ARTICLE TEMPLATE

TSCAPE: Time Series Clustering with Curve Analysis and Projection on an Euclidean space

Jeremy RENAUD[ab] , Raphaël Couturier[a], Christophe Guyeux[a], and Benoit Courjal[b]

[a]Universite de Franche-Comte, CNRS, institut FEMTO-ST, Belfort, France; [b]Caduciel, Voray sur l'Ognon, France

ARTICLE HISTORY
Compiled January 26, 2024

ABSTRACT

The ever-growing use of digital systems has led to the accumulation of vast datasets, particularly time series, depicting the temporal evolution of variables and systems. Analyzing these time series, essential for predictions, anomaly detection, and behavioral studies, presents a tremendous challenge due to their inherent complexity and heterogeneity. Addressing an industrial need in the pharmaceutical wholesale sector, this paper introduces a new clustering method for time series: TSCAPE. The TSCAPE method uses a distance matrix calculated using dynamic time warping, followed by multidimensional scaling to project time series into a 2D Euclidean space, where similar patterns are nearby, thus improving the last clustering stage by K-Means. Unlike conventional techniques, this approach, based on clustering of representation of distances in a euclidean plane rather than on the curve shape, directly enhances the efficiency of the clustering process. The methodology exhibits significant potential for diverse applications, beyond pharmaceutical sales, accommodating varied data types and irregular time series shapes. The research compares multiple variants and proposes metrics to assess their effectiveness on two open-access datasets. The results demonstrate the method's superiority over "only distance comparison clustering techniques", like dynamic time warping and K-Means, with future prospects aimed at predictive applications and refinement of the clustering process by exploring alternative, more powerful clustering algorithms.

KEYWORDS
Time series clustering ; Curves similarity ; Dynamic time warping ; Euclidean Embedding representation ; Neighborhood distance clustering

1. Introduction

Nowadays, the use of digital systems has become ubiquitous, resulting in the generation of vast datasets that can be stored and analyzed to extract pertinent information. Within this pool of data lie time series, delineating the evolution of a system or variable over time, including trajectories, sales curves, or temperature fluctuations. The exploration of such time series offers substantial insights beneficial for predictive analytics, behavioral studies, and anomaly detection within the systems producing and using these data.

However, the analysis of time series data, given its complex and heterogeneous na-

CONTACT Raphaël Couturier Email: raphael.couturier@univ-fcomte.fr

ture, is an intricate task, which complicates its analysis. For instance, companies may aim to identify anomalies in store performance, categorize products based on their sales behavior, or detect failures in a production line. As a result, researchers are in constant pursuit of methodologies to cluster these time series, thereby simplifying their analysis. Similar endeavors were initiated, for instance to address the industrial requirements of pharmaceutical wholesale distributors. Specifically, in the pharmaceutical domain, predictions concerning coherent product groupings, indicating clusters of products displaying similar sales behaviors, lack adequate classifications beyond the pre-established Anatomical Therapeutic Chemical (ATC) classification by the World Health Organization for medications. This gap gave rise to the notion of clustering products with similar sales trends as a potential solution, aiming to generalize predictions across entire product groups by ensuring homogeneous sales behaviors across all members of a same group.

This paper introduces a new clustering method designed to enhance predictions without requiring comprehensive product usage context. The approach involves computing a distance matrix among products that have been grouped together, by using a dynamic time-warping algorithm to analyze time series. Subsequently, the representation of each time series is projected onto a 2D Euclidean space using a multidimensional scaling algorithm. The proximity of points on the plane indicates a similar time series, thereby facilitating the clustering process. To sum up the approach, we propose (1) to compute a time series distance matrix by using dynamic time warping, then (2) to transform the time series data into a set of points by applying multidimensional scaling on these pairwise distances, and finally (3) to apply a K-Means clustering algorithm on this cloud of points.

Our primary goal is to cluster time series sharing the most comparable patterns and relying on an examination of curve shape similarities within the time series. Directly applying a clustering method to time series is intricate, considering the need to cluster data in dimensions equivalent to the size of the studied time series. Thus, our approach entails clustering based on a representation of the similarities between these time series, modeled by distances between points projected onto an Euclidean plane. With this approach, each entity within a cluster should exhibit the closest proximity to other members within the same cluster while maintaining a substantial distance from individuals in other clusters. This clustering strategy necessitates creating clusters centered around their centroids. The method's advantages lie in conducting clustering in a reduced-dimensional space, then grouping time series displaying low similarity distances, and in its intrinsic explicability.

One of the method's merits is its divergence from direct clustering based on curve shape; instead, clustering through a representation of distances on a Euclidean plane is performed. By clustering points in close proximity, assurance is gained regarding the similarity of the represented curves, thereby streamlining the clustering process. This innovative methodology enables one to efficiently manage a large volume of time series without a predetermined classification, holding significant potential for industrial and data science applications. Additionally, the adaptability of our method spans various data types, encompassing sensor data, financial data, and meteorological data. It accommodates time series exhibiting diverse shapes, whether regular or irregular. Moreover, our approach provides a clear graphical depiction of time series groups, making it easier to understand results.

Following the review of the existing body of knowledge, the subsequent sections of the article will detail the developed clustering method, outlining each step and comparing multiple potential variations. The article will introduce the used datasets, will describe the data processing methods, and will also present the metrics need to compare the different methods. Additionally, comparative elements will be suggested, and the article will conclude by addressing future works and potential areas for enhancement.

2. Related Works

2.1. Comparison Techniques for Time Series

The paper addresses the analysis of time series curves' shapes to group them based on similarities. The literature presents various techniques to compare time series, exemplified in the work of Liao (2005). The authors categorize three main clustering types for time series: "raw-data-based" in Golay et al. (1998), using fuzzy logic clustering on functional magnetic resonance imaging data; "feature-based" in Shaw and King (1992), employing cluster analysis to identify local oscillators; and "model-based" in Kalpakis et al. (2001), comparing Linear Predictive Coding cepstrum of ARIMA time series. Techniques leveraging Euclidean distances (*e.g.*, Elmore and Richman (2001)) involve creating a similarity matrix through Euclidean distance for Principal Component Analysis (PCA). However, the investigation into more efficient techniques, like the so-called Dynamic Time Warping (DTW) proposed by Sakoe and Chiba (1978), remains less explored.

2.2. Distance Comparison Techniques in Time Series Analysis

Dynamic Time Warping (DTW) is a well-known method for calculating the smallest distance between two time-series, allowing flexibility in aligning points to account for variations in speed and time. DTW has found major applications in time series clustering tasks and beyond.

DTW's efficiency in time series clustering is relevant in various studies, showcasing its adaptability and usefulness in pattern recognition, as illustrated in Petitjean et al. (2011). DTW's application extended to avian studies where it identified patterns in bird songs, offering insights into bird communication and behavior in the work of Jančovič et al. (2013).

