	**97.49**	*86.80*	83.26
Oats	04.80	04.70	12.10	*50.00*	00.00	03.70	**65.00**
Soybean-notill	32.40	37.87	37.50	30.04	*49.07*	29.60	**81.89**
Soybean-mintill	53.90	*56.36*	35.70	34.82	50.55	52.90	**85.54**
Soybean-clean	15.30	15.68	*32.10*	15.68	25.13	13.20	**80.27**
Wheat	62.30	32.95	47.40	**94.63**	*92.20*	29.90	87.81
Woods	72.80	*87.90*	50.00	42.76	56.92	84.10	**89.86**
Buildings-grass-trees-drives	28.40	29.43	4.300	15.54	*49.22*	12.60	**82.84**
Stone-steel-towers	*73.60*	13.77	00.00	10.57	**86.02**	00.00	73.12
OA	36.84	33.21	20.32	38.01	*48.09*	35.44	**85.08**
AA	39.06	40.64	11.54	34.51	*43.64*	32.36	**81.55**
Kappa	0.303	0.272	0.095	0.315	*0.432*	0.291	**0.832**
NMI	0.423	0.425	0.132	0.422	*0.505*	0.4155	**0.711**
Time (s)	*16.85*	1640	17.50	61.20	753.4	56.77	**4.69**

As shown in the previous plots (Figs. 4–6) regardless of the HSI, when the number of segments is small, the computation time is high, and the precision is low because most of the spectral signatures taken as points within the sample are clustered by SSC. When the number of segments is large, the precision decreases, and the computation time is optimal because not enough points are taken within the sample to learn the structure of the clusters present, and therefore, the clustering that is carried out using minimal residual is imprecise. In addition, we can observe that the highest precision is obtained when the number of points belonging per segment $\rho \in [15,35]$.

4.3 *Visual Maps and Quantitative Results*

In Tables 3–5, we present quantitative results of the proposed approach (F4SC) and the baseline algorithms for the Indian Pines, Salinas, and Pavia University images, respectively. For the state-of-the-art algorithms with which we compare our model and perform a random initialization, the results shown in the tables are the average of running this method 10 times. In addition, the optimal value for each metric is shown in bold, and the second-best result is in italics. Following each table, the visual maps of the three methods that obtained the best OA are shown, ordered from highest to lowest, in Figs. 7–9.

Table 4 Quantitative results for Salinas, where $N_v = 7000$ in the proposed method.

Class	SR-SSC	EnSC	SSCOMP	ESC-FFS	3DS-SSC	SSSC	F4SC
Brocoli_green_weeds_1	00.00	00.00	04.50	00.00	00.00	00.00	**89.50**
Brocoli_green_weeds_2	51.90	64.30	29.30	*99.40*	**100.0**	62.20	89.16
Fallow	*93.30*	**100.0**	55.00	00.00	26.19	19.00	86.54
Fallow_rough_plow	96.10	03.70	*99.00*	**99.13**	85.06	04.40	89.45
Fallow_smooth	74.30	61.40	85.60	**94.39**	*91.23*	35.30	88.76
Stubble	99.80	*99.90*	62.20	90.47	**100.0**	71.20	91.89
Celery	64.00	98.00	18.40	**99.55**	*98.66*	00.00	90.89
Grapes untrained	63.90	72.90	64.50	62.07	42.17	*86.80*	**88.68**
Soil vineyard develop	82.70	00.60	71.80	*91.01*	**91.93**	03.70	89.54
Corn senesced green weeds	63.60	*64.20*	18.50	56.92	61.14	29.60	**88.77**
Lettuce romaine 4wk	4.90	00.00	01.80	00.20	**98.51**	52.90	*84.83*
Lettuce romaine 5wk	91.40	39.10	22.50	**95.58**	*94.17*	13.20	85.47
Lettuce romaine 6wk	00.00	84.00	00.00	**91.37**	00.00	29.90	*85.26*
Lettuce romaine 7wk	38.70	*93.40*	11.10	92.24	**100.0**	84.10	85.61
Vineyard untrained	50.00	51.00	45.30	59.75	*74.72*	12.60	**86.87**
Vineyard vertical trellis	*98.50*	96.6	01.20	10.84	**100.0**	00.00	86.81
OA	56.98	56.18	33.36	69.24	*72.20*	70.76	**88.55**
AA	54.79	56.23	29.45	65.20	*72.80*	60.05	**87.99**
Kappa	0.523	0.519	0.279	0.656	*0.695*	0.660	**0.873**
NMI	0.71	0.744	0.527	0.739	**0.812**	*0.809*	0.752
Time (s)	*102.83*	5987	248.9	894.3	4453	2339.1	**25.71**

Table 5 Quantitative results for Pavia University, where $N_v = 10300$ in the proposed method.

