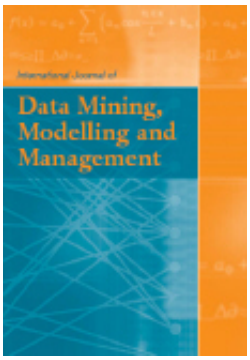


International Journal of Data Mining, Modelling and Management

ISSN online: 1759-1171 - ISSN print: 1759-1163
https://www.inderscience.com/ijdmmm

Sorting paired points: a dissimilarity measure based on sorting of series

Wallace Anacleto Pinheiro, Ricardo Q.A. Fernandes, Ana Bárbara Sapienza Pinheiro

DOI: 10.1504/IJDMMM.2025.10065723

Article History:

Received:	05 January 2024
Last revised:	13 April 2024
Accepted:	27 April 2024
Published online:	25 February 2025

Sorting paired points: a dissimilarity measure based on sorting of series

Wallace Anacleto Pinheiro* and
Ricardo Q.A. Fernandes

Systems Development Center,
Brazilian Army,
Brasília, DF, Brazil
Email: wallaceapinheiro@gmail.com
Email: ricardo.fernandes@eb.mil.br
*Corresponding author

Ana Bárbara Sapienza Pinheiro

Department of Tropical Medicine,
Brazilian University,
Brasília, DF, Brazil
Email: absapienza@gmail.com

Abstract: We propose a new dissimilarity measure, sorting different time series and measuring their absolute and relative degree of disorganisation. This work compares this strategy with the state-of-the-art of dissimilarities or similarities measures, such as DTW, maximal information coefficient (MIC) and complexity-invariant distance (CID). Two clustering algorithms, one deterministic and one non-deterministic, K-means and hierarchical, allow us to analyse their results. To infer the accuracy, we use two different indexes, maximal HITS, and adjusted Rand index. The results of the experiments, over 128 different datasets, demonstrate that the proposed approach provides more accurate results for different domains using the proposed metrics.

Keywords: clustering; similarity; time series; entropy; sorting.

Reference to this paper should be made as follows: Pinheiro, W.A., Fernandes, R.Q.A. and Pinheiro, A.B.S. (2025) 'Sorting paired points: a dissimilarity measure based on sorting of series', *Int. J. Data Mining, Modelling and Management*, Vol. 17, No. 1, pp.1–25.

Biographical notes: Wallace Anacleto Pinheiro obtained his Doctorate in Systems and Computer Engineering in 2010 from the Federal University of Rio de Janeiro (UFRJ). From 2010 to 2014, he was a Professor at the Military Engineering Institute (IME). In 2019, he completed his Post-doctorate at UFRJ. He also worked at the Brazilian Army Systems Development Center (CDS) for many years. Currently, he is engaged in databases, data mining, and artificial intelligence.

Ricardo Q.A. Fernandes completed his Post-doctoral studies at George Mason University in 2016, specialising in the field of Command and Control. His professional goals involve a diverse range of interests, with a particular focus on data mining, artificial intelligence, and command and control methodologies.

Ana Bárbara Sapienza Pinheiro is a Postgraduate in Neonatal Intensive Care from Universidade Federal Fluminense (UFF) in 2003. She obtained her Master's degree in Tropical Medicine in Molecular Biology from the University of Brasília (UnB) in 2019. Currently, her interests involve epidemiology, biostatistics, and information technologies focused on health.

1 Introduction

Dissimilarity and similarity measures are used in several areas to solve different problems. One of their relevant applications involves assisting in clustering strategies. These measures apply concepts related to correlation, association, and entropy to provide different visions of data relations. They deal with monotonic and non-monotonic functions and linear and nonlinear dependence between variables. Some are restricted to specific domains, and few can provide good results in other contexts. In this scenario, time series frequently provide data from several areas that are non-monotonic, having nonlinear dependence, demanding adaptable and generic approaches.

A time series can be defined as an ordered sequence of observations usually through time (Mastrangelo et al., 2013). It also can be interpreted as a set of ordered observations on a quantitative characteristic of a phenomenon at equally spaced time points (Ivanović and Kurbalija, 2016). We can compare time series to detect their similarity to provide better analysis. In this sense, it is possible to use different similarity/dissimilarity measures, such as DTW, maximal information coefficient (MIC), mutual information (MI), and complexity-invariant distance (CID), among others. Time series classification uses different techniques in the area of data mining. Similarity functions are also important in a number of approaches (Lin et al., 2022; Yang et al., 2022).

In this work, we propose a new dissimilarity measure that uses the concept of sorting and difference between values point to point, providing a malleable and comprehensive strategy. We compare our approach to the most used and relevant techniques applied to the clustering time series. Therefore, we apply these measures to two well know clustering algorithms: k-means and hierarchical, and we use a set of 64 time series datasets from different domains to compare the results. Besides, we use two different metrics to rate the results obtained by each technique. To compare the results, we use two different approaches: one compares the proposed algorithm to each other algorithm (relative approach); the other compares all the best results of each algorithm generating a ranking.

2 Background and related works

Dissimilarities and similarities support clustering strategies (Xu and Tian, 2015; Pham et al., 2011) applied to time series (Liao, 2005). In this context, we can find different

techniques (Liao, 2005; Aghabozorgi et al., 2015). Other authors have compared some of these techniques (Wang et al., 2013). However, research is still needed seeking further improvements in this area.

Euclidean distance is the sum of the smallest distance between paired points of two time series of the same size (Berthold and H"oppner, 2016). Despite its simplicity, Euclidean distance continues to get accurate results (Serrà and Arcos, 2014). Manhattan distance measures the sum of the absolute difference of the x-axis and y-axis values of paired points in Euclidean space (Strauss and von Maltitz, 2017). To be considered a distance, dissimilarities obey the triangular inequality or, in some cases, the relaxed triangular inequality (Batista et al., 2014; Niu et al., 2020).

Pearson's similarity or correlation estimates the linear dependence between two variables. Spearman's similarity, differently from Pearson's similarity, is a non-parametric measure that values the intensity and direction of the relation between non-parametric ordinal variables. Similarly, Kendall's similarity allows valuing the ordinal correlation between two non-parametric variables. The normalised values of these three similarities measures range from -1 to 1. Positive values are related to the direct relationships, and negative values identify the inverse relationships (Kumari et al., 2012; Miot, 2018). Another similarity measure, cosine similarity (Shirkhorshidi et al., 2015; Park et al., 2020), uses the angle between vectors, allowing to deal with high-dimensional spaces. All these similarities can be transformed in dissimilarity measures considering that higher values of inverse relationships represent more dissimilar series. In this context, frequently, it is used the conversion of 1 minus the similarity value.

Other different strategies used to measure the association between series involve mutual information (MI) (Cover and Thomas, 2005; Zhang et al., 2018) and a derived measure, maximal information coefficient (MIC) (Reshef et al., 2011). MI is derived from the concept of entropy and measures the information that one variable has about another variable. It is a kind of measure of dependence between two variables, always non-negative, with its normalised version results (strength between input variables) ranging from 0 to 1. MIC uses MI to measure the association. It draws grids with different resolutions, using a heuristic, on the scatter plot of the variables and calculates for every pair of integers (x, y) the highest possible MI obtained by any x-by-y grid applied to the data. Then, they use this set of MI to calculate MIC. MIC, as well as MI, is very versatile, being used for monotonic and non-monotonic associations.

On the other hand, dynamic time warping (DTW) approach can provide a dissimilarity measure that uses dynamic programming to find the optimal alignment between time series (Berndt and Clifford, 1994; Jeong et al., 2011; Wang et al., 2013; Keogh and Ratanamahatana, 2005; Silva and Batista, 2016). This technique yielded relevant results in different scenarios of application. DTW and variations (Lahreche and Boucheham, 2021; Abanda et al., 2019) provide accurate results for long and short time series. Due to its good results in different contexts, DTW is often used as a benchmark. Then, it is relevant to compare the results of new algorithms to this strategy. While this, UCR time series datasets (Lahreche and Boucheham, 2021; Abanda et al., 2019; Dau et al., 2018) are also used in many works, allowing us to compare their results. In our experimental comparison, we use DTW and UCR time series to compare our results to other techniques.

It is common sense in the literature that a good strategy to treat amplitude or offset invariance is z-normalisation. Some authors consider that a good indicator of a

dissimilarity measure is the invariance it can achieve. For example, offset or amplitude invariance, phase invariance, local scaling invariance, uniform scaling invariance, etc. From that idea, CID emerged (Batista et al., 2014; Niu et al., 2020). However, in the real world, the differences in the series may be crucial to classify them correctly. Therefore, evaluating the difference among time series involves giving the right value to the differences than despising them. Sometimes, the differences are, for example, just noise or some delay in time, and sometimes not. Even if contributing just a little, they should count toward the final value. Otherwise, invariance may reduce accuracy.

