Pattern Recognition Letters
journal homepage: www.elsevier.com

Spatiotemporal *k*-means

Olga Dorabiala[a,**], Devavrat Vivek Dabke[b], Jennifer Webster[c], Nathan J. Kutz[a,d], Aleksandr Aravkin[a]

[a]*Department of Applied Mathematics, Seattle, 98195, WA, USA*
[b]*Level Ventures, New York, NY, USA*
[c]*Pacific Northwest National Laboratory, 1100 Dexter Ave N, Suite 500, Seattle 98109, WA, USA*
[d]*Department of Electrical and Computer Engineering, Seattle 98195, WA, USA*

ABSTRACT

Spatiotemporal data is increasingly available due to emerging sensor and data acquisition technologies that track moving objects. Spatiotemporal clustering addresses the need to efficiently discover patterns and trends in moving object behavior without human supervision. One application of interest is the discovery of moving clusters, where clusters have a static identity, but their location and content can change over time. We propose a two phase spatiotemporal clustering method called *spatiotemporal k-means* (ST*k*M) that is able to analyze the multi-scale relationships within spatiotemporal data. By optimizing an objective function that is unified over space and time, the method can track dynamic clusters at both short and long timescales with minimal parameter tuning and no post-processing. We begin by proposing a theoretical generating model for spatiotemporal data and prove the efficacy of ST*k*M in this setting. We then evaluate ST*k*M on a recently developed collective animal behavior benchmark dataset and show that ST*k*M outperforms baseline methods in the low-data limit, which is a critical regime of consideration in many emerging applications. Finally, we showcase how ST*k*M can be extended to more complex machine learning tasks, particularly unsupervised region of interest detection and tracking in videos.

1. Introduction

The widespread use of sensor and data acquisition technologies, including IOT, GPS, RFID, LIDAR, satellite, and cellular networks allows for, among other applications, the continuous monitoring of the positions of moving objects of interest. These technologies create rich spatiotemporal data that is found across many scientific and real-world domains including ecological studies of collective animal behavior, the surveillance of large groups of people for suspicious activity, and traffic management (Kalnis et al., 2005; Vieira et al., 2009; Jeung et al., 2008). Often, the data collected is large and unlabeled, motivating the development of unsupervised learning methods that can efficiently extract information about object behavior with no human supervision.

Clustering is one of the primary goals of unsupervised learning. As such, it has become a critical data mining tool for gaining insight from unlabeled data by grouping objects based on some similarity measure (Bishop & Nasrabadi, 2006; James et al., 2013). Spatial clustering refers to the analysis of static data with features that describe spatial location, while spatiotemporal clustering adds time as a feature, and algorithms have to consider both the spatial and the temporal neighbors of objects in order to extract useful knowledge (Birant & Kut, 2007). There are a handful of spatiotemporal clustering classes, some of which track events or object trajectories, but our focus is on moving clusters, where clusters have a static identity, but their location and content can change over time. The moving cluster problem is especially useful in applications where it is essential to know whether individuals form loose and temporary associations or stable, long-term ones. Applications such as surveillance, transportation, environmental and seismology studies, and mobile data analysis can be considered within this mathematical framework (Ansari et al., 2020).

The mathematical formulation for the moving cluster problem is significantly more challenging than for stationary clustering. Most approaches first cluster in space and then aggregate the results over time, as opposed to minimizing a unified objective function. It has been shown that this post-processing ap-

[**] Corresponding author:
e-mail: `OlgaD400@uw.edu` (Olga Dorabiala)

proach can lead to erroneous results (Chen et al., 2015). Further, the most popular approaches are built upon density-based clustering methods, which are sensitive to hyperparameter tuning and do not explicitly track cluster centers (Bhattacharjee & Mitra, 2021). Finally, while some existing methods operate well on large data, tracking objects over thousands of time steps or more, they exhibit poor performance in the low-data limit, when dynamics are being inferred from either a small number of individuals or over very short windows of time.

We propose a two phase spatiotemporal, unsupervised clustering method, *spatiotemporal k-means* (ST*k*M), for the moving cluster problem that addresses the aforementioned shortcomings. Phase 1 identifies the loose associations between objects by outputting an assignment for each point at every time step, with the flexibility for points to change clusters between time steps. The clustering objective function provides a unified formulation over space and time and less hyperparameter tuning compared to existing methods. It also provides the functionality to directly track cluster paths without post-processing, allowing ST*k*M is to identify long-term point behavior, even in a dynamic environment. Phase 2 can be optionally applied to the cluster assignment histories from Phase 1 to output stable, long-term associations. In fact, Phase 2 can be applied to any method that outputs an assignment for each point at every time step. The combination of Phase 1 and Phase 2 allows us to analyze the multi-scale relationships within spatiotemporal data. We introduce ST*k*M, theoretically demonstrate the efficacy of the algorithm, evaluate its performance against existing methods on the moving cluster problem, and highlight the use of ST*k*M for more sophisticated machine learning applications.

2. Related Work

Spatiotemporal data generally record an object state, an event, or a position in space, over a period of time. Spatiotemporal clustering can be divided into six classes: event clustering, geo-referenced data item clustering, geo-referenced time-series clustering, trajectory clustering, semantic-based trajectory data-mining, and moving clusters (Ansari et al., 2020). Some of the most prominent algorithms, such as ST-DBSCAN and ST-OPTICS belong to the second classification (Birant & Kut, 2007; Agrawal et al., 2016). Unfortunately, they require four and six input parameters, respectively, heavily influencing the quality of clusters, and they do not provide meaningful cluster centers for analysis. The hyper-parameter tuning of the ten aforementioned parameters becomes critically important for achieving reasonable performance.

The algorithms in this paper are concerned with the final classification scheme: moving clusters. A moving object is defined by a set of sequences $\langle id, \mathbf{x}, t \rangle$, where the variable *id* is the unique identifier for each point, t is time, and $\mathbf{x}$ is a vector whose components contain the spatial attributes, i.e. the x and y coordinates (Ansari et al., 2020). Moving clusters have identities (separate from *id* above) that do not change over time, although their positions and content may change. The prototypical example is a herd of animals, where individual animals can enter or leave the herd at any given time.

