Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrics and Operational Research

A performance evaluation of the Swale distance measure for univariate time series clustering

Mette Beekman (581484)

Supervisor:	Durieux, J
Second assessor:	Welz, MFO
Date final version:	1st July 2024

Abstract

This paper presents a replication of the research conducted by Holder, Middlehurst and Bagnall (2024) in which the performance of elastic distance measures on univariate time series is compared. The distance measures that are subject to this study are: the benchmark Euclidean distance (ED), Dynamic time warping (DTW), Weighted DTW (WDTW), Move-split-merge (MSM), Longest common subsequence(LCSS), Edit distance with real penalty (ERP) and Time-warping edit (TWE). The distance measures performances were compared utilising partitional clustering algorithms k-means and k-medoids clusters and evaluated with the performance measures cluster accuracy, rand index and mutual information . In addition, an extra distance measure, the Sequence weighted alignment (Swale) scoring model, was compared and tested for performance. The results demonstrated that for k-means clustering, MSM was the best performing distance measure. For k-medoids, ERP performed the best. These findings differ from those obtained by Holder et al. (2024), where it was concluded that MSM was the best performing distance measure for both k-means and k-medoids clustering. With the additional Swale distance, there was a significant decline in performance for both clustering algorithms.

1 Introduction

Clustering is an unsupervised data analysis technique that groups data into clusters. As defined by Fasulo (1999) clustering is the process of organizing objects in groups where the members are similar. Or as Aghabozorgi, Shirkhorshidi and Wah (2015) explained, homogeneous grouping that can be achieved without the need for advanced definitions of group members. An object of interest may, for instance, be a time series. A time serie is a sequence of a specified number of observations m. Time series can be univariate, wherein the elements of the series are scalars. In the case of multivariate time series, the elements are vectors. The elements of the time series, x_i, are given by the vector $(x_1, \cdots, x_m)$ (Holder et al., 2024). The clusters may be defined as $C = (C_1, \cdots, C_k)$. With the increased power of storage, a great number of additional time series data has become available. For example, financial data such as exchange rates and stock prices. In addition, biomedical measurements, such as blood pressure and biometric data for facial recognition (Aghabozorgi et al., 2015).

A number of studies have been conducted on the subject of time series clustering. In order to identify groups of similar items, the degree of similarity within each group must be determined. The optimal clusters are those with the highest similarity within groups and the lowest between groups (Aghabozorgi et al., 2015). In order to quantify the degree of similarity between two entities, distance measures are employed. The objective of this research is to build upon the findings of Holder et al. (2024). In their research, they compared ten elastic distance measures on 112 time series datasets and evaluated their performance using performance metrics, including accuracy, rand index, and mutual information. The extension of this research is an additional elastic distance, the Sequence weighted alignment (Swale) scoring model, as proposed by Marteau (2008) and compare this with the distances proposed by Holder et al. (2024). Salarpour and Khotanlou (2018) explored the Swale scoring model with the use of hierarchical clustering on multivariate time series datasets. However, the potential of the Swale scoring model as a distance measure for partitional clustering has yet been unexplored. This

research will contribute to the field by comparing the performance of another distance measures for partitional clustering, with the aim of identifying a more effective distance.

By comparing distance measures, we aim to answer the following question: *Which elastic distance measure performs the best for partitional clustering?* To identify the optimal distance measure, this research compares the performance of elastic distances using both k-means and k-medoids cluster algorithms. To answer the primary research question, this study also aims to answer two sub-questions: *Which elastic distance measure performs the best for k-means clustering? Which elastic distance measure performs the best for k-medoids clustering?* In order to ascertain whether the utilisation of the Swale scoring model as a distance measure enhances the accuracy of cluster identification in comparison to the elastic distance measures proposed by Holder et al. (2024), it is necessary to address another sub-question: *Does the Swale scoring model distance measure perform better than the elastic distance measures proposed by Holder et al. (2024)?*

The identification of the optimal distance measure is of relevance, as it would facilitate more accurate cluster analysis in univariate time series datasets. Cluster analysis can be employed in a multitude of analytical contexts. For instance, clustering can be utilized in the prediction of stock prices and the analysis of molecular biology data (Babu, Geethanjali & Satyanarayana, 2012; Nugent & Meila, 2010).

The results of this research show that in terms of the best performing measure the move-split-average distance measure performs the best when utilising a k-means clusterer. For k-medoids clustering the Edit distance with real penalty performs the best. Furthermore, the research similar to Holder et al. (2024) show that the initialisation method does not influence the performance comparison of the distance measures. Moreover, the performance results of the Swale distance illustrates the Swale distance underperforming the other distances. Finally, the performance analysis, shows the number of clusters to influence which distance measure performs best, however, more research with more computational power is necessary to credibly state this.

The remainder of this paper is organised as follows. Section 2 discusses the findings in literature that are related to this research. Section 3 discusses the dataset used in this research. Section 4 explains the methods used. Section 5 presents the results obtained and the paper concludes with answering the research questions in Section 6.

2 Literature review

In contrast to classification, clustering is an unsupervised machine learning tool. Patterns can be identified without the need for labelled data. Aghabozorgi et al. (2015) call it the solution to classify data when there is no knowledge of classes. Since the emergence of clustering algorithms, they have been used for data analysis in various fields. For example, it has been used in biology, Huang and Pan (2006) explored distance cluster analysis to cluster microarray gene expression data. Cluster analysis for stock price prediction has also been explored (Babu et al., 2012).

In these studies, the most commonly used clustering algorithms are the partitional and hierarchical algorithms. Partitional clustering algorithms partition data into an priori specified unkown number of classes (Holder et al., 2024). Because of the priori specified unknown number

of classes, partitional clustering is computationally efficient. Algorithms that do not require the specific priori set number are the hierarchical clustering algorithms. These algorithms build clusters by merging and splitting clusters based on similarity measures, constructing a dendogram. The starting point is either one cluster with all data points or multiple clusters with only one data point (Sonagara & Badheka, 2014). The constructed dendogram is then cut to obtain k number of clusters. Hierarchical clustering algorithms are known to be computationally heavy. Given the restricted computational power and the classified datasets, this research focusses on partitional clustering algorithms, namely, k-means and k-medoids. K-means is a partiotional clustering algorithm that begins with a predefined number of initialised centroids, which is the mean of a cluster (MacQueen et al., 1967; Reddy & Vinzamuri, 2018). The algorithm employs a chosen measure to form clusters. Subsequently, the centroids within the clusters are updated. The process of forming new clusters and updating centroids is repeated iteratively. Since k-means is a greedy algorithm, the centroids will eventually stop changing. This is known as a local minimum, which is a point in the search space where there is no further improvement. In order to account for the local minima, the clusterings can be restarted on a number of occasions with random initialisations. After each restart, the cluster that performs the best in terms of an unsupervised performance measure is retained (Holder et al., 2024). A K-medoids clustering algorithm employs a comparable algorithm to k-means, however, k-medoids considers the actual data points in the cluster as centroids, which are referred to as medoids. This is the data point that is located most centrally within the cluster. K-medoids are more robust to outliers as stated by Reddy and Vinzamuri (2018). In the context of k-means clustering, the centroid is defined as the mean of a cluster, which is susceptible to the influence of outliers. In k-medoids, the centroid, the most central located data point in the cluster, is less affected by outliers. The computational heaviness of k-medoids clustering is greater than that of k-means clustering due to the necessity of calculating all pairwise similarities to form clusters, in contrast to just the similarity with the mean.

In clustering, the performance of the distance measure is defined by the accuracy of the cluster estimation. A review by Holder et al. (2024) showed that Move-split-merch performs the best in term of cluster accuracy, when examining univariate time series datasets. Salarpour and Khotanlou (2018) adopted a different approach with their research indicating that for multivariate time series datasets the Swale scoring model performed the best when looking at average rank. Since k-means clustering cannot be used for multivariate time series clustering, hierarchical cluster algorithms were used. The research demonstrated that the Swale scoring model outperformed other known scoring models such as Longest common subsequence(LCSS) and Edit distance on real sequences (EDR). Other distance measures that have been widely investigated in the literature are Dynamic time warping (DTW), and its weighted and derative version (Holder et al., 2024). The Marteau (2008) research explored the time warp edit (TWE) distance and compared it with LCSS, DTW and the Edit distance with real penalty (ERP). The results demonstrated that the TWE proved to be an effective measure.

3 Data

In this research the time series data from the University of California (UCR) [1] [2]archive is used (Dau et al., 2018). The archive contains 128 univariate time series datasets. In the Holder et al. (2024) paper, research was conducted using 112 datasets. The datasets conducted are time series datasets without missing values and time series of equal length. The Fungi dataset was also deleted from the original 128 datasets, resulting in a total of 112 complete datasets. Each dataset contains a train and test set. The number of elements in these sets varies per time series dataset.

In this research, the decision has been made not to utilise all 112 time series datasets due to the high computation times. As demonstrated by Holder et al. (2024), the performance of the distance measures vary with the number of classes. A dataset of 61 is chosen consisting of time series datasets with varying number of classes. The dataset is split into four groups. Group A: 31 time series datasets with 2 clusters. Group B: 22 time series datasets with 3-5 clusters. The datasets in group A and B have maximum of 1000 time series. Group C: 6 datasets with 6-10 clusters and Group D: 2 datasets with 11-15 clusters. Group C and D have both datasets with maximum of 200 time series. The decision to select less datasets in group C and D is due to the extra computational time when the number of clusters increases. This is the reason why only six and two datasets were selected. All the datasets have time series with a length less than 2000, this is also chosen due to the computation time. The data set allows for a range of research to be conducted in order to determine the optimal distance measure. Since, we are comparing the distance measures the data in this research employed will be normalised. In order to normalise the data a normalising function in `tsml-eval` Python code provided by Holder et al. (2024) will be used.