Another interesting strategy is order-preserving Wasserstein distance (OPWD) for sequence matching (Su and Hua, 2017), which effectively handles local temporal distortion and periodic sequences with arbitrary starting points, this method considers the calculation of sequence distance as an optimal transport problem, with two temporal regularisation terms that smooth out the problem. Additionally, a prior distribution regularisation utilising the KL-divergence inhibits transport between instances with far temporal positions.

Usually, clustering strategies use some of the measures presented previously to solve several problems (Xu and Tian, 2015; Pham et al., 2011), including time series grouping (Liao, 2005). Hierarchical clustering (Murtagh and Contreras, 2011, 2017; Ah-Pine, 2018; Yuan et al., 2022) commonly uses agglomeration methods to group data because they are less expensive than divisive methods. Another clustering technique is K-means (Hartigan and Wong, 1979; Pérez-Ortega et al., 2018; Kogan et al., 2005; Yuan et al., 2022), which originally minimises the average squared Euclidean distance of data from their centres. For time series, the centre is given by a set of points belonging to the time series of reference (named in the literature as a prototype), representing the centre. It is possible to apply K-means using different dissimilarity or similarity measures.

Time series from the same group tend to have similar formats; therefore, their degree of disorganisation generally tends to be consistent. Consequently, a measure that can better reflect this criterion tends to yield superior results. We propose a generic strategy that outperforms other researched strategies, but it is not universally superior across all researched datasets, which would indicate a specific solution prone to overfitting. Instead, it excels in a significant portion of the researched scenarios. Therefore, up to now, theoretically, there is no solution that can universally outperform others in all situations.

3 A new dissimilarity measure based on disorganisation

The proposed measure considers the difference between the values of time series paired points, as well as the relative disorganisation between these time series, using sorting. This strategy is related to the concept of entropy, used in MI and MIC measures, but the process to reach the results and the found values are different. Initially, we consider the following definitions:

$$\mathcal{S} = (s_1, s_2, \ldots, s_n) \tag{1}$$

$$\mathcal{T} = (t_1, t_2, \ldots, t_n) \tag{2}$$

where $\mathcal{S}$ and $\mathcal{T}$ are time series, s_i is a value of an element of $\mathcal{S}$ at the position i, t_i is a value of an element of $\mathcal{T}$at the position i, $i \in \{1, 2, ..., n\}$. Both time series

have the same size equals n. As a premise, we consider that series with similar values and behaviours (synchronism of relative increments or decrements) should have lower dissimilarity. For this, the proposed dissimilarity considers two dimensions.

The first dimension provides the absolute degree of disorganisation between the series and their respective growing versions. The second dimension provides the absolute degree of disorganisation between the series and their respective decreasing versions.

To build a measure for disorganisation between the series, we use sorted versions of the original time series $\mathcal{S}$ and $\mathcal{T}$: namely, their respective growing and decreasing versions:

$$\mathcal{U} = (u_1, u_2, \ldots, u_n) \tag{3}$$

$$\mathcal{V} = (v_1, v_2, \ldots, v_n) \tag{4}$$

$$\mathcal{W} = (w_1, w_2, \ldots, w_n) \tag{5}$$

$$\mathcal{Z} = (z_1, z_2, \ldots, z_n) \tag{6}$$

where

- $\mathcal{U}$ is the growing version of $\mathcal{S}$
- $\mathcal{V}$ is the growing version of $\mathcal{T}$
- $\mathcal{W}$ is the decreasing version of $\mathcal{S}$
- $\mathcal{Z}$ is the decreasing version of $\mathcal{T}$.

To evaluate those disorganisation degrees, we use the concept of correspondent indexes. The indexes of a time series represent a value that occurs at a given time. Correspondent indexes from different time series represent the values that occur in the same time instant.

The establishment of corresponding indexes is realised by a bijective function from a given time series index into another time series index. To define that bijective function, we need to synchronise, in some way, the indexes of both time series. It means that we would need to know the time distribution of the value collection for both time series previously. To avoid this dependency on information outside the time series values, we suppose that whenever two time series have the same size, their indexes are in direct correspondence. Thus, from the source time series, s_i corresponds to t_i, $i \in \{1, 2, ..., n\}$.

But, when we sort the time series, we will need to manipulate the original corresponding indexes. For that reason, we provide an explicit representation of the permutations of the corresponding indexes of the original series. Thus,

- α is the permutation from $\mathcal{U}$ indexes into $\mathcal{S}$ indexes
- β is the permutation from $\mathcal{V}$ indexes into $\mathcal{T}$ indexes
- γ is the permutation from $\mathcal{W}$ indexes into $\mathcal{S}$ indexes
- δ is the permutation from $\mathcal{Z}$ indexes into $\mathcal{T}$ indexes.

With this representation, the index i in $\mathcal{U}$ corresponds to $\alpha(i)$ in the time series $\mathcal{S}$. In other words, $u_i = s_{\alpha(i)}$. Using a similar reasoning $v_i = t_{\beta(i)}$. Leveraging the principle that permutations are bijective functions, we know that $s_i = u_{\alpha^{-1}(i)}$ as well as $t_i = v_{\beta^{-1}(i)}$. Since we assume the indexes of $\mathcal{S}$ and $\mathcal{T}$ are in direct correspondence, the index i in $\mathcal{T}$ is also in correspondence with $\alpha^{-1}(i)$ in $\mathcal{U}$. But it does not mean that neither $u_i = t_{\alpha(i)}$ nor $v_i = t_{\alpha(i)}$ because s_i may differ from t_i.

For our formalisation, we consider that given two time series of size n, two permutations of $(1, \ldots, n)$, and the absolute value function abs, we define the $\triangle$ and $\diamond$ operators as (we divide the values by $1/n$ to get the mean value of each series):

$$\begin{aligned}\triangle(\mathcal{S}, \mathcal{T}, \alpha, \beta) \equiv (1/n) * \sum_{i=0}^{n} abs((s_{\alpha(i)} - t_{\alpha(i)}) * (1 + abs(\alpha(i) - i))) \\ + (1/n) * \sum_{i=0}^{n} abs((s_{\beta(i)} - t_{\beta(i)}) * (1 + abs(\beta(i) - i)))\end{aligned} \tag{7}$$

$$\diamond(\mathcal{S}, \mathcal{T}, \alpha, \beta) \equiv (1/n) * \sum_{i=0}^{n} abs((s_{\alpha(i)} - t_{\beta(i)}) * (1 + abs(\alpha(i) - \beta(i)))) \tag{8}$$

Now, we can express the equations that calculate our measurement of disorganisation between time series through a set of factors.

Considering two series $\mathcal{S}$, $\mathcal{T}$, their sorted versions, $\mathcal{U}$, $\mathcal{V}$, $\mathcal{W}$ and $\mathcal{Z}$, and the permutations α, β, γ and δ as given above; we define two factors: the *sorted positions factor* (spf) and the *reverse positions factor* (rpf):

$$spf = \triangle(\mathcal{S}, \mathcal{T}, \alpha, \beta) \tag{9}$$

$$rpf = \triangle(\mathcal{S}, \mathcal{T}, \gamma, \delta) \tag{10}$$

In this context, spf evaluates the sum of all absolute differences between a value of $\mathcal{S}$ and its correspondent in $\mathcal{T}$, multiplied by the sum of the weights related to the positions of the $\mathcal{S}$ and $\mathcal{T}$ values compared to their growing ordering versions. On the other hand, rpf considers the decreasing order. These two factors allow evaluating the degree of disorganisation of one series related to another through α, β, γ, and δ expressions. Series that are more similar present smaller relative disorganisation and, consequently, smaller values to these factors.

The following two factors are designed to measure the relative degree of disorganisation between ordered and reverse-ordered series values and positions. It evaluates the sum of all absolute differences between the values of the ordered series or reversed ordered series multiplied by the differences between the original positions of these values. Thus, the second dimension provides two factors: sort factor (sf) and reverse factor (rf).

$$sf = \diamond(\mathcal{S}, \mathcal{T}, \alpha, \beta) \tag{11}$$

$$rf = \diamond(\mathcal{S}, \mathcal{T}, \gamma, \delta) \tag{12}$$

Thus, sort factor (sf) and reverse factor (rf) allow the evaluation of the degree of disorganisation of both series related to their ordered or reverse-ordered versions. Series that are more similar to their sorted or reverse-sorted versions present smaller values.

Finally, to quantify the dissimilarity between the series, we sum the values resulting from the possible combinations of the factors. Thus, the proposed measure, named sorting paired points (SPP), is given by:

$$SPP = w1 * spf + w2 * sf + w3 * rpf + w4 * rf \tag{13}$$

In this work, we set the weights $w1 = w2 = w3 = w4 = 1$ for all tests. By assigning equal weights to all factors, we aim to balance sensitivity (via the sum of different factors evaluating disorganisation) in both direct and reverse ordering. In future works, we will explore applying different weights to these factors to assess parameters sensitivity.

