Sparse Subspace Clustering: Algorithm, Theory, and Applications

Elhamifar, E. and Vidal, R.

Presentation for Stat 572

Jerry Wei
Mentor: Vincent Roulet

Department of Statistics
University of Washington

Subspace Clustering Problem

High-dimensional data pose challenges to classical clustering methods

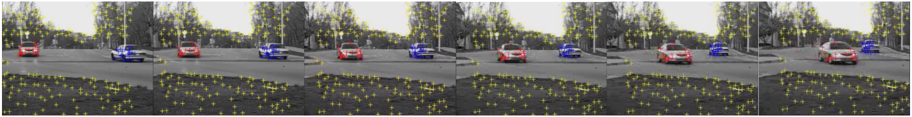
Key observation: Data in a class or category lie in a low-dimensional subspace

Subspace Clustering Problem

High-dimensional data pose challenges to classical clustering methods

Key observation: Data in a class or category lie in a low-dimensional subspace

Examples: rigidly moving object in a video



The video has frames $f : 1, \ldots, F$, and a set of N feature points $\{\boldsymbol{x}_{fi} \in \mathbb{R}^2\}_{i=1}^{N}$ is tracked across the frames. For analysis, each feature trajectory $\boldsymbol{y}_i$ is taken as a data point, where $\boldsymbol{y}_i$ is obtained by stacking the feature points x_{fi} in the video as

$$\boldsymbol{y}_i \equiv [\boldsymbol{x}_{1i}^T, \boldsymbol{x}_{2i}^T, \ldots, \boldsymbol{x}_{Fi}^T]^T \in \mathbb{R}^{2F}$$

Subspace Clustering Problem

Goal:

- find the number of subspaces and their dimensions
- find a basis for each subspace
- group the sample points into subspaces

Remark: Extension of traditional clustering: clustering and dimension reduction simultaneously

- Rigid motion: an affine subspace of dimension at most 3
- Images of a subject with fixed pose and varying illumination: lie close to a linear subspace of dimension 9.

Prior Work on Subspace Clustering

- **Iterative methods** Alternate between assigning points to subspaces and fitting a subspace to each cluster. (K-subspaces, Median K-flats)
- **Algebraic approaches** Based on factorization of the data matrix. (GPCA)
- **Iterative statistical methods** Parametric assumption on data distribution, iterate between clustering and subspace estimation by EM algorithm. (Random Sample Consensus)
- **Information-theoretic statistical approaches** Minimizes the information-theoretic cost to fit a mixture of degenerate Gaussian to the points, up to a given distortion. (Agglomerative Lossy Compression)
- **Spectral clustering-based methods** Construct a similarity graph based on data information and then apply spectral clustering. (Local Subspace Affinity, Low-Rank Subspace Clustering) **Sparse Subspace Clustering (SSC)**

Notations

Let $\{\mathcal{S}_l\}_{l=1}^{n}$ be an arrangement of n linear subspaces of dimensions $\{d_l\}_{l=1}^{n}$ lying in $\mathbb{R}^D$.
Let $\{\boldsymbol{y}_i\}_{i=1}^{N}$ be the collection of N data points that are free of noises and lie in the union of the n subspaces.
Let N_l be the number of samples points in subspace $\mathcal{S}_l$ and the data matrix

$$\boldsymbol{Y} \equiv [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N] = [\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n]\boldsymbol{\Gamma}$$

where $\boldsymbol{Y}_l \in \mathbb{R}^{D \times N_l}$ is a matrix of all the points in $\mathcal{S}_l$ with $N_l > d_l$ and $\boldsymbol{\Gamma}$ is an unknown permutation matrix.

Self-expressiveness property

Definition

Each data point in a union of subspaces can be efficiently reconstructed by a combination of other points in the dataset.

Subspace-Sparse Representation: represented by a few other points in the same subspace

Self-expressiveness property

Definition

Each data point in a union of subspaces can be efficiently reconstructed by a combination of other points in the dataset.