Most approaches to the moving cluster problem first cluster in space and then aggregate the results over time. Kalnis et. al proposed running DBSCAN at every time step and defined a moving cluster criteria to associate clusters in successive time steps (Kalnis et al., 2005). This approach was later extended to the discovery of convoys consisting of at least some points that exist near one another for a minimum number of consecutive time steps (Jeung et al., 2008). Other work identified flocks of objects that stay together for a given window of time (Vieira et al., 2009). The commonality between these approaches was a requirement for moving clusters to exist in some fixed number of consecutive time steps. In practice, points can split apart and come back together, motivating the proposal of swarms, where a minimum number of objects travel together for at least some proportion of time steps (Li et al., 2010). Contrastingly, Chen et. al proposed an extension of DBSCAN that incorporates a novel spatiotemporal distance function, where points' distances are their spatial distances from one another if they are temporal neighbors and zero otherwise (Chen et al., 2015). Their four step process performs even in the presence of noise and missing data, but, like ST-DBSCAN, requires extensive hyperparameter tuning.

Though substantial work has been done to develop various spatiotemporal clustering techniques, the performance of these methods is rarely compared against one another and implementations are not open source. Recognizing that there was no unified and commonly used experimental dataset and protocol, Cakmak et. al proposed a benchmark for detecting moving clusters in collective animal behavior (Cakmak et al., 2021). They generate realistic synthetic data with ground truth, and present state-of-the-art baseline methods. Their implemented algorithms extend spatial clustering methods by first assessing whether a data point is density reachable from another data point with respect to both space and time and then employing a splitting and merging process (Peca et al., 2012). Additionally, ST*k*M (based on a pre-print of this paper) has been extended to the more abstract metric case involving graphs (Dabke & Dorabiala, 2023b,a, 2024).

3. Spatiotemporal *k*-means

Drawing inspiration from approaches that define unique spatiotemporal distance metrics, we propose a clustering objective function that provides a unified formulation over space and time and predicts cluster membership for each point at every time step (Izakian et al., 2012; Chen et al., 2015). We build upon the *k*-means algorithm, so that cluster centers are explicitly tracked and there are fewer parameters to tune. In a single pass of Phase 1, without post-processing, point membership and dynamic cluster center paths are output. We provide an optional secondary phase that can extract stable, long-term clusters.

3.1. Phase 1: Loose, Temporary Associations

The first phase of our method captures loose associations, with points having the flexibility to change clusters. We propose a temporal extension of the *k*-means objective function. We focus on *k*-means, because of its simplicity, speed, and scalablity.

Also, unlike density-based methods, k-means explicitly identifies cluster centers, giving us the ability to directly track the movement of our k clusters. The objective is shown in (1).

$$\min_{\mathbf{C},\mathbf{W}} \sum_{i=1}^{N}\sum_{j=1}^{k}\sum_{t=1}^{T} w_{t,j,i}\|\mathbf{x}_{t,i}-\mathbf{c}_{t,j}\|^2+\lambda\|\mathbf{c}_{t,j}-\mathbf{c}_{t+1,j}\|^2 \quad \text{where } \mathbf{W}_{t,:,i}\in\Delta_1 \qquad (1)$$

The matrix $\mathbf{X} \in \mathbb{R}^{T\times m\times N}$ contains N data points and the matrix $\mathbf{C} \in \mathbb{R}^{T\times m\times k}$ contains k cluster centers, both of spatial dimension m being tracked over T time steps. The matrix $\mathbf{W} \in \mathbb{R}^{T\times k\times N}$ contains auxiliary weights that map the assignment of points to clusters over time. Instead of restricting the entries of $\mathbf{W}$ to the discrete set $\{0, 1\}$, we allow them to vary over the closed interval $[0, 1]$. This relaxation is used by fuzzy versions of k-means and gives the user a way to quantify the extent of each point's membership to a cluster (Nayak et al., 2015) . The second term in (1) associates cluster centers between time frames automatically, as opposed to through post processing, as in (Kalnis et al., 2005; Jeung et al., 2008; Vieira et al., 2009; Peca et al., 2012; Cakmak et al., 2021). Cluster centers maintain their identity because they are penalized for moving apart, where the parameter $\lambda \in [0, 1]$ controls the extent of the penalty. Objective (1) requires all points to exist at every time step. To ensure this criteria is satisfied, data can be divided into time intervals, missing spatial information in an interval can be augmented using interpolation, and intervals with multiple spatial coordinates can be reduced through averaging.

Problem (1) can be solved using alternating minimization. The centers are updated using the Gauss-Seidel step in (2), and unlike fuzzy versions of k-means, which update the weights with an explicit formula based on points' distances from cluster centers, we use Proximal Alternating Minimization (PAM), as shown in (3) (Nayak et al., 2015). PAM gives us control over how quickly weights are updated and can be thought of as a proximal regularization of the Guass-Seidel scheme (Attouch et al., 2010). PAM is guaranteed to converge as long as $d_k > 1.0$. In practice, we set $d_k = 1.1$.

$$\mathbf{c}_{t,j}^{k+1} = \frac{\sum_{i=1}^{n} w_{t,j,i}\mathbf{x}_{t,i} + n\lambda\mathbf{c}_{t+1,j}}{\sum_{i=1}^{n}(w_{t,j,i} + \lambda)} \qquad (2)$$

$$w_{t,j,i}^{k+1} = proj_{\Delta_1}\Big(w_{t,j,i} - \frac{1}{d_k}\|\mathbf{x}_{t,i} - \mathbf{c}_{t,j}\|^2\Big) \qquad (3)$$

Since both point membership and clusters are tracked throughout the clustering process we can directly visualize the paths of dynamic clusters, a feature that without post-processing is unavailable with any existing method. The top row of Figure 1 displays ground truth versus predicted cluster paths using Phase 1 of STkM on a synthetic dataset containing three long-term moving clusters. Though cluster paths are not identified perfectly because we do dynamic prediction on clusters with static membership, STkM is still able to pick up the general trends of cluster movement. Even in a dynamic environment, we do not completely lose information about long-term cluster behavior.

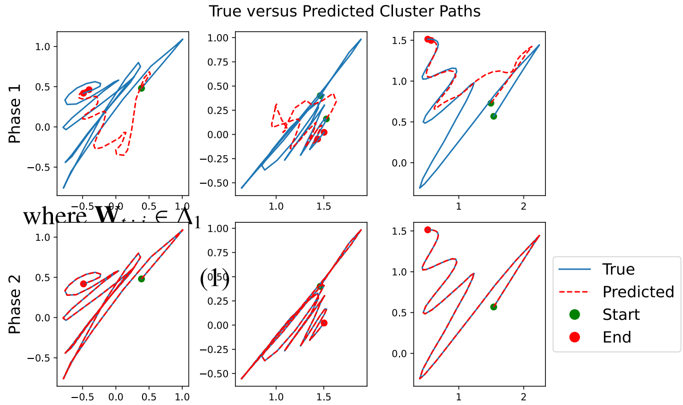


Fig. 1. True static versus predicted cluster paths from Phase 1 and Phase 2 of ST*K*M . After Phase 1, ST*k*M identifies general trends of cluster movement, even when allowing points to switch clusters over time. In Phase 2, ST*K*M correctly identifies the true static cluster paths.