3.1 Complexity

For the complexity analysis, we suppose to have two time series of size n. Since we need to sort each time series, we have at least $\mathcal{O}(n * \log(n))$ complexity. Given two time series, the factors calculations are taken after the sorting phase. Therefore the factors calculations are independent of the sorting phase in the perspective of complexity analysis. Each factor is separately calculated using $\triangle$ and $\diamond$ operators. Both operators depend on the complexity of the evaluation of the permutations. We consider the permutation mappings, and their reverse mappings are made during the sorting phase and have $(O)(1)$ complexity for evaluation. Given a permutation α, it means $\alpha(i)$ is evaluated at constant time. Therefore, the operators $\triangle$ and $\diamond$ have the $\mathcal{O}(n)$ complexity because of the size of their sums. We conclude the resulting complexity is $\mathcal{O}(n * \log(n))$.

3.2 SPP and time series classification

It is a challenge to reach a balance in the application of invariance. Time series may vary in many ways, and selecting what is more important is challenging. Before comparing SPP to a more diverse set of dissimilarities and datasets, the following example compares the proposed dissimilarity, using hierarchical clustering, to three other dissimilarities: Euclidean distance, DTW and CID. The first dissimilarity is one of the most used nowadays, the second is one of the most competitive strategies, and the third one, which is also very competitive, strongly applies the concept of invariance, providing an alternative approach. Figure 1 shows a set of image silhouettes of sea animals and the time series, extracted from these images using radial scanning (Keogh et al., 2006). It also shows the identifier of each object to allow their comparison.

In this scenario, we evaluate the dissimilarities of the time series using the four strategies based on their dissimilarities matrices. Tables 1, 2, 3 and 4 present the results of the dissimilarities of Euclidean, CID, DTW and SPP strategies, respectively. We normalised the table values, dividing them by the minimum value of each strategy to make the interpretation easier. Notice that SPP values tend to decrease between similar time series and increase as we compare different series.

Figure 1 Silhouettes of sea animals and their correspondent time series, extracted using radial scanning (see online version for colours)

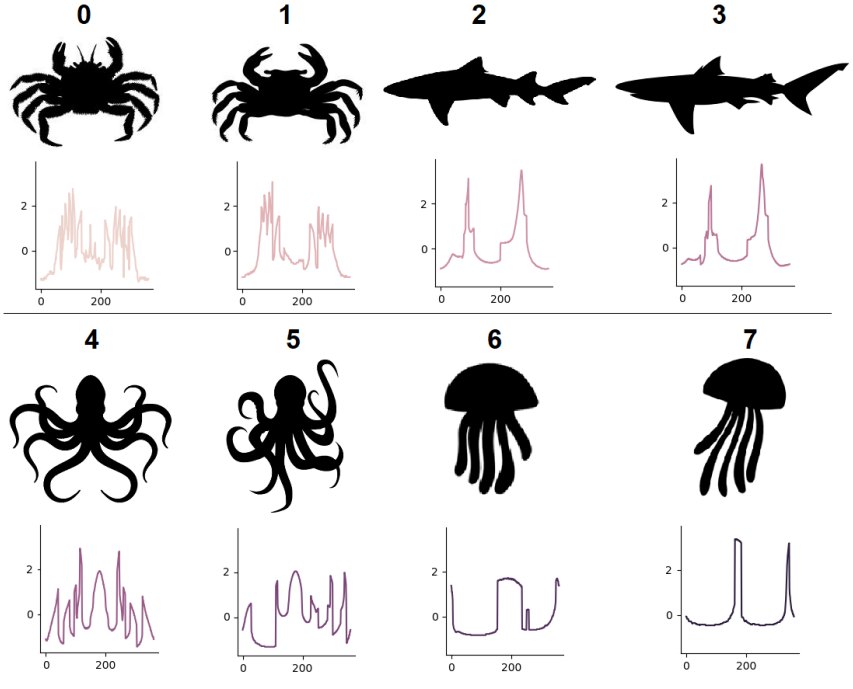


Table 1 Euclidean distance matrix

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
0		2.383	2.494	2.47	3.466	3.974	4.085	4.017
1			2.312	2.29	3.467	4.096	4.24	4.194
2				1.	3.714	4.129	4.039	4.187
3					3.561	3.987	4.101	4.143
4						2.582	3.294	2.953
5							2.64	2.328
6								2.699
7								

Table 2 CID matrix

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
0		3.011	5.621	6.02	4.079	5.848	8.639	6.066
1			3.812	4.085	3.442	4.411	6.561	4.634
2				1.	6.576	5.846	3.98	5.778
3					6.819	6.106	4.37	6.184
4						2.985	5.473	3.503
5							3.508	2.209
6								3.496
7								

Figure 2 Hierarchical clustering using Euclidean distance of time series from sea animals (see online version for colours)

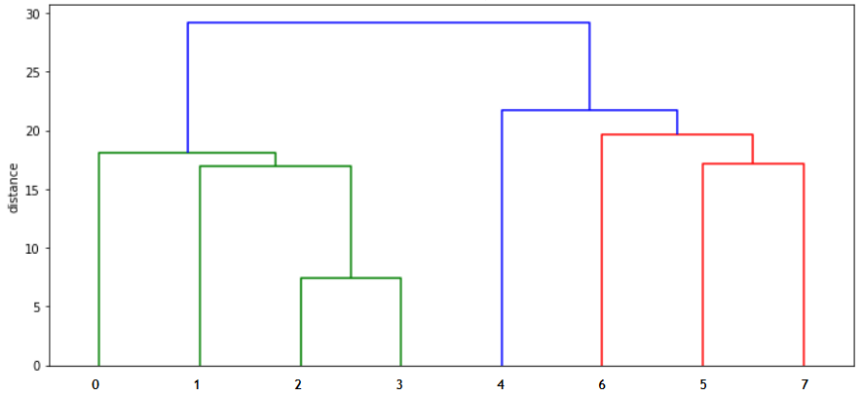


Figure 3 Hierarchical clustering using CID of time series from sea animals (see online version for colours)

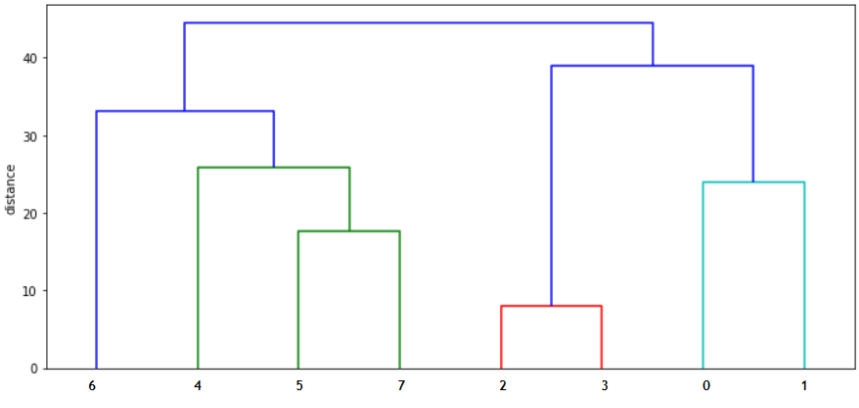


Figure 4 Hierarchical clustering using DTW of time series from sea animals (see online version for colours)

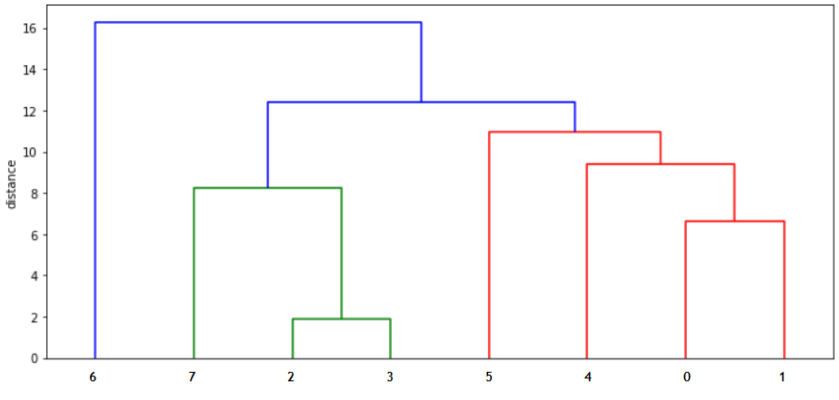


Figure 5 Hierarchical clustering using SPP of time series from sea animals (see online version for colours)

