K-Subspaces for Sequential Data

Wubin Sheng and John Lipor
Department of Electrical and Computer Engineering
Portland State University
{wubin,lipor}@pdx.edu

***Abstract*—We study the problem of clustering high-dimensional temporal data such as video sequences of human motion, where points that arrive sequentially in time are likely to belong to the same cluster. State-of-the-art approaches to this problem rely on the union-of-subspaces model, where points lie near one of K unknown low-dimensional subspaces. We propose the first approach to sequential subspace clustering based on the popular K-Subspaces (KSS) formulation, which we refer to as Temporal K-Subspaces (TKSS). We show how sequential information can be incorporated into the KSS problem and provide an efficient algorithm for approximate minimization of the resulting cost function, proving convergence to a local minimum. Results on benchmark datasets show that TKSS achieves state-of-the-art performance, obtaining an accuracy increase of over 10% compared to existing methods.**

I. Introduction

In recent years, there has been an explosion of algorithms capable of efficiently clustering high-dimensional data. In contrast to earlier approaches that involve dimensionality reduction followed by clustering, algorithms for *subspace clustering* seek to simultaneously learn the intrinsic low-dimensional structure as well as the clusters themselves [1]. These methods assume the data lie on a union of low-rank subspaces, each corresponding to a distinct cluster/class, and have achieved excellent performance on computer vision tasks including face recognition, object tracking, and motion segmentation [2]–[5].

While existing subspace clustering methods adequately incorporate low-dimensional structure into the clustering process, most ignore additional structure in the data. For example, when performing human motion segmentation from video, the goal is to cluster video sequences so that frames containing the gesture are grouped together [6] (see Fig. 1 for example gestures). In this case, the data have a meaningful ordering, and sequential frames are more likely to belong to the same cluster.

To address such problems, several methods have been developed to perform *temporal subspace clustering*, i.e., subspace clustering in light of the temporal structure in the data. These function by learning an encoding of the data such that similar encodings are obtained (a) for points lying in the same subspace and (b) for those that are temporally similar. The first goal is typically accomplished by leveraging the self-expressiveness property, which states that points belonging to the same subspace are more easily encoded by other points within the same subspace, and is the basis for numerous subspace clustering algorithms [2], [8]–[18]. The second goal is accomplished through a variety of regularizers based on enforcing structure in the learned coding matrix [19], [20], applying Laplacian regularization [6], [21], and block-diagonal regularization [22]. While these have achieved strong results on motion segmentation datasets, they all rely on the self-expressiveness formulation laid out in [2], [8], [9]. More recently, authors have shown that the simple K-Subspaces (KSS) algorithm outperforms encoding-based approaches when initialized properly [23]–[25]. Further, methods based on KSS scale linearly in the number of data points, making them amenable to large-scale clustering.

Fig. 1: Example gestures from the Ballet dataset.

In this paper, we propose a novel approach to temporal subspace clustering based on the KSS algorithm. Like KSS (and K-means), our algorithm alternates between a subspace learning step and clustering step. We show that the sequential structure in the data can be easily incorporated into both steps, and that the resulting algorithm converges to a local minimum. Further, we propose a simple and natural method for initializing our algorithm. We show that our method outperforms the state-of-the-art on benchmark motion segmentation datasets while maintaining a low computational cost.

II. Problem Formulation & Related Work

Consider a collection of N vectors $x_1, \ldots, x_N \in \mathbb{R}^D$, and let $X \in \mathbb{R}^{D \times N}$ be the matrix obtained by stacking these vectors as columns. The goal of subspace clustering is to label points in an unknown union of K subspaces according to nearest subspace. The union-of-subspaces model assumes that if x_i lies in cluster k, then it can be approximated as $x_i = U_k z_i$, where $U_k \in \mathbb{R}^{D \times d}$ has $d \ll D$ orthonormal columns that span the corresponding d-dimensional subspace, and $z_i \in \mathbb{R}^d$ denotes the low-dimensional representation of x_i.

In temporal clustering, we assume that the points are ordered such that sequential points are more likely to belong to the same cluster. When this assumption holds, we write

$X = \left[X^{(1)}\ X^{(2)}\ \ldots\ X^{(K)}\right]$, where $X^{(k)} \in \mathbb{R}^{D \times N_k}$ denotes the matrix containing the N_k points belonging to cluster k.