3.2. Phase 2: Stable, Long-term Associations

Phase 2 of STKM identifies the long-lived associations between data points, and the output is a single assignment of static clusters containing points that have the most similar spatiotemporal characteristics. Because Phase 2 uses dynamic clusters to inform decisions about long-term behavior, the clusters predicted by Phase 2 are more accurate than methods that directly find static clusters. To run Phase 2 on the output of Phase 1, we first need to extract cluster assignment histories, which we define as the arg max over the rows of $\mathbf{W}$, so that the vector $\mathbf{a}_r \in \mathbb{R}^T$ contains the assignment of point r at each time t. We use Hamming distance, denoted as $\mathcal{H}(\mathbf{a}_r, \mathbf{a}_s)$, to quantify the extent of difference between two vectors $\mathbf{a}_r, \mathbf{a}_s$. Then the similarity can be defined as $\mathrm{sim}(\mathbf{a}_r, \mathbf{a}_s) = 1 - \frac{\mathcal{H}(\mathbf{a}_r,\mathbf{a}_s)}{T}$. We can create a similarity matrix $\mathbf{A}$, where $A_{r,s}$ contains the similarity between the cluster assignment histories of points r and s, and run agglomerative clustering on $\mathbf{A}$ to output k long-term clusters.

Row two of Figure 1, compares the predicted versus ground truth long-term cluster paths of the moving objects from the previous section, and we observe that Phase 2 identifies the paths perfectly. We can also combine results from Phase 1 and Phase 2 to gain insights about the multi-scale behavior of moving objects e.g. which long-term clusters are most stable, which points switch clusters most often, etc.

4. Theoretical Analysis

4.1. Overview

We define the *correlated random walk* model: given a collection of particles with each particle belonging to a unique cluster, the particles are performing random walks that are correlated within a cluster and uncorrelated without. We would like to analyze how STkM performs on this system. While cluster membership may rightfully change over time in spatiotemporal data, we make the assumption that each particle fits into a unique "correct" cluster in order to illustrate the utility of STkM.

4.2. Definitions

Let $T \in \mathbb{N}$ be end time for our simulation and k be the total number of clusters; each cluster has n_i particles (where $n_i \geq 1$) and the total number of particles is $n \triangleq n_1 + \cdots + n_k$, so $n \geq k$.

Let X_i^t represent the position of particle $i \in [n]$ at time $t \in \{0\} \cup [T]$ where[1], by construction, $X_i^t \in \mathbb{R}^d$. All elements in cluster 1 are indexed $\{1, \ldots, n_1\}$, those in cluster 2 are indexed $\{n_1 + 1, \ldots, n_1 + n_2\}$, and so on; $a\colon [n] \to [k]$ maps indices to their clusters (see Section Appendix A.2 for details).

We further construct our *displacements* Y_i^t. Let $W_i^t, Z_j^t \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$ where[2] $i \in [n], j \in [k]$. We can thus write $Y_i^t \triangleq \sqrt{q}\cdot W_i^t + \sqrt{p}\cdot Z_{a(i)}^t$ where $q \triangleq 1-p$. These displacements form a set of standard normal vectors that are independent across different timesteps and different clusters, but have correlation p within a cluster at the same time. Equivalently, $Y_i^t \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$ with the condition $\operatorname{Cor}\left(Y_i^s, Y_j^t\right) = p$ if $a(i) = a(j), s = t$ and 0 otherwise. Proposition Appendix A.1 shows equivalence.

Remark (Correlation is Covariance)**.** *Since the variance of each Y_i^s is* 1*, we know that* $\operatorname{Cor}\left(Y_i^s, Y_j^t\right) = \operatorname{Cov}\left(Y_i^s, Y_j^t\right)$

4.3. System Dynamics

To define the dynamics of this system, we let

$$X_i^t \triangleq \begin{cases} \mathbf{0} & t = 0 \\ X_i^{t-1} + Y_i^{t-1} & t > 0 \end{cases} \tag{4}$$

4.4. Key Results

First, ST*k*M works correctly over time in this context. In particular, points in the same cluster are more likely to be closer together and ST*k*M exactly optimizes for this case. We introduce Theorem 4.1 to explain this behavior.

Theorem 4.1. *In expectation, intracluster distances are smaller than intercluster distances.*

Proof. Following directly from Lemma Appendix A.3, observe that the correlation between two points in a cluster is strictly less than the correlation between two points in different clusters. □

Next, we establish a bound on closeness within a cluster. Theorem 4.2 bounds the total distance that a set of points within a cluster can drift. If a center is chosen within a cluster, then this theorem applies directly where q is simply[3] $1 - p$.

Theorem 4.2. *For $\epsilon > 0$, the probability that all particles in a cluster are within distance D of a chosen particle is at least $1-\epsilon$ when $D = \frac{1}{\epsilon} \cdot c \cdot n_i \sqrt{tq}$. The constant c depends on the ambient dimension, and n_i is the number of particles in the cluster.*

Proof. By Lemma Appendix A.3, the expected distance of a particle from any particle is $c \cdot \sqrt{tq}$, where c is a constant that depends on the ambient dimension[4]. By Markov's inequality[5], the probability that this distance is greater than D is $\frac{1}{D} \cdot c \cdot \sqrt{tq}$. By the Union Bound, the probability α that at least one particle is more than distance D away from a cluster center is at most $n_i \cdot \frac{1}{D} \cdot c \cdot \sqrt{tq}$. Let $D = n_i \cdot \frac{1}{\epsilon} \cdot c \cdot \sqrt{tq}$, where $\epsilon > 0$. By substitution, we see that $\alpha \leq \epsilon$. Finally, we note that by the law of total probability, the probability that no particles are more than distance D away from any point is simply $1 - \epsilon$. □

5. Experiments

5.1. Methodology

To experimentally validate the performance of STG*k*M, we use the benchmark dataset proposed by Cakmak et. al (Cakmak et al., 2021). Their benchmark is based on three collective animal behavior models and contains 3,600 spatiotemporal datasets of sizes ranging from 600 up to 520,000, where size is calculated as $T \times n$. The datasets track static clusters, where points do not change cluster membership over time. During our evaluation, we focus on the datasets that have size between 800 and 35,000, of which there are 1,034. We do so, because we are particularly interested in performance in the low-data domain, where either we have few objects or few time steps from which to infer behavior.