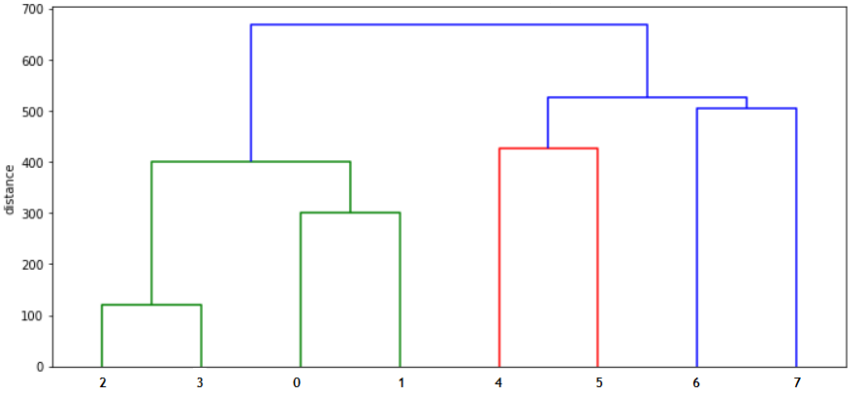


Table 3 DTW matrix

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
0		3.535	5.772	6.117	4.544	5.677	9.485	8.023
1			4.87	5.206	5.481	6.07	8.402	7.395
2				1.	7.167	6.658	9.092	4.622
3					6.922	5.781	8.798	4.138
4						5.753	9.504	7.454
5							7.552	7.878
6								7.991
7								

Table 4 SPP matrix

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
0		2.479	3.507	3.656	4.598	5.383	6.216	6.167
1			2.997	3.113	4.566	5.552	6.162	6.401
2				1.	4.696	5.954	5.633	5.581
3					4.752	5.736	5.477	5.354
4						3.515	4.565	4.891
5							3.846	4.096
6								4.107
7								

Considering the SPP dissimilarity matrix, the lowest value of the intersection of each row and column identifies the most similar time series from the objects, which are: 0 and 1; 2 and 3; 4 and 5; 6 and 7. Thus, initially, we have four groups of time series. It is also possible to say that the time series 0 and 1 are more like 2 and 3 than the others. Furthermore, time series 4 and 5 are more like 6 and 7 than the others. Finally, we can connect a hypothetical group composed of time series 0, 1, 2, and 3 to a hypothetical group composed of time series 4, 5, 6, and 7. According to the SPP dissimilarity matrix,

it also would be acceptable to say that time series 2, 3 and 4, 5 are the most similar time series inside these two groups.

Therefore, a matrix comparing all these time series among them, starting at the pair 0-1 and finishing at pair 6-7, will have an increasing dissimilarity among the pairs from the smaller pair values to the higher pair values.

We also use hierarchical clustering to group these time series using average linkage. Figures 2, 3, 4 and 5 show the dendrograms.

In the dendrograms, the results for Euclidean distance, DTW and CID do not follow common sense. Mainly when the time series have some delay between them (phase variance) and noise. However, SPP has correctly related the images, producing an assertive dendrogram.

4 Metrics

This research uses two metrics to analyse the correct grouping of series in clusters: maximal HITS and adjusted RAND index.

4.1 Maximal HITS

The metric used in this work to rate the elements correctly grouped in the clusters, named maximal HITS (MHITS), tries to maximise the number of hits considering the clusters formed by the algorithms when compared to the reference clusters. Given the set of time series $\{\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{F}\}$ and the correct corresponding clustering *Ref_Clusters* = $[[\mathcal{A}, \mathcal{B}, \mathcal{C}], [\mathcal{D}], [\mathcal{E}, \mathcal{F}]]$; we need to compare the clustering given by the different similarity functions. We give here an example of the list of clusters found by some clustering strategy that will be compared to the correct pattern: Found_Clusters = $[[\mathcal{A}], [\mathcal{B}, \mathcal{C}, \mathcal{E}], [\mathcal{D}, \mathcal{F}]]$.

Initially, there is no correspondence between the cluster of *Ref_Clusters* and *Found_Clusters*. To find the correspondence of the cluster, we use a strategy that tries to maximise the number of time series in the correspondent clusters of *Ref_Clusters* and *Found_Clusters*. Taking as base the cluster parameters, we build a correspondence matrix where each row corresponds to a cluster of *Ref_Clusters*, and each column corresponds to a cluster of *Found_Clusters*. The value of each position in the matrix corresponds to the number of series in the cluster corresponding to that column in that row. The correspondence matrix is given by [[1, 2, 0], [0, 0, 1], [0, 1, 1]]

Then, we select the position that contains the maximum value. In the case of a tie, the selected value is the one that corresponds to the lowest sum of the values of the row and column of the matrix that crosses with those tied values. Following, the row and column that cross the selected value are deleted, forming a new lower-order matrix. The process is repeated, selecting the next maximum value. After reaching the matrix with just one element, it is necessary to sum the value of this element with the maximum values selected in the process. This result corresponds to the number of elements correctly grouped by the algorithm. In the example, the first maximum value is (row 1, column 2) equal to 2. The new matrix is presented as follows [[0, 1], [0, 1]].

The next maximum value is 1, and the final element is 0. Therefore, the number of elements correctly grouped is 3. The result of this metric divides this value by the number of series grouped. In the case presented previously, $3/6 = 0.5$.

This solution is not optimum. The optimum solution involves an exponential complexity algorithm. In this case, it would be necessary to check the maximum values of all correspondence matrix columns and selects the value related to the lowest sum of the values of the row and column of the matrix that crosses with it. In the case of a tie, it should select the maximum value.

4.2 Adjusted Rand index

The adjusted Rand index (ARI) evaluates the level of agreement between groups (Kuncheva and Hadjitodorov, 2004; Santos and Embrechts, 2009) ranging from 1 (total agreement) to -1 (no agreement). The first set of clusters contains the reference groups that will be compared to the second set of clusters obtained by a clustering process. It provides a correction by chance to the Rand index (Rand, 1971), which is given by:

$$Rand = 2 * (p + q)/n * (n - 1) \tag{14}$$

where

- p equals the number of pairs that are in the same related clusters considering clusters of reference and clusters found by some clustering strategy
- q equals the number of pairs that are in not related clusters considering clusters of reference and clusters found by some clustering strategy
- n equals the number of series in the clusters.

5 Experimental comparison

The experimental platform used a laptop, Intel Core i7-2670QM, memory of 12 GB, and hard disk of 500 GB. We used Python (van Rossum, 1995) to execute all dissimilarity measures.

The UCR time series classification archive (Dau et al., 2018) provides several time series that allow testing and comparing clustering algorithms. These time series are z-normalised and grouped in 128 datasets, classified into 19 different domains (types). They cover various issues to test the robustness of the algorithms. Each dataset is divided into two groups: train and test.

We divided this group of 128 datasets into two subgroups of 64 datasets to facilitate the execution of the experiment. The first subgroup covers 11 different domains containing about 56,000 time series. The second subgroup includes 13 domains containing about 134,000 datasets.

Table 5 shows the first subgroup. For each dataset, it shows the name, the number of time series, the length of each time series, the number of classes (representing the number of clusters), and the domain (type) of the time series. The last column identifies the best results for SPP, where: 1 represents the best result for K-means applying MHITs, 2 represents the best result for K-means applying ARI, 3 represents the best results for hierarchical using MHITs, and 4 represents the best results for hierarchical using ARI.

We analyse ten different measures, applied to the first subgroup presented previously, to make a comprehensive analysis of the results: CID (Batista et al., 2014; Niu et al., 2020), cosine (Shirkhorshidi et al., 2015; Park et al., 2020), DTW (Berndt and Clifford, 1994; Jeong et al., 2011; Wang et al., 2013; Keogh and Ratanamahatana, 2005; Silva and Batista, 2016), Euclidean (Serrà and Arcos, 2014; Wang et al., 2013; Berthold and H"oppner, 2016), MI (Cover and Thomas, 2005; Zhang et al., 2018), MIC (Reshef et al., 2011), Manhattan (Strauss and von Maltitz, 2017), Kendall, Pearson, Spearman (Kumari et al., 2012; Miot, 2018), and SPP.