Subspace clustering has been a topic of increasing interest since the initial formulations described in [3], with recent approaches obtaining strong theoretical guarantees [24]–[27] and state-of-the-art performance on many image clustering tasks while maintaining relatively low computational complexity [15], [18], [23].

Existing methods for temporal subspace clustering rely on the self-expressiveness property described above. In [19], the authors introduce Ordered Subspace Clustering (OSC), which follows the self-expressiveness formulation with an additional penalty to encourage sequential points to have the same coefficients in the coding matrix Z. In [22], the authors add an additional regularizer to OSC to encourage block diagonal structure in Z, yielding further improvements. The Temporal Subspace Clustering (TSC) algorithm [21] extends this approach in two ways. First, temporal regularization is induced by Laplacian regularization, which is more general and allows the algorithm to encourage multiple sequential points to be clustered together. Second, instead of regressing the points using the data vectors themselves, a dictionary is learned simultaneously, allowing for a more flexible representation. This method outperforms OSC significantly, and Laplacian regularization continues to be the basis for state-of-the-art methods in temporal subspace clustering. A Bayesian approach is considered in [28]; this further improves on TSC but requires the appropriate selection of prior distributions and has a high computational complexity. The work of [20] extends the Low-Rank Representation algorithm [8] to include temporal structure and presents an efficient optimization method for solving the resulting formulation, though these results lag behind those of TSC. To the best of the authors' knowledge, the current state-of-the-art in unsupervised temporal subspace clustering is achieved by the Graph Constrained Data Representation (GCDR) algorithm [6]. Like TSC, this algorithm learns both a dictionary and a representation matrix and relies on Laplacian regularization to encourage sequential clustering. The key addition is that GCDR includes an additional graph penalization that encourages the pairwise similarities between the true data points (in X) and their learned approximations to be similar. The authors demonstrate through an ablation study that this penalization has a significant impact on clustering performance.

While these methods have made significant advances on the problem of temporal subspace clustering, they all rely on ADMM or similar alternating optimization techniques, followed by the Normalized Cut algorithm, which can incur a large computational cost. Further, they are exclusively variations on the self-expressiveness formulation, whose performance lags behind KSS-type methods for very close subspaces [24]. Hence, there is a need to consider other problem formulations that may reduce computation time and succeed where existing methods fall short.

Recent work has considered a more direct approach to the subspace clustering problem via variations on the K-Subspaces algorithm [29]–[31], which seeks to solve

$$\min_{\mathcal{C},\mathcal{U}} \sum_{k=1}^{K} \sum_{i \in c_k} \left\|x_i - U_k U_k^T x_i\right\|_2^2, \tag{1}$$

where c_k denotes the indices of points belonging to the kth cluster, $\mathcal{C} = \{c_1, \ldots, c_K\}$ denotes the set of estimated clusters, and $\mathcal{U} = \{U_1, \ldots, U_K\}$ denotes the corresponding set of orthonormal subspace bases. Beginning with an initial set of subspace bases, the KSS algorithm alternates between clustering via nearest subspace and learning the subspace corresponding to each cluster via PCA. Although (1) is NP-hard even to approximate within any finite factor [32], it has been shown to achieve strong performance under certain initialization conditions [23]–[25], [32].

III. Temporal K-Subspaces

In this section, we present our approach to temporal subspace clustering, which we refer to as *Temporal K-Subspaces* (TKSS). Like KSS, TKSS alternates between a subspace learning step and a clustering step and maintains a computational complexity that is linear in the number of data points. We discuss the local convergence of the proposed approach and present a method of initializing TKSS that performs well on sequential data.

We wish to perform subspace clustering while encouraging temporally sequential points to be clustered together. Define the KSS loss for a single point x_i assigned to the subspace spanned by U_k to be

$$\mathcal{L}_{\text{KSS}}(x_i, U_k) = \left\|x_i - U_k U_k^T x_i\right\|_2^2.$$

As stated above, the KSS loss encourages each point to be near its assigned subspace. Next, we wish to encourage that each x_i is clustered together with its s nearest sequential neighbors. Let $\mathcal{N}_i$ be the indices of sequential neighbors of the ith point

$$\mathcal{N}_i = \left\{ j \in [N] : |i - j| \leq \frac{s}{2} \right\}.$$

We then define the sequential loss for the point x_i as the sum of distances from each sequential neighbor to the subspace defined by U_k

$$\mathcal{L}_{\text{Seq}}(x_i, U_k) = \sum_{j \in \mathcal{N}_i} \left\|x_j - U_k U_k^T x_j\right\|_2^2. \tag{2}$$