Additionally, Longest Common SubSequence (LCSS), as used in Vlachos et al. (2002), operates by identifying the longest subsequence common to two timeseries, providing a robust measure of similarity. Edit distance with Real Penalty (ERP), see Chen and Ng (2004), offers a technique computing distance between time series while accounting for their alignment and amplitude difference. Complexity-Invariant Distance (CID), proposed by Batista et al. (2014), focuses on capturing the complexity pattern of time series, rendering it robust against length and scaling variations.

Furthermore, Shape-Based Measures like Slope Distance (cf. Kamalzadeh et al. (2020)) and Procrustes Shape Analysis, like in the work of Andreella et al. (2023), was used to align matrices. These techniques provide different methodologies to compare time series and are used in various fields, contributing significantly to pattern recognition in An et al. (2023), signal processing in Liu et al. (2023), data analysis (Ejegwa and Agbetayo (2023)), and identification (Hittmeir et al. (2022)).

2.3. Embedding and Dimensionality Reduction Techniques

Embedding and dimensionality reduction techniques play a pivotal role in visualizing high-dimensional data into lower dimensions while preserving crucial information.

MultiDimensional Scaling (MDS) is a fundamental technique used to transform distance matrices into point clouds on predefined Euclidean planes, as proposed in Cox and Cox (2008). Its applications span various fields, showcasing its versatility and efficiency. For instance, MDS was applied in profiling analysis, unveiling distinct patterns in individuals' profiles, such as in Ding (2000). Additionally, MDS was used for predicting bankruptcy, providing a succinct visualization of companies' financial statuses, like in the work of Neophytou and Molinero (2004). MDS was also used to analyze the financial health of banks, offering insights into the banking sector's dynamics, see, *e.g.*, Mar-Molinero and Serrano-Cinca (2001). Furthermore, MDS found application in virus behavior analysis providing a structured understanding of virus behavior patterns, cf. Lopes et al. (2016). The technique often collaborates with clustering algorithms, as evident in the study analyzing killer whale songs in Brown et al. (2006), employing MDS as part of an unsupervised grouping methodology leveraging a sequence of algorithms.

In addition to MDS, several other dimensionality reduction algorithms exist with distinct features and applications. t-SNE (t-Distributed Stochastic Neighbor Embedding) specializes in visualizing high-dimensional data by minimizing the divergence between data points in different dimensions, see Van der Maaten and Hinton (2008). Isomap (Isometric Mapping, Tenenbaum et al. (2000)) maintains geometric relationships in the high-dimensional space, preserving local properties in the low-dimensional representation. Locally Linear Embedding (LLE), as in Roweis and Saul (2000), focuses on local linearity, projecting data points in a way that maintains the relationships between neighboring points. UMAP (Uniform Manifold Approximation and Projection) like in the work of Becht et al. (2019) is adept at preserving both local and global structures in the data space. PCA (Principal Component Analysis, used for instance in Jolliffe and Cadima (2016)) and KernelPCA are widely used linear techniques that project data onto a new subspace to maximize variance, offering a simplified yet informative representation, and can be used before a clustering algorithm as in Mohammadi et al. (2022).

These techniques provide diverse strategies for dimensionality reduction and are applicable in a myriad of domains, contributing significantly to data visualization and analysis.

2.4. Clustering Algorithms

Clustering encompasses various algorithms, each with distinct methodologies and applications in data analysis. K-means, as in Agarwal and Mustafa (2004), is a pervasive technique relying on centroids, involving iterative assignment and replacement of cluster centroids. In contrast, HDBSCAN (Malzer and Baum (2020)) focuses on the similarity between cluster members, prioritizing neighborhood association over cluster center proximity. This leads to the creation of more precise yet unequal-sized clusters, a feature particularly advantageous for non-uniformly distributed data. While HDBSCAN excels in handling clusters based on similarity, it may encounter challenges with non-compact point clouds and data points that do not easily adhere to traditional cluster formations.

Several other notable clustering algorithms deserve attention. DBSCAN (Density-

Based Spatial Clustering of Applications with Noise, Ester et al. (1996)) identifies clusters based on dense regions separated by areas of lower density. OPTICS (Ordering Points To Identify the Clustering Structure), proposed by Ankerst et al. (1999), is an extension of DBSCAN, providing a hierarchical clustering representation. Mean Shift (Derpanis (2005)) uses a non-parametric technique to identify clusters by locating density maxima in the data space. This algorithm has recently been improved with the use of GPUs like in You et al. (2022). Agglomerative Hierarchical Clustering (Day and Edelsbrunner (1984)) operates by successively merging individual data points or clusters based on their similarity, creating a dendrogram to illustrate the merging process.

Finally, BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies, Zhang et al. (1996)) uses a hierarchical clustering approach to handle large datasets, producing a compact summary for subsequent analysis. Spectral Clustering, like in Abbasimehr and Bahrini (2022); Lee and Park (2023); Ng et al. (2001), operates on similarity graphs and spectral theory to partition data into clusters. Lastly, Affinity Propagation (Frey and Dueck (2007)) uses message passing to find the most appropriate examples of data points for clustering, balancing the quality of the example and the number of clusters.

These algorithms demonstrate diverse approaches and have applications in various domains, catering to different types of datasets and structures.

3. TSCAPE description

The idea of our solution is decomposed into 3 steps, see below (Figure 1). In the first step, the distance is computed by pairwise comparison of time series curves to create a similarity matrix. Subsequently, this distance matrix is input to multidimensional scaling, which projects a representation of the time series onto a 2D space while preserving the distances described by the matrix. It is on this 2D representation that a clustering algorithm is applied to achieve the grouping of similar time series.

In the following subsections, we will detail each step of the methodology and the algorithms used in these steps.

3.1. Distance comparison with Dynamic Time Warping (DTW)

Initiating the comparison process involves a pairwise assessment of all the time series representing sales curves. It is possible that certain sales curves exhibit similar behavior but are slightly time-shifted. To address this issue, employing the dynamic time-warping algorithm enables the comparison of curves without being impeded by minor variations in their shapes. Dynamic Time Warping (DTW) is relevant in this context as it accommodates temporal shifts between sequences. It aligns the sequences by warping them in the time dimension, allowing for a flexible matching process, ideal to compare time series with inherent temporal distortions. The dynamic time warping calculates the smallest distance between two time series. Unlike a simple Euclidean distance calculation, it does not compare point to point the two time series and offers a degree of flexibility to align the points. This algorithm ($DTW(x, y)$) is applied to the data (P) compared two by two, to create a matrix M of size $N \cdot N$, N corresponding to the number of time series to be clustered. M is defined as follows: $M(i, j) = DTW(i, j)$ with i,j belonging to P. The $DTW(X, Y)$ algorithm is defined as follows. For two time