Cakmak et. al measure clustering quality with adjusted mutual information (AMI) score and report execution time for a handful of baseline methods. The methods all output dynamic clusters, but AMI compares the dynamic cluster assignments against a static ground truth. To avoid this mismatch, we compare the ground truth against stable clusters derived from the full assignment histories. To this end, we use Phase 2 of ST*k*M to extract long-term clusters, not only from Phase 1's output, but also from the baseline methods. Then we report what we refer to as long-term AMI, which compares the predicted versus ground truth static clusters. We divide our data into groups based on size (e.g. 800-3000, 3000-6000, etc.) and report results as the median and average of long-term AMI for each range of sizes. We note that in (Cakmak et al., 2021), during the cluster merging process, points that cannot be assigned to a cluster are given the same label, resulting in an erroneous association of unassigned points as a single cluster during the calculation of AMI. In order to avoid this interpretation, we give them all given unique labels during evaluation.

5.2. Parameter Selection

All of the baseline methods have at least four parameters that need to be defined: frame size, frame overlap, ϵ_1, and ϵ_2. These correspond to the number of time steps that belong to a single

[1] The notation $[c]$ represents the set $\{1, \ldots, c\} \subset \mathbb{N}$

[2] $\mathbf{0}$ is the 0 element (origin) of $\mathbb{R}^d$, $\mathbb{I}_d$ is the $d \times d$ identity matrix

[3] if the point is not within the cluster, then $q = 1$ because $p = 0$.

[4] This lemma applies here because we have selected a cluster, which all have the same correlation with a point, whether or not it is in the cluster.

[5] Markov's Inequality applies because the distance is non-negative and its expectation is well-defined.

frame, the number of time steps that frames overlap when associating clusters between frames, and the spatial and temporal distances that define whether a point is density reachable from the current one. All of the methods except for ST-DBSCAN also take as input the true number of clusters k. In their experiments, Cakmak et. al arbitrarily fix frame size to be 100 and frame overlap to be 10. All of the methods use the default value $\epsilon_1 = 0.50$, except for ST-DBSCAN, which searches for $\epsilon_1 \in [0.01, 0.05]$. Grid search is used to find the optimal remaining parameters that achieve the highest accuracy measure against the ground truth (Cakmak et al., 2021). In an unsupervised setting, one cannot tune parameters to maximize accuracy based on a ground truth. We argue that the performance of the baseline methods in Cakmak et. al is therefore unrealistic and avoid parameter tuning in our experiments.

In contrast, Phase 1 of our method requires only two parameters: λ, which controls the extent of the penalty that indirectly discourages points from switching clusters, and k, the true number of clusters. The parameter λ is confined to the range $[0, 1]$, and the meaning of its value is intuitive. We seek to create a similar, intuitive interpretation of the baseline methods' ϵ_2, the temporal distance a point is density reachable from the current one. We define $\epsilon_2 = \alpha t$, where t is the total number of time steps in the data and $\alpha \in [0, 1]$ is some given proportion. This formulation gives us a principled approach to choosing ϵ_2, as opposed to choosing a unique value for each dataset.

Because we know that the ground truth clusters do not allow points to switch clusters, we set both λ and α fairly high. We run all of the methods on each set of data with $\lambda, \alpha = [0.60, 0.80, 1.00]$. For the baseline methods, we fix the remaining parameters as follows: frame size = 100, frame overlap = 10, $\epsilon_1 = 0.50$ for all of the methods, except for ST-DBSCAN where $\epsilon = 0.05$, and k is set to the true number of clusters. Any other parameters in the baseline methods are set to their default values. Since we run each method three times using different parameters on each dataset, we obtain metrics for 3,102 runs of each method. We then report the aggregate of long-term AMI for each method on every range of dataset sizes.

5.3. Results

Figure 2 displays the performance of all baseline methods on the benchmark data in terms of long-term AMI. STkM, ST-Agglomerative, ST-KMeans, and ST-BIRCH score almost identically in terms of their median scores, but STkM maintains the highest averages over all datasets. As expected, as dataset size increases, more information can be extracted either due to more time steps or more point interactions, and the accuracy of the top methods increases. It is only in datasets under size 10,000 that we observe median scores noticeably smaller than 1.0. Across almost all sizes, the boxplots for STkM in Figure 2 have the tightest interquartile ranges, the shortest tails, and the most condensed outliers, demonstrating that STkM has the lowest variability and most consistent performance. This result implies that the short-term relationships detected by STkM are the most informative in identifying long-lived point relationships.

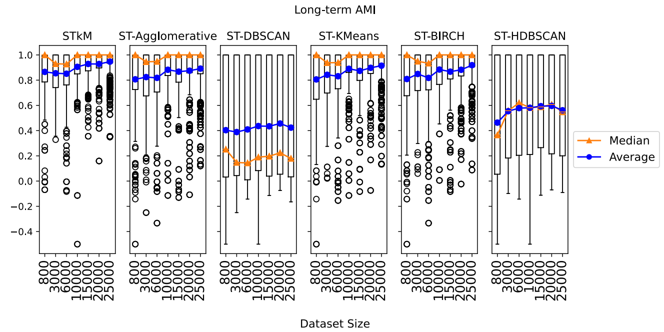
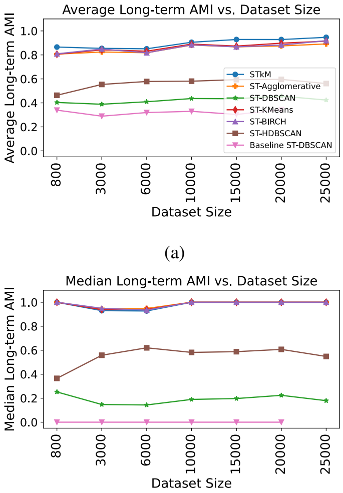


Fig. 2. Boxplots of long-term AMI scores for various methods over different dataset sizes. Median scores are shown in orange and average scores in blue. Boxplots for STkM have the top median and average scores, smallest interquartile ranges, shortest tails, and least dispersed outliers, demonstrating that STkM's performance is the best and most consistent.