With these definitions in mind, we define the TKSS cost function as the weighted combination of KSS loss and sequential loss

$$\min_{\mathcal{C},\mathcal{U}} \sum_{k=1}^{K} \sum_{i \in c_k} \mathcal{L}_{\text{KSS}}(x_i, U_k) + \lambda \mathcal{L}_{\text{Seq}}(x_i, U_k). \tag{3}$$

Below, we show that this cost can be approximately minimized via an alternating procedure similar to KSS. Finally, we note that in the case where $\lambda = 0$, the above reduces to the original KSS algorithm.

Algorithm 1 Temporal K-Subspaces (TKSS)

Input: dataset X, number of subspaces K, subspace dimension d, sequential weight parameter λ, number of sequential neighbors s

Output: subspace bases $\mathcal{U} = \{U_1, \ldots, U_K\}$, cluster estimates $\mathcal{C} = \{c_1, \ldots, c_K\}$

```
 1: initialize clusters $c_1, \ldots, c_k$
 2: while not converged do
 3:     for $k = 1, \ldots, K$ do
 4:         form weight matrix $W^{(k)}$ according to (6)
 5:         $U_k \leftarrow d$ principal singular vectors of matrix $XW^{(k)}$
 6:     end for
 7:     assign clusters $c_1, \cdots, c_K$ according to (8)
 8: end while
 9: $\mathcal{U} \leftarrow \{U_1, \ldots, U_K\}$
10: $\mathcal{C} \leftarrow \{c_1, \ldots, c_K\}$
11: return $\mathcal{U}$ and $\mathcal{C}$
```

A. Approximate Minimization of the TKSS Cost

The TKSS cost (3) is approximately minimized in an alternating fashion, where the subspace learning step is efficiently solved using the singular value decomposition (SVD), and points are assigned to clusters in order to minimize the cost at each iteration.

We first describe the subspace estimation step, assuming an initial set of clusters are obtained (we describe one such initialization in Sec. III-C below). We first define $n_k(j)$ to be the number of times the point x_j appears as a sequential neighbor of a point in the kth cluster, i.e.,

$$n_k(j) = |\{i \in c_k : j \in \mathcal{N}_i\}|, \tag{4}$$

where $|S|$ denotes the cardinality of the set S. Now note that for a candidate basis U, the TKSS cost for the kth cluster can be written as

$$\sum_{i \in c_k} \left(\left\| x_i - UU^T x_i \right\|_2^2 + \lambda \sum_{j \in \mathcal{N}_i} \left\| x_j - UU^T x_j \right\|_2^2 \right)$$
$$= \sum_{i=1}^{N} \left\| x_i - UU^T x_i \right\|_2^2 \left(\mathbb{1}\{i \in c_k\} + \lambda n_k(i) \right), \tag{5}$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function. From (5), we see that the TKSS cost applies a weight to each point that depends on whether that point (a) lies in the corresponding cluster or (b) is a sequential neighbor of one or more points in the cluster. Further, we see that points are weighted more heavily if they are sequential neighbors of multiple points in the cluster. To minimize (5) over U, define the diagonal weight matrix $W^{(k)} \in \mathbb{R}^{N \times N}$ with diagonal elements

$$w_{ii}^{(k)} = \sqrt{\mathbb{1}\{i \in c_k\} + \lambda n_k(i)}. \tag{6}$$

The subspace estimation step is then equivalent to solving

$$U_k = \underset{\substack{U \in \mathbb{R}^{D \times d} \\ U^T U = I_d}}{\arg\min} \left\| XW^{(k)} - UU^T XW^{(k)} \right\|_F^2, \tag{7}$$

for which the solution corresponds to the left singular vectors of the matrix $XW^{(k)}$ corresponding to the largest d singular values.

After the subspace learning step, points are assigned to clusters in order to minimize the TKSS cost (3). This is done by evaluating the weighted combination of residual from the point to a given subspace as well as the residuals of its sequential neighbors, so that the kth cluster is

$$c_k = \left\{ i \in [N] : k = \arg\min_{l} \mathcal{L}_{\text{KSS}}(x_i, U_l) + \lambda \mathcal{L}_{\text{Seq}}(x_i, U_l) \right\}. \tag{8}$$

The algorithm then alternates between subspace learning and cluster assignment until convergence (guaranteed below). Pseudocode for the TKSS algorithm is given in Alg. 1.