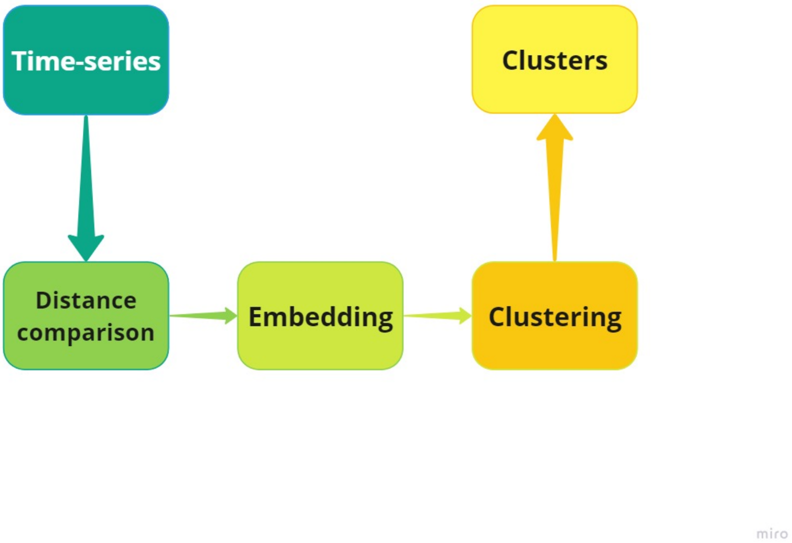

Figure 1. The different steps of our solution. The pairwise comparison of time series is performed to derive their similarity distances, subsequently transforming the resultant similarity matrix into a point projection representing the time series. This representation is then used in the formation of clusters comprising the closest points through the application of a clustering algorithm.

series X, Y of respective size $N1$ and $N2$, the first step is to create a distance matrix D of size $N1 \cdot N2$ with:

$$D(k,p) = |Xk - Yp| + min\{D(k-1,p), D(k,p-1), D(k-1,p-1)\}.$$

The next step is to recreate the path between $D(N1-1, N2-1)$ and $D(0,0)$ with the smallest distance value. For this purpose, one adds the value of the cells that were crossed, using as a rule of displacement : $Forcurrentcell = D(k,p)\ Nextcell = min\{D(k-1,p), D(k,p-1), D(k-1,p-1)\}$.

Once the matrix M was created and filled, it was passed as input to the multidimensional scaling algorithm.

Tests have also been performed by replacing the classical dynamic time warping algorithm by a soft_DTW as in Cuturi and Blondel (2017), which is a variant of the DTW algorithm with an additional ”gamma” parameter (Table 1).

3.2. Embedding: MultiDimensional Scaling (MDS)

MDS is an algorithm used in data science to visualize dissimilarities between objects in a dataset by mapping them to a lower-dimensional space while preserving as much as possible their original pairwise distances. The primary goal of MDS is to represent high-dimensional data in a reduced number of dimensions, typically two or three, to facilitate visualization and interpretation.

3.2.1. Mathematical Formulation

Consider a dataset of n objects, where the dissimilarities between each pair of objects are represented by a symmetric $n \times n$ dissimilarity matrix D. MDS aims to embed these objects into a lower-dimensional space, typically p dimensions.

The objective of MDS is to find a configuration of points $X = \{x_1, x_2, ..., x_n\}$ in a p-dimensional space such that the Euclidean distances between these points best approximate the dissimilarities in the original matrix D.

The stress function in classical MDS, also known as Kruskal's stress, is designed to minimize the difference between the original dissimilarities D_{ij} and the reconstructed dissimilarities using the embedded points x_i and x_j in the reduced space.

The stress function is given by:

$$E(X) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij}(d_{ij} - \|x_i - x_j\|)^2$$

where:

- $E(X)$ is the stress function to be minimized,
- w_{ij} is a weight associated with the dissimilarity between objects i and j,
- d_{ij} is the original dissimilarity between objects i and j,
- $\|x_i - x_j\|$ represents the Euclidean distance between points x_i and x_j in the lower-dimensional space.

The minimization problem is solved using optimization techniques such as gradient descent, eigendecomposition, or nonlinear optimization methods.

3.2.2. Operational Details

1. **Input**: The input to MDS is a dissimilarity matrix representing the pairwise dissimilarities between objects.

2. **Computing Distances**: Calculate the pairwise Euclidean distances from the dissimilarity matrix.

3. **Optimization**: Minimize the stress function by finding the optimal configuration of points in the lower-dimensional space.

4. **Visualization**: Visualize the embedded points in 2D or 3D space to explore the relationships between objects.

Multidimensional Scaling serves several essential purposes in the preprocessing of time series data for clustering analysis. Primarily, MDS endeavors to maintain the original relationships between time series by conserving pairwise distances or dissimilarities as closely as possible during dimensionality reduction. This preservation of proximity ensures the retention of critical information crucial for subsequent clustering. Additionally, MDS helps to reduce noise by filtering out extraneous details in the time series data, highlighting the more substantial patterns and structures, thereby enhancing the efficiency of subsequent clustering algorithms. Moreover, by representing time series data in a reduced-dimensional space, MDS allows for a more effective application of clustering algorithms, generating more insightful and accurate clusters as it reflects the original relationships between the data points. Finally, MDS contributes to computational efficiency by reducing the complexity of subsequent clustering algorithms through dimensionality reduction, making the clustering process more computationally manageable and facilitating the application of algorithms struggling with higher dimensions.

3.2.3. Use in TSCAPE

The second step of TSCAPE is to apply a multidimensional scaling algorithm on the distance matrix generated by the DTW. This algorithm allows, by using a distance matrix, to create points in an Euclidean space of dimension d. Each point represents a time series contained in the matrix M. If $d \leq 3$, visualization of the positions of the points is possible graphically. But this is not a limiting bound. The last step is to perform a clustering on these points.

3.3. Clustering with K-means

For the clustering step, a few algorithms have been applied on the points representing the time series.

The K-Means algorithm is a widely used clustering method in data analysis to partition a dataset into K distinct clusters. It operates by minimizing the sum of squared distances between data points and their assigned cluster centers, thus forming K clusters around centers called centroids.

3.3.1. Mathematical Formulas

Let $X = \{x_1, x_2, ..., x_n\}$ be a set of n data points and K the number of clusters to be formed.

1. **Initialization of Centroids**: Randomly select K points as initial centroids $C = \{c_1, c_2, ..., c_K\}$.

2. **Assignment of Points to Clusters**: For each point x_i, calculate the distance to each centroid c_j and assign x_i to the cluster whith the nearest centroid. This can be formulated as:

$$\text{Cluster}(x_i) = \arg\min_j ||x_i - c_j||^2$$

3. **Updating the Centroids**: Recalculate the centroids of each cluster as the mean of all points assigned to that cluster:

$$c_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i$$

where S_j is the set of points assigned to cluster j.

4. Repeat steps 2 and 3 until convergence, i.e., until there is no change in the allocation of points to clusters or until a predefined tolerance is reached.