(a)

(b)

Fig. 3. (a) Average long-term AMI trendlines. (b) Median long-term AMI trendlines. The top methods perform almost identically in terms of median scores, but STkM achieves the highest average long-term AMIs on datasets of all sizes. All methods that use short-term information to inform long-term predictions perform better than Baseline ST-DBSCAN.

Figure 3 provides a closeup of average and median long-term AMI trendlines, and also includes the results of baseline ST-DBSCAN, which is a popular spatiotemporal clustering method that produces solely static cluster assignments. The remaining methods, which utilize Phase 2 of STkM to generate static cluster assignments, outperform baseline ST-DBSCAN, suggesting that a two phase approach that uses short-term behavior to inform long-term relationships, captures moving object behavior more accurately. Overall, STkM achieves the highest long-term AMI on 70% of datasets. Table 1 shows the long-term AMI scores for each of the tested methods averaged over all 3,102 runs, and we observe that STkM achieves the highest score. Although STkM demonstrably outputs more informative moving cluster labels and more accurate long-term cluster labels, the trade-off is it's runtime. STkM runs slowest and scales worst out of all methods tested, as seen in Figure 4. An improvement could come from decreasing the number of iterations in STkM.

Table 1. Average Long-term AMIs for all methods over all datasets.

	ST*k*M	ST-Agglomerative	ST-DBSCAN	ST-KMeans	ST-BIRCH	ST-HDBSCAN
Average Long-term AMI	**.90**	.86	.42	.87	.87	.57

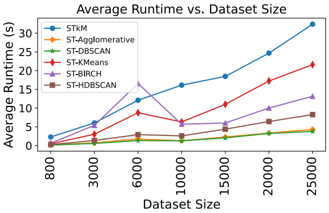


Fig. 4. Average runtime versus dataset size. ST*k*M scales poorest in terms of runtime, compared to other methods.

5.4. *Machine Learning*

Thus far, we have shown ST*k*M's ability to cluster moving objects of the simplest kind: points traveling in two dimensions. However, moving objects can be much more complex; they can be any evolving, high-dimensional feature vectors. Since ST*k*M sets the benchmark in the two dimensional case, we seek to apply it to more interesting machine learning applications, such as region of interest (ROI) detection and tracking in videos.

Variants of *k*-means have successfully been applied in the context of image segmentation in literature dating all the way back to the 1980s (Coleman & Andrews, 1979; Pappas & Jayant, 1989). Over the years, approaches have become more sophisticated, experimenting with pre- and post-processing, ensembling, and the integration of clustering objectives into the functions being optimized by neural networks (Dhanachandra et al., 2015; Ji et al., 2019; Kim et al., 2020). Extracting ROIs in videos is much more challenging. Most methods use deep learning to extract ROIs on a frame-by-frame basis and aggregate them over time, as in (Wu et al., 2019). However, the aggregation is done over consecutive or short time windows, thereby failing to capture a global perspective (Lu et al., 2019). This is where we believe STG*k*M could be of value.

One approach for region of interest tracking in videos, would be to directly apply ST*k*M to the pixels in a video, where each pixel has a feature vector that captures evolving RGB channels. Unfortunately, ST*k*M will not scale well to hundreds of thousands of points, and three features may not be discriminative enough to generate meaningful clusters. Our approach is to instead use a pre-trained CNN on each video frame to generate "super-pixels" that summarize the important features in each grid box, simultaneously enriching the feature space and diminishing dataset size. The process is described formally below.

5.4.1. *Lifting an image model to video*

For a given w, length l, and number of channels d, the space of images is $\mathbb{R}^{w\times l\times d}$. In this setting, we will assume that we have an oracle neural network that maps images to some latent space. In particular, given latent dimension n, we will assume the existence of a neural network N such that $N\colon \mathbb{R}^{w\times l\times d} \to \mathbb{R}^{w\times l\times n}$. Given a *movie* $(x^t)_{t\in[T]}$ where $x^t \in \mathbb{R}^{w\times l\times d}, T \in \mathbb{N}$, we can construct a set of spatiotemporal points. Namely, $p_i^t \in \mathbb{R}^n$ such that $p_i^t \triangleq N\left(x^t\right)_{\sigma(i)}$, where σ is a bijection for re-indexing such that $\sigma\colon [w\cdot l] \to [w]\times[l]$, e.g., $i \mapsto (\lceil i/w \rceil, i \mod l)$. With this set of points, we use ST*k*M to cluster the movie pixels.

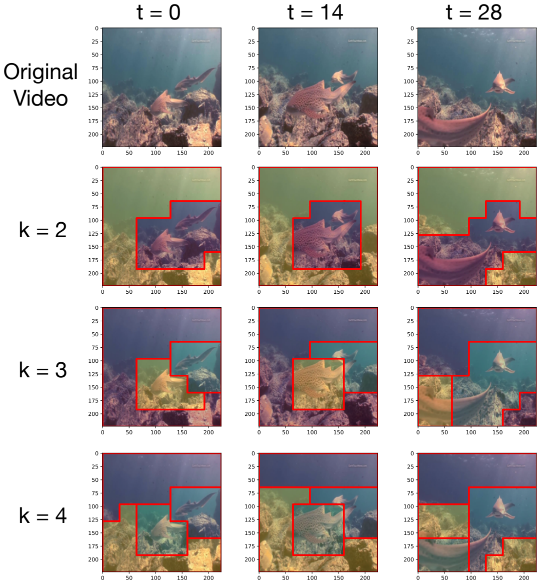


Fig. 5. Output of our region of interest detection pipeline on a video of swimming fish using varied values of k in ST*k*M. We achieve background/foreground separation with $k = 2$, separate the fish and the water when $k = 3$, and cluster coral, water, and fish seperately when $k = 4$.

Figure 5 shows the output of our proposed pipeline on a video of swimming fish. We run each frame of the video through a pre-trained ResNet 50, with the final layer removed. The output is a 7x7 grid of "super-pixels" that capture the important features in each grid box. We flatten the grids and run Phase 1 of ST*k*M on the resulting vectors. When $k = 2$, we achieve foreground/background separation, assigning the fish and the background to different clusters. When $k = 3$, the individual fish are separated from each other and the background. When $k = 4$, the clusters correspond to coral, open water, and individual fish. The cluster bounding boxes are not precise; particularly when $k = 3$ and $k = 4$, small parts of the fish are separated into different clusters. It may be worth experimenting with different CNN backbones or "super-pixel" granularity, but we leave a principled study and evaluation of ST*k*M for region of interest detection for future work. For now, we emphasize the potential of ST*k*M to be used for this task with no specialized transfer learning, labeled data, or training time.