The computational complexity of TKSS is dominated by the subspace learning step, which requires a SVD on the $D \times N$ matrix $XW^{(k)}$. However, since we only require the top d singular vectors, this procedure has computational complexity $O(NDd)$. For each iteration of TKSS, we perform one such SVD per class, so in the case where we run T iterations, the computational complexity of our method is $O(TKNDd)$, making it scale linearly with the number of data points. For comparison, TSC and its variants have a computational complexity of $O(TNr^2)$, where T is the number of ADMM iterations and r is the number of vectors in the learned dictionary. While this scales linearly with N, TSC also requires the use of the Normalized Cut algorithm to obtain clusters, which has a computational complexity that is quadratic in N.

B. Local Convergence of TKSS

The optimization problem (3) is non-convex, with the first global convergence results of similar problems appearing only very recently in [25]. Here we observe that the proposed TKSS algorithm converges to a local minimum. The proof follows in a fashion very similar to that of [29]. Note that TKSS proceeds by alternating between a subspace learning step and a cluster assignment step. During the subspace learning step, the resulting subspace is the global optimum of (5). Hence, this step cannot increase the overall objective. Similarly, the cluster assignment step assigns each point to the subspace that minimizes the distance to it and its sequential neighbors; this again cannot increase the overall objective. Since there are a finite number of ways the points in X can be assigned, and since the objective function (3) is bounded below by zero, the TKSS algorithm must terminate at some clustering that is locally optimal.

C. Initialization

As observed in [23]–[25], obtaining a good initial set of subspaces (or clusters) is key to achieving strong performance with KSS-type algorithms. In [25], the authors show that a simple procedure that involves performing spectral clustering on the Gram matrix of the data yields correct clustering of the data under certain assumptions. In our setting, we

Algorithm	Keck	Weizmann	MAD	UT	Ballet
OSC	49.1	38.3	24.4	75.3	31.6
TSC	49.3	75.8	75.7	84.1	50.3
GCDR	78.6	85.0	83.0	87.0	(not reported)
TKSS	88.9	83.5	84.5	83.0	73.1

TABLE I: Clustering accuracy (%) on benchmark human motion segmentation datasets. The proposed TKSS algorithm is among the top two performers for all but the UT dataset.

wish to begin with an initialization that encourages sequential neighbors to be clustered together. With this in mind, we follow the simple approach of dividing the dataset into K balanced clusters of sequential points. For example, for a dataset with $K = 2$ clusters and $N = 100$ points, the initial clusters would be $c_1 = \{1, \ldots, 50\}$ and $c_2 = \{51, \ldots, 100\}$. We show in Sec. IV that this approach obtains state-of-the-art performance on benchmark datasets, even though these datasets are not balanced.

IV. Empirical Results

In this section, we demonstrate the strong performance of TKSS on human motion segmentation datasets commonly used as benchmarks for temporal subspace clustering. We compare the clustering accuracy of TKSS with OSC [19], TSC [22], and GCDR [6]. For OSC and TSC, we use the code provided by the authors. Unless otherwise stated, for OSC, TSC, and TKSS, we perform hyperparameter selection via the Optuna framework [33] and report the best performance among 50 hyperparameter configurations. For GCDR, we compare only to the reported values in [6], since author code is not yet available.

We consider the Keck [34], Weizmann [35], MAD [36], UT [37], and Ballet [7] datasets. Each dataset consists of video sequences of human subjects performing various gestures, with further details reported in [38]. These datasets contain between 440 - 2048 total points with the number of clusters ranging from 7-10. For the Keck, Weizmann, MAD, and UT datasets, we use the data provided by the authors of [38], extracting HoG features [39] to obtain 324-dimensional feature vectors from each frame, then concatenate all frames into a single video sequence. For the Ballet dataset, we follow the procedure of [28], obtaining 300-dimensional feature vectors via Orthogonal Matching Pursuit [40].