Subspace-Sparse Representation: represented by a few other points in the same subspace

Specifically, each data point $\boldsymbol{y}_i \in \cup_{l=1}^{n} \mathcal{S}_l$ can have the representation

$$\boldsymbol{y}_i = \boldsymbol{Y}\boldsymbol{c}_i, c_{ii} = 0$$

where $\boldsymbol{c}_i \equiv (c_{i1}, c_{i2}, \ldots, c_{iN})^T$ is the vector of constant coefficients

Goal: $\boldsymbol{c}_i$ has a few non-zero entries corresponding to data points in the same subspace as $\boldsymbol{y}_i$

Sparse Optimization Problem

The optimization problem:

$$\min ||\boldsymbol{c}_i||_q \quad s.t. \quad \boldsymbol{y}_i = \boldsymbol{Y}\boldsymbol{c}_i, c_{ii} = 0$$

where $\boldsymbol{c}_i$ is a N dimensional vector of constants on the weights for the other data points and $||.||_q$ is the l_q norm.
Ideally, want to use l_0 norm, but is NP-hard

Sparse Optimization Problem

The optimization problem:

$$\min ||\boldsymbol{c}_i||_q \quad s.t. \quad \boldsymbol{y}_i = \boldsymbol{Y}\boldsymbol{c}_i, c_{ii} = 0$$

where $\boldsymbol{c}_i$ is a N dimensional vector of constants on the weights for the other data points and $||.||_q$ is the l_q norm.
Ideally, want to use l_0 norm, but is NP-hard

Use l_1 norm the convex problem is: For all $i = 1, \ldots, N$,

$$\min ||\boldsymbol{c}_i||_1 \quad s.t. \quad \boldsymbol{y}_i = \boldsymbol{Y}\boldsymbol{c}_i, c_{ii} = 0$$

or in matrix form,

$$\min ||\boldsymbol{C}||_1 \quad s.t. \quad \boldsymbol{Y} = \boldsymbol{Y}\boldsymbol{C}, diag(\boldsymbol{C}) = 0$$

where $\boldsymbol{C} \equiv [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N] \in \mathbb{R}^{N \times N}$ is the matrix whose i-th column corresponds to the representation coefficient for $\boldsymbol{y}_i$.

Clustering using Sparse Coefficients

The optimization gives the sparse coefficient matrix $\boldsymbol{C}$ as the output.
Use the matrix to construct a graph, and apply spectral clustering gives the final clustering result.

Weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$, where the N data points compose the vertices $\mathcal{V}$, and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ denotes the set of edges between nodes, $\boldsymbol{W}$ denotes the weights for the edges.

$\boldsymbol{W} = |\boldsymbol{C}| + |\boldsymbol{C}|^T, \boldsymbol{W}_{ij} = |c_{ij}| + |c_{ji}|.$

Sparse Subspace Clustering Algorithm

Algorithm : Sparse Subspace Clustering (SSC)

Input: A set of points $\{\boldsymbol{y}_i\}_{i=1}^N$ lying in a union of n linear subspaces $\{\mathcal{S}_l\}_{l=1}^n$.

1. Solve the sparse optimization program
2. Normalize the columns of the resulting coefficients matrix $\boldsymbol{C}$ as $\boldsymbol{c}_i \leftarrow \frac{\boldsymbol{c}_i}{||\boldsymbol{c}_i||_\infty}$
3. Form a similarity graph with N nodes representing the data points. Set the weights on the edges between the nodes by $\boldsymbol{W} = |\boldsymbol{C}| + |\boldsymbol{C}|^T$
4. Apply spectral clustering to the similarity graph.

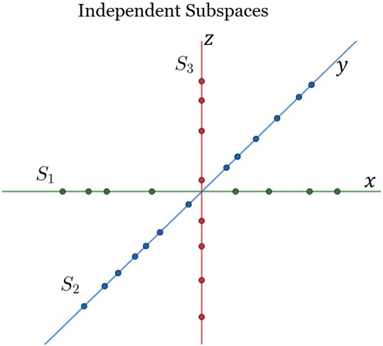
Output: Segmentation of the data: $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_n$

Subspace-Sparse Recovery Theory

Subspace-Sparse Representation: represented by a few other points in the same subspace

Definition (Independent Subspaces)

A collection of subspaces $\{\mathcal{S}_i\}_{i=1}^n$ is said to be independent if $dim(\oplus_{i=1}^n \mathcal{S}_i) = \sum_{i=1}^n dim(\mathcal{S}_i)$ where $\oplus$ denotes the direct sum operator.



