Under consideration for publication in Knowledge and Information
Systems

Making Clustering in Delay-Vector Space Meaningful

Jason R. Chen

Department of Information Engineering
Research School of Information Science and Engineering
College of Engineering and Computer Science
The Australian National University
Canberra, ACT, 0200, Australia
Jason.Chen@anu.edu.au

Abstract. Sequential time series clustering is a technique used to extract important features from time series data. The method can be shown to be the process of clustering in the Delay-Vector space formalism used in the Dynamical systems literature. Recently, the startling claim was made that sequential time series clustering is meaningless. This has important consequences for a significant amount of work in the literature, since such a claim invalidates these work's contribution. In this paper, we show that sequential time series clustering is not meaningless, and that the problem highlighted in these works stem from their use of the Euclidean distance metric as the distance measure in the delay vector space. As a solution, we consider quite a general class of time series, and propose a regime based on two types of similarity that can exist between delay vectors, which give rise naturally to an alternative distance measure to Euclidean distance in the delay vector space. We show that, using this alternative distance measure, sequential time series clustering can indeed be meaningful.

Keywords: delay space, clustering, time series, sequential time series clustering, STS clustering

1. Introduction

Data miners are often interested in extracting features from a time series of data (J.F.Roddick and M.Spiliopoulou, 2002). For example, consider a time series

$$X = x_t | t = 1, \ldots, n \tag{1}$$

Received xxx
Revised xxx
Accepted xxx

where t is the time index and n is the number of observations in the series. Such a time series could represent the closing price of a particular stock in the stock market, or the value returned by a sensor on a mobile robot, etc. Clustering is a technique that is very often proposed as the means for extracting features from time series data, for example in Feng and Huang (2005), T.Oates (1999), Babcock et al. (2003) and many others (see E.Keogh et al. (2003)). When the value of n is relatively small, and the interest is in comparing a set of complete time series produced by some event or process, clustering proceeds in the same way as for the conventional clustering of discrete objects; group together into the same cluster all similar time series. When n is large, it is impractical to conduct "whole series" clustering, and indeed, often we wish to find repeating features in a single time series, or in a number of time series produced by the same process. Here subsequence clustering using the sliding windows technique is a commonly proposed approach. If X is our time series, and $w < n$ is the window length, a single subsequence z_p is extracted as

$$z_p = x_{p-w+1}, x_{p-w+2}, \ldots, x_{p-1}, x_p \tag{2}$$

and a sequence Z of subsequences can be formed using the sliding windows technique by simply forming the set $Z = z_p | p = w, \ldots, n$. Subsequence time series clustering then proceeds by forming k clusters, each containing "similar" z_p, using whichever of the many clustering algorithms that are available (Berkhin, 2002). Subsequence time series clustering (from here on referred to as STS-clustering) is a widely used technique in the data mining community, often as a subroutine of some other technique or method. For example, E.Keogh et al. (2003) notes the use of this technique in the data mining areas of rule discovery [G.Das et al. (1998), S.K.Harms, J.Deogun and T.Tadesse (2002), S.K.Harms, S.Reichenbach, S.E.Goddard, T.Tadesse and W.J.Waltman (2002), M.L.Hetland and P.Saetrom (2002), X.Jin et al. (2002), T.Mori and K.Uehara (2001), R.Osaki et al. (n.d.)], indexing [C.Li et al. (1998), N.Radhakrishnan et al. (2000)], classification [P.Cotofrei (2002), P.Cotofrei and K.Stoffel (2002)], prediction [C.Schittenkopf et al. (2000), P.Tino et al. (2000)], and anomaly detection [T.Yairi et al. (2003)].

Amazingly, the validity of sequential time series clustering as a data mining technique has recently been called into question (E.Keogh et al., 2003). This has important consequences for work we have just surveyed, since such a claim may show it to be invalid. The conclusion in E.Keogh et al. (2003) is based on the finding that STS-clustering produces sine-type waves as cluster representatives. To help clarify our discussion of their result, we conduct a simple experiment.