Class	SR-SSC	EnSC	SSC-OMP	ESC-FFS	3DS-SSC	SSSC	F4SC
Asphalt	62.10	70.00	52.90	64.75	*76.90*	61.07	**87.27**
Meadows	83.90	76.00	**87.70**	24.01	48.01	77.09	*86.71*
Gravel	00.00	00.00	27.20	00.28	*74.71*	01.21	**90.14**
Trees	48.40	66.00	69.60	71.63	**91.53**	58.11	*87.24*
Painted metal sheets	80.90	58.00	00.00	**99.47**	00.00	00.00	*91.08*
Bare soil	37.70	23.00	20.30	30.62	*45.54*	20.23	**84.73**
Bitumen	00.00	08.00	03.10	*91.42*	00.00	13.23	**91.50**
Self-blocking bricks	37.80	05.00	27.30	*88.51*	87.32	13.72	**90.47**
Shadows	78.30	00.00	00.90	**99.04**	00.00	03.75	*92.08*
OA	43.46	32.98	39.36	44.03	*55.98*	42.70	**87.86**
AA	51.13	37.18	30.11	*66.30*	47.12	34.30	**88.18**
Kappa	0.346	0.224	0.284	0.370	*0.479*	0.245	**0.843**
NMI	0.519	0.477	0.413	0.515	*0.623*	0.368	**0.708**
Time (s)	*264.1*	5185	730.3	4169	5172	5272	**40.59**

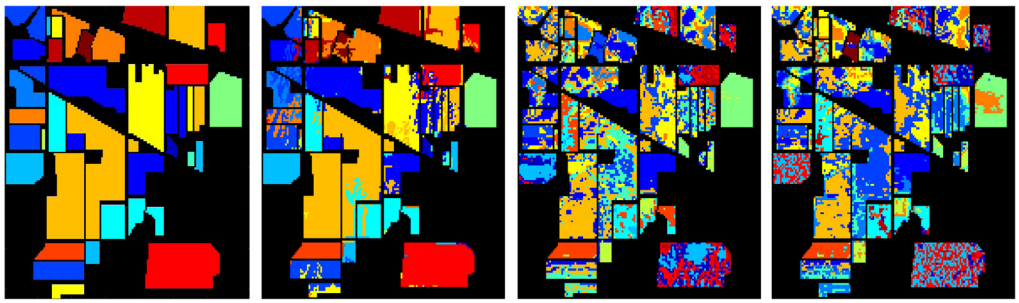

Fig. 7 Visual maps for Indian Pines: (a) ground truth, (b) F4SC, (c) ESC-FFS, and (d) SR-SSC.

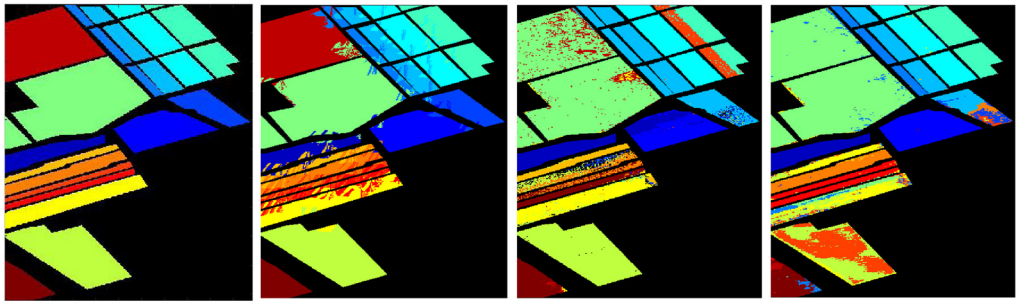

Fig. 8 Visual maps for Salinas: (a) ground truth, (b) F4SC, (c) SSSC, and (d) SR-SSC.