Subspace-Sparse Recovery Theory

Theorem (Independent Subspaces Recovery)

Consider a collection of data points drawn from n independent subspaces $\{\mathcal{S}_i\}_{i=1}^n$ of dimensions $\{d_i\}_{i=1}^n$. Let $\boldsymbol{Y}_i$ denote N_i data points in $\mathcal{S}_i$, where $\mathrm{rank}(\boldsymbol{Y}_i) = d_i$, and let $\boldsymbol{Y}_{-i}$ denote data points in all subspaces except $\mathcal{S}_i$. Then, for every $\mathcal{S}_i$ and every nonzero $\boldsymbol{y}$ in $\mathcal{S}_i$, the l_q-minimization program

$$\begin{bmatrix} \boldsymbol{c}^* \\ \boldsymbol{c}_-^* \end{bmatrix} = \mathrm{argmin} \left\| \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix} \right\|_q \quad s.t. \quad \boldsymbol{y} = [\boldsymbol{Y}_i \quad \boldsymbol{Y}_{-i}] \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix}$$

for $q < \infty$, recovers a subspace-sparse representation, i.e., $\boldsymbol{c}^* \neq \mathbf{0}$ and $\boldsymbol{c}_-^* = \mathbf{0}$

Practical Extension: Data Nuisances

Data points contaminated with sparse outlying entries and noise

$$\boldsymbol{y}_i = \boldsymbol{y}_i^0 + \boldsymbol{e}_i^0 + \boldsymbol{z}_i^0$$

where $\boldsymbol{y}_i^0$ is the error-free part of observation and lies perfect on the underlying subspace, $\boldsymbol{e}_i^0 \in \mathbb{R}^D$ is sparse outlying entries, and $\boldsymbol{z}_i^0 \in \mathbb{R}^D$ is the noise component.

Utilizing self-expressiveness property of $\boldsymbol{y}_i^0$

$$\boldsymbol{y}_i = \sum_{j \neq i} c_{ij} \boldsymbol{y}_j + \boldsymbol{e}_i + \boldsymbol{z}_i$$

$$\boldsymbol{e}_i = \boldsymbol{e}_i^0 - \sum_{j \neq i} c_{ij} \boldsymbol{e}_j^0$$

$$\boldsymbol{z}_i = \boldsymbol{z}_i^0 - \sum_{j \neq i} c_{ij} \boldsymbol{z}_j^0$$

Practical Extension: Data Nuisances

Let matrices $\boldsymbol{E}$ and $\boldsymbol{Z}$ have $\boldsymbol{e}_i$ and $\boldsymbol{z}_i$ as columns, we have the matrix representation:

$$\boldsymbol{Y} = \boldsymbol{YC} + \boldsymbol{E} + \boldsymbol{Z}, \quad diag(\boldsymbol{C}) = 0$$

And hence the optimization program for sparse representation becomes

$$\min ||\boldsymbol{C}||_1 + \lambda_e ||\boldsymbol{E}||_1 + \frac{\lambda_z}{2} ||\boldsymbol{Z}||_F^2$$
$$s.t. \quad \boldsymbol{Y} = \boldsymbol{YC} + \boldsymbol{E} + \boldsymbol{Z}, \quad diag(\boldsymbol{C}) = 0$$

Practical Extension: Affine Subspace

Data point $\boldsymbol{y}_i$ in an affine subspace $\mathcal{S}_l$ with dimension d_l can be written as an affine combination of $d_l + 1$ other points from $\mathcal{S}_l$.