4 Methodology

As previously stated, this research will build upon the work of Holder et al. (2024). The research compares the performance of elastic distance measures using the k-means and k-medoids clustering algorithms, as outlined in sections 4.1.1 and 4.1.2. The performance of elastic distance measures is evaluated for normalised data. In the clustering process, the number of clusters is equivalent to the number classes in the classification dataset, similar as in Holder et al. (2024).

4.1 Cluster methods

As previously stated, this research will examine the performance of distance measures when utilising k-means and k-medoids clustering algorithms. The k-means clustering is executed with the same `aeon` default parameters employed by Holder et al. (2024), namely random initialisation, a maximum of 300 iterations, 10 restarts and the centroid average used as the mean. The initialisation algorithm, maximum iterations and restarts employed for k-medoids are identical to those utilised for k-means.

[1] https://www.cs.ucr.edu/∼eamonn/time_series_data_2018/
[2] https://timeseriesclassification.com/

4.1.1 K-means clustering

MacQueen et al. (1967) introduced k-means clustering as a partitioning process that is relatively efficient when considering within-cluster variance. The algorithm partitions the data set into a specified number of clusters, designated by the parameter k. Initially, the centroids are established, a process that will be more thoroughly explained in section 4.1.3. Secondly, the distance between each data point and the centroids is calculated using distance measure. Upon measurement of the distance, the data point is assigned to the cluster with the smallest distance to the centroid of the cluster. When the data point is added to the cluster, the centroid of the cluster is updated to reflect the mean value of all the data points within the cluster. This process is repeated until the centroids of the clusters cease to undergo change, or until the maximum number of iteration has been reached.

4.1.2 K-medoids clustering

K-medoids clustering introduced by Kaufman and Rousseeuw (2009) is an algorithm that employs medoids instead of means. This means that, in contrast to the conventional approach of utilising the mean of the cluster as the centroid, the k-medoids algorithm employs a data point within the cluster as the centroid. The data point in question is the one with the smallest total distance to the other data points in the cluster. This makes the algorithm comparable to the k-means algorithm, the only difference being the updating procedure. As Holder et al. (2024) observed, an advantage of k-medoids is that only the pairwise distance matrix is necessary, not the mean, to determine the clusters. This makes it an efficient algorithm. Furthermore, they elucidate that in their research they utilise as proposed by Kaufman and Rousseeuw (2009), the Partition around medoids (PAM) approach to avoid complete enumeration, given that complete enumeration can be computationally demanding. This present study will investigate the same methodology for k-medoids clustering, to reduce computational time. For k-medoids the same stopping criteria are considered as for k-means. Once the maximum number of iterations has been reached, or when the inertia falls below a predefined tolerance, indicating that their has been no significant change in the clusters, convergence can be assumed, as proposed by Ikotun, Ezugwu, Abualigah, Abuhaija and Heming (2023).

4.1.3 Initialisation

K-means clustering is a the oldest and most widely used partitional clustering algorithm (Holder et al., 2024). Nevertheless, several studies have demonstrated that the results are influenced by the initialisation. In order to ascertain the impact of these initialisation parameters on performance, they are modified and the influence on performance is evaluated. Holder et al. stated that increasing the number of iterations is unlikely to affect performance, as most distances and sets converge within twenty iterations. Consequently, the number of iterations will remain unaltered throughout this research. Another parameter that requires consideration is the number of restarts. Unfortunately, due to the lack of computational resources, it will not be possible to increase the number of restarts. Finally, the initialization algorithm is likely to influence the performance, which is why this will be explored in the research. The research will investigate

three initialisation algorithms: random, k-means++ and Forgy's initialisation.

The process of random initialisation involves the selection of random points from the datasets as initial centres, as explained by Holder et al. (2024). It is likely that the points will be located in dense regions due to the random nature of the selection process. In order to identify the optimal cluster, the model can be rerun multiple times. This process allows for the logical outcome to be constructed.

The k-means++ algorithm proposed by Arthur, Vassilvitskii et al. (2007), initiates dispersed clusters. By selecting a bias towards separate centres, a scatter of centres is created. Initially, a centre is selected randomly from the dataset. Subsequently, the centres are selected by calculating the inverse probability of the minimum distance between cases and the previous selected centres. The minimum distance for case x and the previous centres is defined as $md(x)$. The probability of selecting x as the new centre is given by Equation 1 (Holder et al., 2024).

$$\frac{md(x)^2}{\sum_{j=1}^{n} md(x_j)^2} \tag{1}$$

The third and final initialisation algorithm to be considered is the Forgy initialisation algorithm. This algorithm was proposed by Forgy (1965) and involves the initialisation of centres by a uniform random assignment of each data point to one of the k clusters. The centroids of the initialised clusters will then be designated as the centres. One limitation of this approach is that it lacks a theoretical foundation, given that the formation of random clusters is not characterised by internal homogeneity (Holder et al., 2024).

4.2 Elastic distance measures

It should be noted that not all ten distances will be explored in this research. The objective is to identify the optimal distance measure. The Holder et al. (2024) research demonstrated that when employing k-means clustering, four distances outperformed the benchmark Euclidean distance. Consequently, these elastic distances will be included in this study, as they are considered the most promising candidates for identifying the optimal distance. The distances under consideration are TWE, Weighted DTW (WDTW), Move-split-merge (MSM) and ERP. The construction of these distances will be explained in the following sections. For k-medoids clustering Holder et al. (2024) demonstrated that DTW outperforms the benchmark Euclidean distance. Furthermore, DTW is a commonly used elastic distance in the literature, which provides a rationale for including this distance in the research.

Moreover, the LCSS distance will be included. As an extension, the Swale scoring model will be added as an additional distance, as detailed in Section 4.2.8. This model employs a similar algorithm to the LCSS (Marteau, 2008). This makes an interesting proposition to compare this measure with the LCSS. The construction of the LCSS distance can be found in Section 4.2.4.

4.2.1 Benchmark: Euclidean distance

In order to provide a benchmark, the Euclidean distance will be considered. The Euclidean distance between two time series can be calculated as shown in Equation 2.

$$d_{ed}(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^{m}(a_i - b_i)^2} \tag{2}$$

This equation is the same equation used in the Holder et al. (2024) research. In the equation $\mathbf{a} = \{a_1, a_2, \ldots, a_m\}$ and $\mathbf{b} = \{b_1, b_2, \ldots, b_m\}$, being the two univariate time series. Middlehurst et al. (2021) demonstrated the Euclidean distance is a poor performing benchmark compared to the elastic distances, for time series classification. Similarly, Lines and Bagnall (2015) observed that elastic distances outperformed Euclidean distances in classification tasks employing a k-nearest neighbours algorithm. Therefore, it is expected that the elastic distances will perform better than the Euclidean distance.

4.2.2 Dynamic time warping (DTW)

The concept of Dynamic time warping (DTW) was initially developed to facilitate the comparison of speed patterns in the field of automatic speed recognition. It has since been subjected to extensive research and has been applied with great success in a number of other areas, including data mining and information retrieval (Müller, 2007).

DTW aligns two time series by identifying the optimal match between them. As outlined by Holder et al. (2024), the distance matrix $\mathbf{M}$ between the two sequences $\mathbf{a}$ and $\mathbf{b}$ with element M_{ij} can be calculated as follow: $M_{ij} = (a_j - b_i)^2$. Equation 3 constructed by Holder et al. (2024) illustrates a warping path.

$$P =< (e_1, f_1), (e_2, f_2), ..., (e_s, f_s) > \tag{3}$$

Holder et al. (2024) stated that a warping path is only valid when it begins at $(1, 1)$ and ends at (m, m), where m represents the length of the sequences. It is not possible for the path to include backtracking, so $0 \leq e_{i+1} - e_i \geq 1$ and $0 \leq fi + 1 - fi \geq 1$ with i being between one and m. The warping path is the collection of indices that form a path through M. The DTW distance is calculated using the warping path with the minimal total distance. The distance of a path can be calculated as shown in Equation 4 (Holder et al., 2024).

$$D_P(\mathbf{a}, \mathbf{b}, M) = \sum_{i=1}^{s} M_{e_i, f_i} \tag{4}$$

Given that s is the length of the path P, there are several possible paths. The DTW is the path with the minimum distance. If we take P^* as the minimum path, the DTW distance can be written as shown in Equation 5.

$$d_{dtw}(\mathbf{a}, \mathbf{b}) = D_{P^*}(\mathbf{a}, \mathbf{b}, M) \tag{5}$$

In this research, the optimal warping path is identified using an algorithm introduced by (Holder et al., 2024). The dynamic programming formulation is described in Appendix A.1 in Algorithm 1. As outlined by Holder et al. (2024), the warping process is a time-consuming process. Consequently, a bounding technique will be employed in this research to reduce the

computational time. The itakura parallelogram bounding technique will be utilised, which generates a parallelogram that minimises the warping at the start and end of the sequence (Holder et al., 2024).