3.3.2. Use in TSCAPE

The clustering is therefore not applied on the curve shapes but on a representation on a Euclidean plane of their dissimilarities. Applying K-means on the MDS-transformed data helps to form distinct clusters by leveraging the reduced, structured representations provided by MDS. The combination allows K-means to operate on the reduced dimensions while aiming to form clusters based on the derived spatial relationships among the time series data points. This integration enables K-means to capture meaningful patterns and similarities in the reduced space created by MDS, potentially leading to more insightful and accurate clustering results for time series data.

The primary advantage of the K-means algorithm compared to other clustering methods, particularly when aiming for highly concentrated clusters, lies in its ability to minimize intra-cluster variance. K-means seeks to minimize the sum of squared distances of points to their respective centroids, efficiently grouping points around centers (centroids) to minimize dispersion within each cluster. In comparison to other clustering algorithms, K-means is effective in forming compact clusters as it explicitly targets the reduction of intra-cluster variance using an iterative approach of reassigning points to clusters based on their proximity to centroids, thus promoting the concentration of data within each cluster. This is particularly relevant for our work as we aim to create clusters with the smallest intra-cluster distances, which would correspond to groups of products with the most similar curves.

4. Experiments

4.1. Data used

The data used for the initial project being proprietary, for this paper we used open-source datasets available on the internet to simulate our sales time series.

- The first one is an open dataset Szrlee (2018) corresponding to data from company exchanges, between 2006 and 2018. For this paper, the column "Volume" (corresponding to the number of shares traded) was used.
- The second dataset corresponds to data of house sales prices according to their district and their number of rooms: Holdings (2019). The data has been grouped by district number, property type, and number of bedrooms: for example, "2603_5_house" will contain all the 5-bedroom houses in district 2603.

Each dataset is transformed to obtain three columns: date, product name, and quantitative values simulating the sales quantity. The data are aggregated by month and by product. This gives us for each product a time series of monthly sales. The missing values over the period are replaced by zero. Before being inputted in Dynamic Time Warning algorithms, the data are also normalized between 0 and 1, to avoid that the volume of sales influences the distance scale of the dissimilarity matrix.

4.2. Metrics

To compare the solutions between them, three metrics were used.

- The silhouette score (SilScore in this paper) checks the density of clusters and their separations. The solution interval for the silhouette score is between 1 (best solution) and -1 (worst one).
- The Calinski-Harabasz score (CHScore in this paper) is somewhat similar: it is a ratio between the sum of inter-cluster variances and the sum of intra-cluster ones. It ranges from 0 (worst) to +infinity (best), and its range is related to the data set.
- The Davies-Bouldin index (DbScore in this paper) measures how similar it is to the closest cluster. It is the average of the maximum ratio between the distance of a point to the center of its cluster and the distance between two cluster centers.

It is between 0 (best) and + infinity (worst).

4.3. Clustering by K-means with DTW metric (tslearn)

In the first step, the data were clusterized with the TimeSeriesKmeans function of the TSlearn Tavenard et al. (2020) Python library. This library is specialized on the treatment of time series. The TimeSeriesKmeans function has an option in its *metric* parameter that directly uses dynamic time warning. The number of clusters has been chosen by an Elbow method (Figures 2 and 3). The elbow method is a technique used in clustering algorithms, such as K-means, to determine the optimal number of clusters in a dataset. It involves plotting the variance or distortion (inertia) against the number of clusters. As the number of clusters increases, the variance typically decreases. The ”elbow” point on the graph signifies the number of clusters where the rate of decline changes abruptly, indicating the point of diminishing returns, helping to select an appropriate number of clusters for the data. The number of clusters was set to 8 for the house datasets and to 10 for the control Market dataset. The parameters of the K-means are the same as those described in Section ”4.4.3 K-Means configuration” of this paper, except for the specific parameter of this solution: *metric*, which was initialized to *dtw*.

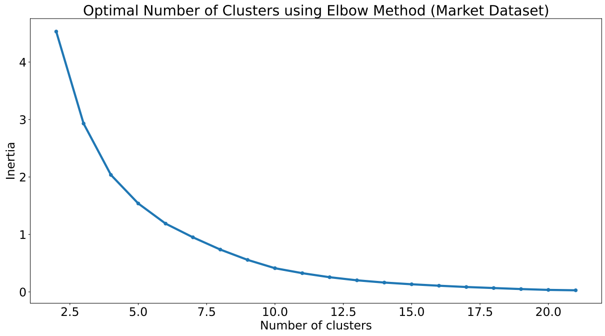

Figure 2. Elbow method for Market dataset

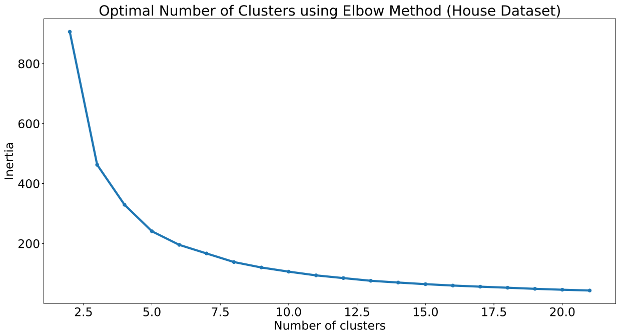

Figure 3. Elbow method for House dataset

The results show that this first solution is not very efficient for our data. The silhouette score is low, which indicates poor cluster coherence (poor cluster separation, or too large distances between individuals in a cluster). Indeed, the sales curves are quite different from each other and therefore the pattern detection does not seem to work very well (Tables 4 and 5).

4.4. Principal Component Analysis (PCA) for Time Series Clustering

Principal Component Analysis (PCA) is a crucial dimensionality reduction technique widely used in time series clustering. In this context, PCA enhances the efficiency of subsequent clustering algorithms, such as K-means. The process involves mean-centering the time series data and extracting principal components that capture the maximum variance.

Following mean-centering, the covariance matrix is computed, and PCA is applied to obtain principal components ordered by variance. The decision on the number of components to retain is crucial, often based on a predefined variance threshold. Subsequently, time series are reconstructed using the selected principal components, resulting in a reduced-dimensional representation.

The reduced representation serves as input for K-means clustering, where time series are grouped into clusters based on their similarity in the reduced feature space. PCA offers advantages such as noise reduction, computational efficiency,and enhanced clustering performance.

Using PCA in conjunction with a clustering algorithm such as K-means appears to be one of the most effective methods for time series clustering, as shown in the works of Li (2019); Singhal and Seborg (2005); Yang and Shahabi (2004). These approaches differ from our work as they do not rely on the comparison of distances between time series. Despite achieving superior results (Table 4 and Table 5), it comes at the cost of the interpretability of dimensionality reduction: in our method, two points are necessarily close because they have a short distance calculated by the DTW, i.e. their curves are similar, as opposed to the PCA method, which reduces dimensions by creating its own axes, making projection analysis more complex.

We also conducted tests by (PCA) into the embedding stage of our methodology, to assess whether it enhances the outcomes (Table 4 and Table 5).