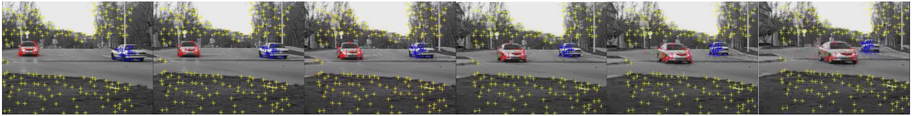
$$\boldsymbol{y}_i = \boldsymbol{Y}\boldsymbol{c}_i, c_{ii} = 0, \mathbf{1}^T \boldsymbol{c}_i = 1$$

where $\boldsymbol{c}_i$ has $d_l + 1$ nonzero entries corresponding to points also in $\mathcal{S}_l$. The more general program:

$$\min ||\boldsymbol{C}||_1 + \lambda_e ||\boldsymbol{E}||_1 + \frac{\lambda_z}{2} ||\boldsymbol{Z}||_F^2$$
$$s.t. \quad \boldsymbol{Y} = \boldsymbol{Y}\boldsymbol{C} + \boldsymbol{E} + \boldsymbol{Z}, \quad diag(\boldsymbol{C}) = 0, \quad \mathbf{1}^T \boldsymbol{C} = \mathbf{1}^T$$

Real Data Experiments: Motion Segmentation

Dataset: Hopkins 155 dataset. Video sequences along with the features extracted and tracked in all the frames.



The video has frames $f : 1, \ldots, F$, and a set of N feature points $\{\boldsymbol{x}_{fi} \in \mathbb{R}^2\}_{i=1}^N$ is tracked across the frames. For analysis, each feature trajectory $\boldsymbol{y}_i$ is taken as a data point, where $\boldsymbol{y}_i$ is obtained by stacking the feature points x_{fi} in the video as

$$\boldsymbol{y}_i \equiv [\boldsymbol{x}_{1i}^T, \boldsymbol{x}_{2i}^T, \ldots, \boldsymbol{x}_{Fi}^T]^T \in \mathbb{R}^{2F}$$

120 videos of two motions ($N = 266$ and $F = 30$)
35 videos of 3 motions ($N = 398$ and $F = 29$)

The most general version of the SSC is implemented. With/without PCA as preprocessing.

Clustering error (%) of different algorithms on the Hopkins 155 dataset

Algorithm	SSC	LSA	LRSC	K-Subspace
2 Motions, without PCA				
Mean	**2.23**	20.88	3.98	28.42
Median	**0**	14.72	**0**	33.72
3 Motions, without PCA				
Mean	**5.78**	21.09	7.96	32.27
Median	**0.95**	22.81	3.40	29.62
2 Motions, with PCA				
Mean	**2.32**	3.01	3.89	19.78
Median	**0**	0.25	**0**	1.62
3 Motions, with PCA				
Mean	5.78	**5.19**	7.88	21.34
Median	**0.95**	1.11	3.39	3.19

Discussion

Summary

- SSC algorithm
- Guarantee for obtaining Subspace-Sparse representation
- Practical Extensions
- Real application on motion segmentation

Advantages:

- Deal with noises, sparse outlying entries, and affine subspaces directly
- Can deal with subspaces with different unknown dimensions

Further Problem:

- Subspace-sparse recovery guarantee for corrupted data
- Theory for graph connectivity
- Application when number of clusters unknown
- Nonlinear extension
- Application to very large dataset

Reference

- Elhamifar, E. and Vidal, R., 2013. Sparse subspace clustering: Algorithm, theory, and applications. IEEE transactions on pattern analysis and machine intelligence, 35(11), pp.2765-2781.
- G. Liu and S. Yan, "Latent Low-Rank Representation for subspace segmentation and feature extraction," 2011 International Conference on Computer Vision, Barcelona, 2011, pp. 1615-1622, doi: 10.1109/ICCV.2011.6126422.

Question?

Graph Connectivity

For subspaces of dimensions greater than or equal to 4, under odd distribution of data, it is possible that points in the same subspace form multiple components.

Consider a regularization term

$$\|\boldsymbol{C}\|_{r,0} \equiv \sum_{i=1}^{N} I(\|\boldsymbol{c}^i\|_2 > 0)$$

where $\boldsymbol{c}^i$ denotes the i-th row of the matrix $\boldsymbol{C}$.
The convex relaxation

$$\|\boldsymbol{C}\|_{r,1} \equiv \sum_{i=1}^{N} \|\boldsymbol{c}^i\|_2$$

The optimization program to consider is

$$\min ||\boldsymbol{C}||_1 + \lambda_r \|\boldsymbol{C}\|_{r,1} \quad s.t. \quad \boldsymbol{Y} = \boldsymbol{YC}, diag(\boldsymbol{C}) = 0$$