4.2.3 Weigted dynamic time warping DTW

In addition to Dynamic time warping, this research will also examine the performance of Weighted dynamic time warping (WDTW), as proposed by Jeong, Jeong and Omitaomu (2011). Unlike Dynamic time warping, in which all points in the path are penalised equally, regardless of the difference between the test and reference point, WDTW penalises points based on the magnitude of the difference. This implies that the penalty will be greater in the case of a large discrepancy than in the case of a minor discrepancy. The distances in distance matrix M can be calculated using the applied weighted penalty, as demonstrated by Equation 6. The variables $\mathbf{a}$ and $\mathbf{b}$ represent the sequences, while w represents the weight.

$$M_{i,j}^{w} = w(|i-j|) \cdot (a_i - b_j)^2 \tag{6}$$

In order to determine the weight, the same logistic function will be employed as was used in the research by Holder et al. (2024). This research employed a logistic weight function proposed by Jeong et al. (2011). When warping a place, the weight can be calculated as follows:

$$w(y) = \frac{w_{\max}}{1 + e^{-g \cdot (a - m/2)}} \tag{7}$$

In this research, the upper bound of the weight, denoted by $wmax$, is set to 1. The parameter g controls the penalty level for large warpings, while m represents the length of the series. The WDTW distance can be constructed using the calculated weight w and distance matrix M, as shown in Equation 8.

$$d_{wdtw}(\mathbf{a}, \mathbf{b}) = D_{P^*}(\mathbf{a},\ \mathbf{b}, M^w) = DTW(\mathbf{a}, \mathbf{b}, M^w) \tag{8}$$

4.2.4 Longest common subsequence(LCSS)

Another distance that will be examined in this research is the Longest common subsequence (LCSS) distance. In this measure the longest sequence is derived by deleting elements from the original elements. This approach was originally developed for aligning sequences of discrete variables, such as DNA (Holder et al., 2024). In order to replicate the results of Holder et al. (2024), the LCSS distance in this research is calculated in the same manner as their study. The algorithm for calculating the $LCSS(\mathbf{a}, \mathbf{b})$ can be found in Appendix A.2 in Algorithm 2. Using this algorithm the distance can be calculated as shown in Equation 9.

$$d_{\mathrm{LCSS}}(\mathbf{a}, \mathbf{b}) = 1 - \frac{\mathrm{LCSS}(\mathbf{a}, \mathbf{b})}{m} \tag{9}$$

With $\mathbf{a}$ and $\mathbf{b}$ being two different time series and m the length of the time series. Vlachos, Kollios and Gunopulos (2002) stated the LCSS is a more robust method for time series classification than DTW, particularly in the presence of noise. However, this research employed

normalised data which reduces noise, and only time series without missing data points are explored. Consequently, it can be reasonably assumed that the enhanced performance of LCSS due to its robustness will not result in an improvement in the outcomes of LCSS in this research.

4.2.5 Edit distance with real penalty (ERP)

The Edit distance with real penalty (ERP) represents an alternative to the Edit distance on real sequences (EDR). As stated in Chen and Ng (2004), the edit distance for penalty is a combination of the L-1 norm and EDR. When considering the ERP distance, a real gap is calculated between two elements that are non gaps. In the event that one of the two elements is a gap, the distance is taken to be a constant value.

The ERP distance employed in this research is identical to that employed by Holder et al. (2024). The ERP distance is using the following formula: $d(a, b) = \sqrt{(a-b)^2}$, to construct the pointwise distance matrix. With **a** and **b** being two time series. Algorithm 3 in Appendix A.3 illustrates the determination of the ERP distance between the time series.

In the algorithm, lines 5 and 7, initialises the edges with a large constant value. In line 12, the minimum value of the cost function is taken from the matching (line 9), insertion (line 10) and deletion (line 11) operations. Furthermore, Chen and Ng (2004) posits that the ERP is a metric, satisfying the triangular inequality. The ERP distance is defined by Equation 10.

$$d_{\text{ERP}}(\mathbf{a}, \mathbf{b}) = \text{ERP}(\mathbf{a}, \mathbf{b}, g, d) \tag{10}$$

4.2.6 Move-split-merge (MSM)

Another distance that will be explored is the Move-split-merge (MSM) distance. This distance is constructed similarly to the previously discussed ERP distance. The Move-split-merge algorithm contains transformations that transforms one time series to another. Stefan, Athitsos and Das (2012) explain instead of using the insert, delete and substitute operations as employed in ERP, MSM employs transformations designated as move, split and merge, as suggested by the name. In the MSM, the substitute operation is referred to as the move operation. When an element is inserted, two operations are performed. First, the split operation, which repeats a selected value twice. The second operation move, which is executed by setting the value of a new element. For the deletion of an element another two operations are performed. First, the element is moved so that it is equal to its predecessor or successor. Second, to delete the element the merge transformation, which is executed by merging two equal elements form one, resulting in the element being deleted.

The costs of these operations can be calculated. Equation 11 shows how the costs are constructed (Holder et al., 2024). If the inserted value x is between the two values of y and z, the cost is a constant c. Otherwise the minimum deviation is added to the constant costs.

$$\text{cost}(x, y, z, c) = \begin{cases} c & \text{if } y \leq x \leq z \text{ or } y \geq x \geq z, \\ c + \min(|x-y|, |x-z|) & \text{otherwise.} \end{cases} \tag{11}$$

The algorithm to calculated the MSM distance is given in Appendix A.4 Algorithm 4. This

algorithm is the same as the algorithm used in the research performed by Holder et al. (2024).

The most significant distinction between this algorithm and the ERP algorithm, as outlined 4.2.5, is that the merge costs that are replaced by the delete cost. Changing these aspects of the algorithm results in the costs depend on the absolute value of the magnitude of the inserted value and the adjacent values (Holder et al., 2024). Thus, not every insertion and deletion are equally in costs. Furthermore, Stefan et al. (2012) posit that the MSM is a metric, satisfying the triangular inequality, symmetry and reflexivity. The distance between two series **a** and **b** is constructed in Equation 12

$$d_{\mathrm{MSM}}(\mathbf{a}, \mathbf{b}) = \mathrm{MSM}(\mathbf{a}, \mathbf{b}, c) \tag{12}$$

4.2.7 Time-warping Edit (TWE)

The final distance proposed by Holder et al. (2024) that is included in this research is the Time-warping edit (TWE). This distance is comprised of three operations, as described by Marteau (2008). The construction of the time warp edit is described in Algorithm 5 in Appendix A.5. Firstly, the delete and insert operations are employed in order to achieve a match. The associated costs are detailed in lines 10 and 11 of Algorithm 5. These costs are equal to the distance between the elements and its predecessor, with a further penalty of λ and stiffness ν. Moreover, the operation of matching is considered. The cost associated with this operation is the sum of the proportional costs corresponding to matching points in the segments. This is calculated in line 9 of the algorithm. Finally, the minimum of the three operations represents the value for the D matrix. Equation 13 is used to calculate the TWE distance (Holder et al., 2024).

$$d_{\mathrm{TWE}}(\mathbf{a}, \mathbf{b}) = \mathrm{TWE}(\mathbf{a}, \mathbf{b}, \nu, \lambda) \tag{13}$$

Given that the parameter λ is comparable with the c parameter of MSM, it is reasonable to choose to set it to the same value of 1. The parameter ν in the TWE distance measure is analogous to the weighting parameter w in the WDTW, therefore ν is set equal to 0.05. The aforementioned values were also employed in the Holder et al. (2024) research.

4.2.8 Extension: Sequence weighted alignment (Swale) scoring model

The research performed by Holder et al. will be extended by adding a distance measure and compare its performance with the other distances. In their research Salarpour and Khotanlou (2018) showed that the Sequence weighted alignment (Swale) scoring model exhibited the highest average rank of similarity measures for multivariate time series. The research was conducted using a dataset comprising 23 publicly available multivariate time series datasets, with varying lengths ranging from 120 to 3,109 and the number of clusters between 2 and 50.

The Swale scoring model is an improvement of the threshold value based scoring models LCSS and EDR (Marteau, 2008). The similarity score is based on both the match reward and mismatch penalties. It also weights penalties relative to one another. These weights are learned by performing experiments on a training dataset. Morse and Patel (2007) proposed that the Swale distance function can be defined as shown in Equation 14.

$$\text{Swale}(\mathbf{R}, \mathbf{S}) = \begin{cases} n * \text{gap}_c & \text{if } m = 0 \\ m * \text{gap}_c & \text{if } n = 0 \\ \text{reward}_m + \text{Swale}(\text{Rest}(\mathbf{R}), \text{Rest}(\mathbf{S})) & \text{if } \forall d, |r_{d,1} - s_{d,1}| \leq \epsilon \\ \max\{\text{gap}_c + \text{Swale}(\text{Rest}(\mathbf{R}), \mathbf{S}), \text{gap}_c + \text{Swale}(\mathbf{R}, \text{Rest}(\mathbf{S}))\} & \text{otherwise} \end{cases} \tag{14}$$

With $\mathbf{R}$ and $\mathbf{S}$ being two time series with length m and n. gap_c is the gap cost and $reward_w$ the match reward and $Rest(\mathbf{R})$ the time series $\mathbf{R}$ with the first element removed.

The EDR algorithm scores the gaps and mismatches but not reward matches, while the LCSS solely focuses on rewarding the matches. Swale rewards matches, as LCSS, but also penalises gap elements. This results in an algorithm that is analogous to the algorithm used for LCSS. The algorithm for the Swale distance measure is shown in Algorithm 6 in the Appendix A.6, which is constructed based on the methods used in the research by Salarpour and Khotanlou (2018).

In the research the parameter values for window, reward and penalty will be tuned. The intervals and tuning methods will be explained in section 4.4. The threshold will be set to the same value as the LCSS. This facilitates the comparison of performances.

4.3 Performance measures

In order to facilitate a comparison of the performances of the distance measures, this research utilises the same cluster metrics as those employed in the Holder et al. (2024) review. The cluster metrics considered are the cluster accuracy (CL-ACC), the rand index (RI) and the mutual information (MI). The RI increases with the number of clusters, therefore, in order to account for this, the adjusted RI (ARI) is considered. The following sections will provide a more detailed discussion of the performance measure.

4.3.1 Cluster accuracy (CL-ACC)

The initial performance measure employed in this research is the cluster accuracy (CL-ACC). The accuracy is calculated by dividing the number of correct predictions by the total number of predictions. The correct predictions are given in the first column of the dataset. The equation used to calculate the cluster accuracy in this research is the cluster accuracy proposed by Holder et al. (2024) and shown in Equation 15.

$$CL - ACC(\mathbf{y}, \hat{\mathbf{y}}) = max_{s \in S_k} \frac{1}{|\mathbf{y}|} \sum_{i=1}^{|\mathbf{y}|} 1 \begin{cases} 1, & y_i = \mathbf{s}(\hat{y}_i) \\ 0, & otherwise \end{cases} \tag{15}$$

In this equation $\mathbf{y}$ is all the known correct clusters and $\hat{\mathbf{y}}$ are the predicted clusters. If the predicted cluster matches the assigned cluster the prediction is correct and the cost will be one. Otherwise, when the prediction is incorrect the cost is zero.