6. Conclusion

We demonstrate that ST*k*M, an unsupuervised two phase spatiotemporal clustering method, is able to capture the multi-scale behavior of moving object data. Phase 1 returns an assignment for each point at every iteration, and provides us the unique ability to directly track cluster centers without any post-processing. This phase minimizes an objective function, that unlike existing methods, is unified in both space and time and requires many fewer parameters to run. Phase 2 can be optionally applied to classify each point into a single long-term cluster. Because Phase 2 infers long-term relationships from short-term ones, Phase 2 results in more accurate static clusters compared to methods that provide exclusively static clusters. The combination of both phases allows us to draw conclusions about the relationships between both points and clusters.

We demonstrate the competitiveness of ST*k*M against existing spatiotemporal clustering methods on a benchmark dataset proposed by Cakmak et. al (Cakmak et al., 2021). All algorithms output dynamic clusters, so we use Phase 2 of ST*k*M to translate them into static clusters for comparison against the ground truth. We show that ST*k*M performs best and most consistently in terms of average and median long-term AMI over all datasets, suggesting that the short-term relationships predicted by ST*k*M are more informative than those of the baseline methods. The tradeoff in using ST*k*M is a slower runtime.

Overall, ST*k*M demonstrably outperforms existing methods on the moving cluster problem. As such, we explore how ST*k*M can be used for more complex machine learning applications and provide evidence that it has the potential to be used as part of an ensemble for region of interest detection in videos. In the future, we intend to explore robust extensions of ST*k*M for handling noise, approaches to estimating the number of clusters k, and further study applications of ST*k*M for computer vision and other more complex machine learning tasks. In a parallel line of work, we have already extended ST*k*M to the more abstract metric case involving graphs (Dabke & Dorabiala, 2023b,a, 2024). With ever increasing information from broad applications such as surveillance, transportation, environmental studies, and mobile data analysis, ST*k*M and other related methods are critical for the unsupervised analysis of spatiotemporal data streams.

Acknowledgements

OD and JNK acknowledge support from the National Science Foundation AI Institute in Dynamic Systems
(grant number: 2112085)

DD would like to thank Albert Azout and acknowledge support from Level Ventures.

References

Agrawal, K., Garg, S., Sharma, S., & Patel, P. (2016). Development and validation of optics based spatio-temporal clustering technique. *Information Sciences*, *369*, 388–401.

Ansari, M. Y., Ahmad, A., Khan, S. S., Bhushan, G. et al. (2020). Spatiotemporal clustering: a review. *Artificial Intelligence Review*, *53*, 2381–2423.

Attouch, H., Bolte, J., Redont, P., & Soubeyran, A. (2010). Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality. *Mathematics of operations research*, *35*, 438–457.

Bhattacharjee, P., & Mitra, P. (2021). A survey of density based clustering algorithms. *Frontiers of Computer Science*, *15*, 1–27.

Birant, D., & Kut, A. (2007). St-dbscan: An algorithm for clustering spatial–temporal data. *Data & knowledge engineering*, *60*, 208–221.

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* volume 4. Springer.

Cakmak, E., Plank, M., Calovi, D. S., Jordan, A., & Keim, D. (2021). Spatio-temporal clustering benchmark for collective animal behavior. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Animal Movement Ecology and Human Mobility* (pp. 5–8).

Chen, X., Faghmous, J. H., Khandelwal, A., & Kumar, V. (2015). Clustering dynamic spatio-temporal patterns in the presence of noise and missing data. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Coleman, G. B., & Andrews, H. C. (1979). Image segmentation by clustering. *Proceedings of the IEEE*, *67*, 773–785.

Dabke, D. V., & Dorabiala, O. (2023a). A novel method for vertex clustering in dynamic networks. In *Complex Networks & Their Applications XII* (pp. 445–456). Springer. doi:10.1007/978-3-031-53499-7_36.

Dabke, D. V., & Dorabiala, O. (2023b). Spatiotemporal graph k-means. In *Proceedings of the Communities in Networks ComNets @ NetSci 2023*.

Dabke, D. V., & Dorabiala, O. (2024). Vertex clustering in diverse dynamic networks. Pre-print, under review.

Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, *54*, 764–771.

Izakian, H., Pedrycz, W., & Jamal, I. (2012). Clustering spatiotemporal data: An augmented fuzzy c-means. *IEEE transactions on fuzzy systems*, *21*, 855–868.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* volume 112. Springer.

Jeung, H., Shen, H. T., & Zhou, X. (2008). Convoy queries in spatio-temporal databases. In *2008 IEEE 24th International Conference on Data Engineering* (pp. 1457–1459). IEEE.

Ji, X., Henriques, J. F., & Vedaldi, A. (2019). Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9865–9874).

Kalnis, P., Mamoulis, N., & Bakiras, S. (2005). On discovering moving clusters in spatio-temporal data. In *International symposium on spatial and temporal databases* (pp. 364–381). Springer.

Kim, W., Kanezaki, A., & Tanaka, M. (2020). Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE Transactions on Image Processing*, *29*, 8055–8068.

Li, Z., Ding, B., Han, J., & Kays, R. (2010). *Swarm: Mining relaxed temporal moving object clusters*. Technical Report ILLINOIS UNIV AT URBANA-CHAMPAIGN DEPT OF COMPUTER SCIENCE.

Lu, X., Wang, W., Ma, C., Shen, J., Shao, L., & Porikli, F. (2019). See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3623–3632).

Nayak, J., Naik, B., & Behera, H. (2015). Fuzzy c-means (fcm) clustering algorithm: a decade review from 2000 to 2014. *Computational intelligence in data mining-volume 2*, (pp. 133–149).

Pappas, T. N., & Jayant, N. S. (1989). An adaptive clustering algorithm for image segmentation. In *International Conference on Acoustics, Speech, and Signal Processing,* (pp. 1667–1670). IEEE.

Peca, I., Fuchs, G., Vrotsou, K., Andrienko, N. V., & Andrienko, G. L. (2012). Scalable cluster analysis of spatial events. *EuroVA@ EuroVis*, *6*, 19–23.

