EVOLUTIONARY SUBSPACE CLUSTERING: DISCOVERING STRUCTURE IN SELF-EXPRESSIVE TIME-SERIES DATA

Abolfazl Hashemi and Haris Vikalo

Department of Electrical and Computer Engineering
University of Texas at Austin, Austin, TX 78712, USA

ABSTRACT

An evolutionary self-expressive model for clustering a collection of evolving data points that lie on a union of low-dimensional evolving subspaces is proposed. A parsimonious representation of data points at each time step is learned via a non-convex optimization framework that exploits the self-expressiveness property of the evolving data while taking into account data representation from the preceding time step. The resulting scheme adaptively learns an innovation matrix that captures changes in self-representation of data in consecutive time steps as well as a smoothing parameter reflective of the rate of data evolution. Extensive experiments demonstrate superiority of the proposed framework overs state-of-the-art static subspace clustering algorithms and existing evolutionary clustering schemes.

***Index Terms*—** subspace clustering, evolutionary clustering, self-expressive models, temporal data, real-time clustering

1. INTRODUCTION

Subspace clustering is the problem of clustering a collection of points that lie on a union of low-dimensional subspaces [1]. This task is typically performed by constructing a similarity matrix that represents the relations among data points and relying on spectral clustering [2] to find a segmentation of the data. To form the similarity matrix, state-of-the-art methods exploit the *self-expressiveness* property [3] of data and identify a so-called subspace preserving similarity matrix that captures similarities among data points originating from the same subspace. In particular, finding such a matrix requires solving the optimization

$$\min_{\mathbf{C}} \|\mathbf{C}\| \quad \text{s.t.} \quad \|\mathbf{X}-\mathbf{X}\mathbf{C}\|_F^2 \leq \epsilon, \quad \text{diag}(\mathbf{C}) = \mathbf{0}, \quad (1)$$

where $\mathbf{X}$ and $\mathbf{C}$ denote the data and representation matrices, respectively. Specifying different matrix norms in the objective function in (1) leads to different subspace clustering schemes. For instance, the sparse subspace clustering (SSC) method in [3, 4] utilizes $\|\cdot\|_1$, while methods based on orthogonal matching pursuit (OMP) [5, 6] and low-rank representations (LRR) enforce minimization of $\|\mathbf{C}\|_0$ and $\|\mathbf{C}\|_*$ (i.e., the nuclear norm or sum of singular values of $\mathbf{C}$), respectively. After $\mathbf{C}$ is found, one defines an affinity (or similarity) matrix $\mathbf{A} = |\mathbf{C}| + |\mathbf{C}|^\top$ and applies spectral clustering [2] to find the clustering solution.

In many problems, besides having structural properties (e.g., union-of-subspaces and self-expressiveness), data is often acquired at multiple points in time and a clustering solution is needed for each time step. In other words, the data points – which are now stored as the columns of the evolving data matrix $\mathbf{X}_t$ – evolve with time. For example, in real-time motion segmentation it is well-known that feature point trajectories associated with motion in a video lie in an affine subspace [7]. Motion during any given short time interval (e.g., a few consecutive frames) is related to the motion in recent past. Therefore, in addition to the union-of-subspaces structure of the video data, there exists an underlying *evolutionary structure* characterizing the motion. Therefore, it is of interest to design and investigate frameworks that exploit both *union-of-subspaces and temporal smoothness* structures to perform fast and accurate clustering, particularly in real-time applications where a clustering solution is required at each time step.

In this paper, we study *evolutionary subspace clustering* – the task of clustering data points that lie on a union of *evolving* subspaces. In particular, we introduce the evolutionary self-expressive model (ESEM), an optimization framework that exploits the self-expressiveness property of data and learns sparse representations while taking into account representations from prior time steps. The task of learning parameters of the ESEM leads to a non-convex optimization problem which is approximately solved via an alternating minimization procedure. In the process of learning data representation, we further tune a smoothing parameter which characterizes the significance of prior data representations, i.e., quantifies similarity of the representation in successive time steps. The smoothing parameter is shown to be reflective of the rate of evolution of the data and signifies the amount of temporal changes in consecutive data snapshots. Following extensive simulations on synthetic datasets and real-world datasets originating from real-time motion segmentation, we demonstrate that the proposed framework significantly improves the performance and decreases computational cost of state-of-the-art *static* subspace clustering algorithms that exploit only the self-expressiveness property of the data.