4.5. TSCAPE Experimentation [1]

To compare both solutions the parameters used are preserved, as well as the number of clusters and the Kmean setting (excluding *"metric = dtw"*).

(1) **Dynamic Time Warping configuration**

The "gamma" parameter of the soft-DTW was tested with different values. The scores of the different values were compared to each other, and also with the results obtained on a classic dynamic time warping (Table 1).

(2) **Multidimensional scaling configuration**

For the multidimensional scaling step, the variation of the dimension parameter (*n_components*) was tested. An example for both datasets, on a space of dimension $d = 2$, is provided in Figures 4 and 5.

Increasing the number of dimensions allows more degrees of freedom to the Multidimensional Scaling algorithm. But looking at Table 2, one can see that increasing the dimension $d \leq 4$ does not improve the different scores.

The MSD algorithm has a *random_state* parameter that allows us to initialize

[1] Source code available at github.com : https://github.com/ml-anon/TSCAPE/.

	Mean of 10 runs with House data			Mean of 10 runs with Market data		
SoftDTW gamma	SilScore Mean	CHScore Mean	Dbscore Mean	SilScore Mean	CHScore Mean	Dbscore mean
DTW	0.46	333.84	0.71	0.44	66.93	0.53
10	0.44	193.23	0.81	0.41	36.12	0.6
0.1	0.47	429.31	0.66	0.38	74.2	0.6
0.01	0.49	343.72	0.67	0.56	437.61	0.33
0.005	0.48	333.2	0.67	0.49	327.53	0.34

Table 1. Variation of the metrics according to the SoftDTW gamma parameters.

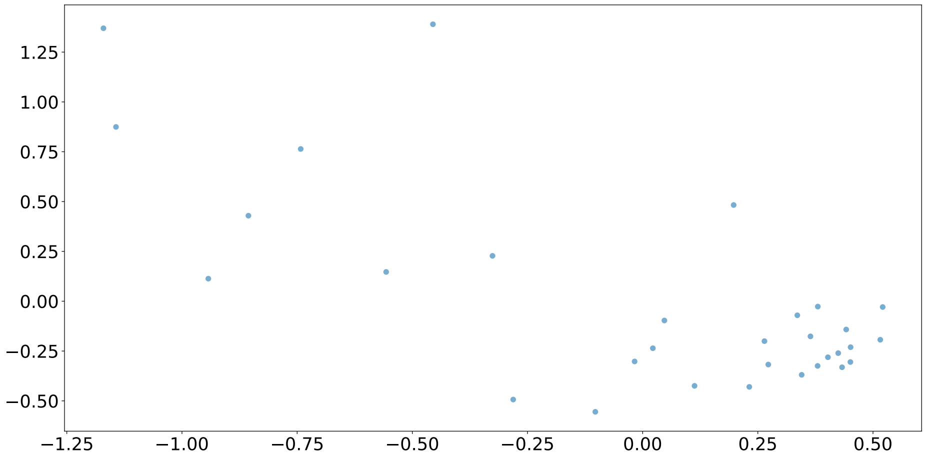

Figure 4. 2D projection with MDS for the Market dataset. Each point corresponds to the representation of a time series. The distance between two points represents the similarity score.

Two points close to each other means that both time series have a similar curve.

	Mean of 10 runs with House data			Mean of 10 runs with Market data		
Ndim	SilScore Mean	CHScore Mean	Dbscore Mean	SilScore Mean	CHScore Mean	Dbscore mean
2	0.49	343.62	0.67	0.52	376.93	0.33
3	0.48	342.99	0.67	0.56	473.74	0.34
4	0.48	343.05	0.67	0.54	456.7	0.36
5	0.48	342.81	0.67	0.54	424.43	0.37
6	0.48	342.9	0.67	0.54	421.57	0.36
7	0.48	342.9	0.67	0.54	432.47	0.36
8	0.48	342.8	0.67	0.54	424.43	0.36
9	0.48	342.83	0.67	0.54	413.05	0.37
10	0.48	342.77	0.67	0.54	432.47	0.36

Table 2. Variation of the metrics according to the number of dimensions of the MDS.

it with different seed values. The variation of this parameter does not seem to change greatly the result (Table 3). The observation of these results shows

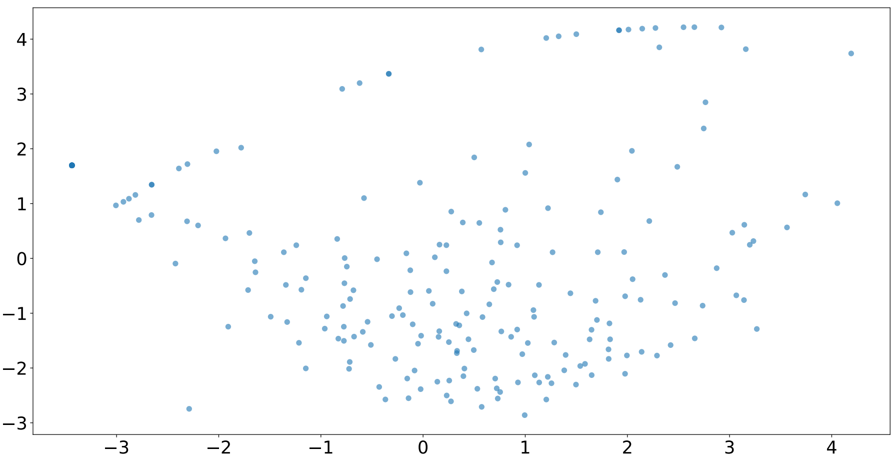

Figure 5. 2D projection with MDS for the House dataset. Each point corresponds to the representation of a time series. The distance between two points represents the similarity score.

Two points close to each other means that both time series have a similar curve.

that changing the initialization seed of the MDS varies only slightly with the clustering result for datasets with many samples. In case of datasets with a few samples, this parameter can be useful to increase the silhouette score.

	Mean of 10 runs with House data			Mean of 10 runs with Market data		
Seed	SilScore Mean	CHScore Mean	Dbscore Mean	SilScore Mean	CHScore Mean	Dbscore mean
2	0.49	343.67	0.67	0.54	450.69	0.35
3	0.49	343.72	0.67	0.54	373.75	0.37
4	0.49	343.69	0.67	0.54	396.38	0.36
5	0.49	343.68	0.67	0.55	464.09	0.34
6	0.49	343.7	0.67	0.54	436.32	0.35
7	0.49	343.71	0.67	0.51	370.92	0.33
8	0.49	343.7	0.67	0.54	390.69	0.36
9	0.49	343.57	0.67	0.51	371.39	0.33
10	0.49	343.72	0.67	0.56	437.61	0.33

Table 3. Variation of the metrics according to the seed number of the MDS.