Definition (Disjoint Subspaces)

A collection of subspaces $\{\mathcal{S}_i\}_{i=1}^n$ *is said to be disjoint if every pair of subspaces intersect only at the origin. In other words, for every pair of subspaces we have* $dim(\mathcal{S}_i \oplus \mathcal{S}_j) = dim(\mathcal{S}_i) + dim(\mathcal{S}_i)$*, where* $\oplus$ *denotes the direct sum operator.*

To characterize two disjoint subspaces, we can have:

Definition (Smallest Principal Angle)

The smallest principal angle between two subspaces $\mathcal{S}_i$ *and* $\mathcal{S}_j$ *, denoted by* θ_{ij} *, is defined as*

$$cos(\theta_{ij}) \equiv max_{\mathbf{v}_i \ \ in \mathcal{S}_i, \mathbf{v}_j \ \ in \mathcal{S}_j} \frac{\mathbf{v}_i^T \mathbf{v}_j}{||\mathbf{v}_i||_2 ||\mathbf{v}_j||_2}$$

Subspace-Sparse Recovery Theory

For Disjoint subspaces, need to study points in the intersection of subspaces. Let $\boldsymbol{x}$ be a non-zero vector in the intersection of $\mathcal{S}_i$ and $\oplus_{j \neq i} \mathcal{S}_j$. Let

$$\boldsymbol{a}_i = argmin ||\boldsymbol{a}||_1 \ \ s.t. \ \ \boldsymbol{x} = \boldsymbol{Y}_i \boldsymbol{a}$$
$$\boldsymbol{a}_{-i} = argmin ||\boldsymbol{a}||_1 \ \ s.t. \ \ \boldsymbol{x} = \boldsymbol{Y}_{-i} \boldsymbol{a}$$

Theorem

Consider a collection of data points drawn from n independent subspaces $\{\mathcal{S}_i\}_{i=1}^n$ of dimensions $\{d_i\}_{i=1}^n$. Let $\boldsymbol{Y}_i$ denote N_i data points in $\mathcal{S}_i$, where $rank(\boldsymbol{Y}_i) = d_i$, and let $\boldsymbol{Y}_{-i}$ denote data points in all subspaces except $\mathcal{S}_i$. The l_1 minimization

$$\begin{bmatrix} \boldsymbol{c}^* \\ \boldsymbol{c}_-^* \end{bmatrix} = argmin \left\| \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix} \right\|_1 \ \ s.t. \ \ \boldsymbol{y} = [\boldsymbol{Y}_i \ \ \boldsymbol{Y}_{-i}] \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix}$$

recovers a subspace-sparse representation, i.e., $\boldsymbol{c}^* \neq \boldsymbol{0}$ and $\boldsymbol{c}_-^* = \boldsymbol{0}$ if and only if

$$\forall \boldsymbol{x} \in \mathcal{S}_i \cap (\oplus_{j \neq i} \mathcal{S}_j), \boldsymbol{x} \neq 0 \implies ||\boldsymbol{a}_i||_1 < ||\boldsymbol{a}_{-i}||_1$$

Parameter Choice

$$\mu_e \equiv min_i max_{j \neq i} ||\boldsymbol{y}_i||_1, \quad \mu_z \equiv min_i max_{j \neq i} |\boldsymbol{y}_i^T \boldsymbol{y}_j|$$

The choose $\lambda_e \geq \alpha_e/\mu_e, \lambda_z \geq \alpha_z/\mu_z$ with $\alpha_e, \alpha_z \geq 1$.

Proposition

Consider the optimization program with noise and sparse outlying entries. Without the noise term $\boldsymbol{Z}$, if $\lambda_e \leq 1/\mu_e$, then there exists at least one data point $\boldsymbol{y}_l$ for which in the optimal solution we have $(\boldsymbol{c}_l, \boldsymbol{e}_l) = (\boldsymbol{0}, \boldsymbol{y}_l)$. That is, the data point is taken entirely as outlying entries and is not represented as combination of other data points. Similarly, without the sparse outlying term $\boldsymbol{E}$, if $\lambda_z \leq \mu_z$, then there exists at least one data point $\boldsymbol{y}_l$ for which $(\boldsymbol{c}_l, \boldsymbol{z}_l) = (\boldsymbol{0}, \boldsymbol{y}_l)$.