2. EVOLUTIONARY SELF-EXPRESSIVE MODEL

Let $\{\mathbf{x}_{t,i}\}_{i=1}^{N_t}$ be a collection of (evolving) real-valued D_t-dimensional data points at time t and let us organize those points in a matrix $\mathbf{X}_t = [\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,N_t}] \in \mathbb{R}^{D_t \times N_t}$. The data points are drawn from a union of n_t evolving subspaces $\{\mathcal{S}_{t,i}\}_{i=1}^{n_t}$ with dimensions $\{d_{t,i}\}_{i=1}^{n_t}$. Without loss of generality, we assume that the columns of $\mathbf{X}_t$, i.e., the data points, are normalized vectors with unit ℓ_2 norm.[1] Due to the underlying union-of-subspaces structure, data satisfies the self-expressiveness property [3] formally stated below.

Definition 1. *A collection of evolving data points $\{\mathbf{x}_{t,i}\}_{i=1}^{N_t}$ satisfies the self-expressiveness property if each point has a linear representation in terms of the other points in the collection, i.e., there exist a representation matrix $\mathbf{C}_t$ such that*

$$\mathbf{X}_t = \mathbf{X}_t\mathbf{C}_t, \quad \text{diag}(\mathbf{C}_t) = \mathbf{0}. \quad (2)$$

[1] As we proceed, for simplicity we may omit the time index.

Subspace clustering attempts to partition $\{\mathbf{x}_{t,i}\}_{i=1}^{N_t}$ into n_t groups such that data points that belong to the same subspace are in the same cluster. To distinguish between different methods, we refer to subspace clustering schemes that find a representation matrix $\mathbf{C}_t$ which satisfies (2) as *static subspace clustering* methods. As stated in Sections 1, in many applications the subspaces and the data points lying on the union of those subspaces evolve over time. Imposing the self-expressiveness property helps exploit the fact that the data points belong to a union-of-subspaces. However, (2) alone does not capture potential evolutionary nature of the data. To this end, we propose to find a representation matrix $\mathbf{C}_t$, for each time t, such that

$$\mathbf{C}_t = \mathbf{U} + \alpha \mathbf{C}_{t-1}, \quad \mathbf{X}_t = \mathbf{X}_t \mathbf{C}_t, \quad \operatorname{diag}(\mathbf{C}_t) = \mathbf{0}, \tag{3}$$

where $\mathbf{U}$ and α are parameters that will be learned. Intuitively, the *innovation representation matrix* $\mathbf{U}$ captures changes in the representation of data points from time $t-1$ to t. The term $\alpha \mathbf{C}_{t-1}$ is the part of temporal representation that carries over from the previous snapshot of the data. The parameter $-1 \leq \alpha \leq 1$ quantifies significance of the previous representation on the structure of the data points at time t (i.e., it quantifies the "memory" of the representation). Intuitively, if the data is static we expect $\alpha = 0$. Conversely, if the temporally evolving data is characterized by a relatively stable subspace representation, we expect α to be closer to 1 or -1.

Since we seek sparse representations, we require the innovation matrix $\mathbf{U}$ to be sparse; moreover, $\operatorname{diag}(\mathbf{U}) = \mathbf{0}$. Consequently, the task of finding $\mathbf{U}$ and α amounts to solving

$$\begin{aligned} \min_{\mathbf{U},\alpha} \quad & \|\mathbf{X}_t - \mathbf{X}_t(\mathbf{U} + \alpha \mathbf{C}_{t-1})\|_F^2 \\ \text{s.t.} \quad & \operatorname{diag}(\mathbf{U}) = \mathbf{0}, \quad \|\mathbf{U}\|_0 \leq k, \quad -1 \leq \alpha \leq 1. \end{aligned} \tag{4}$$

Parameter k determines sparsity level of the innovation representations. Since each point in $\mathcal{S}_i$ can be written in terms of at most d points in $\mathcal{S}_i$, we typically set $k \leq d$ (assuming the subspaces have the same dimension).