4.3.2 Rand index (RI)

The second measure employed to assess the performance of distance measures is the Rand index (RI) proposed by Rand (1971). Holder et al. (2024) denote that the RI is used to assess the degree of similarity between two sets of labels, between the known truth set of labels and the predicted labels by the clustering method. With the RI being the the correct labelled pairs divided by the total pairs. Since, the RI has a limited factor, the score inflates when the number of clusters increases, this research will only use the adjusted Rand index (ARI) as a performance measure. The ARI measure proposed by Hubert and Arabie (1985) adjusts the RI based on expected scores, which are produced by a purely random model. This compensates for the inflation which makes the ARI a more reliable measure to judge the performance.

4.3.3 Mutual information (MI)

The final performance measure is the mutual information (MI) proposed by Cover (1999). This measure employs an entropy function to measure the agreement of the predicted and true labelled clusters (Holder et al., 2024). In addition to the MI, two further versions of the MI will be considered in this research. Firstly, the normalised mutual information (NMI) which normalises the mutual information by rescaling it to an interval between one and zero. Secondly, the adjusted mutual information (AMI) will be considered. This adjustment is made to account for class distribution.

4.4 Tuning the Swale parameters

In order to answer the research question, the parameters of the Swale distance needed to be tuned. This research replicates the findings of Holder et al. (2024), which gave a strong reason to choose the same epsilon as used and tuned in their research. Based on the findings of Salarpour and Khotanlou (2018), a window of 0.2, 0.4, 0.6, 0.8 and 1.0 was selected. In their research, they utilised a window with a minimum of 2 percent of the reward. However, this research explores the window with starting value 0.2. As a maximum value, the window was set equal to the standard deviation. Upon examination of the data, it was observed that the standard deviations were all around 0.99. Given that the maximum window utilized in this research was 1.0 for the other distances, the maximum value for the window parameter of the Swale distance was set equal to one.

In their research Salarpour and Khotanlou (2018) used 50 times the standard deviation as their reward. However, to explore the influence of the reward on the performance, an interval of 10, 20, 30, 40 and 50 times the standard deviation will be considered. The gap cost, is also based on the Salarpour and Khotanlou (2018) research. In their research they used a interval from zero to the reward value with 5 equally distanced values. In our research the same will be done. The gap cost is defined as the gap value multiplied by the reward. In light of the preference for the inclusion of a gap cost, the smallest gap value considered is 0.2. The gap value interval is 0.2, 0.4, 0.6, 0.8 and 1.0, with 1.0 meaning that the gap cost and reward value are equal. Since the gap cost is the gap value times the reward.

5 Results

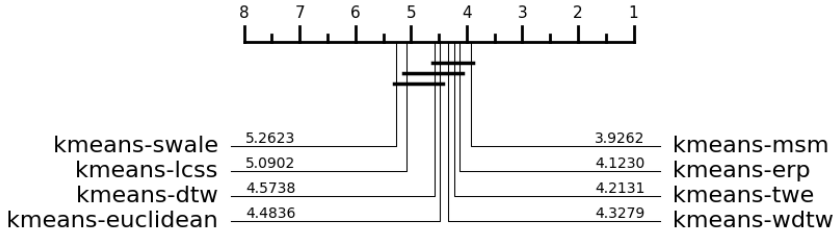
The results in this research were obtained using the `tsml-eval` Python code by Bagnall and contributors (2024) [3], which utilises the `aeon` package. This package was also employed in the Holder et al. (2024) research. Firstly, the performance of distance measures utilising k-means clustering will be explored. Subsequently, the performance will be evaluated when looking at k-medoids clustering. Additionally, the impact of different initialisation methods will be evaluated to find the effect on the best performing distance measure. Furthermore, the extended Swale distance measure will be explored and optimised. Finally, a performance analysis will be conducted to explore differences in performance when dealing with problems involving a large and small number of clusters.

5.1 K-means clustering results

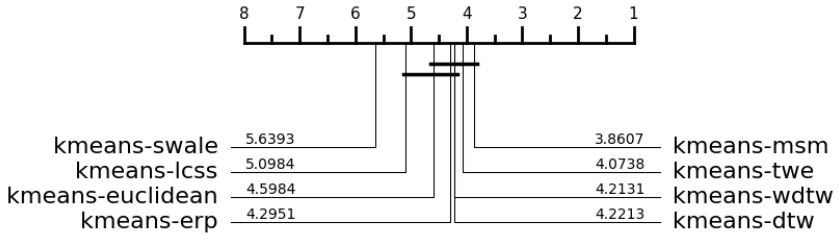
The initial results were obtained through the application of k-means clustering. The results were obtained using the `tsml-eval` Python code with the `aeon 0.8.1` version. The clusters were run for all the datasets and distances. Following the clustering process, the performance metrics were employed to assess the outcomes. For all these performance measures, the critical differences diagrams were plotted. The results can be found in Figure 1.

The results demonstrate when examining clustering accuracy, the Move-split-merge (MSM) distance measure outperforms the other distance measures. When exploring the other performance measures the MSM outperforms as well. With these results it can be concluded that the Move-split-merge distance measure performs the best. Additionally, the TWE performs well, although it consistently underperforms the MSM distance measure, the MSM does not significantly outperforms the TWE. A comparison of the distances with the benchmark, the Euclidean distance, reveals that the LCSS consistently underperforms, however, not significantly. The DTW also underperforms in two of the five performance measures, for the accuracy and adjusted rand index. This outcome was anticipated. The results of the study by Holder et al. (2024) also demonstrated that when employing k-means clustering, the DTW exhibited inferior performance relative to the Euclidean benchmark distance. Their findings indicated that the DTW demonstrated suboptimal performance when evaluated all the performance measures. Nevertheless, in this research DTW only underperformed for two performance measures. A possible explanation for the discrepancy in performance could be the difference in datasets employed, which could potentially influence the results. The performance analysis conducted in the Holder et al. (2024) study showed that the performance of the Euclidean distance increases with the number of clusters. As previously mentioned this research's dataset contains mainly small number of cluster clustering problems. Which could be an explanation of the DTW showing poorer performance in the Holder et al. (2024) research when comparing with the Euclidean distance measure. Although the results are not an exact replication of the Holder et al. (2024) result, when comparing the distances in question, we have employed similar results. The MSM distance is the best performing distance with the ERP, TWE and WDTW performing better than the benchmark. The Swale distance results in Figure 1 will be discussed in Section 5.4.

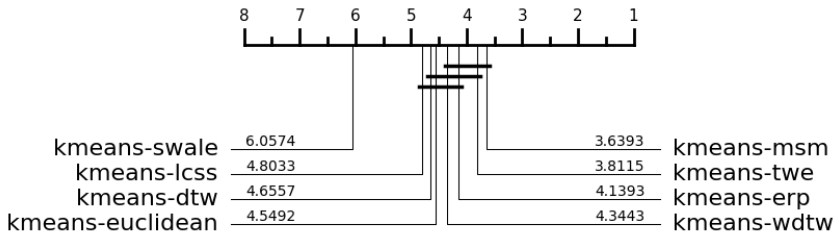
[3] https://github.com/time-series-machine-learning/tsml-eval



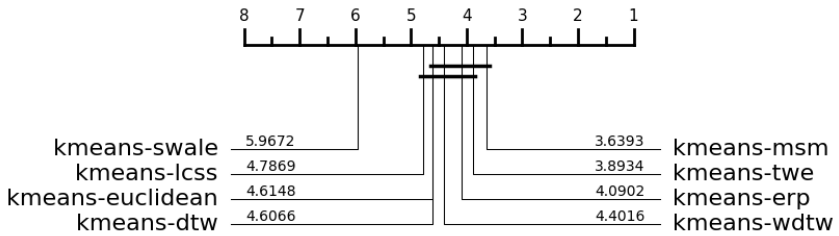
(a) Accuracy



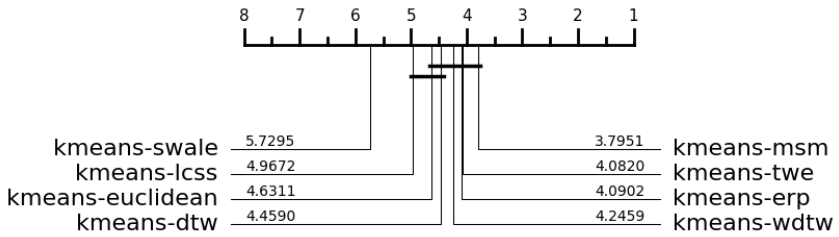
(b) ARI



(c) Mutual information



(d) AMI



(e) NMI

Figure 1: Critical difference diagrams of five performance measures for k-means clustering using eight different elastic distances. The higher to the right better the performance. The horizontal bar indicates there no significant difference in performance in terms of the Friedman test.

5.2 K-medoids clustering

Following the k-means, the distances were also compared with the performance of the distances utilising k-medoids clustering. The k-medoids results were obtained utilising the `tsml-eval` Python code, with the `aeon 0.8.1` version package. For all these performance measures, the critical differences diagrams were plotted. The results can be found in Figure 2.

Upon examination of the results, it can be concluded that the ERP distance measure performs optimally in the k-medoids context. The ERP distance measure outperforms the other distances in every performance measure. A deviating result when looking at the replicated research performed by Holder et al. (2024). In their research, both the MSM and TWE demonstrate superior performance to the ERP distance measure. The result in their research that shows superior performance of the ERP is when considering datasets where the number of clusters is between six and ten. Given our research merely focuses on datasets with little number of clusters, Holder et al. (2024) results can not be a explanation for the difference in our results. However, we do see that the ERP never significantly outperforms the MSM, when looking at the horizontal lines in the diagrams.