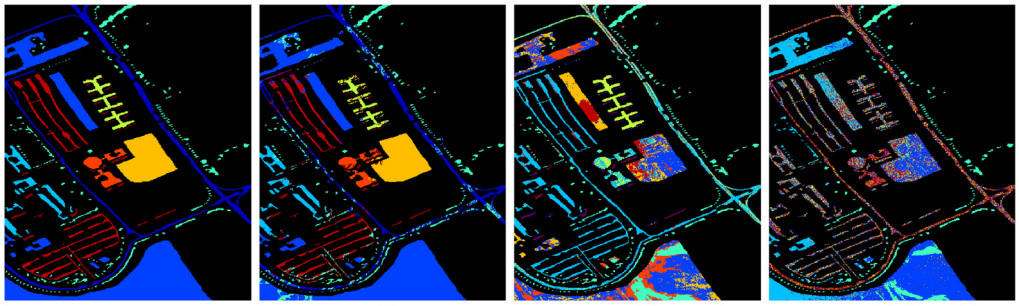

Fig. 9 Visual maps for Pavia University: (a) ground truth, (b) F4SC, (c) SR-SSC, and (d) SSSC.

Table 6 Results of deep learning approaches in the HSI Indian Pines.

	F3D	FST	GCN	MSC	F4SC
OA	**96.98**	*95.83*	95.42	94.28	85.05
AA	**96.85**	*95.68*	95.28	94.08	81.55
Kappa	**0.968**	*0.957*	0.954	0.941	0.832
Time (s)	*274.3*	321.7	322.4	389.5	**4.69**

Table 7 Results of deep learning approaches in the HSI Salinas.

	F3D	FST	GCN	MSC	F4SC
OA	**97.65**	96.75	*96.84*	96.42	88.55
AA	**97.52**	*96.89*	96.63	96.24	87.99
Kappa	**0.974**	*0.969*	0.967	0.963	0.873
Time (s)	289.6	364.1	*210.9*	394.4	**25.71**

4.4 *Comparison with Deep Learning Methods*

Due to the great advances that artificial intelligence has had in the last decade, in this section, we show some works in which deep learning is used for subspace clustering in HSIs. In Tables 6 and 7, we compare our results with the following works: fused 3-D deep neural networks (F3D),[35] fast 3D-CNN (FST),[36] graph convolutional neural network (GCN),[37] and multiscale 3D convolutional neural network (MSC).[38]

As can be noticed in the previous tables, the methods based on deep learning obtain the highest precision when assigning the clusters belonging to an HSI. However, even with their high precision, these methods have a high inference time compared with F4SC. Our method can be seen as a trade-off between precision and computation time, being efficient in both aspects.

5 Conclusions

In this work, we presented an efficient subspace clustering algorithm for HSIs that can handle large-scale datasets and take advantage of spectral image's neighboring spatial information to boost the clustering accuracy. Our method considers the spatial information present in the scene

by detecting edges and superpixels and then dividing the HSI into multiple segments. Next, our method performs the SSC algorithm and minimal residual within each segment. Finally, the method merges the individual results to obtain the full HSI clustering. The experimental results with real datasets show that our approach is remarkably efficient compared with other algorithms that we evaluated. Specifically, we achieved an increase in the OA of 36%, 32%, and 16% for the Indian Pines, Pavia University, and Salinas datasets, respectively, compared with the state-of-the-art algorithms. Furthermore, we show that the execution time of our method can be up to 50 times faster than some methods based on deep learning.