Table 5 Time series features

ID	*Type*	*Name*	*Train*	*Test*	*Class*	*Size*	*SPP-best*
1	Image	Adiac	390	391	37	176	
2	Image	ArrowHead	36	175	3	251	1 2 4
3	Spectro	Beef	30	30	5	470	1
4	Image	BeetleFly	20	20	2	512	1 2
5	Image	BirdChicken	20	20	2	512	3 4
6	Simulated	BME	30	150	3	128	
7	Sensor	Car	60	60	4	577	
8	Simulated	CBF	30	900	3	128	
9	Traffic	Chinatown	20	345	2	24	
10	Sensor	ChlorineConcentration	467	3,840	3	166	2
11	Sensor	CinCECGTorso	40	1,380	4	1,639	
12	Spectro	Coffee	28	28	2	286	3 4
13	Device	Computers	250	250	2	720	4
14	Motion	CricketX	390	390	12	300	3
15	Motion	CricketY	390	390	12	300	
16	Motion	CricketZ	390	390	12	300	1 2
17	Image	DiatomSizeReduction	16	306	4	345	3
18	Image	DistalPhalanxOutlineAgeGroup	400	139	3	80	2
19	Image	DistalPhalanxOutlineCorrect	600	276	2	80	3
20	Image	DistalPhalanxTW	400	139	6	80	
21	ECG	ECG200	100	100	2	96	
22	ECG	ECGFiveDays	23	861	2	136	3 4
23	Image	FaceAll	560	1,690	14	131	3 4
24	Image	FaceFour	24	88	4	350	
25	Image	FacesUCR	200	2,050	14	131	
26	Image	Fish	175	175	7	463	2
27	HRM	Fungi	18	186	18	201	3 4
28	Motion	GunPoint	50	150	2	150	3 4
29	Motion	GunPointAgeSpan	135	316	2	150	
30	Motion	GunPointMaleVersusFemale	135	316	2	150	1 2 3 4
31	Motion	GunPointOldVersusYoung	135	316	2	150	3 4
32	Motion	Haptics	155	308	5	1,092	
33	Motion	InlineSkate	100	550	7	1,882	3 4
34	EPG	InsectEPGRegularTrain	62	249	3	601	1 2 3 4
35	Sensor	ItalyPowerDemand	67	1,029	2	24	3

Table 5 Time series features (continued)

ID	*Type*	*Name*	*Train*	*Test*	*Class*	*Size*	*SPP-best*
36	Sensor	Lightning2	60	61	2	637	3 4
37	Sensor	Lightning7	70	73	7	319	1 2 3 4
38	Simulated	Mallat	55	2,345	8	1,024	
39	Image	MedicalImages	381	760	10	99	
40	Image	MiddlePhalanxOutlineAgeGroup	400	154	3	80	
41	Sensor	MoteStrain	20	1,252	2	84	4
42	Spectro	OliveOil	30	30	4	570	
43	Image	OSULeaf	200	242	6	427	
44	Hemodynamics	PigAirwayPressure	104	208	52	2,000	4
45	Sensor	Plane	105	105	7	144	
46	Power	PowerCons	180	180	2	144	
47	Image	ProximalPhalanxOutlineAgeGroup	400	205	3	80	
48	Simulated	SmoothSubspace	150	150	3	15	3
49	Sensor	SonyAIBORobotSurface1	20	601	2	70	
50	Sensor	SonyAIBORobotSurface2	27	953	2	65	3 4
51	Spectro	Strawberry	613	370	2	235	
52	Image	SwedishLeaf	500	625	15	128	
53	Image	Symbols	25	995	6	398	
54	Simulated	SyntheticControl	300	300	6	60	
55	Motion	ToeSegmentation1	40	228	2	277	
56	Sensor	Trace	100	100	4	275	
57	ECG	TwoLeadECG	23	1,139	2	82	3 4
58	Motion	UWaveGestureLibraryX	896	3,582	8	315	3 4
59	Motion	UWaveGestureLibraryY	896	3,582	8	315	
60	Motion	UWaveGestureLibraryZ	896	3,582	8	315	
61	Spectro	Wine	57	54	2	234	3 4
62	Motion	Worms	181	77	5	900	
63	Motion	WormsTwoClass	181	77	2	900	3 4
64	Image	Yoga	300	3,000	2	426	

Pearson, Spearman and Kendall used *scipy.stats* library (Virtanen et al., 2020), DTW used *dtaidistance* library (Meert and Craenendonck, 2018), MI used *sklearn.metrics.cluster* library (Pedregosa et al., 2011; Buitinck et al., 2013), MIC used *minepy* libray (Albanese et al., 2013). While this, Euclidean, Manhattan, and cosine used an implementation based on (Shirkhorshidi et al., 2015). Finally, CID used an implementation base on (Batista et al., 2014).

To compare the results, we applied two clustering algorithms using Python language: K-means, using a vector implementation (Snell, 2017), and hierarchical, using the package *scipy.cluster.hierarchy* (Virtanen et al., 2020).

The measures used to compare the results are MHITS and ARI. The mean values from train and test time series, using these measures, corresponding to the values presented in this work. For K-means algorithm, which is a non-deterministic algorithm, we executed each method 3 times, and we used the mean of these values in the experiment.

Figure 6 Comparison of algorithms using K-means and MHITS (see online version for colours)

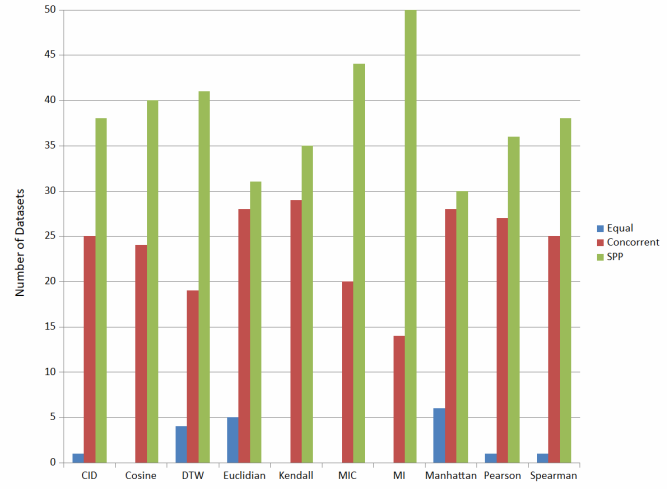


Figure 7 Comparison of algorithms using K-means and ARI (see online version for colours)

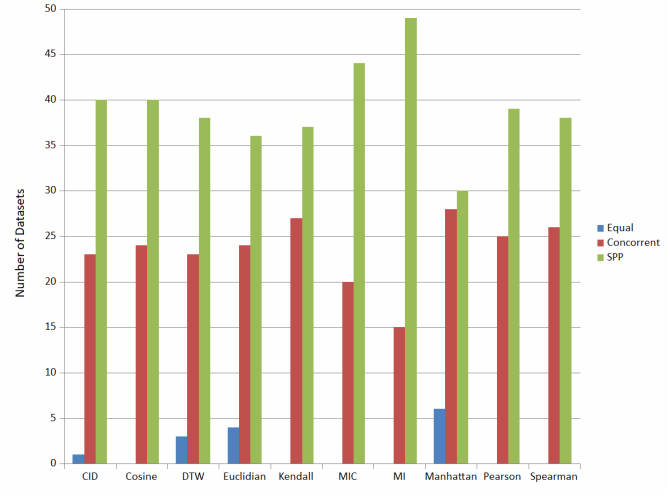


Figure 6 compares SPP to the other ten strategies using K-means and MHITs. For each algorithm, it presents the number of datasets where the concurrent algorithm gets better results (red bars), the number of datasets where SPP gets better results (green bars), and the number of ties (blue bars). These relative comparisons show that SPP provides a good approach when different sources are considered.

Figure 7 compares SPP to other strategies using K-means and ARI. It is possible to notice that both measures, MHITS, presented previously, and ARI, provide close results.

Figure 8 uses hierarchical clustering and MHITs to compare SPP against other strategies. Using these approaches, SPP also demonstrates to get better results when compared relatively to other algorithms.