Tests have been performed with other dimensionality reduction algorithms. The t-SNE algorithm is one of the best methods in this category. It was used just after the MDS to see if it was able to place the points corresponding to the products more efficiently than the MDS. The results (Table 4 and Table 5) show a slight improvement in silhouette score for datasets with many samples. It seems that when the number of samples and the size of the matrix M is larger, the MDS has fewer degrees of freedom to place the points between them, and therefore offers more convergent results whatever the seed. It can then be assisted by the t-SNE to improve these results (Figures 6 and 7).

Tests were also done by replacing the MDS algorithm with a Spectral Embedding such as in Belkin and Niyogi (2003). Although the results were better on some configurations, this did not allow the best solution to be surpassed (Table 4 and Table 5).

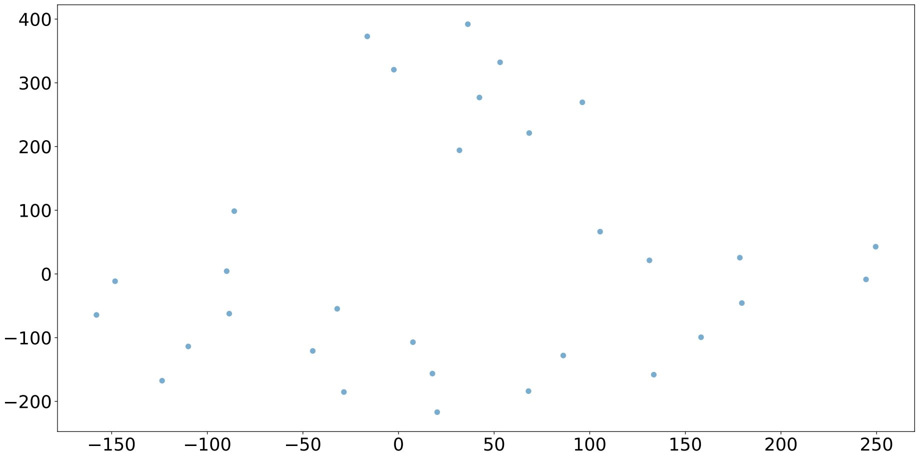

Figure 6. 2D projection with MDS followed by point replacement using the t-SNE algorithm for the Market dataset. The distance between two points represents the similarity score.

Each point corresponds to the representation of a time series. Two points close to each other mean that the both time series have a similar curve.

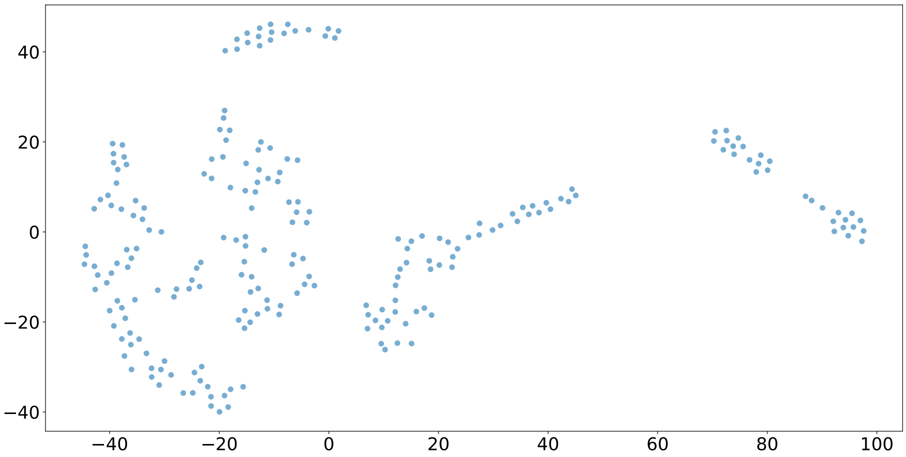

Figure 7. 2D projection with MDS followed by point replacement using the t-SNE algorithm for the House dataset. The distance between two points represents the similarity score.

Each point corresponds to the representation of a time series. Two points close to each other mean that both time series have a similar curve.

(3) **K-means configuration**

The parameters of the K-means are the same as for the tslearn part: *max_iter* was set to 400, *n_init* to 100, *init* to *K-means++*, and finally *random_state* was initialized and varied from 0 to 20. The number of clusters is also the same as defined in the tslearn part of this paper: (8 for the house datasets and 10 for the control Market dataset). All the points are indeed in a cluster, but sometimes some points close to each other are in 2 different clusters. This is due to the K-means clustering method, it is the only problematic consequence of this choice. This problem can be corrected by increasing the number of clusters enough to try to better distribute the points in each group, at the expense of the size of the number of individuals per cluster, and of the ease of analysis (Figures 8 and 9).

Experiments were conducted using K-Medoids and Spectral clustering algorithms. Replacing the K-means algorithm in our solution with K-Medoids led to some improvements in certain metrics for specific datasets. However, when used solely after DTW, our solution remained the most effective. On the other hand, the Spectral Clustering algorithm did not improve our solution (Table 4 and Table 5).

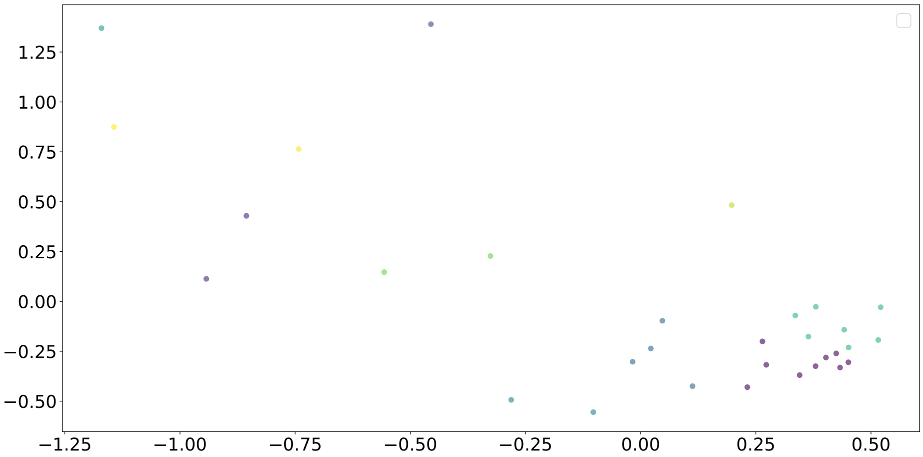

Figure 8. Clustering for the Market dataset. Clusters are represented by different colors: two points of the same color are in the same cluster.