Vieira, M. R., Bakalov, P., & Tsotras, V. J. (2009). On-line discovery of flock patterns in spatio-temporal data. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (pp. 286–295).

Wu, H., Chen, Y., Wang, N., & Zhang, Z. (2019). Sequence level semantics aggregation for video object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9217–9225).

Appendix A. Supplementary Material

Appendix A.1. Code

Code can be found at:
`https://github.com/OlgaD400/STKM`

Appendix A.2. Indexing Function

We can assume that we have a surjective assignment function $a\colon [n] \to [k]$ that assigns each particle to a cluster. Therefore, our assignment function takes the form

$$a(i) \triangleq \min\left\{k' \;\middle|\; i \geq \sum_{j=1}^{k'} n_j\right\}$$

For intuition, note that we are asking for the first cluster such that the total number of points within clusters "so far" do not exceed a given input index i.

Appendix A.3. Intermediate Results and Proofs

We use the following notation:

- $[c]$ represents the set $\{1, \ldots, c\} \subset \mathbb{N}$
- $\mathbf{0}$ is the 0 element (origin) of $\mathbb{R}^d$
- $\mathbb{I}_d$ is the $d \times d$ identity matrix

Proposition Appendix A.1. *Let $W_i^t, Z_j^t \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$ where $i \in [n], j \in [k]$. Then, $Y_i^t = \sqrt{q} \cdot W_i^t + \sqrt{p} \cdot Z_{a(i)}^t$ where $q \triangleq 1 - p$ if and only if $Y_i^t \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$ with the condition* $\operatorname{Cor}\left(Y_i^s, Y_j^t\right) = p$ *if $a(i) = a(j)$, $s = t$ and* 0 *otherwise.*

Proof. Assume $Y_i^t = \sqrt{q} \cdot W_i^t + \sqrt{p} \cdot Z_{a(i)}^t$. The sum of two i.i.d. normal distributions is also a normal distribution and the mean and variance of the sum is the sum of the means and variances. Therefore, Y_i^t has mean $\sqrt{q} \cdot \mathbf{0} + \sqrt{p} \cdot \mathbf{0} = \mathbf{0}$; it has variance

$$\begin{aligned}
\operatorname{Var}\left[Y_i^t\right] &= \operatorname{Var}\left[\sqrt{q} \cdot W_i^t + \sqrt{p} \cdot Z_{a(i)}^t\right] \\
&= q \operatorname{Var}\left[W_i^t\right] + p \operatorname{Var}\left[Z_{a(i)}^t\right] \\
&= q\mathbb{I}_d + p\mathbb{I}_d \\
&= \mathbb{I}_d
\end{aligned}$$

Thus, $Y_i^t \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_d)$.

Now, we prove the other direction: by construction, the various displacements are uncorrelated at different time steps, so we just need to verify $\operatorname{Cor}\left(Y_i^t, Y_j^t\right)$ as

$$\begin{aligned}
\operatorname{Cor}\left(Y_i^t, Y_j^t\right) &= \operatorname{Cor}\left(\sqrt{q} \cdot W_i^t + \sqrt{p} \cdot Z_{a(i)}^t,\ \sqrt{q} \cdot W_j^t + \sqrt{p} \cdot Z_{a(j)}^t\right) \\
&= \operatorname{Cor}\left(\sqrt{q} \cdot W_i^t,\ \sqrt{q} \cdot W_j^t\right) \\
&\quad + \operatorname{Cor}\left(\sqrt{q} \cdot W_i^t,\ \sqrt{p} \cdot Z_{a(j)}^t\right) \\
&\quad + \operatorname{Cor}\left(\sqrt{p} \cdot Z_{a(i)}^t,\ \sqrt{q} \cdot W_j^t\right) \\
&\quad + \operatorname{Cor}\left(\sqrt{p} \cdot Z_{a(i)}^t,\ \sqrt{p} \cdot Z_{a(j)}^t\right) \\
&= 0 + 0 + 0 + \operatorname{Cor}\left(\sqrt{p} \cdot Z_{a(i)}^t,\ \sqrt{p} \cdot Z_{a(j)}^t\right) \\
&= p \operatorname{Cor}\left(Z_{a(i)}^t, Z_{a(j)}^t\right) \\
&= p\mathbb{1}_{a(i)=a(j)}
\end{aligned}$$

□

Lemma Appendix A.2 (Sequence Correlation is Maintained)**.**

$$\operatorname{Cor}\left(X_i^t, X_j^t\right) = \begin{cases} p & a(i) = a(j) \\ 0 & a(i) \neq a(j) \end{cases}$$

Proof. With i, j such that $a(i) \neq a(j)$, X_i^t, X_j^t are sums of i.i.d. random variables, so their correlation is 0.

If we select i, j such that $a(i) = a(j)$, let us compute

$$\begin{aligned}
\operatorname{Cov}\left(X_i^t, X_j^t\right) &= \operatorname{Cov}\left(\sum_{r=0}^{t-1} Y_i^r, \sum_{s=0}^{t-1} Y_j^s\right) \\
&= \sum_{r,s} \operatorname{Cov}\left(Y_i^r, Y_j^s\right) \\
&= \sum_{r=s} \operatorname{Cov}\left(Y_i^r, Y_j^s\right) + \sum_{r \neq s} \operatorname{Cov}\left(Y_i^r, Y_j^s\right) \\
&= \sum_{r=s} p + \sum_{r \neq s} 0 \\
&= t \cdot p
\end{aligned}$$

Using the same logic, we can compute

$$\begin{aligned}
\operatorname{Var}\left[X_i^t\right] &= \operatorname{Cov}\left(\sum_{r=0}^{t-1} Y_i^r, \sum_{s=0}^{t-1} Y_i^s\right) \\
&= \sum_{r,s} \operatorname{Cov}\left(Y_i^r, Y_i^s\right) \\
&= \sum_{r=s} \operatorname{Cov}\left(Y_i^r, Y_i^s\right) + \sum_{r \neq s} \operatorname{Cov}\left(Y_i^r, Y_i^s\right) \\
&= \sum_{r=s} 1 + \sum_{r \neq s} 0 \\
&= t
\end{aligned}$$

Putting this together, we can compute

$$\begin{aligned}
\operatorname{Cor}\left(X_i^t, X_j^t\right) &= \frac{\operatorname{Cov}\left(X_i^t, X_j^t\right)}{\sqrt{\operatorname{Var}\left[X_i^t\right] \cdot \operatorname{Var}\left[X_j^t\right]}} \\
&= \frac{t \cdot p}{\sqrt{t \cdot t}} \\
&= p
\end{aligned}$$

□

From here, we study what happens to the distance between two particles over time. In particular, we want to characterize the distribution of $\delta(i, j, t)$ where

$$\delta(i, j, t) \triangleq \|X_i^t - X_j^t\|$$

From now, we assume $\|\cdot\|$ is that standard Euclidean distance, but ideally our results and proofs would not depend on the norm selected or would at least work for any L^p norm.