Table 6 Time series features

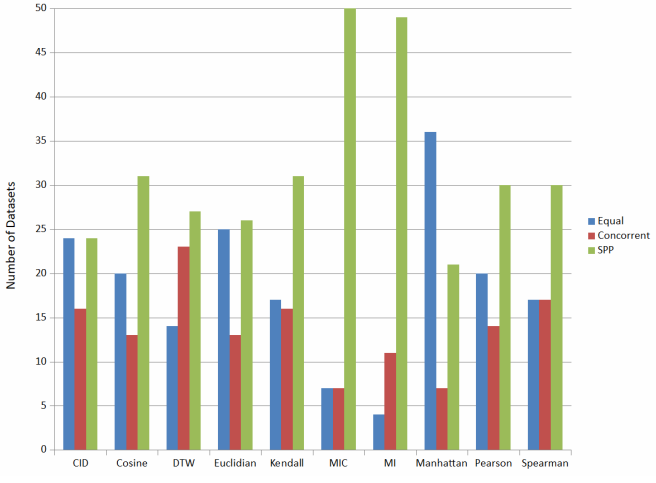
ID	*Type*	*Name*	*Train*	*Test*	*Class*	*Size*	*SPP-best*
65	Device	ACSF1	100	100	10	1,460	1 2
66	Sensor	AllGestureWiimoteX	300	700	10	Vary	
67	Sensor	AllGestureWiimoteY	300	700	10	Vary	
68	Sensor	AllGestureWiimoteZ	300	700	10	Vary	
69	Image	Crop	7,200	16,800	24	46	1 2
70	Sensor	DodgerLoopDay	78	80	7	288	1 2
71	Sensor	DodgerLoopGame	20	138	2	288	1 2
72	Sensor	DodgerLoopWeekend	20	138	2	288	
73	Sensor	Earthquakes	322	139	2	512	
74	ECG	ECG5000	500	4,500	5	140	1 2
75	Device	ElectricDevices	8,926	7,711	7	96	
76	EOG	EOGHorizontalSignal	362	362	12	1,250	
77	EOG	EOGVerticalSignal	362	362	12	1,250	
78	Spectro	EthanolLevel	504	500	4	1,751	1 2
79	Image	FiftyWords	450	455	50	270	1 2
80	Sensor	FordA	3,601	1,320	2	500	
81	Sensor	FordB	3,636	810	2	500	
82	Sensor	FreezerRegularTrain	150	2,850	2	301	
83	Sensor	FreezerSmallTrain	28	2,850	2	301	1 2
84	Trajectory	GestureMidAirD1	208	130	26	360	1 2
85	Trajectory	GestureMidAirD2	208	130	26	360	
86	Trajectory	GestureMidAirD3	208	130	26	360	
87	Sensor	GesturePebbleZ1	132	172	6	Vary	1 2
88	Sensor	GesturePebbleZ2	146	158	6	Vary	1 2
89	Spectro	Ham	109	105	2	431	
90	Image	HandOutlines	1,000	370	2	2,709	
91	Image	Herring	64	64	2	512	
92	Device	HouseTwenty	34	101	2	3,000	1 2
93	EPG	InsectEPGSmallTrain	17	249	3	601	
94	Sensor	InsectWingbeatSound	220	1,980	11	256	1 2
95	Device	LargeKitchenAppliances	375	375	3	720	1 2
96	Spectro	Meat	60	60	3	448	
97	Traffic	MelbournePedestrian	1,200	2,450	10	24	1 2
98	Image	MiddlePhalanxOutlineCorrect	600	291	2	80	
99	Image	MiddlePhalanxTW	399	154	6	80	1 2
100	Image	MixedShapesRegularTrain	500	2,425	5	1,024	1 2
101	Image	MixedShapesSmallTrain	100	2,425	5	1,024	1 2
102	ECG	NonInvasiveFetalECGThorax1	1,800	1,965	42	750	1 2
103	ECG	NonInvasiveFetalECGThorax2	1,800	1,965	42	750	1 2
104	Image	PhalangesOutlinesCorrect	1,800	858	2	80	1 2
105	Sensor	Phoneme	214	1,896	39	1,024	
106	Sensor	PickupGestureWiimoteZ	50	50	10	Vary	1 2
107	Hemodynamics	PigArtPressure	104	208	52	2,000	1 2
108	Hemodynamics	PigCVP	104	208	52	2,000	

Table 6 Time series features (continued)

ID	*Type*	*Name*	*Train*	*Test*	*Class*	*Size*	*SPP-best*
109	Device	PLAID	537	537	11	Vary	1
110	Image	ProximalPhalanxOutlineCorrect	600	291	2	80	
111	Image	ProximalPhalanxTW	400	205	6	80	1 2
112	Device	RefrigerationDevices	375	375	3	720	1 2
113	Spectrum	Rock	20	50	4	2,844	
114	Device	ScreenType	375	375	3	720	1 2
115	Spectrum	SemgHandGenderCh2	300	600	2	1,500	1
116	Spectrum	SemgHandMovementCh2	450	450	6	1,500	
117	Spectrum	SemgHandSubjectCh2	450	450	5	1,500	
118	Sensor	ShakeGestureWiimoteZ	50	50	10	Vary	1 2
119	Simulated	ShapeletSim	20	180	2	500	1
120	Image	ShapesAll	600	600	60	512	
121	Device	SmallKitchenApplianc.	375	375	3	720	1
122	Sensor	StarLightCurves	1,000	8,236	3	1,024	
123	Motion	ToeSegmentation2	36	130	2	343	
124	Simulated	TwoPatterns	1,000	4,000	4	128	
125	Simulated	UMD	36	144	3	150	
126	Motion	UWaveGestureLibraryAll	896	3,582	8	945	1 2
127	Sensor	Wafer	1,000	6,164	2	152	
128	Image	WordSynonyms	267	638	25	270	1 2

Similarly, Figure 9 compares SPP to other strategies using hierarchical clustering and ARI. Again, the results are similar to the results using MHITS.

Figure 8 Comparison of algorithms using hierarchical clustering and MHITS (see online version for colours)



Analysing the obtained results, the comparison agrees in a general way, using MHITS and ARI. Considering both metrics, SPP gets better outcomes when compared to each

other strategies. Adopting a different perspective, we consider the sum of the absolute best results considering all values obtained, both clustering algorithms (K-means and hierarchical clustering) and both metrics (MHITs and ARI). Figure 10 shows the results of this approach. It is possible to infer an absolute ranking from the values, where SPP appears in the first position, followed by DTW, in the second position, and CID, in the third position. This analysis shows that SPP gets a higher number of best values considering an absolute comparison of all results.

Figure 9 Comparison of algorithms using hierarchical clustering and ARI (see online version for colours)

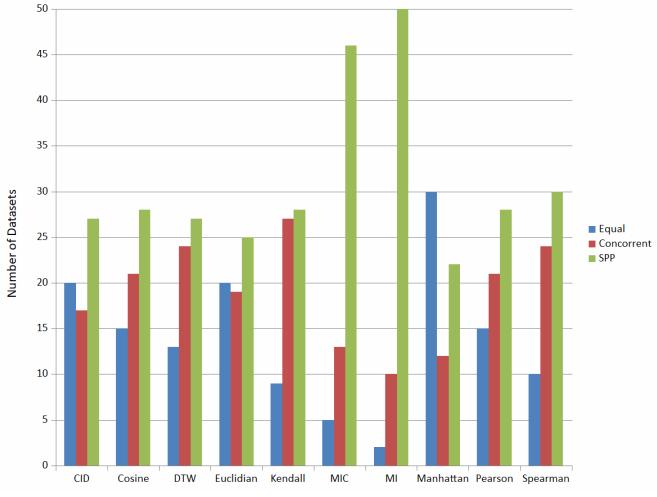
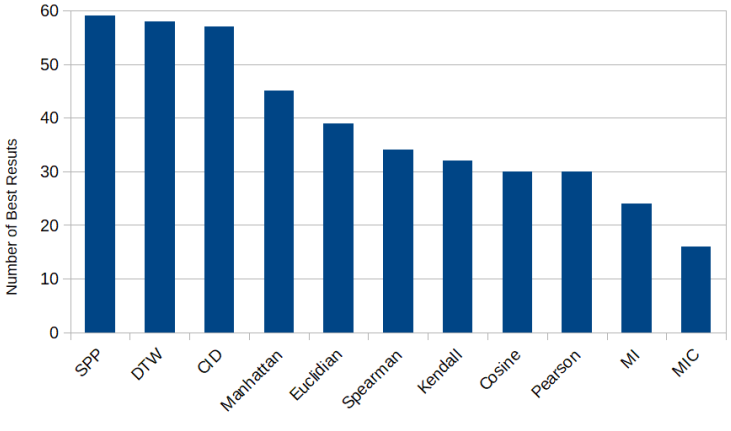


Figure 10 Comparison of the best results considering all clustering algorithms and metrics (see online version for colours)



At this point, it is important to estimate the error of the overall results until now, considering the different datasets. For this, we use the margin of error (MOE) formula (Weiers, 2010):

$$MOE = z * \sqrt{((p * (1-p))/n)} \tag{15}$$

where

- z represents the z-score (the desired confidence level);
- p represents the sample proportion;
- n represents the sample size (number of datasets).

Assuming that the p is 0.5 (conservative strategy), the confidence level of the total population is 90% (z-score is 1.645 for this case), and the sample size is 64 for a large population (more than 20,000). Consequently, the estimated MOE percentage is 12.25%.

Considering that the number of best results of SPP is 59, then MOE is 7.3. Therefore the results of SPP range from 66.3 to 51.7. DTW and CID are technically tied. Manhattan, which corresponds to the fourth best-placed algorithm, has a MOE of 5.52. Its results range from 50.52 to 39.48 (it is not tied with SPP). Then, only three algorithms have similar results. Because of that, we conducted a second phase of experiments using another subgroup of datasets, shown in Table 6. The goal is to evaluate the consistency of these three algorithms in different scenarios. In this subgroup of datasets, some time series have missing values. Thus, we use a known common technique that involves replacing the missing values with the mean of the time series. The last column identifies the best results for SPP, where: 1 represents the best result for K-means applying MHITs, and 2 represents the best result for K-means applying ARI. We apply the top three (SPP, CID e DTW) from the ten initial dissimilarities measures (considering the first subgroup previously presented) using only K-means clustering algorithm (Snell, 2017), executed 3 times for each method. We continue applying MHITS and ARI to estimate the results for this stage of the experiment.