The resulting clustering accuracy for each dataset is given in Table I. We first note that the accuracies for OSC and TSC are sometimes significantly higher than those reported elsewhere, due to our more thorough selection of hyperparameters. Selecting hyperparameters in an unsupervised manner is considered in [41], and these results highlight the importance of parameter selection in comparing algorithms for subspace clustering. From Table I, we see that TKSS is among the top two performers for all but the UT dataset, though strong performance is still achieved in this case. On the Keck dataset, TKSS achieves an accuracy of 88.9%, outperforming the closest competitor (GCDR) by over 10%. On the Ballet dataset, TKSS outperforms TSC by over 20%,

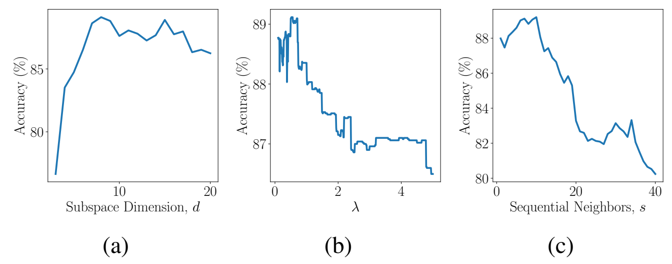

Fig. 2: Sensitivity analysis on Keck dataset. Accuracy as a function of (a) subspace dimension, (b) weight of sequential loss λ, and (c) number of sequential neighbors s.

indicating a dramatic increase over the state-of-the-art. The Keck and Ballet datasets are those with the largest number of points N. This conforms to previous observations that KSS-type algorithms perform best when there are many points per subspace, whereas algorithms based on the self-expressiveness formulation achieve stronger performance when there are few points per class [24]. Hence, we see that TKSS either performs on par with the state-of-the-art or obtains a dramatic performance improvement in the case of large datasets.

Given the difficulty of tuning hyperparameters, we next investigate the sensitivity of TKSS to the subspace dimension d, weight of sequential loss λ, and number of sequential neighbors s. Fig. 2 shows the accuracy as a function of these parameters on the Keck dataset, varying each parameter while keeping the other two fixed to their optimal values. The figure shows that TKSS exhibits stable performance over a wide range of parameter values. The number of sequential neighbors has the greatest impact on performance, with $s \leq 10$ resulting in stable performance, and an accuracy reduction of nearly 10% when too many sequential neighbors are included. Although not shown due to lack of space, similar results hold for the other datasets considered.

V. Conclusions & Future Work

In this work, we considered the problem of subspace clustering on sequential data, where sequentially-arriving points are likely to belong to the same cluster. We presented the first known approach to sequential clustering based on the popular K-Subspaces algorithm, proving that this algorithm converges to a local minimum and providing an initialization procedure that yields strong empirical performance. We demonstrated the efficacy of our proposed approach on benchmark datasets, as well as its robustness to the choice of hyperparameters.

The development of other initialization methods that do not assume perfectly balanced classes could lead to further performance benefits. Such methods could respect the sequential neighbors of a given point while also seeking out points that are likely to belong to the same subspace. Further, it would be of interest to apply the results of [25] to obtain global convergence results for TKSS. Doing so would require the development of an appropriate model for sequential UoS data, as well as the analysis of any proposed initialization method.

References

[1] R. Vidal, Y. Ma, and S. S. Sastry, *Generalized principal component analysis*. Springer, 2016, vol. 5.

[2] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

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

[4] T. Wu, "Graph regularized low-rank representation for submodule clustering," *Pattern Recognition*, vol. 100, p. 107145, 2020.

[5] Q. Li, Z. Sun, Z. Lin, R. He, and T. Tan, "Transformation invariant subspace clustering," *Pattern Recognition*, vol. 59, pp. 142–155, 2016.

[6] M. Dimiccoli, L. Garrido, G. Rodriguez-Corominas, and H. Wendt, "Graph constrained data representation learning for human motion segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1460–1469.

[7] Y. Wang and G. Mori, "Human action recognition by semilatent topic models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 10, pp. 1762–1774, 2009.

[8] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 663–670.

[9] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, and S. Yan, "Robust and efficient subspace segmentation via least squares regression," *Computer Vision–ECCV 2012*, pp. 347–360, 2012.

[10] Q. Li, W. Liu, and L. Li, "Affinity learning via a diffusion process for subspace clustering," *Pattern Recognition*, vol. 84, pp. 39–50, 2018.

[11] H. Chen, W. Wang, and X. Feng, "Structured sparse subspace clustering with within-cluster grouping," *Pattern Recognition*, vol. 83, pp. 107–118, 2018.

[12] N. Passalis and A. Tefas, "Discriminative clustering using regularized subspace learning," *Pattern Recognition*, vol. 96, p. 106982, 2019.