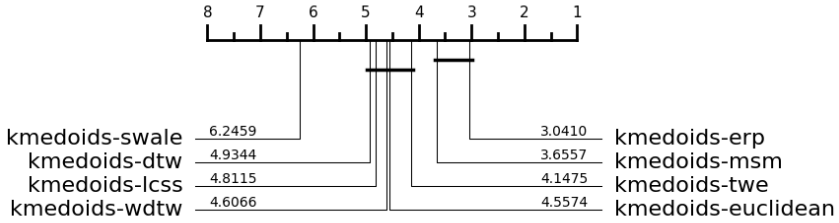
Upon further examination of the results, it can be observed that for k-medoids clustering, overall the same distance measures demonstrate superior performance compared to the Euclidean benchmark. As demonstrated in the Holder et al. (2024) research, the DTW distance measure outperforms the Euclidean benchmark in k-medoids clustering measures, in contrast to k-means clustering. In this research this is observed for three of the five performance measures. However, the DTW never significantly under- or outperforms the Euclidean distance. With regard to the DTW, another noteworthy comparison with the replicated research is the WDTW, which demonstrated superior performance relative to the unweighted version. This is similarly observed for three of the five performance measures in the results of this research.

In conclusion, the ERP distance performs the best when considering k-medoids clustering. However, the MSM distance does not significantly underperform. Comparing the results with those of Holder et al. (2024), it can be seen that the same distance measures outperform the Euclidean distance. However, when considered relative to each other, the ERP performs better than the other distance measures in this research.

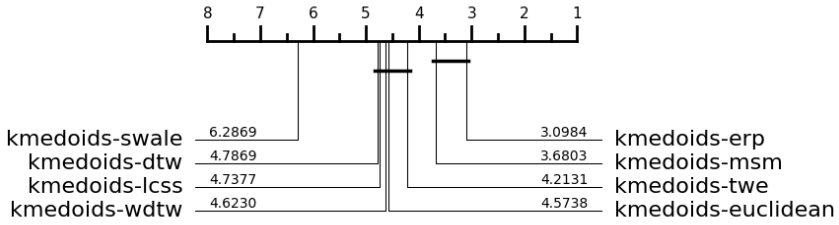
5.3 Initialisation results

In order to ascertain the impact of initialisation on the performance of the distances the initialisation k-means++ and k-medoids++ were explored. The kmeans++ results were obtained with the use of the `tsml-eval` python code with the `aeon 0.8.1` package. Another initialisation method considered was the Forgy's initialisation. Interestingly, this was only possible to use for the k-means, with the provided `tsml-eval` code of Holder et al. (2024). Therefore, this is limited the investigation to explore Forgy's initialisation for the k-means clustering algorithm only. Important for these results is that the code was run in a different `aeon` version, the `aeon 0.4.0` version, as the newer version did not include the Forgy's initialisation. To obtain the values for the k-medoids++ the same old `aeon 0.4.0` version was employed.

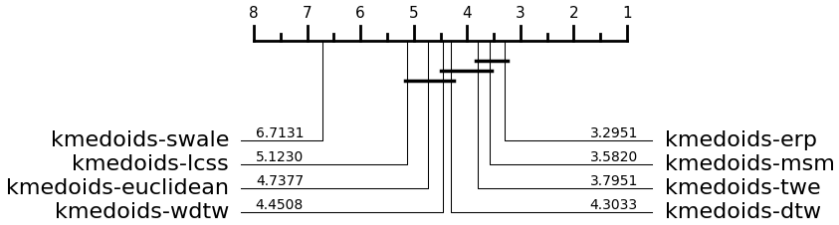
The results for the different initialisation of the methods for k-means and k-medoids are presented in Table 1. The other performance measures the results can be found in Appendix B.



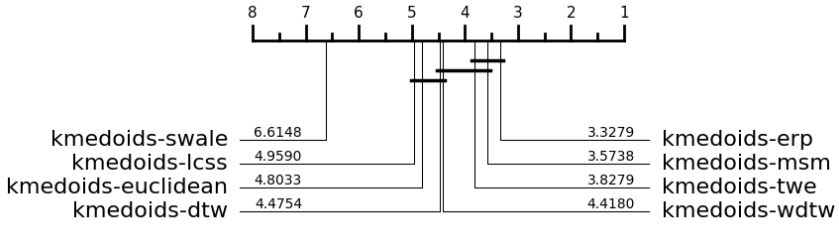
(a) Accuracy



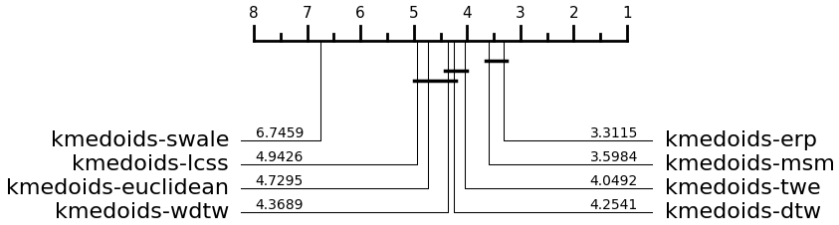
(b) ARI



(c) Mutual information



(d) AMI



(e) NMI

Figure 2: Critical difference diagrams of five performance measures for k-medoids clustering using eight different elastic distances. The higher to the right better the performance. The horizontal bar indicates there no significant difference in performance in terms of the Friedman test.

When examining the results, it can be concluded that when looking at clustering accuracy the initialisation method does not affect the best performing distance measure. There is a discrepancy in the values of the critical differences, yet the initialisation method does not influence the performance comparisons. This align with the findings of Holder et al. (2024), who concluded that the initialisation method affects performance, yet not when comparing distance measures.

	K-means			Kmedoids	
	Random	Kmeans++	Forgy	Random	Kmedoids++
MSM	**3.9262**	**3.4754**	**3.5164**	3.6557	3.5656
TWE	4.2131	4.0000	3.8852	4.1475	4.0000
ERP	4.1230	3.6967	3.6639	**3.0410**	**3.4180**
WDTW	4.3279	3.9836	3.9262	4.6066	3.7377
DTW	4.5738	4.0820	4.2705	4.9344	3.8361
ED	4.4836	4.2131	4.0820	4.5574	4.5656
LCSS	5.0902	4.5492	4.6557	4.8115	4.8770

Table 1: Critical difference between distance seven elastic distance measures considering performance measure clustering accuracy for different initialisation methods

5.3.1 Tuning Swale

In order to identify the optimal Swale distance, the parameters were tuned. As previously outlined in the methodology, the parameters that were modified were the window, reward, and the value of the cost gap. However, during the parameter tuning process, some challenges were encountered. One of the difficulties encountered was that for lower gap cost values, it was not possible to identify the optimal number of clusters within the set up of the maximum of 300 iterations and 10 restarts. Consequently, in this research, only gap cost values of 0.6, 0.8 and 1.0 times the reward were considered.

The window, reward and cost gap variables were initially tuned using k-means clustering. The average clustering accuracy of clustered datasets for all distances was then evaluated, and the results demonstrated that the combination of a gap cost with a size of 0.2 times the reward and a window of 1.0 resulted in the most accurate predicted clusters. In this tuning process, the reward size was also considered. However, the tuning results indicated that the size of the reward does not affect the accuracy of the clusterings, only the relationship between the reward and the gap cost. Thus, the value for reward was taken 50 times the standard deviation, in accordance with the methodology proposed by Salarpour and Khotanlou (2018).

Secondly, the Swale parameters were tuned for k-medoids clustering. Since, for k-means the research concluded that the reward value did not affect the performance. The reward value was taken as the same value used in the Salarpour and Khotanlou (2018) research, namely 50 times the standard deviation of the time series dataset. The remaining parameter values were determined using the same configuration as for the k-means, with the initial step being the identification of the window and gap cost. The results of the parameter tuning demonstrated that a gap of 0.8 and a window value of 0.6 yielded the highest overall clustering accuracy when considering the Swale distance measure for k-medoids clustering. A gap value of 0.8 signifies

that the gap costs are 0.8 times the reward. In Table 2 an overview of the tuning results are given.

Parameters	Interval	K-means	K-medoids
window	{0.2, 0.4, 0.6, 0.8, 1.0}	0.2	0.6
reward	{10, 20, 30, 40, 50}	50	50
gap value	{0.6, 0.8, 1.0}	1.0	0.8

Table 2: Parameter tuned for Swale distance when using k-means and k-medoids clustering.

5.4 Extension results

This paper builds upon the findings of Holder et al. (2024) by examining the Swale distance measure in greater depth. This distance is incorporated into the `aeon 0.8.1` package for the purpose of evaluating its performance. In order to ascertain whether the Swale performance is influenced by the clustering algorithm employed, the measure was tested using both k-means and k-medoids clustering.

As demonstrated in section 5.3 the initialisation method does not significantly affect the distance measure performances. Therefore the Swale distance measure performance has only been explored using the random initialisation. The results for the performance of the Swale distance when utilising k-means clustering are presented in Figure 1. A review of the performance results indicates that the Swale distance measure exhibits a suboptimal performance compared to all other evaluated distance measures. In terms of clustering accuracy, the Swale does not exhibit a notable deficiency in performance relative to the LCSS, DTW, and benchmark Euclidean distance, as evidenced by the horizontal line in the diagram. However, when considering the other performance measures, it becomes evident that the Swale distance measure significantly underperforms in comparison to the other distances. The Swale distance underperformance compared to the LCSS is not necessarily what we expected. The Swale distance penalises the gap as well as rewards the match, with this design we would expect the Swale to outperform the LCSS. However, this is not what can be observed from the results.

Upon examination of the k-medoids clustering results, it again becomes evident that the Swale exhibits a suboptimal performance compared to the other distances. A comparison of the k-means and k-medoids clustering results indicates that the performance of Swale is better when using k-means clustering.

Overall, the findings of this research indicate that the Swale distance is not an optimal distance measure for clustering univariate time series. This is in contrast to the findings of Salarpour and Khotanlou (2018), who indicated that for hierarchical clustering, the Swale distance measure was the most effective, outperforming the other distance measures considered in this research.