Figure 11 compares SPP to the CID and DTW, using K-means and MHITs. For each algorithm, it presents the number of datasets where the CID and DTW algorithm gets better results (red bars), and the number of datasets where SPP gets better results (green bars). There are no ties in this case.

Using the same approach, Figure 12 compares SPP to the CID and DTW, using K-means and ARI (which results are very similar to MHITs). These comparisons show that SPP is consistently superior to the other two strategies, considering the subgroup of time series used.

Considering the sum of the absolute best results for this second subgroup of time series, and both metrics (MHITs and ARI), we can generate Figure 13. This analysis shows that SPP also gets a higher number of best values considering an absolute comparison of these results.

Regarding to the error estimation of this second dataset subgroup, considering the estimated MOE percentage is 12.25% (equal to the first subgroup experiment), we have that the number of best results of SPP is 60, then the MOE is 7.35. Therefore the outcomes of SPP range from 67.35 to 52.65. The second-best algorithm got 38 best results. Then, it ranges from 42.7 to 33.4. Consequently, SPP got better results than the other two algorithms, even considering the margin of error.

Generally speaking, the experiments demonstrated that SPP provides a strategy capable of dealing with time series from different sources and domains. Moreover, considering the number of tested datasets, it gets better results than the other ten strategies analysed.

Figure 11 Comparison of SPP, CID and DTW using K-means and MHITS (see online version for colours)

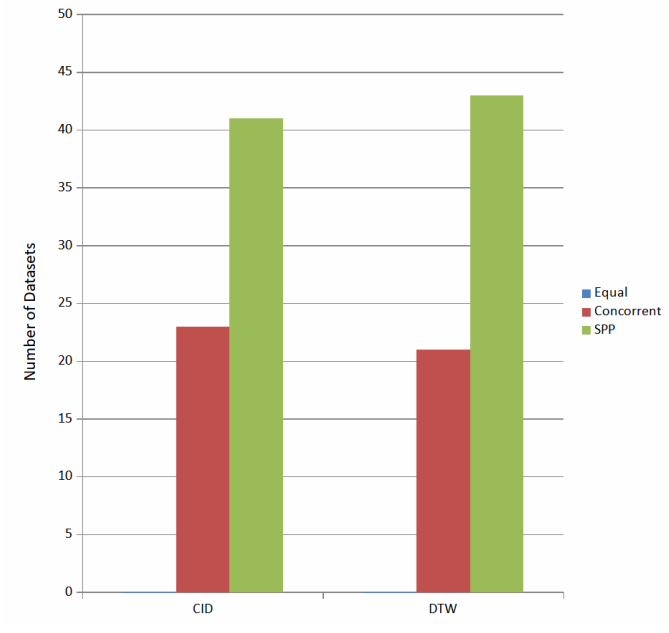


Figure 12 Comparison SPP, CID and DTW using K-means and ARI (see online version for colours)

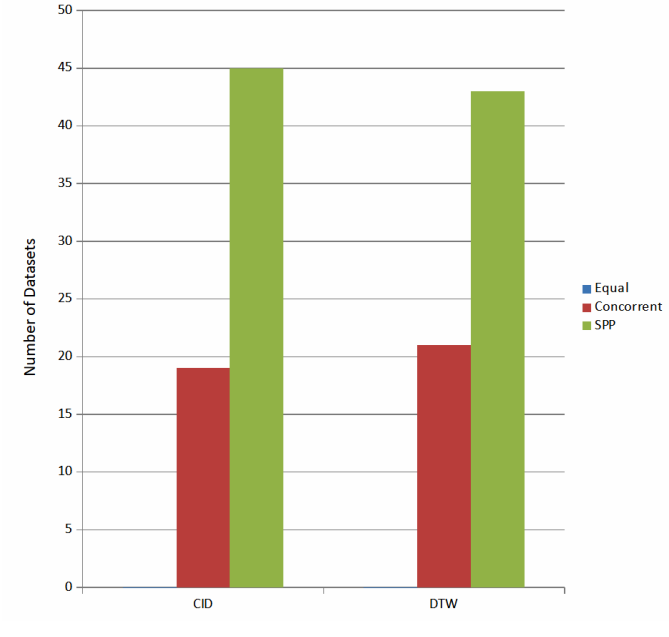


Figure 13 Comparison of the best results considering all clustering algorithms and metrics (see online version for colours)

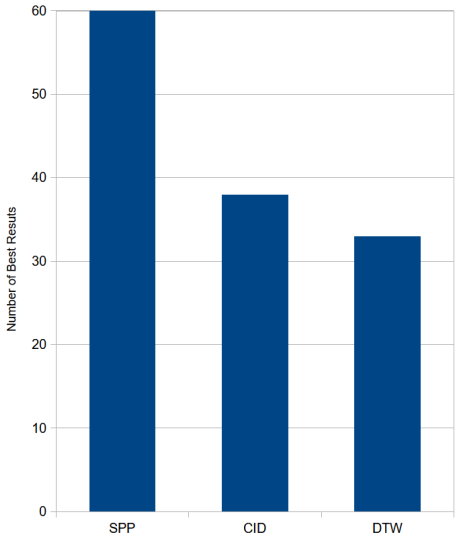


Table 7 SPP best results per type

Types (>2 datasets)	*Quantity of datasets*	*Better results*
Device	9	88,89%
ECG	6	83,33%
Hemodynamics	3	66,67%
Image	32	50,00%
Motion	17	52,94%
Sensor	30	46,67%
Simulated	8	25,00%
Spectro or spectrum	12	41,67%
Trajectory	3	33,33%
Average		*54.28%*

A complementary analysis of the experiments considers the number of elements in the datasets and the average size of the time series. The first subgroup contains a smaller number of series (about 56,000 against 134,000) as a smaller average series size (approximately 375 against 740). In this first subgroup, despite the advantage of the proposed strategy, two other methods obtain close results (CID and DTW). Meanwhile, SPP obtains far superior results to these two strategies when we consider the second subgroup. This fact may imply that the SPP strategy becomes even more differentiated when are considered larger datasets with long series.

We also tried to test OPWD algorithm in our experiments (Su and Hua, 2017). The algorithm was available originally in MATLAB code, then we converted it to Python in order to compare it to other algorithms analysed in this paper. However, this proposal did not reach results for some datasets (it only returned results for 91 datasets after 5 days

running in our hardware). Comparing OPWD and SSP considering only the returned results and ARI and MHITS metrics, we have that SPP wins by 83 against 82, ocurring 17 tied results. Another interesting result was that the average time of OPWD was more than ten times the average time of SPP. Consequently, in spite of the fact that OPWD got similar results to SPP for ARI and MHITS metrics, it got poor results in terms of average time.

An additional analysis involves verifying, by dataset type, the percentage of datasets where the SPP measure achieved the best results in any test (ARI or MHITS in hierarchical or K-means), considering only dataset types with more than two datasets. Table 7 establishes the basis of this analysis.

Table 7 shows that the SPP measure achieved the best results on average in 54% of cases compared to all measures combined when considering different types of datasets. The best result was observed for datasets of device type (88%), while the worst results were for simulated datasets (25%).

6 Conclusions

This work presents a new measure that uses the disorganisation of series, considering the growing and decreasing sorted series. This strategy provides an enhanced scattering of dissimilarity when compared to other strategies. We test our strategy using two different clustering strategies: K-means and hierarchical clustering. The experimental results, based on 128 time series datasets from different sources, show that the proposed approach can be used to group time series from different areas, having, on average, better results. The main contributions of our work are:

1 proposing a new dissimilarity measure based on sorting of series values that allows comparing time series consistently

2 analysing a set of ten relevant measures with very recent algorithms implementations, besides the proposed strategy, in the context of time series clustering

3 evaluating the results using a comprehensive set of time series from different domains (that present a different number of classes, sizes, and formats)

4 comparing the results obtained by leading approaches, employing relative and absolute values, against the proposed dissimilarity measure, using two different clustering algorithms and two distinct metrics that allow rating the formed clusters

5 demonstrating the superiority of the proposed strategy against all algorithms tested, considering the results obtained by our strategy compared to each concurrent algorithm and the set of time series used in the experiments.

In future works, we intend to apply other strategies using different weights for the facts used to calculate the measure. This way, our strategy is going to improve the relevance of these factors according to features of the time series, such as sizes, variations, invariances, and other parameters of the time series.

References

Abanda, A., Mori, U. and Lozano, J.A. (2019) 'A review on distance based time series classification', *Data Min. Knowl. Disc.*, Vol. 33, pp.378–412.

Aghabozorgi, S., Shirkhorshidi, A.S. and Wah, T.Y. (2015) 'Time-series clustering – a decade review', *Information Systems*, Vol. 53, pp.16–38.

Ah-Pine, J. (2018) 'An efficient and effective generic agglomerative hierarchical clustering approach', *Journal of Machine Learning Research*, Vol. 19, No. 42, pp.1–43.