We refer to (4) as the evolutionary self-expressive model (ESEM). Note that due to cardinality constraint, (4) is a non-convex optimization problem even though the objective and other constraints are convex in both $\mathbf{U}$ and α.

2.1. Alternating minimization solvers

We propose to find $\mathbf{U}$ and α in an alternating fashion. In particular, given $\mathbf{U}_{t-1}$, the innovation representation matrix found at time $t-1$, we can find the smoothing parameter according to

$$\alpha = \underset{-1 \leq \bar{\alpha} \leq 1}{\arg\min} \|\mathbf{X}_t - \mathbf{X}_t(\mathbf{U}_{t-1} + \bar{\alpha} \mathbf{C}_{t-1})\|_F^2. \tag{5}$$

The objective function in (5) is unimodal and convex; in our implementation, we rely on the golden-section search algorithm [8] to efficiently find α. Once α is found, we arrive at the representation learning step which requires solving

$$\begin{aligned} \min_{\mathbf{U}} \quad & \|\mathbf{X}_t - \mathbf{X}_t(\mathbf{U} + \alpha \mathbf{C}_{t-1})\|_F^2 \\ \text{s.t.} \quad & \operatorname{diag}(\mathbf{U}) = \mathbf{0}, \quad \|\mathbf{U}\|_0 \leq k, \end{aligned} \tag{6}$$

which is a non-convex optimization problem due to the cardinality constraint. Let $\widetilde{\mathbf{X}}_t = \mathbf{X}_t - \alpha \mathbf{X}_t \mathbf{C}_{t-1}$. Then,

$$\min_{\mathbf{U}} \quad \|\widetilde{\mathbf{X}}_t - \mathbf{X}_t \mathbf{U}\|_F^2 \quad \text{s.t.} \quad \operatorname{diag}(\mathbf{U}) = \mathbf{0}, \quad \|\mathbf{U}\|_0 \leq k. \tag{7}$$

The optimization problem (7) is clearly related to static subspace clustering with sparse representation (cf. (1)), and in general to the compressed sensing problems [9]. Therefore, similar to static sparse subspace clustering schemes [3–6, 10], we can employ compressed sensing approaches such as basis pursuit (BP) [11] (or the related LASSO [12]), orthogonal matching pursuit (OMP) [13], and orthogonal least squares (OLS) [14] algorithms to find a suboptimal innovation matrix $\mathbf{U}$ in polynomial time.

Specifically, for the BP-based representation learning strategy, we consider the convex relaxation program

$$\min_{\mathbf{U}} \quad \|\mathbf{U}\|_1 + \frac{\lambda}{2} \|\widetilde{\mathbf{X}}_t - \mathbf{X}_t \mathbf{U}\|_F^2 \quad \text{s.t.} \quad \operatorname{diag}(\mathbf{U}) = \mathbf{0}, \tag{8}$$

which can be solved using any convex solver (e.g. schemes based on ADMM [15]). Here, $\lambda > 0$ is a regularization parameter that determines sparsity level of the innovation representations.

For the OMP-based strategy, to learn the representation for each data point $\mathbf{x}_{j_t}$, $j \in [N]$, we define an initial residual vector $\mathbf{r}_0 = \widetilde{\mathbf{x}}_{j_t}$ and greedily select k data points indexed by $\mathcal{A}_k = \{i_1, \ldots, i_k\} \subset [N]$ that contribute to the representation of $\mathbf{x}_{j_t}$ according to

$$i_\ell = \underset{i \in [N] \setminus \mathcal{A}_{\ell-1} \cup \{j\}}{\arg\max} |\mathbf{r}_{\ell-1} \mathbf{x}_{i_t}|^2, \tag{9}$$