Face Clustering Results

Algorithm	SSC	LSA	RANSAC	LRSC	KSubspace
2 Subjects					
Mean	**1.87**	33.47	37.45	11.41	45.41
Median	**0**	46.09	39.06	11.25	46.88
Time	57.5	13.7	4240.4	2.0	149.2
3 Subjects					
Mean	**3.30**	53.03	49.39	13.97	59.18
Median	**1.04**	51.06	50.52	13.87	59.90
Time	81.1	19.5	6815.0	3.1	536.0
5 Subjects					
Mean	**4.32**	58.82	62.10	21.58	67.59
Median	**2.50**	57.19	63.12	21.56	67.19
Time	135.8	29.4	12721.3	11.2	1115.9
8 Subjects					
Mean	**5.89**	57.41	787	34.73	72.00
Median	**4.59**	57.81	5415	34.37	71.58
Time	216.0	65.0	15901.7	19.5	2030.1
10 Subjects					
Mean	**7.40**	56.04	71.4	51.06	72.03
Median	**5.63**	60.47	72.5	50.78	72.34
Time	326.0	95.3	16393.7	59	3248.0

ADMM Procedure

Algorithm 2: ADMM Procedure to solve sparse-optimization program
Initialization: Set maxIter $= N$ and errThres $= \varepsilon$. $k = 0$, Terminate = False.
Initialize $\boldsymbol{C}^{(0)}, \boldsymbol{E}^{(0)}, \boldsymbol{A}^{(0)}, \boldsymbol{\Delta}^{(0)}, \boldsymbol{\delta}^{(0)}$ to zero.
While (Terminate == False) **do**

1 Update $\boldsymbol{A}^{(k+1)}$ by solving

$$(\lambda_z \boldsymbol{Y}^T\boldsymbol{Y} + \rho \boldsymbol{I} + \rho \mathbf{1}\mathbf{1}^T)\boldsymbol{A}^{(k+1)} = \lambda_z \boldsymbol{Y}^T(\boldsymbol{Y} - \boldsymbol{E}^{(k+1)}) + \rho(\mathbf{1}\mathbf{1}^T + \boldsymbol{C}^{(k)}) - \mathbf{1}\boldsymbol{\delta}^{(k)T} - \boldsymbol{\Delta}^{(k)}$$

2 Update $\boldsymbol{C}^{(k+1)} = \boldsymbol{J} - diag(\boldsymbol{J})$, where $\boldsymbol{J} = \mathcal{T}_{\frac{1}{\rho}}(\boldsymbol{C}^{(k+1)} + \boldsymbol{\Delta}^{(k)}/\rho)$,

3 Update $\boldsymbol{E}^{(k+1)} = \mathcal{T}_{\frac{\lambda_e}{\lambda_z}}(\boldsymbol{Y}\boldsymbol{A}^{(k+1)} - \boldsymbol{Y})$,

4 Update $\boldsymbol{\Delta}^{(k+1)} = \boldsymbol{\Delta}^{(k)} + \rho(\boldsymbol{A}^{(k+1)} - \boldsymbol{C}^{(k+1)})$,

5 Update $\boldsymbol{\delta}^{(k+1)} = \boldsymbol{\delta}^{(k)} + \rho(\boldsymbol{A}^{(k+1)T}\mathbf{1} - \mathbf{1})$,

6 **if** $(max\{||\boldsymbol{A}^{(k+1)} - \boldsymbol{C}^{(k+1)}||_\infty, ||\boldsymbol{A}^{(k+1)T}\mathbf{1} - \mathbf{1}||_\infty, ||\boldsymbol{A}^{(k+1)} - \boldsymbol{A}^{(k)}||_\infty, ||\boldsymbol{E}^{(k+1)} - \boldsymbol{E}^{(k)}||_\infty\} \leq \varepsilon$ or $k+1 \geq$ maxIter):
  Terminate = True
end if

7 $k = k + 1$,

end while Output: Sparse coefficient matrix $\boldsymbol{C} = \boldsymbol{C}^{(k)}$