Albanese, D., Filosi, M., Visintainer, R., Riccadonna, S., Jurman, G. and Furlanello, C. (2013) 'Minerva and Minepy: a C engine for the MINE suite and its R, Python and MATLAB wrappers', *Bioinformatics*, Vol. 29, No. 3, pp.407–408.

Batista, G.E.A.P.A., Keogh, E.J., Tataw, O. and de Souza, V.M.A. (2014) 'CID: an efficient complexity-invariant distance for time series', *Data Mining and Knowledge Discovery*, Vol. 28, No. 3, pp.634–669.

Berndt, D.J. and Clifford, J. (1994) 'Using dynamic time warping to find patterns in time series', *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pp.359–370.

Berthold, M.R. and Höppner, F. (2016) *On Clustering Time Series Using Euclidean Distance and Pearson Correlation*, arXiv:1601.02213.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B. and Varoquaux, G. (2013) 'API design for machine learning software: experiences from the scikit-learn project', *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp.108–122.

Cover, T.M. and Thomas, J.A. (2005) *Elements of Information Theory*, John Wiley & Sons, Inc., Hoboken, NJ, USA, ISBN: 9780471748823.

Dau, H.A., Keogh, E., Kamgar, K., Yeh, C-C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, C., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G. and HexagonML (2018) *The UCR Time Series Classification Archive*, HexagonML [online] https://www.cs.ucr.edu/~eamonn/time_series_data_2018/ (accessed 5 January 2024).

Hartigan, J.A. and Wong, M.A. (1979) 'Algorithm as 136: a k-means clustering algorithm', *The Journal of the Royal Statistical Society, Series C (Applied Statistics)*, Vol. 28, No. 1, pp.100–108.

Ivanović, M. and Kurbalija, V. (2016) 'Time series analysis and possible applications', *39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp.473–479.

Jeong, Y-S., Jeong, M.K. and Omitaomu, O.A. (2011) 'Weighted dynamic time warping for time series classification', *Pattern Recognition*, Vol. 44, No. 9, pp.2231–2240.

Keogh, E. and Ratanamahatana, C.A. (2005) 'Exact indexing of dynamic time warping', *Knowledge and Information Systems*, Vol. 7, No. 3, pp.358–386.

Keogh, E., Wei, L., Xi, X., Lee, S. and Vlachos, M. (2006) 'Lb_keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures', *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB)*, pp.882–893.

Kogan, J., Teboulle, M. and Nicholas, C. (2005) 'Data driven similarity measures for k-means like clustering algorithms', *Information Retrieval*, Vol. 8, No. 2, pp.331–349.

Kumari, J.N.S., Chen, H.S. et al. (2012) 'Evaluation of gene association methods for coexpression network construction and biological knowledge discovery', *PLOS One*, Vol. 7, p.11.

Kuncheva, L. and Hadjitodorov, S. (2004) 'Using diversity in cluster ensembles', *IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, Vol. 2, pp.1214–1219.

Lahreche, A. and Boucheham, B. (2021) 'A fast and accurate similarity measure for long time series classification based on local extrema and dynamic time warping', *Data Expert Systems with Applications*, Vol. 168, No. 114374, pp.1–12.

Lin, W., Wu, X., Wang, Z., Wan, X. and Li, H. (2022) 'Topic network analysis based on co-occurrence time series clustering', *Mathematics*, Vol. 10, No. 16, pp.1–17.

Mastrangelo, C.M., Simpson, J.R. and Montgomery, D.C. (2013) 'Time series analysis', *Encyclopedia of Operations Research and Management Science*, pp.1546–1552, Springer US, Boston, MA, ISBN: 978-1-4419-1153-7.

Meert, W. and Craenendonck, V. (2018) *Time Series Distances: Dynamic Time Warping (DTW)*, Zenodo [online] https://doi.org/10.5281/zenodo.1314205 (accessed 10 July 2020).

Miot, H.A. (2018) 'Correlation analysis in clinical and experimental studies', *J. Vasc. Bras.*, Vol. 17, No. 4, pp.275–279.

Murtagh, F. and Contreras, P. (2011) 'Algorithms for hierarchical clustering: an overview', *Data Mining and Knowledge Discovery*, Vol. 2, No. 1, pp.86–97.

Murtagh, F. and Contreras, P. (2017) 'Algorithms for hierarchical clustering: an overview II', *Data Mining and Knowledge Discovery*, Vol. 7, No. 6, pp.1–27.

Niu, H., Xu, K. and Wang, W. (2020) 'A hybrid stock price index forecasting model based on variational mode decomposition and LSTM network', *Applied Intelligence*, Vol. 50, pp.4296–4309.

Park, K., Hong, J.S. and Kim, W. (2020) 'A methodology combining cosine similarity with classifier for text classification', *Applied Artificial Intelligence*, Vol. 34, No. 5, pp.396–411.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011) 'Scikit-learn: machine learning in Python', *Journal of Machine Learning Research*, Vol. 12, pp.2825–2830.

Pérez-Ortega, J., Almanza-Ortega, N.N. and Romero, D. (2018) 'Balancing effort and benefit of k-means clustering algorithms in big data realms', *PLOS One*, Vol. 13, No. 9, pp.1–19.

Pham, M.C., Cao, Y., Klamma, R. and Jarke, M. (2011) 'A clustering approach for collaborative filtering recommendation using social network analysis', *JUCS – Journal of Universal Computer Science*, Vol. 17, No. 4, pp.583–604.

Rand, W.M. (1971) 'Objective criteria for the evaluation of clustering methods', *Journal of the American Statistical Association*, Vol. 66, No. 336, pp.846–850.

Reshef, D.N., Reshef, Y.A., Finucane, H.K., Grossman, S.R., McVean, G., Turnbaugh, P.J. and Sabeti, P.C. (2011) 'Detecting novel associations in large data sets', *Science (New York, N.Y.)*, Vol. 334, No. 6062, pp.1518–1524.

Santos, J.M. and Embrechts, M. (2009) 'On the use of the adjusted Rand index as a metric for evaluating supervised classification', *Artificial Neural Networks – ICANN 2009*, Vol. 5769, pp.175–184.

Serrà, J. and Arcos, J.L. (2014) 'An empirical evaluation of similarity measures for time series classification', *Knowledge-Based Systems*, Vol. 67, pp.305–314.

Shirkhorshidi, A.S., Aghabozorgi, S. and Wah, T.Y. (2015) 'A comparison study on similarity and dissimilarity measures in clustering continuous data', *PLOS One*, Vol. 10, p.12.

Silva, D.F. and Batista, G.E.A.P.A. (2016) 'Speeding up all-pairwise dynamic time warping matrix calculation', *Proc. SIAM Int. Conf. Data Min.*, Society for Industrial and Applied Mathematics, pp.837–845.

Snell, J. (2017) *Optimizing k-means Clustering for Time Series Data*, New Relic News and Products [online] https://dzone.com/articles/optimizing-k-means-clustering-for-time-series-data (accessed 2 January 2024).

Strauss, T. and von Maltitz, M.J. (2017) 'Generalising ward's method for use with manhattan distances', *PLOS One*, Vol. 12, No. 1, pp.1–21.

Su, B. and Hua, G. (2017) 'Order-preserving Wasserstein distance for sequence matching', *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2906–2914.

van Rossum, G. (1995) *Python Tutorial*, Technical report, CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam.

Virtanen, P., Gommers, R., Oliphant, T.E. and Contributors (2020) 'SciPy is an open-source scientific computing library for the Python programming language', *Nature Methods*, Vol. 17, No. 3, pp.261–272.

Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P. and Keogh, E. (2013) 'Experimental comparison of representation methods and distance measures for time series data', *Data Mining and Knowledge Discovery*, Vol. 26, pp.275–309.

Liao, T.W. (2005) 'Clustering of time series data – a survey', *Pattern Recognition*, Vol. 38, No. 11, pp.1857–1874.

Weiers, R.M. (2010) *Introduction to Business Statistics*, 7th ed., Cengage Learning, Mason, OH, ISBN: 053845217X.

Xu, D. and Tian, Y. (2015) 'A comprehensive survey of clustering algorithms', *Annals of Data Science*, Vol. 2, pp.165–193.

Yang, B., Kommuri, D.N.L., Sarva, S., Ercal, G. and Onal, S. (2022) 'Clustering time series data with applications to gait analysis', *Proceedings of IIE Annual Conference*, Norcross, GA, pp.1–6.

Yuan, M., Zobel, J. and Lin, P. (2022) 'Data driven similarity measures for k-means like clustering algorithms', *Information Retrieval*, Vol. 25, No. 3, pp.239–268.

Zhang, P., Gao, W. and Liu, G. (2018) 'Feature selection considering weighted relevancy', *Applied Intelligence*, Vol. 48, pp.4615–4625.