where $\ell \in [k]$. The residual vector is updated according to $\mathbf{r}_\ell = \mathbf{P}_{\mathcal{A}_\ell}^{\perp} \widetilde{\mathbf{x}}_{j_t}$, where $\mathbf{P}_{\mathcal{S}}^{\perp} = \mathbf{I} - \mathbf{X}_{\mathcal{S}} \mathbf{X}_{\mathcal{S}}^{\dagger}$ is the projection operator onto the orthogonal complement of the subspace spanned by the columns of $\mathbf{X}_{\mathcal{S}}$, and $\mathbf{X}_{\mathcal{S}}^{\dagger} = \left(\mathbf{X}_{\mathcal{S}}^{\top} \mathbf{X}_{\mathcal{S}}\right)^{-1} \mathbf{X}_{\mathcal{S}}^{\top}$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{X}_{\mathcal{S}}$. Once $\mathcal{A}_k$ is determined, the innovation representation is computed as the least square solution $\mathbf{u}_j = \mathbf{X}_{\mathcal{A}_{kt}}^{\dagger} \widetilde{\mathbf{x}}_{j_t}$.

The OLS-based representation learning strategy is similar to that of OMP, except with a modified selection criterion:

$$i_\ell = \underset{i \in [N] \setminus \mathcal{A}_{\ell-1} \cup \{j\}}{\arg\max} \frac{|\mathbf{r}_{\ell-1} \mathbf{x}_{i_t}|^2}{\|\mathbf{P}_{\mathcal{A}_{\ell-1}}^{\perp} \mathbf{x}_{i_t}\|_2^2}. \tag{10}$$

Once both $\mathbf{U}$ and α are found, the desired representation matrix $\mathbf{C}_t$ is computed by $\mathbf{C}_t = \mathbf{U} + \alpha \mathbf{C}_{t-1}$.

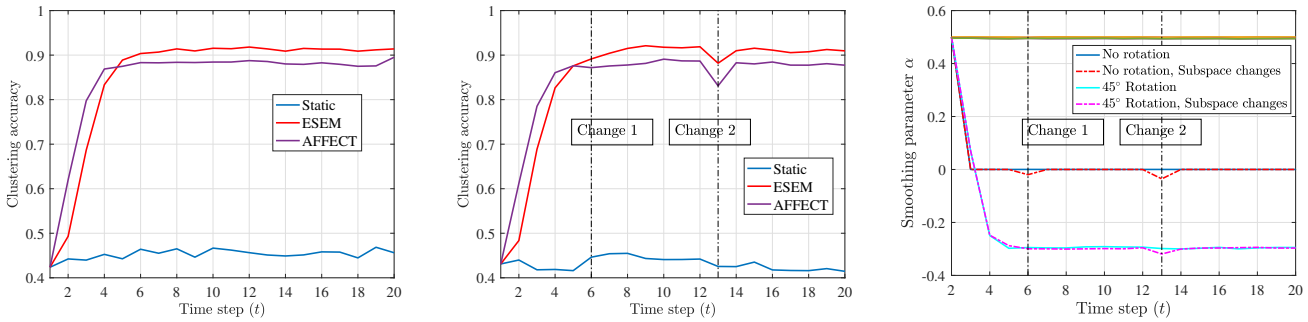
Remark 1. In practice, it may happen that some of the data points vanish over time while new data points are introduced. Our proposed framework readily deals with such scenarios. Let $\mathcal{T}$ denote the set of indices of data points introduced at time t that were not present at time $t-1$. To incorporate these points into the model, we expand $\mathbf{C}_{t-1}$ by inserting all-zero vectors in rows and columns indexed by $\mathcal{T}$. New data points do not play a role in the temporal representations of other data points but they may participate in the innovation representation matrix $\mathbf{U}$. Finally, let $\overline{\mathcal{T}}$ denote the set of indices of data points that were present at time $t-1$ but have vanished at time t; those points are removed from the model by excluding rows and columns of $\mathbf{C}_{t-1}$ that are indexed by $\overline{\mathcal{T}}$.

2.2. Complexity analysis

Since it takes $\mathcal{O}(N^2)$ to evaluate the objective function in (5), the complexity of finding the smoothing parameter using the golden-section search is $\mathcal{O}(N^2)$.