6 Appendix A. Estimation of the Number of Subspaces

Typically, to test the performance of SC-based algorithms, it is assumed that the labels or ground truth are known. However, in practice, the number of clusters k_i could be unknown. In such a case, the value of k_i can be estimated using a rank selection technique that selects the rank and sparsity tuning parameters simultaneously for hyperspectral data.[39] Das et al.[40] proposed estimating the number of clusters present in the HSI by choosing the optimal threshold value between the higher and smaller eigenvalues of the matrix covariance of the HSI. These approaches are interesting, but they estimate the number of clusters using direct information from the spectral image. Therefore, we decided to use the approach proposed by Von Luxburg,[21] which estimates the number of clusters using the Laplacian matrix $\mathbf{A}$:

$$k_i \approx \arg\max_{i\in[u_e-1]}(\lambda_{i+1}-\lambda_i), \tag{9}$$

where $\lambda_1 \leq \lambda_2, \cdots, \leq \lambda_{u_e}$ are the eigenvalues of the normalized Laplacian of the graph given by matrix $\mathbf{A}$ described in Sec. 2.

References

1. C. V. Correa et al., "Multiple snapshot colored compressive spectral imager," *Opt. Eng.* **56**(4), 041309 (2016).
2. K. Sanchez, C. Hinojosa, and H. Arguello, "Supervised spatio-spectral classification of fused images using superpixels," *Appl. Opt.* **58**(7), B9–B18 (2019).
3. N. Patel et al., "Study of crop growth parameters using airborne imaging spectrometer data," *Int. J. Remote Sens.* **22**(12), 2401–2411 (2001).
4. B. Hörig et al., "Hymap hyperspectral remote sensing to detect hydrocarbons," *Int. J. Remote Sens.* **22**(8), 1413–1422 (2001).
5. M. T. Eismann, A. D. Stocker, and N. M. Nasrabadi, "Automated hyperspectral cueing for civilian search and rescue," *Proc. IEEE* **97**(6), 1031–1055 (2009).
6. A. Y. Ng et al., "On spectral clustering: analysis and an algorithm," in *Adv. Neural Inf. Process. Syst.*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds., MIT Press (2002).
7. E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *IEEE Conf. Comput. Vision and Pattern Recognit., 2009. CVPR 2009.*, IEEE, pp. 2790–2797 (2009).
8. E. Elhamifar and R. Vidal, "Sparse subspace clustering: algorithm, theory, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(11), 2765–2781 (2013).
9. C. A. Hinojosa et al., "Hyperspectral image segmentation using 3D regularized subspace clustering model," *J. Appl. Remote Sens.* **15**(1), 016508 (2021).
10. C. Hinojosa, J. Bacca, and H. Arguello, "Coded aperture design for compressive spectral imaging subspace clustering," *IEEE J. Sel. Top. Sign. Process.* **12**(6), 1589–1600 (2018).
11. C. A. Hinojosa, J. Bacca, and H. Arguello, "Spectral imaging subspace clustering with 3-D spatial regularizer," in *Digital Hologr. and Three-Dimens. Imaging*, Optical Society of America, p. JW5E–7 (2018).
12. H. Zhang et al., "Spectral–spatial sparse subspace clustering for hyperspectral remote sensing images," *IEEE Trans. Geosci. Remote Sens.* **54**(6), 3672–3684 (2016).