4.6. Results

In our methodology, the first two steps involving dynamic time warping and multidimensional scaling represent the similarity between two time-series by establishing a distance between two points. Our goal is to group time series that have the most similar curves, necessitating the grouping of the closest points. Each individual within a cluster should be as close as possible to the other members of the same cluster and ideally far from the members of other clusters. This implies the creation of clusters around their centroids, a task efficiently performed by the K-means algorithm. Metrics such as the silhouette score, Calinski-Harabasz score, and Davies-Bouldin index are relevant to evaluate data clustering when seeking to group the closest points. The silhouette score measures the similarity of points within clusters and their dissimilarity from other clusters. The Calinski-Harabasz score evaluates cluster compactness and

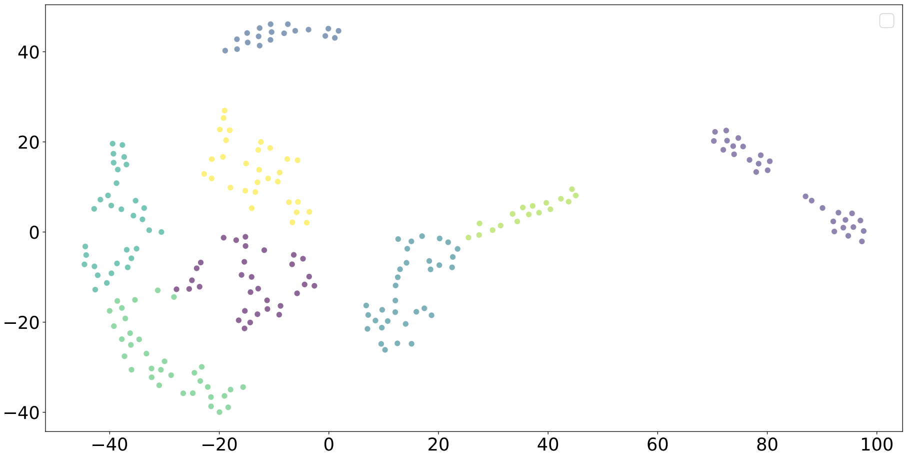

Figure 9. Clustering for the House dataset. Different colors represent clusters: two points of the same color are in the same cluster.

separation, while the Davies-Bouldin index quantifies both intra-cluster dispersion and inter-cluster separation. These metrics offer a comprehensive evaluation of clusters in terms of intra-cluster cohesion, inter-cluster separation, and differentiation between clusters.

The silhouette score of the clusters proves the improvement our solution TSCAPE brought (Table 4 & Table 5). The addition of multidimensional scaling improves the silhouette score compared to a solution composed only of dynamic time warping and K-means (with equal parameter settings). Adding a t-SNE algorithm between MDS and clustering improves the silhouette score in some cases. Replacing the DTW algorithm by its Soft-DTW enables one to improve the results with certain ”gamma” parameter values.

5. Conclusion

Through this paper, we have presented a technique to cluster time series. This clustering is based on the analysis and comparison of curve shapes. The different techniques could be tested by comparing the silhouette scores, the Calinski-Harabasz score and the Davies-Bouldin index, of all the clusters. The results showed that the solution proposed in this paper is more efficient in creating clusters than the existing methods based on dynamic time warping and K-Means. Although our method is less effective than the PCA-based methods, it provides better interpretation in terms of dimension reduction. The next step of this work is to use these clusters to make predictions by grouping products in the context of data from a wholesaler-retailer for French pharmacies. Clustering improvement works can also be done, for example by testing other clustering algorithms more powerful than the K-means one.

Dataset House	SilScore mean	CHScore mean	Dbscore mean	SilScore STD	CHScore STD	Dbscore STD
DTW + MDS + K-means	0.46	333.86	0.72	0	0	0
DTW + MDS + t-SNE +K-means	0.53	563.74	0.61	0.002	35.33	0.005
DTW + K-means	0.39	226.08	0.88	5.55	2.84	0
DTW + t-SNE + K-means	0.43	160.26	0.77	0.03	12.51	0.04
DTW + Spectral Embeding + K-means	0.42	149.11	0.67	0	0	0
DTW + SpectralEmbeding + t-SNE + K-means	0.52	449.2	0.68	0.01	17	0.005
SoftDTW + Spectral Embeding + K-means	0.43	186.05	0.71	0	0	0
SoftDTW + SpectralEmbeding + t-SNE + K-means	0.55	818.68	0.63	0	52.31	0.02
SoftDTW + K-means	0.41	198.74	0.88	0	0	0
SoftDTW + t-SNE + K-means	0.56	1404.53	0.61	0.01	289.89	0.03
SoftDTW + MDS +K-means	0.49	343.62	0.67	0	0	0
SoftDTW + MDS + t-SNE + K-means	0.56	924.1	0.58	0	93.34	0
SoftDTW + MDS + K-Medoids	0.40	279.49	1.0	0	0	0
SoftDTW + MDS + t-SNE + K-Medoids	0.59	750.82	0.48	0	0	0
SoftDTW + MDS + SpectralClustering	0.40	255.62	0.79	0	0	0
SoftDTW + MDS + t-SNE + SpectralClustering	0.26	125.53	1.95	0	0	0
SoftDTW + SpectralClustering	0.35	179.13	1.11	0	0	0
SoftDTW + K-Medoids	0.37	257.70	0.98	0	0	0
SoftDTW + PCA + K-Means	0.53	469.35	0.60	0	0	0
SoftDTW + PCA + t-SNE + K-Means	**0.64**	1061.51	**0.41**	0	0	0
PCA + K-means	0.62	**2337.83**	0.41	0	69.31	0.04
tslearn	0.053	13.65	5.49	0	0.34	0.08

Table 4. Benchmark (means of 10 runs) of all solutions and options of this paper for House dataset (the best solution is in bold).

References

Abbasimehr, H. and Bahrini, A. (2022). An analytical framework based on the recency, frequency, and monetary model and time series clustering techniques for dynamic segmentation. *Expert Systems with Applications*, 192:116373.

Agarwal, P. K. and Mustafa, N. H. (2004). K-means projective clustering. In *Proceedings of the Twenty-Third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '04, page 155–165, New York, NY, USA. Association for Computing Machinery.

An, S., Bhat, G., Gumussoy, S., and Ogras, U. (2023). Transfer learning for human activity recognition using representational analysis of neural networks. *ACM Transactions on Computing for Healthcare*, 4(1):1–21.

Andreella, A., De Santis, R., Vesely, A., and Finos, L. (2023). Procrustes-based distances for