The cost of using BP-based strategy to learn the innovation representation matrix $\mathbf{U}$ in τ iterations of the interior-point method is $\mathcal{O}(\tau D N^3)$. However, by using an efficient ADMM implementation the complexity can be reduced to $\mathcal{O}(\tau_m D^2 N^2)$ where τ_m denotes the maximum number of iterations of the ADMM algorithm.

Since greedy schemes require search over $\mathcal{O}(N)$ D-dimensional data points in k iterations, the complexity of learning innovation representation matrix using OMP and OLS methods is $\mathcal{O}(kDN^2)$ and $\mathcal{O}(kD^2N^2)$, respectively. The complexity of the OLS-based



(a) (b) (c)

Fig. 1: (a) Comparison of clustering accuracy on simulated rotating random subspaces data with 45° rotation. (b) Comparison of clustering accuracy on simulated rotating random subspaces data with 45° rotation and subspace changes. (c) Comparison of temporal changes in the value of smoothing parameter α for various settings of rotating random subspaces data.

method can further be reduced using the accelerated OLS (AOLS) algorithm, introduced in [16]. In contrast to OLS that greedily selects data points according to (10), AOLS efficiently builds a collection of orthogonal vectors to represent the basis of $\mathbf{P}^{\perp}_{\mathcal{A}_{\ell-1}}$ in order to reduce the cost of projection involved in (10). In addition, AOLS anticipates future selections via choosing L data points in each iteration, where $L \geq 1$ is an adjustable hyper-parameter. Therefore, in our implementations, we employ the AOLS strategy instead of OLS to learn the innovation matrix $\mathbf{U}$. More details regarding implementation of ESEM are included as part of the extended manuscript [17].

2.3. Dealing with outliers and missing entries

The evolving data may contain outliers or missing entries at some or all of the time steps. The proposed framework allows for application of convex relaxation methods to handle such cases. Specifically, let $\mathbf{E}$ denote a sparse matrix containing outliers, and let Ω denote the set of observed entries of the corrupted data $\mathbf{X}^c_t$. Define the operator $\mathcal{P}_\Omega : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$ as the orthogonal projector onto the span of matrices having zero entries on $[D] \times [N]\backslash\Omega$, but agreeing with $\mathbf{X}^c_t$ on entries indexed by the set Ω. Prior to employing greedy representation learning methods, we identify outliers and values of the missing entries by solving the convex program

$$\begin{aligned} \min_{\mathbf{X}_t,\mathbf{E}} \quad & \|\mathbf{X}_t\|_* + \lambda_e\|\mathbf{E}\|_1 \\ \text{s.t.} \quad & \mathcal{P}_\Omega(\mathbf{X}^c_t) = \mathcal{P}_\Omega(\mathbf{X}_t), \quad \mathbf{X}^c_t = \mathbf{X}_t + \mathbf{E}. \end{aligned} \tag{11}$$

Then we can apply the ESEM framework using any of the greedy representation learning methods to process the "clean" data $\mathbf{X}_t$, ultimately finding the representations and clustering results.

Compared to greedy methods, BP-based approach enjoys joint representation learning and corruption elimination. That is, by letting $\widetilde{\mathbf{X}}_t = \mathbf{X}_t - \alpha\mathbf{X}_t\mathbf{C}_{t-1}$ we may solve

$$\begin{aligned} \min_{\mathbf{X}_t,\mathbf{U},\mathbf{E}} \quad & \|\mathbf{U}\|_1 + \frac{\lambda}{2}\|\widetilde{\mathbf{X}}_t - \mathbf{X}_t\mathbf{U}\|^2_F + \lambda_x\|\mathbf{X}_t\|_* + \lambda_e\|\mathbf{E}\|_1 \\ \text{s.t.} \quad & \mathcal{P}_\Omega(\mathbf{X}^c_t) = \mathcal{P}_\Omega(\mathbf{X}_t), \quad \mathbf{X}^c_t = \mathbf{X}_t + \mathbf{E}, \quad \operatorname{diag}(\mathbf{U}) = \mathbf{0}, \end{aligned} \tag{12}$$

to simultaneously learn the innovation, detect the outliers, and complete the missing entries.