5.5 Performance analysis

To ascertain whether certain distance measures are more suitable for assessing distinct clustering problems, such as those involving a limited number of clusters or those with a larger number of

clusters. The data was divided into four groups, each with a different number of clusters. The evaluation of these datasets yielded the following results. The results are presented in Table 3 for the k-means clustering.

It is noteworthy that the Swale distance exhibits comparable performance to LCSS and superior performance to DTW and WDTW when evaluating datasets group A, characterised by 1-2 clusters, in terms of average rank for performance measure accuracy. However, this enhanced performance is not statistically significant. Upon closer examination of the data, the Swale distance measure demonstrates similar or superior performance to the LCSS distance measure across three of four distinct datasets. In dataset B, it exhibits suboptimal performance. It is important to note that the results for datasets C and D should be interpreted with caution, as the datasets are relatively small. In contrast, datasets A and B include a greater number of time series datasets, thereby increasing the reliability of the results. Further research is required to ascertain whether the Swale distance is indeed more effective than the LCSS in small or high number of cluster problems in order to draw valid conclusions. With regard to the other distance measures, it can be observed that the ERP performs most effectively in clustering problems with a small number of clusters, which is consistent with the findings reported by Holder et al. (2024). For datasets in group B, the WDTW is the most successful distance measure, which is also in accordance with the results of the replicated research by Holder et al. (2024).

		MSM	TWE	ERP	WDTW	DTW	Euclidean	LCSS	Swale
A	(1-2 clusters)	4.1000	4.3333	**3.9667**	4.9667	4.9667	4.1667	4.7500	4.7500
B	(3-5 clusters)	3.8913	4.3043	4.5217	**3.5870**	4.3478	4.7174	4.6987	6.0217
C	(6-10 clusters)	4.0000	4.1667	3.5833	3.7500	**3.3333**	5.0000	7.6667	4.5000
D	(11-15 clusters)	**1.5000**	**1.5000**	3.5000	5.0000	5.0000	5.0000	8.0000	6.5000

Table 3: K-means clustering critical difference of eight distance measure considering cluster accuracy performance for four different groups of datasets.

In addition, the performance of the distances for the four datasets is evaluated in the context of k-medoids clustering. The results are presented in Table 4. When examining the performance, it can be seen that the ERP performs better in clustering problems with the low number of clusters. This is in contrast to the findings for k-means, not inline with the results obtained for k-medoids clustering by Holder et al. (2024), where the TWE and MSM were identified as the most performing distances for the first two groups of datasets. Another noteworthy outcome is that, when considering the rank of the clustering accuracy, the DTW distance measure performs the best for the dataset with six to ten clusters, Holder et al. (2024) found the ERP the best performing distance measure in this group of datasets. One possible explanation for this is that the datasets group C is very small and therefore is the analysis not reliable. This also applies to the analysis for datasets group D.

The results of this performance analysis indicate that the number of clusters in the clustering problem does affect which distance performs the best. However, to make a credible statement, further research on larger datasets is necessary, particularly for the Swale distance measure.

		MSM	TWE	ERP	WDTW	DTW	Euclidean	LCSS	Swale
A	(1-2 clusters)	3.5500	4.3000	**3.3500**	4.8333	5.7500	4.2167	4.5833	5.4167
B	(3-5 clusters)	4.0000	3.9783	**2.7391**	4.8043	4.5000	4.8478	4.2391	6.8913
C	(6-10 clusters)	3.5833	4.0833	3.0000	3.0833	**2.5000**	5.0000	7.2500	7.5000
D	(11-15) clusters	**1.5000**	4.0000	2.0000	3.5000	5.0000	7.5000	7.5000	6.5000

Table 4: K-medoids clustering critical difference of eight distance measure considering cluster accuracy performance for four different groups of datasets.

6 Conclusion

This study has examined the performance of distance measures for partitional clustering k-means and k-medoids. The analysis commenced with a replication of the results obtained in the research performed by Holder et al. (2024). Due to the different datasets, the replications of this research are not identical. However, it is anticipated that the replication will draw the same conclusions. Following the replication, the impact of initialisation methods on distance measures performances was explored. Subsequently, this study builds upon the findings of Holder et al. (2024) by incorporating an additional distance measure to assess its performance. This additional distance measure is the Swale distance measure, which has previously demonstrated promising results in empirical studies conducted by Salarpour and Khotanlou (2018).

The replication analysis demonstrated that among the distance measures, the ERP and MSM exhibited the optimal performance. With regard to k-means clustering, the MSM demonstrated superior performance across all performance measures. Moreover, when considering k-medoids clustering, the ERP is the most effective distance measure, with the MSM performing almost as well, the MSM never significantly underperforms the ERP.

Furthermore, the analysis of the impact of initialisation methods revealed that the initialisation does affect the outcome of the performance measures. However, it does not influence the comparison of the distance measures. Therefore, it can be concluded that the initialisation method does not influence the results when answering the question of which distance measure performs the best. Consequently, the remaining analysis results were obtained using the random initialisation method.

To test the performance of the extension distance, Swale, the parameters reward, window and gap cost were tuned. With the tuned parameters, the Swale distance measure performance was evaluated. The results demonstrated that the Swale distance measure underperforms the other distance measures. For k-means as well as for k-medoids clustering. From these results it can therefore be concluded that for univariate time series the Swale distance is not a good distance measure when using partitional clustering k-means and k-medoids. So, when answering the subquestion does the Swale distance measure outperform the elastic distance measures proposed by Holder et al. (2024), it can be answered negatively. In answering the two remaining sub-questions, it can be concluded that when considering k-means clustering, the MSM distance measure performs best. Additionally, when considering k-medoids clustering, the ERP distance measure performs best. However, it is important to consider that the type of clustering datasets as well as the clustering algorithm have an influence on which distance measure performs best.

In conclusion, this research builds upon the findings of Holder et al. (2024) by replicating the results and extending the analysis to compare the performance of different elastic distance measures. This comparison was conducted to identify the optimal distance measure for partitioning clustering algorithms when clustering univariate time series. The findings indicate that the MSM is the best performing for k-means and ERP for k-medoids clustering. Furthermore, the Swale distance measure is not a suitable distance measure for this purpose.

For future research, it would be beneficial to evaluate distances on additional datasets. In this research project, the computational power required was a significant limitation. In particular, the performance analysis was not particularly reliable due to the limited number of datasets. The reliability of the performance analysis could be enhanced by evaluating additional datasets. Furthermore, it would be beneficial to investigate the distances using a greater number of restarts. In this research, the number of restarts was limited to 10 due to computational power. Additionally, literature indicates that a greater number of restarts are typically conducted. This would enhance the credibility of the research and potentially lead to improved tuned variables for the Swale distance, resulting in a more effective distance measure.

References

Aghabozorgi, S., Shirkhorshidi, A. S. & Wah, T. Y. (2015). Time-series clustering–a decade review. *Information systems*, *53*, 16–38.

Arthur, D., Vassilvitskii, S. et al. (2007). k-means++: The advantages of careful seeding. In *Soda* (Vol. 7, pp. 1027–1035).

Babu, M. S., Geethanjali, N. & Satyanarayana, B. (2012). Clustering approach to stock market prediction. *International Journal of Advanced Networking and Applications*, *3*(4), 1281.

Bagnall, T. & contributors. (2024). *tsml-eval: Evaluation tools for time series machine learning algorithms.* `https://github.com/time-series-machine-learning/tsml-eval`. (Accessed: 2024-04-29)

Chen, L. & 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* (pp. 792–803).

Cover, T. M. (1999). *Elements of information theory.* John Wiley & Sons.

Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., ... Hexagon-ML (2018, October). *The ucr time series classification archive.* `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`. (Accessed: 2024-04-29)

Fasulo, D. (1999). *An analysis of recent work on clustering algorithms* (Tech. Rep.). Technical report.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, *21*, 768–769.

Holder, C., Middlehurst, M. & Bagnall, A. (2024). A review and evaluation of elastic distance functions for time series clustering. *Knowledge and Information Systems*, *66*(2), 765–809.

Huang, D. & Pan, W. (2006). Incorporating biological knowledge into distance-based clustering analysis of microarray gene expression data. *Bioinformatics*, *22*(10), 1259–1268.

Hubert, L. & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*, 193–218.

Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B. & Heming, J. (2023). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, *622*, 178–210.

Jeong, Y.-S., Jeong, M. K. & Omitaomu, O. A. (2011). Weighted dynamic time warping for time series classification. *Pattern recognition*, *44*(9), 2231–2240.

Kaufman, L. & Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis.* John Wiley & Sons.

Lines, J. & Bagnall, A. (2015). Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, *29*, 565–592.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297).

Marteau, P.-F. (2008). Time warp edit distance with stiffness adjustment for time series matching. *IEEE transactions on pattern analysis and machine intelligence*, *31*(2), 306–318.

Middlehurst, M., Large, J., Flynn, M., Lines, J., Bostrom, A. & Bagnall, A. (2021). Hive-cote 2.0: a new meta ensemble for time series classification. *Machine Learning*, *110*(11),

3211–3243.

Morse, M. D. & Patel, J. M. (2007). An efficient and accurate method for evaluating time series similarity. In *Proceedings of the 2007 acm sigmod international conference on management of data* (pp. 569–580).

Müller, M. (2007). Dynamic time warping. *Information retrieval for music and motion*, 69–84.

Nugent, R. & Meila, M. (2010). An overview of clustering applied to molecular biology. *Statistical methods in molecular biology*, 369–404.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, *66*(336), 846–850.

Reddy, C. K. & Vinzamuri, B. (2018). A survey of partitional and hierarchical clustering algorithms. In *Data clustering* (pp. 87–110). Chapman and Hall/CRC.

Salarpour, A. & Khotanlou, H. (2018). An empirical comparison of distance measures for multivariate time series clustering. *International Journal of Engineering*, *31*(2), 250–262.

Sonagara, D. & Badheka, S. (2014). Comparison of basic clustering algorithms. *Int. J. Comput. Sci. Mob. Comput*, *3*(10), 58–61.