13. H. Zhai et al., "A new sparse subspace clustering algorithm for hyperspectral remote sensing imagery," *IEEE Geosci. Remote Sens. Lett.* **14**(1), 43–47 (2016).
14. S. Huang, H. Zhang, and A. Pižurica, "Semisupervised sparse subspace clustering method with a joint sparsity constraint for hyperspectral remote sensing images," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **12**(3), 989–999 (2019).
15. X. Peng et al., "A unified framework for representation-based subspace clustering of out-of-sample and large-scale data," *IEEE Trans. Neural Networks Learn. Syst.* **27**(12), 2499–2512 (2015).
16. C. You et al., "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 3928–3937 (2016).
17. C. You, D. Robinson, and R. Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 3918–3927 (2016).
18. N. Gauraha, "Introduction to the lasso," *Resonance* **23**(4), 439–464 (2018).
19. C. You et al., "Scalable exemplar-based subspace clustering on class-imbalanced data," in *Proc. Eur. Conf. Comput. Vision (ECCV)*, pp. 67–83 (2018).
20. M. Abdolali, N. Gillis, and M. Rahmati, "Scalable and robust sparse subspace clustering using randomized clustering and multilayer graphs," *Signal Process.* **163**, 166–180 (2019).
21. U. Von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.* **17**(4), 395–416 (2007).
22. B. Nasihatkon and R. Hartley, "Graph connectivity in sparse subspace clustering," in *CVPR 2011*, IEEE, pp. 2137–2144 (2011).
23. H. Zhai et al., "Kernel sparse subspace clustering with a spatial max pooling operation for hyperspectral remote sensing data interpretation," *Remote Sens.* **9**(4), 335 (2017).
24. E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*, Vol. **201**, Prentice Hall Englewood Cliffs (1998).
25. H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscipl. Rev.: Comput. Stat.* **2**(4), 433–459 (2010).
26. E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *Proc. Seventh ACM SIGKDD Int. Conf. Knowl. Discov. and Data Mining*, pp. 245–250 (2001).
27. O. R. Vincent et al., "A descriptive algorithm for Sobel image edge detection," in *Proc. Inf. Sci. & IT Educ. Conf. (InSITE)*, Informing Science Institute California, Vol. 40, pp. 97–107 (2009).
28. R. Achanta et al., "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(11), 2274–2282 (2012).
29. M. Annergren, A. Hansson, and B. Wahlberg, "An ADMM algorithm for solving l1 regularized MPC," in *IEEE 51st IEEE Conf. Decision and Control (CDC)*, IEEE, pp. 4486–4491 (2012).
30. J. Wright et al., "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(2), 210–227 (2009).
31. M. Graña, M. Veganzons, and B. Ayerdi, "Hyperspectral remote sensing scenes," 2020, http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes.
32. T. Lillesand, R. W. Kiefer, and J. Chipman, *Remote Sensing and Image Interpretation*, John Wiley & Sons (2015).
33. Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 1356–1363 (2015).
34. J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**(6), 679–698 (1986).
35. A. Sellami et al., "Fused 3-D spectral-spatial deep neural networks and spectral clustering for hyperspectral image classification," *Pattern Recognit. Lett.* **138**, 594–600 (2020).
36. M. Ahmad et al., "A fast and compact 3-D CNN for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.* 1–5 (2020).
37. A. Qin et al., "Spectral–spatial graph convolutional networks for semisupervised hyperspectral image classification," *IEEE Geosci. Remote Sensi. Lett.* **16**(2), 241–245 (2018).

38. M. He, B. Li, and H. Chen, "Multi-scale 3d deep convolutional neural network for hyperspectral image classification," in *IEEE Int. Conf. Image Process. (ICIP)*, IEEE, pp. 3904–3908 (2017).
39. B. Rasti, M. O. Ulfarsson, and J. R. Sveinsson, "Hyperspectral subspace identification using sure," *IEEE Geosci. Remote Sens. Lett.* **12**(12), 2481–2485 (2015).
40. S. Das, A. Routray, and A. K. Deb, "Noise robust estimation of number of endmembers in a hyperspectral image by eigenvalue based gap index," in *8th Workshop Hyperspectral Image and Signal Process.: Evol. in Remote Sens. (WHISPERS)*, IEEE, pp. 1–5 (2016).

Jhon Lopez received his BSc Eng degree in systems engineering and informatics from the Universidad Industrial de Santander, Colombia, in 2020. Currently, he is pursuing his MSc degree in computer science at the Universidad Industrial de Santander. His research interests include deep optics, high-dimensional image processing, unsupervised learning, computer vision, and deep learning.

Carlos Hinojosa received his BSc Eng and MSc degrees in systems engineering and informatics from the Universidad Industrial de Santander, Colombia, in 2015 and 2018, respectively. Currently, he is pursuing his PhD in computer science at the Universidad Industrial de Santander. His research interests include computational imaging, high-dimensional image processing, unsupervised learning, optical code design, compressive imaging, computer vision, and sparse representation.

Henry Arguello received his BSc Eng degree in electrical engineering and MSc degree in electrical power from the Universidad Industrial de Santander, Bucaramanga, Colombia, in 2000 and 2003, respectively, and his PhD in electrical engineering from the University of Delaware, USA, in 2013. Currently, he is an associate professor with the Department of Systems Engineering, Universidad Industrial de Santander. His research interests include high-dimensional signal processing, optical imaging, compressed sensing, and computational imaging.