3. NUMERICAL TESTS

We compare performance of the proposed ESEM framework to that of static subspace clustering schemes and the state-of-the-art evolutionary clustering strategy of AFFECT [18]. AFFECT does not exploit the fact that the data points lie on a union of low dimensional subspaces We found that AFFECT performs poorly compared to other schemes (including static algorithms) when using the default RBF kernels as affinity matrices. Hence, in all experiments we use the representation learning methods introduced in Section 2 for all evolutionary algorithms including AFFECT.

3.1. Synthetic data: rotating random subspaces

Motivated by motion segmentation task [7] characterized by rotational and transitional motions, we consider the following scenario of rotating subspaces where we repeat each experiment for 150 trials.

At time $t = 1$, we construct $n = 10$ linear subspaces in $\mathbb{R}^D$, $D = 10$, each with dimension $d = 6$ by choosing their bases as the top d left singular vectors of a random Gaussian matrix in $\mathbb{R}^{D\times D}$. Then, we sample $N = 500$ data points, 50 from each subspace, by projecting random Gaussian vectors to the span of each subspace. Note that, in this setting, all the subspaces are distributed uniformly at random in the ambient space and all data points are uniformly distributed on the unit sphere of each subspace. According to the analysis in [6, 19, 20], this in turn implies that the subspace preserving property and performance of representation learning methods based on BP, OMP, and AOLS is similar. However, we intentionally construct relatively low number of data points as compared to the dimension of subspaces and the dimension of the ambient space; this creates a challenging setting for static subspace clustering. After constructing the subspaces at time $t = 1$, we evolve the subspaces by rotating their basis 45° around a random vector and project the data points $\mathbf{X}_1$ on the span of the rotated subspaces to obtain the samples $\mathbf{X}_2$. We continue this process for $T = 20$ time steps.

We next consider an experiment where in addition to rotation, at time $t = 6$ data points generated from subspace $\mathcal{S}_{10}$ are absorbed by subspace $\mathcal{S}_9$. That is, at $t = 6$ we project $\mathbf{X}_{\mathcal{S}_{10}5}$ to the span of $\mathcal{S}_9$. At time $t = 13$, these data points are separated from $\mathcal{S}_9$ and lie once again on $\mathcal{S}_{10}$. Hence, for $6 \leq t \leq 12$ the effective number of subspaces is $n = 9$ and there are 100 data points lying in $\mathcal{S}_9$.

The clustering accuracy results for these two experiments are illustrated in Fig. 1 (a) and Fig. 1 (b). As we can see, the static SSC-OMP algorithm performs poorly compared to ESEM and AFFECT. Since ESEM and AFFECT exploit the evolutionary behavior of the data points, after a few time steps their accuracy significantly increases. We further observe that as time passes the proposed ESEM framework achieve better accuracy than AFFECT as it exploit the self-expressiveness property of the data points in the representation learning process rather than simply combining current and prior rep-

Table 1: Comparison of static and evolutionary subspace clustering on real-time motion segmentation dataset. Best results for each row are in boldface fonts.

	Static			AFFECT			ESEM		
Learning method	error (%)	SI (%)	runtime (s)	error (%)	SI (%)	runtime (s)	error (%)	SI (%)	runtime (s)
BP	10.76	86.3	**46.16**	**9.86**	**87.8**	47.35	10.25	87.1	49.80
OMP	31.66	79.7	1.80	14.47	86.7	3.31	**7.97**	**87.4**	**0.82**
AOLS ($L = 1$)	16.27	84.4	4.08	9.27	**90.9**	5.39	**7.63**	90.5	**1.96**
AOLS ($L = 2$)	8.54	90.3	3.75	6.17	**93.1**	5.17	**5.67**	**93.1**	**1.82**
AOLS ($L = 3$)	6.97	92.3	3.14	5.92	93.9	4.28	**5.79**	**94.0**	**1.71**

resentations to enforce the self-expressiveness property.