Stefan, A., Athitsos, V. & Das, G. (2012). The move-split-merge metric for time series. *IEEE transactions on Knowledge and Data Engineering*, *25*(6), 1425–1438.

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

A Elastic distance measures algorithms

In the sections below the algorithm employed in this research for all the elastic distances will be showed. WDTW is the only distance measure does not have a algorithm since it is obtained using the DTW distance measure algortihm with a restricted w. The distance measure algorithms of DTW, LCSS, ERP, MSM, TWE and Swale will be showed below.

A.1 DTW algoritm

Algorithm 1 illustrates the methodology employed to calculate the Dynamic time warping distance between two time series, denoted by **a** and **b**. The input parameters are the window, which is set to one, and the pointwise distance matrix M. The algorithm returns the DTW distance, which is the final element of the matrix C. The algorithm is identical to that employed in the replicated research by Holder et al. (2024).

Algorithm 1 DTW ($\mathbf{a}, \mathbf{b}$, (both series of length m), w (window proportion, default value $w \leftarrow 1$), M (pointwise distance matrix))

Let C be an $(m+1) \times (m+1)$ matrix initialized to zero, indexed from zero.
for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to m **do**
    if $|i - j| < w \cdot m$ **then**
      $C_{i,j} \leftarrow \mathbf{M}_{i,j} + \min(C_{i-1,j-1}, C_{i-1,j}, C_{i,j-1})$
    end if
  end for
end for
return $C_{m,m}$

A.2 LCSS algorithm

The following algorithm is used to calculate the Longest common subsequence(LCSS). The input parameters are the two time series, denoted by **a** and **b**. Furthermore, the threshold, denoted by ϵ, is an input. In this research, the threshold is set at 0.05, which is similar to the replicated research by Holder et al. (2024). The algorithm is identical to that employed in the replicated research. Returned is the last element of matrix L, this is not the LCSS distance.

Algorithm 2 LCSS (a, b, (both series of length m), ϵ (equality threshold))

Let L be an $(m+1) \times (m+1)$ matrix initialized to zero, indexed from zero.
for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to m **do**
    if $|a_i - b_j| < \epsilon$ **then**
      $L_{i,j} \leftarrow L_{i-1,j-1} + 1$
    else
      $L_{i,j} \leftarrow \max(L_{i-1,j}, L_{i,j-1})$
    end if
  end for
end for
return $L_{m,m}$

A.3 ERP algorithm

The algorithm described in Algorithm 3 constructs the edit with real penalty (ERP) distance. The algorithm returns the distance between series **a** and **b** . With a penalty value of 0.05. The pointwise distance function is represented by the symbol d. The algorithm employed to ascertain the ERP distance is identical to that described in the Holder et al. (2024) review.

Algorithm 3 ERP (**a**, **b**, (both series of length m), g (penalty value), d (pointwise distance function))

Let E be an $(m+1) \times (m+1)$ matrix initialized to zero, indexed from zero.
for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to m **do**
    if $i = 0$ **then**
      $E_{i,j} \leftarrow \sum_{k=1}^{m} d(b_k, g)$
    else if $j = 0$ **then**
      $E_{i,j} \leftarrow \sum_{k=1}^{m} d(a_k, g)$
    else
      $match \leftarrow E_{i-1,j-1} + d(a_i, b_j)$
      $insert \leftarrow E_{i-1,j} + d(a_i, g)$
      $delete \leftarrow E_{i,j-1} + d(g, b_j)$
      $E_{i,j} \leftarrow \min(match, insert, delete)$
    end if
  end for
end for
return $E_{m,m}$

A.4 MSM algorithm

The algorithm obtained to determine the Move-split-merge (MSM) distance is constructed in Algorithm 4, This algorithm is analogous to the algorithm employed in the Holder et al. (2024) research. The algorithm is used to calculate the distance between two given time series, designated as **a** and **b**. In order to determine the distance, the algorithm also utilises the parameters c, which represent the minimum cost, in our research 1.0, and d, the pointwise distance function. These are provided as inputs to the algorithm. The algorithm then returns the MSM distance, which is the final element of the D matrix.

A.5 TWE algorithm

The Time-warping edit (TWE) distance is constructed in Algorithm 5. The algorithm determines the TWE distance between two time series, denoted by **a** and **b**. The algorithm accepts three input parameters: the edit cost, denoted by λ; the warping penalty factor and is set to 1.0, denoted by ν and is set to 0.05; and the pointwise distance function, denoted by d. The algorithm ultimately returns the TWE distance, which is the final element in matrix D.

Algorithm 4 MSM (a, b (both series of length m), c (minimum cost), d (pointwise distance function))

Let D be an $m \times m$ matrix initialized to zero.
$D_{1,1} = d(a_1, b_1)$
for $i \leftarrow 2$ to m **do**
  $D_{i,1} = D_{i-1,1} + \text{cost}(a_i, a_{i-1}, b_1, c)$
end for
for $i \leftarrow 2$ to m **do**
  $D_{1,i} = D_{1,i-1} + \text{cost}(b_i, a_1, b_{i-1}, c)$
end for
for $i \leftarrow 2$ to m **do**
  for $j \leftarrow 2$ to m **do**
    $match \leftarrow D_{i-1,j-1} + d(a_i, b_j)$
    $insert \leftarrow D_{i-1,j} + \text{cost}(a_i, a_{i-1}, b_j, c)$
    $delete \leftarrow D_{i,j-1} + \text{cost}(b_j, b_{j-1}, a_i, c)$
    $D_{i,j} \leftarrow \min(match, insert, delete)$
  end for
end for
return $D_{m,m}$

Algorithm 5 TWE (a, b (both series of length m), λ (edit cost), ν (warping penalty factor), d (pointwise distance function))

Let D be an (m + 1) × (n + 1) matrix initialized to 0
$D_{0,0} = 0$
for i $\leftarrow 1$ to m **do**
  $D_{i,0} = \infty$
  $D_{0,i} = \infty$
end for
for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to n **do**
    $match \leftarrow D_{i-1,j-1} + d(a_i, b_j) + d(a_{i-1}, b_{j-1}) + 2\nu(|i - j|)$
    $delete \leftarrow D_{i-1,j} + d(a_i, a_{i-1}) + \lambda + \nu$
    $insert \leftarrow D_{i,j-1} + d(b_j, b_{j-1}) + \lambda + \nu$
    $D_{i,j} \leftarrow \min(match, delete, insert)$
  end for
end for
return $D_{m,n}$

A.6 Swale algorithm

The final algorithm employed in this research is Algorithm 6. This algorithm returns the Swale distance between two time series, designated as **a** and **b**. The input to this algorithm besides the time series comprises three elements: $reward$, which is the match reward; $gapCost$, which is the cost of a gap; and finally, the equality threshold, denoted by the symbol ϵ, set at 0.05. The algorithm returns the Swale distance, which is the final element of the S matrix.

Algorithm 6 Swale (a, b, (both series of length m), reward, gapCost, ϵ (equality threshold))

Let L be an $(m+1) \times (m+1)$ matrix initialized to zero, indexed from zero.
$S_{0,j} = n \times gapCost$, $\forall j$
$S_{i,0} = m \times gapCost$, $\forall i$
for $i \leftarrow 1$ to m **do**
  for $j \leftarrow 1$ to m **do**
    if $|a_i - b_j| < \epsilon$ **then**
      $S_{i,j} \leftarrow S_{i-1,j-1} + reward$
    else
      $S_{i,j} \leftarrow \max(S_{i-1,j} + gapCost, S_{i,j-1} + gapCost)$
    end if
  end for
end for
return $S_{m,m}$

B Initialisation results

For the k-means and k-medoids clustering with random initialisation method can be found in Figure 1 and 2 in the main report. For the other initialisation methods the results can be found in the appendix sections below.

B.1 k-means clustering with kmeans++ initialisation

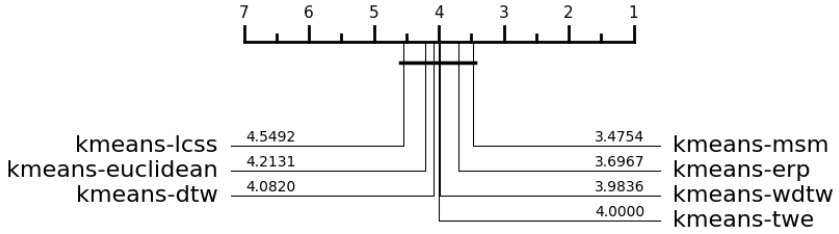
In Figure 3 the results of k-means clustering with the kmeans++ initialisation method for the five different performance measures are shown.

B.2 k-means clustering with forgy initialisation

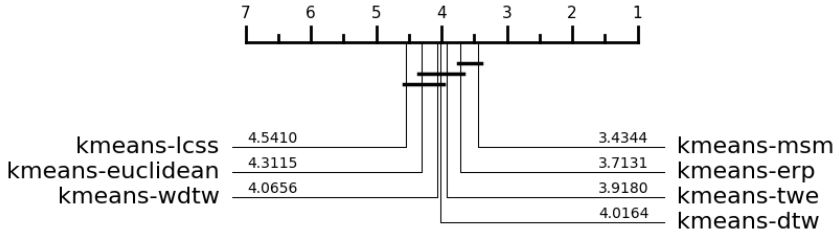
In Figure 4 the results of k-means clustering with the Forgy's initialisation method for the five different performance measures are shown.