Dataset	SilScore	CHScore	Dbscore	SilScore	CHScore	Dbscore
Market	mean	mean	mean	STD	STD	STD
DTW + MDS + K-means	0.43	66.92	0.53	0	0	1.11
DTW + MDS + t-SNE + K-means	0.35	61.05	0.73	0.01	4.83	0.02
DTW + K-means	0.37	90.04	0.69	0	0	0
DTW + t-SNE + K-means	0.34	96.63	0.75	0.01	3.17	0.03
DTW + Spectral Embeding + K-means	0.44	34.92	0.49	0	0	0
DTW + SpectralEmbeding + t-SNE + K-means	0.45	60.53	0.62	0.01	5.29	0
SoftDTW + Spectral Embeding + K-means	0.45	35.27	0.62	0	0	0
SoftDTW + Spectral Embeding + t-SNE + K-means	0.4	48.13	0.66	0.02	4.8	0.03
SoftDTW + K-means	0.33	138.65	0.45	0	0	0
SoftDTW + t-SNE + K-means	0.42	94.98	0.64	0.02	4.91	0.02
SoftDTW + MDS + K-means	0.52	376.93	0.33	0	0	0
SoftDTW + MDS + t-SNE + K-means	0.36	114.68	0.75	0	3.48	0.02
SoftDTW + MDS + K-Medoids	0.37	514.15	0.35	0	0	0
SoftDTW + MDS + t-SNE + K-Medoids	0.26	104.06	0.80	0	0	0
SoftDTW + MDS + SpectralClustering	0.24	64.64	1.17	0	0	0
SoftDTW + MDS + t-SNE + SpectralClustering	0.01	26.40	1.64	0	0	0
SoftDTW + SpectralClustering	0.14	61.65	1.69	0	0	0
SoftDTW + K-Medoids	0.32	470.21	0.39	0	0	0
SoftDTW + PCA + K-Means	0.47	577.95	**0.31**	0	0	0
SoftDTW + PCA + t-SNE + K-Means	0.35	138.90	0.72	0	0	0
PCA + K-means	**0.65**	**1062.85**	0.55	0	0	0
tslearn	0.27	21.95	1.07	0.01	0.64	0.06

Table 5. Benchmark (means of 10 runs) of all solutions and options of this paper for Market dataset (the best solution is in bold).

exploring between-matrices similarity. *Statistical Methods & Applications*, pages 1–16.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). Optics: Ordering points to identify the clustering structure. *ACM Sigmod record*, 28(2):49–60.

Batista, G. E., Keogh, E. J., Tataw, O. M., and De Souza, V. M. (2014). Cid: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, 28:634–669.

Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I. W., Ng, L. G., Ginhoux, F., and Newell, E. W. (2019). Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, 37(1):38–44.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396.

Brown, J. C., Hodgins-Davis, A., and Miller, P. J. O. (2006). Classification of vocalizations of killer whales using dynamic time warping. *The Journal of the Acoustical Society of America*, 119(3):EL34–EL40.

Chen, L. and Ng, R. (2004). On the marriage of lp-norms and edit distance. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 792–803.

Cox, M. A. A. and Cox, T. F. (2008). *Multidimensional Scaling*, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cuturi, M. and Blondel, M. (2017). Soft-DTW: a differentiable loss function for time-series. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 894–903. PMLR.

Day, W. H. and Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1:7–24.

Derpanis, K. G. (2005). Mean shift clustering. *Lecture Notes*, 32:1–4.

Ding, C. S. (2000). Profile analysis: Multidimensional scaling approach. *Practical Assessment, Research, and Evaluation*, 7(1):16.

Ejegwa, P. A. and Agbetayo, J. M. (2023). Similarity-distance decision-making technique and its applications via intuitionistic fuzzy pairs. *Journal of Computational and Cognitive Engineering*, 2(1):68–74.

Elmore, K. L. and Richman, M. B. (2001). Euclidean distance as a similarity metric for principal component analysis. *Monthly Weather Review*, 129(3):540 – 549.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *science*, 315(5814):972–976.

Golay, X., Kollias, S., Stoll, G., Meier, D., Valavanis, A., and Boesiger, P. (1998). A new correlation-based fuzzy logic clustering algorithm for fmri. *Magnetic resonance in medicine*, 40(2):249–260.

Hittmeir, M., Mayer, R., and Ekelhart, A. (2022). Distance-based techniques for personal microbiome identification. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, pages 1–13.

Holdings, H. (2019). House property sales time series. data retrieved from Kaggle, `https://www.kaggle.com/datasets/htagholdings/property-sales?resource=download`.

Jančovič, P., Köküer, M., Zakeri, M., and Russell, M. (2013). Unsupervised discovery of acoustic patterns in bird vocalisations employing dtw and clustering. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5.

Jolliffe, I. T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202.

Kalpakis, K., Gada, D., and Puttagunta, V. (2001). Distance measures for effective clustering of arima time-series. In *Proceedings 2001 IEEE international conference on data mining*, pages 273–280. IEEE.

Kamalzadeh, H., Ahmadi, A., and Mansour, S. (2020). Clustering time-series by a novel slope-based similarity measure considering particle swarm optimization. *Applied Soft Computing*, 96:106701.

Lee, Y. and Park, S. (2023). Spectral clustering of weighted variables on multi-omics data. *The Korean Journal of Applied Statistics*, 36(3):175–196.

Li, H. (2019). Multivariate time series clustering based on common principal component analysis. *Neurocomputing*, 349:239–247.

Liao, T. W. (2005). Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874.

Liu, B., Liu, C., Zhou, Y., Wang, D., and Dun, Y. (2023). An unsupervised chatter detection method based on ae and merging gmm and k-means. *Mechanical Systems and Signal Processing*, 186:109861.

Lopes, A. M., Andrade, J. P., and Tenreiro Machado, J. (2016). Multidimensional scaling analysis of virus diseases. *Computer Methods and Programs in Biomedicine*, 131:97–110.

Malzer, C. and Baum, M. (2020). A hybrid approach to hierarchical density-based cluster selection. In *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 223–228.

Mar-Molinero, C. and Serrano-Cinca, C. (2001). Bank failure: a multidimensional scaling approach. *The European Journal of Finance*, 7(2):165–183.

Mohammadi, Y., Miraftabzadeh, S. M., Bollen, M. H., and Longo, M. (2022). An unsupervised learning schema for seeking patterns in rms voltage variations at the sub-10-minute time scale. *Sustainable Energy, Grids and Networks*, 31:100773.

Neophytou, E. and Molinero, C. M. (2004). Predicting corporate failure in the uk: A multidimensional scaling approach. *Journal of Business Finance & Accounting*, 31(5-6):677–710.

Ng, A., Jordan, M., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14.

Petitjean, F., Ketterlin, A., and Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern recognition*, 44(3):678–693.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326.

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49.

Shaw, C. and King, G. (1992). Using cluster analysis to classify time series. *Physica D: Nonlinear Phenomena*, 58(1-4):288–298.

Singhal, A. and Seborg, D. E. (2005). Clustering multivariate time-series data. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 19(8):427–438.

Szrlee (2018). Djia 30 stock time series. data retrieved from Kaggle, https://www.kaggle.com/datasets/szrlee/stock-time-series-20050101-to-20171231?resource=download.

Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., and Woods, E. (2020). Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6.

Tenenbaum, J. B., Silva, V. d., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Vlachos, M., Kollios, G., and Gunopulos, D. (2002). Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering*, pages 673–684. IEEE.

Yang, K. and Shahabi, C. (2004). A pca-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM international workshop on Multimedia databases*, pages 65–74.

You, L., Jiang, H., Hu, J., Chang, C. H., Chen, L., Cui, X., and Zhao, M. (2022). Gpu-accelerated faster mean shift with euclidean distance metrics. In *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 211–216. IEEE.

Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25(2):103–114.