Next, we investigate value of α, i.e., the smoothing parameter of ESEM and AFFECT in the aforementioned experiments to further assess which scheme more accurately captures the evolutionary nature of the subspaces. Fig. 1 (c) illustrates changes in the value of α over time, where in addition to the above two experiments we consider the scenario where subspaces are not rotating.[2] The figure indicates that the smoothing parameter of AFFECT is approximately 0.5 regardless of the changes happening in the evolution of the subspaces (see the curves at $\alpha = 0.5$ in the figure). Note that if the data is static, we expect $\alpha = 0$ for the ESEM framework and $\alpha = 1$ for AFFECT. In contrast to AFFECT's smoothing parameter, the values of α for the ESEM framework quickly converged to the anticipated levels (i.e. 0); note that we initialized α as 0.5. Fig. 1 (c) further suggest that the smoothing parameter of ESEM noticeably changes from the set values at times $t = 6, 13$. This indicates ESEM is able to detect the subspace changes taking place at $t = 6, 13$, while AFFECT fails to detect whether the subspaces are rotating.

These results suggest that not only ESEM improves the performance of static subspace clustering algorithms when the data is evolving, but it is superior to state-of-the-art evolutionary clustering strategies in the considered settings. Furthermore, smoothing parameter of ESEM is meaningful and interpretable, and properly adapts to the underlying evolutionary behavior of the subspaces.

3.2. Real-world data: Real-time motion segmentation

In real-time motion segmentation, the video sequence is often received as a stream of frames and it is desirable to identify multiple rigidly moving objects in the video [21, 22]. In the real-time setting, the t^{th} snapshot of $\mathbf{X}_t$ (a time interval consisting of multiple video frames) is of dimension $2F_t \times N_t$, where N_t is the number of trajectories at t^{th} time interval, F_t is the number of video frames received in t^{th} time interval, n_t is the number of rigid motions at t^{th} time interval, and $F = \sum_t F_t$ denotes the total number of frames. Real-time motion segmentation falls within the scope of evolutionary subspace clustering since the received video sequence is naturally characterized by temporal properties; at t^{th} time interval, the trajectories of n_t rigid motions lie in a union of n_t low-dimensional subspaces in $\mathbb{R}^{2F_t}$, each with the dimension of at most $d_t = 3n_t$ [23].

In contrast to the real-time motion segmentation, clustering in offline settings is performed on the entire sequence, i.e., $\mathbf{X} = [\mathbf{X}_1^\top, \ldots, \mathbf{X}_T^\top]^\top$. Therefore, one expects to achieve more accurate segmentation in the offline settings [4, 24]. However, offline motion segmentation cannot be used in scenarios where some motions vanish or new motions appear in the video, or in cases where a real-time motion segmentation solution is desired.

[2]For illustrative purposes, only curves corresponding to values of α for ESEM are descried in the legend of Fig. 1 (c).

We consider Hopkins 155 database [7] which consists of 155 video sequences with 2 or 3 motions in each video (corresponding to 2 or 3 low-dimensional subspaces). Unlike the majority of prior work (e.g. [4, 24]) that process this data set in an offline setting, we consider the following real-time scenario: each video is divided into T data matrices $\{\mathbf{X}_t\}_{t=1}^T$ such that $F_t \geq 2n$ for a video with n motions. Then, we apply PCA on $\mathbf{X}_t$ and take its top $D = 4n$ left singular vectors prior to representation learning.

We benchmark the proposed framework by comparing it to static subspace clustering and AFFECT; the former applies subspace clustering at each time step independently from the previous clustering results while the latter applies spectral clustering [2] on the weighted average of affinity matrices $\mathbf{A}_t$ and $\mathbf{A}_{t-1}$. Under the default choices for the affinity matrix, i.e. negative squared Euclidean distance or its exponential form, AFFECT achieves a clustering error of 21.96 and 44.15 percent, respectively, which as we present next is inferior even to the static subspace clustering algorithms. Hence, to fairly compare the performance of different evolutionary clustering strategies, we employ BP, OMP, and AOLS with $L = 1, 2, 3$ to learn the representations for all schemes, including AFFECT.