Lemma Appendix A.3 (Expectation, Variance of Distance)**.** *Given two particles X_i^t and X_j^t*

$$\mathbb{E}\left[\delta(i, j, t)\right] \in \Theta\left(\sqrt{tqd}\right) \tag{A.1}$$

$$\operatorname{Var}\left[\delta(i, j, t)\right] \to tq \qquad \text{(as } d \to \infty\text{)}$$

In particular, if $a(i) \neq a(j)$ and thus $q = 1$, then

$$\mathbb{E}\left[\delta(i,j,t)\right] \in \Theta\left(\sqrt{td}\right) \qquad \text{(A.2)}$$
$$\mathrm{Var}\left[\delta(i,j,t)\right] \to t \qquad (\text{as } d \to \infty)$$

In particular, when $d = 2$

$$\mathbb{E}\left[\delta(i,j,t)\right] \approx 2.5066 \cdot \sqrt{tq}$$
$$\mathrm{Var}\left[\delta(i,j,t)\right] \approx 0.8584 \cdot tq$$

and when $d = 3$ then

$$\mathbb{E}\left[\delta(i,j,t)\right] \approx 3.1915 \cdot \sqrt{tq}$$
$$\mathrm{Var}\left[\delta(i,j,t)\right] \approx 0.9070 \cdot tq$$

Proof. Since each component of every X^t is independent, we will begin by analyzing the random vector component-wise. We will write $X^t_{i,l}$ to refer to the l^{th} component of X^t_i and related quantities, where $l \in [d]$. First, note that $X^t_{i,l} - X^t_{j,l}$ equals

$$\sum_{r=0}^{t-1}\left[\sqrt{q}\cdot W^r_{i,l} + \sqrt{p}\cdot Z^r_{a(i),l} - \sqrt{q}\cdot W^r_{j,l} - \sqrt{p}\cdot Z^r_{a(j,l)}\right]$$

In the case that $a(i) = a(j)$, we see that $Z^t_{a(i)} = Z^t_{a(j)}$. In the case that $a(i) \neq a(j)$, we can write $q = 1, p = 0$, so in either case, we can write

$$\begin{aligned} X^t_{i,l} - X^t_{j,l} &= \sum_{r=0}^{t-1}\left[\sqrt{q}\cdot W^r_{i,l} - \sqrt{q}\cdot W^r_{j,l}\right] \\ &= \sqrt{q}\sum_{r=0}^{t-1}\left[W^r_{i,l} - W^r_{j,l}\right] \\ &= \sqrt{2tq}\sum_{r=0}^{t-1}\frac{1}{\sqrt{2t}}\left[W^r_{i,l} - W^r_{j,l}\right] \end{aligned}$$

We can observe that

$$\sum_{r=0}^{t-1}\frac{1}{\sqrt{2t}}\left[W^r_{i,l} - W^r_{j,l}\right] \sim \mathcal{N}(0,1)$$

so by letting $\overline{Z}_{ij,l} \triangleq \sum_{r=0}^{t-1}\frac{1}{\sqrt{2t}}\left[W^r_{i,l} - W^r_{j,l}\right]$, we can write $X^t_{i,l} - X^t_{j,l} = \sqrt{2tq}\cdot\overline{Z}_{ij,l}$ and we can now proceed to put everything together. Namely, note that

$$\begin{aligned} \left\|X^t_i - X^t_j\right\| &= \sqrt{\sum_{l=1}^{d}\left(X^t_{i,l} - X^t_{j,l}\right)^2} \\ &= \sqrt{\sum_{l=1}^{d}\left(\sqrt{2tq}\cdot\overline{Z}_{ij,l}\right)^2} \\ &= \sqrt{2tq}\sqrt{\sum_{l=1}^{d}\left(\overline{Z}_{ij,l}\right)^2} \end{aligned}$$

which directly implies that

$$\kappa\cdot\|X^t_i - X^t_j\| \sim \mathcal{X}(d)$$

where $\kappa = 1/\sqrt{2tq}$. The expectation of the $\mathcal{X}$ distribution is well-known and implies that

$$\begin{aligned} \mathbb{E}\left[\|X^t_i - X^t_j\|\right] &= \sqrt{2tq}\cdot\mathbb{E}\left[\kappa\cdot\|X^t_i - X^t_j\|\right] \\ &= \sqrt{2tq}\cdot\mu_d \end{aligned}$$

where $\mu_d \to \sqrt{d - \frac{1}{2}} \in O(\sqrt{d})$, but for small values of d, we know that

$$\mu_d = \begin{cases} \sqrt{2\pi}\frac{2^{1-d}(d-1)!}{\left(\left(\frac{d}{2}-1\right)!\right)^2} & d \text{ even} \\ \\ \sqrt{2}\frac{\left(\frac{d-1}{2}\right)!}{\frac{2^{2-d}\sqrt{\pi}(d-2)!}{\left(\frac{d-1}{2}-1\right)!}} & d \text{ odd} \end{cases}$$

In particular,

$$\mu_2 = \sqrt{\frac{\pi}{2}} \approx 1.2533$$

$$\mu_3 = \sqrt{\frac{8}{\pi}} \approx 1.5958$$

so

$$\sqrt{2}\cdot\mu_2 \approx 2.5066$$
$$\sqrt{3}\cdot\mu_3 \approx 3.1915$$

Moreover, we can compute that

$$\begin{aligned} \mathrm{Var}\left[\|X^t_i - X^t_j\|\right] &= 2tq\,\mathrm{Var}\left[\kappa\cdot\|X^t_i - X^t_j\|\right] \\ &= 2tq(d - \mu_d^2) \end{aligned}$$

which approaches tq when d is large, since $(d - \mu_d^2) \to \frac{1}{2}$ and in particular

$$2(2 - \mu_2^2) \approx 0.8584$$
$$2(3 - \mu_3^2) \approx 0.9070$$

□