One of the experiments in (E.Keogh et al., 2003) was the clustering of the Cylinder-Bell-Funnel data set, which is made up three features of the form shown in Figure 1. For reasons that will become clear later, we construct a particular version of this time series. In particular, the onset and duration within the 128 data point window of each feature in the original time series of Keogh et. al. were subject to some variability. Here we construct a version of this time series where the features have fixed onset and duration. We then construct the final time series in the same way by concatenating each of the features into a single "base" time series, and then concatenating a number (say 20) of these base time series into the final time series. Given there are three features in Figure 1, each with length 128, if we then conduct STS-clustering on this time series with window length $w = 128$, and the number of clusters $k = 3$, we might hope to recover the Cylinder, Bell, and Funnel features as cluster representatives. In fact the result

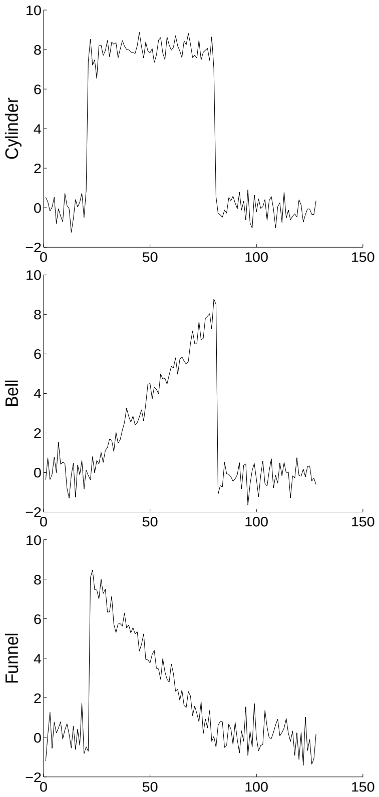


Fig. 1. The Cylinder, Bell and Funnel features

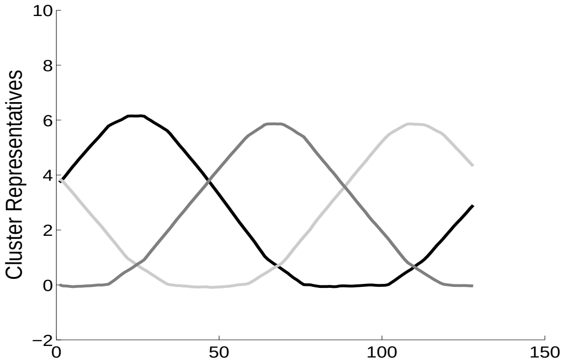


Fig. 2. Non-intuitive result of subsequence clustering using the sliding windows technique on a data series of concatenated features from Figure 1

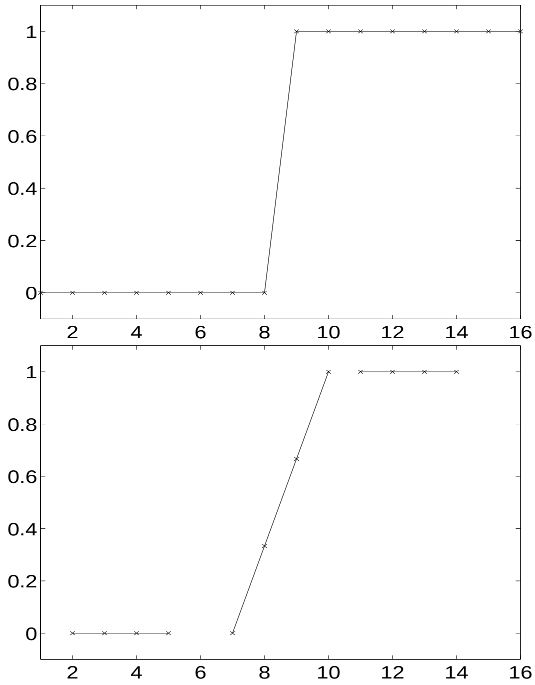


Fig. 3. A simple time series containing a single step (top), and the cluster representatives of a three-cluster STS-clustering of this data (bottom)

(using the k-means clustering method) is the three sine-type wave features shown in Figure 2. That is, for the version of the Cylinder-Bell-Funnel time series with fixed feature onset and duration, the same non-intuitive sine-type wave result is obtained. This is not suprising, since E.Keogh et al. (2003) report that sine-type waves were produced by STS-clustering no matter what clustering algorithm, number of clusters, or data set was used. Their conclusion that STS-clustering is meaningless followed, since, if any data set produces the same type of cluster representatives, then the output produced by the method is independent of its input, and hence is meaningless.