The clustering error, Silhouette index (SI), and running time of various schemes are presented in Table 1; there, the results are averaged over all sequences and all time intervals excluding the initial time interval $t = 1$. As we can see from the table, the proposed ESEM framework is superior to AFFECT in terms of clustering error for majority of the representation learning methods. In addition, ESEM achieves lower running time than static and AFFECT strategies for the case of using OMP and AOLS as the representation learning methods and is more suitable for online applications.

4. CONCLUSION

We formulated the problem of evolutionary subspace clustering that is concerned with organizing a collection of data points that lie on a union of low-dimensional temporally evolving subspaces. By relying on the self-expressiveness property, we proposed a non-convex optimization framework that enables learning parsimonious representations of data points at each time step while taking into account representations from the preceding time step by introducing an innovation representation matrix as well as a smoothing parameter that are updated alternatively. Extensive studies on synthetic and real-world datasets illustrated superiority of the proposed framework over existing methods.

5. REFERENCES

[1] René Vidal, "Subspace clustering," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 52–68, Mar. 2011.

[2] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al., "On spectral clustering: Analysis and an algorithm," in *the Advances in Neural Information Processing Systems (NIPS)*, 2001, vol. 14, pp. 849–856.

[3] Ehsan Elhamifar and René Vidal, "Sparse subspace clustering," in *Conf. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 2790–2797.

[4] Ehsan Elhamifar and Rene Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2765–2781, Nov. 2013.

[5] Eva L Dyer, Aswin C Sankaranarayanan, and Richard G Baraniuk, "Greedy feature selection for subspace clustering," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2487–2517, Sep. 2013.

[6] Chong You, Daniel Robinson, and René Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *Conf. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 3918–3927.

[7] Roberto Tron and René Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," in *Conf. Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2007, pp. 1–8.

[8] Jack Kiefer, "Sequential minimax search for a maximum," *the American Mathematical Society*, vol. 4, no. 3, pp. 502–506, Sep. 1953.

[9] David L Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[10] Abolfazl Hashemi and Haris Vikalo, "Accelerated sparse subspace clustering," *arXiv preprint arXiv:1711.00126*, Oct. 2017.

[11] Scott Shaobing Chen, David L Donoho, and Michael A Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, Feb. 2001.

[12] Robert Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, Jan. 1996.

[13] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Asilomar Conf. Signals, Syst. and Computers*. IEEE, 1993, pp. 40–44.

[14] Sheng Chen, Stephen A Billings, and Wan Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. Journal of Control*, vol. 50, no. 5, pp. 1873–1896, Nov. 1989.

[15] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[16] Abolfazl Hashemi and Haris Vikalo, "Accelerated orthogonal least-squares for large-scale sparse reconstruction," *Digital Signal Process.*, vol. 82, pp. 91–105, Nov. 2018.

[17] Abolfazl Hashemi and Haris Vikalo, "Evolutionary self-expressive models for subspace clustering," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 6, Dec. 2018.

[18] Kevin S Xu, Mark Kliger, and Alfred O Hero III, "Adaptive evolutionary clustering," *Data Mining and Knowledge Discovery*, vol. 28, no. 2, pp. 304–336, Mar. 2014.

[19] Mahdi Soltanolkotabi and Emmanuel J Candes, "A geometric analysis of subspace clustering with outliers," *The Annals of Statistics*, pp. 2195–2238, Aug. 2012.

[20] Mahdi Soltanolkotabi, Ehsan Elhamifar, and Emmanuel J Candes, "Robust subspace clustering," *The Annals of Statistics*, vol. 42, no. 2, pp. 669–699, Apr. 2014.

[21] Stephen M Smith and John M Brady, "ASSET-2: Real-time motion segmentation and shape tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 814–820, Aug. 1995.

[22] Robert T Collins, Yanxi Liu, and Marius Leordeanu, "Online selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1631–1643, Aug. 2005.

[23] Carlo Tomasi and Takeo Kanade, "Shape and motion from image streams under orthography: A factorization method," *Int. Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, Nov. 1992.

[24] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.