[13] Y. Sui, G. Wang, and L. Zhang, "Sparse subspace clustering via low-rank structure propagation," *Pattern Recognition*, vol. 95, pp. 261–271, 2019.

[14] Y. Chen, X. Xiao, and Y. Zhou, "Multi-view subspace clustering via simultaneously learning the representation tensor and affinity matrix," *Pattern Recognition*, vol. 106, p. 107441, 2020.

[15] S. Baek, G. Yoon, J. Song, and S. M. Yoon, "Deep self-representative subspace clustering network," *Pattern Recognition*, vol. 118, p. 108041, 2021.

[16] Y. Xu, S. Chen, J. Li, L. Luo, and J. Yang, "Learnable low-rank latent dictionary for subspace clustering," *Pattern Recognition*, vol. 120, p. 108142, 2021.

[17] X. Si, Q. Yin, X. Zhao, and L. Yao, "Consistent and diverse multi-view subspace clustering with structure constraint," *Pattern Recognition*, vol. 121, p. 108196, 2022.

[18] Y. Qin, H. Wu, J. Zhao, and G. Feng, "Enforced block diagonal subspace clustering with closed form solution," *Pattern Recognition*, p. 108791, 2022.

[19] S. Tierney, J. Gao, and Y. Guo, "Subspace clustering for sequential data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1019–1026.

[20] J. Zheng, P. Yang, G. Shen, S. Chen, and W. Zhang, "Enhanced low-rank constraint for temporal subspace clustering and its acceleration scheme," *Pattern Recognition*, vol. 111, p. 107678, 2021.

[21] S. Li, K. Li, and Y. Fu, "Temporal subspace clustering for human motion segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4453–4461.

[22] F. Wu, Y. Hu, J. Gao, Y. Sun, and B. Yin, "Ordered subspace clustering with block-diagonal priors," *IEEE transactions on cybernetics*, vol. 46, no. 12, pp. 3209–3219, 2015.

[23] C. Lane, B. Haeffele, and R. Vidal, "Adaptive online k-subspaces with cooperative re-initialization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[24] J. Lipor, D. Hong, Y. S. Tan, and L. Balzano, "Subspace clustering using ensembles of k-subspaces," *Information and Inference: A Journal of the IMA*, vol. 10, no. 1, pp. 73–107, 2021.

[25] P. Wang, H. Liu, A. M.-C. So, and L. Balzano, "Convergence and recovery guarantees of the k-subspaces method for subspace clustering," *arXiv preprint arXiv:2206.05553*, 2022.

[26] M. Soltanolkotabi and E. J. Candes, "A Geometric Analysis of Subspace Clustering with Outliers," *The Annals of Statistics*, vol. 40, no. 4, pp. 2195–2238, 2012.

[27] ——, "Robust Subspace Clustering," *The Annals of Statistics*, vol. 42, no. 2, pp. 669–699, 2014.

[28] B. Gholami and V. Pavlovic, "Probabilistic temporal subspace clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3066–3075.

[29] P. S. Bradley and O. L. Mangasarian, "k-Plane clustering," *Journal of Global Optimization*, vol. 16, pp. 23–32, 2000.

[30] P. Tseng, "Nearest q-flat to m points," *Journal of Optimization Theory and Applications*, vol. 105, no. 1, pp. 249–252, 2000.

[31] P. K. Agarwal and N. H. Mustafa, "K-means projective clustering," in *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2004, pp. 155–165.

[32] A. Gitlin, B. Tao, L. Balzano, and J. Lipor, "Improving k-subspaces via coherence pursuit," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 6, pp. 1575–1588, 2018.

[33] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.

[34] Z. Jiang, Z. Lin, and L. Davis, "Recognizing human actions by learning and matching shape-motion prototype trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 533–547, 2012.

[35] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[36] D. Huang, S. Yao, Y. Wang, and F. D. L. Torre, "Sequential max-margin event detectors," in *European conference on computer vision*. Springer, 2014, pp. 410–424.

[37] M. S. Ryoo and J. K. Aggarwal, "Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 1593–1600.

[38] L. Wang, Z. Ding, and Y. Fu, "Learning transferable subspace for human motion segmentation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 886–893.

[40] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[41] J. Lipor and L. Balzano, "Clustering quality metrics for subspace clustering," *Pattern Recognition*, vol. 104, p. 107328, 2020.