B.3 k-medoids clustering with kmedoids++ initialisation

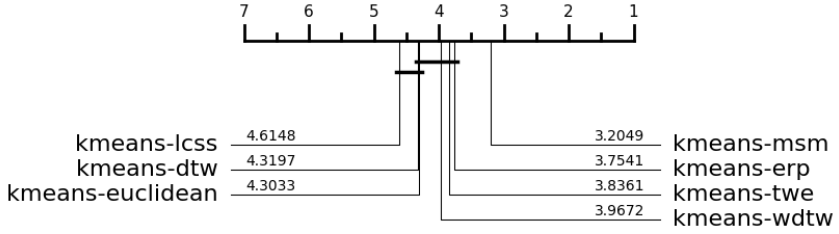
The performance measure results for k-medoids clustering with kmedoids++ initialisation are presented in Figure 5



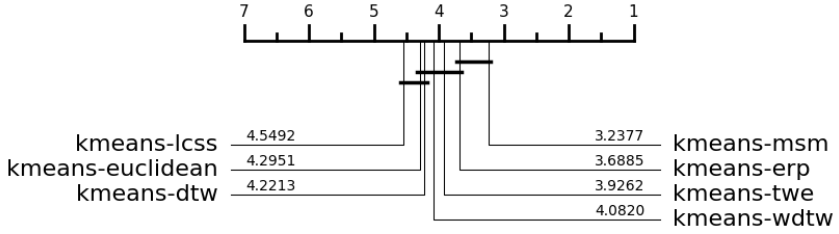
(a) Accuracy



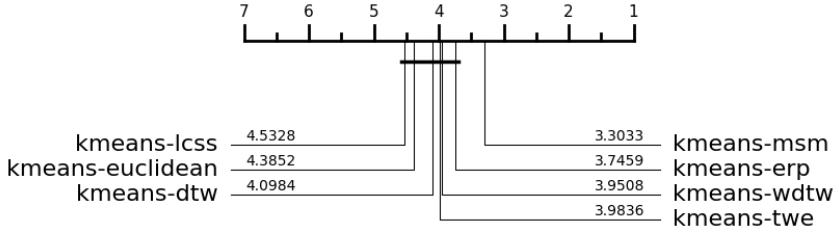
(b) ARI



(c) Mutual information

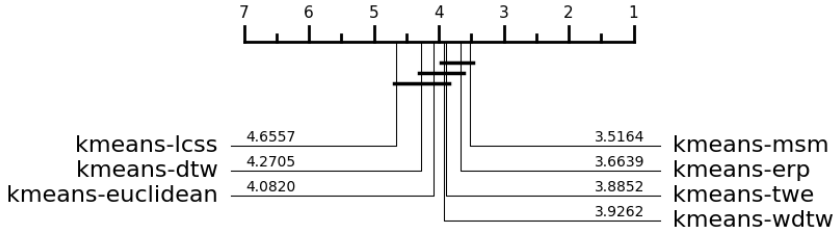


(d) AMI

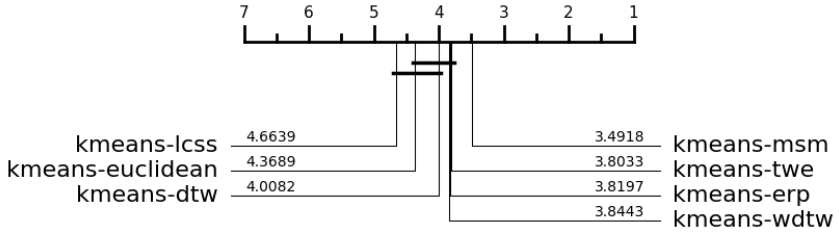


(e) NMI

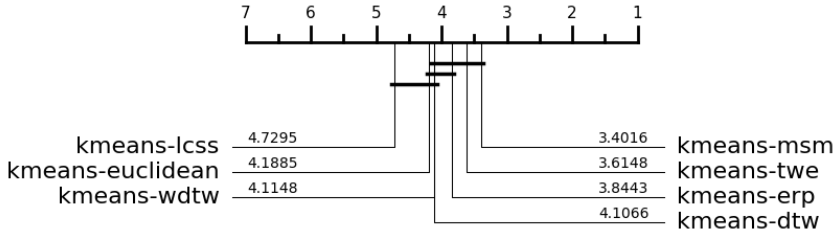
Figure 3: Critical difference diagrams of five performance measures for k-means clustering for seven different elastic distances obtained with the kmeans++ initialisation method. The higher to the right better the performance. The horizontal bar indicates there no significant difference in performance in terms of the Friedman test.



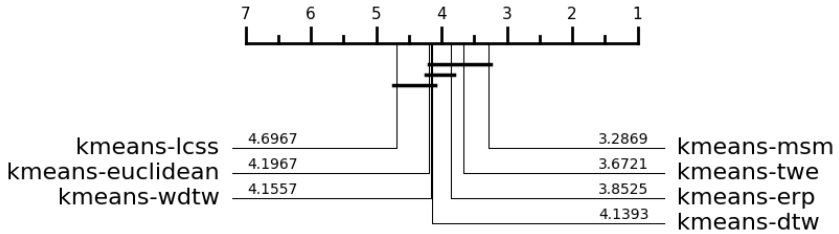
(a) Accuracy



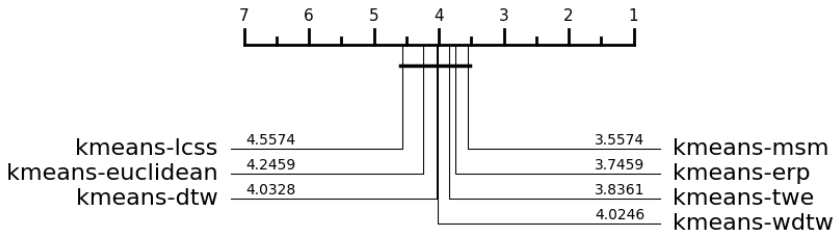
(b) ARI



(c) Mutual information

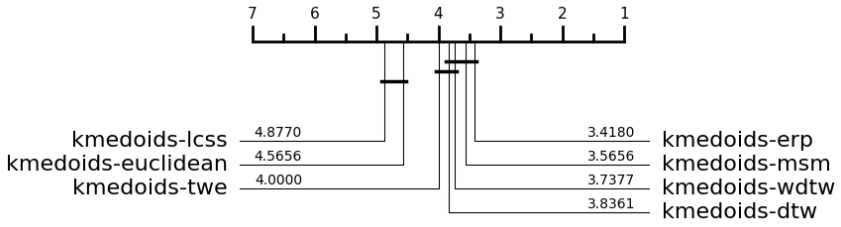


(d) AMI

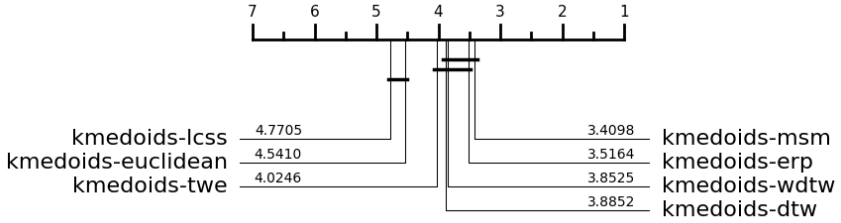


(e) NMI

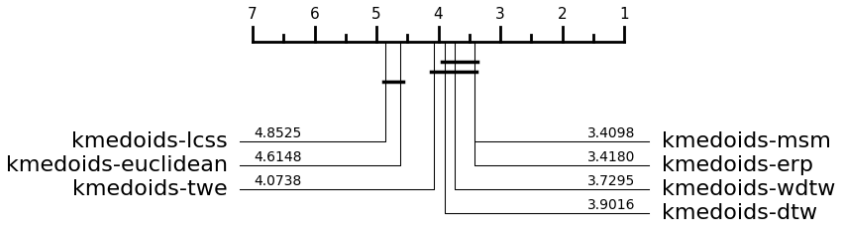
Figure 4: Critical difference diagrams of five performance measures for k-means clustering for seven different elastic distances obtained with the Forgy's initialisation method. The higher to the right better the performance. The horizontal bar indicates there no significant difference in performance in terms of the Friedman test.



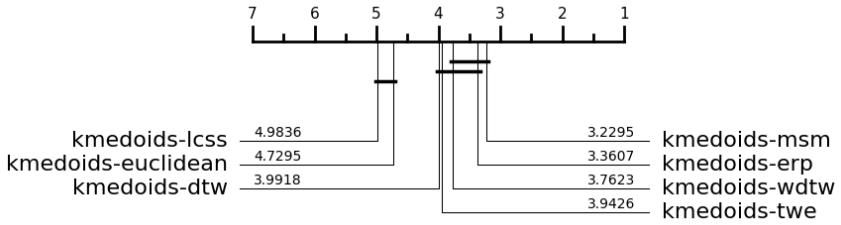
(a) Accuracy



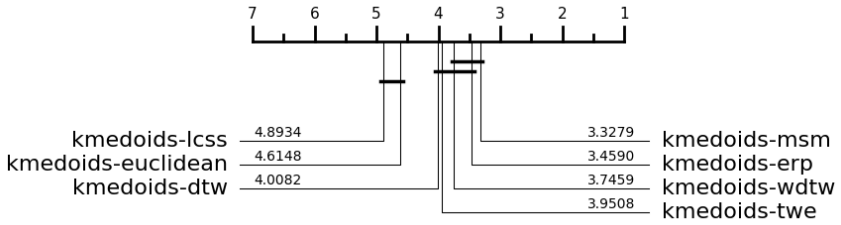
(b) ARI



(c) Mutual information



(d) AMI



(e) NMI

Figure 5: Critical difference diagrams of five performance measures for k-medoids clustering for seven different elastic distances obtained with the kmedoids++ initialisation method. The higher to the right better the performance. The horizontal bar indicates there no significant difference in performance in terms of the Friedman test.

C Programming code

The results in this research were obtained with the use of Python packages `aeon` and the `tsml-eval` package provided by Holder et al. (2024). In this process two Python version with different `aeon` version packages were utilised. Python 3.10 was utilised to obtain the results of the k-means clustering with Forgy's initialisation and the results for k-medoids clustering with k-medoids++ as initialisation. The rest of the results were obtained using Python 3.12. The results were obtained by running the code on a Apple M2 computer equipped with 8GB RAM. A full description of how to obtain the results showed in this report can be found in the attached PDF file called *Thesis_code_description.pdf*