In this paper we propose a solution to the above dilemma. We show that two fundamental errors in the work in STS-clustering to date have been made. The first concerns how clusters are formed by the clustering method chosen, while the second concerns how the cluster centre (i.e. the representative of the cluster) is chosen. We will show that both errors are the result of how distance in the subsequence vector space is measured, and that once distance is measured in an appropriate way, STS-clustering can indeed be meaningful.

2. Selecting a Cluster Representative

We commence our investigation by exposing the reason why the cluster representatives in Figure 2 are so smooth. Given the original time series is full of sharp edges, it seems strange that the final cluster representatives are smooth sine-type

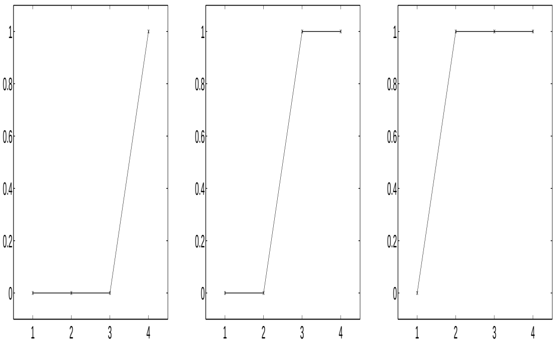


Fig. 4. The three members of the step cluster shown in Figure 3 (bottom)

waves. Assume we have a very simple data series consisting of 17 points which form a single step (as shown in Figure 3 (top)). If we apply STS-clustering to this data series, using the k-means clustering method, with the number of clusters $k = 3$, and with a window length $w = 4$, we obtain the cluster representatives shown in Figure 3 (bottom), where these representatives have been placed along the horizontal axis to coincide with the features they represent in the data series. At first glance, the clustering looks sensible, since the three representatives would seem to represent the three main features in the original data: when the signal is low, when it is high, and the step. However, a curious property of the step cluster representative is that it has been smoothed. Where the step in the original data was completed in one time unit of the horizontal axis, the step in the step-feature cluster representative now takes three time units. Figure 4 shows the three subsequence members of the step cluster. Note that, if cluster $C_j, j = 1 \ldots k$ contains r_j members $z_i, i = 1 \ldots r_j$, the k-means clustering algorithm determines a cluster representative as the point $\bar{z_j}$ which minimises

$$J = \sum_{i=1}^{r_j} d(z_i, \bar{z_j}) \tag{3}$$

where $d(.,.)$ (from now on) denotes Euclidean distance, and where, from an implementation point of view, $\bar{z_j}$ can be calculated simply as the mean of all members in C_j (i.e. $1/r_j \sum_{i=1}^{r_j} z_i$). Then, the mean of the three cluster members in Figure 4 is exactly the step-feature cluster representative shown in Figure 3. It seems then that Equation 3 does not provide a valid means for determining a cluster representative, since it gives rise to a curious “smoothing effect”, and this might be the reason why we get smoothed sine-type waves as cluster representatives in Figure 2. Toward exploring this possibility, we note that if we take the *medoid* member of the cluster (i.e. as per Equation 3, but where $\bar{z_j} \in C_j$) we are guaranteed to retain the sharpness of any feature [1].

We turn now again to the non-intuitive results of Figure 2. Maybe the clustering of the data series in this figure is correct, but that in determining each cluster representative according to Equation (3), the smoothing effect has degenerated the cluster representatives into smoothed sine-type waves. Conceivably,

[1] although we make no claim that it will form a balanced representation of the cluster members

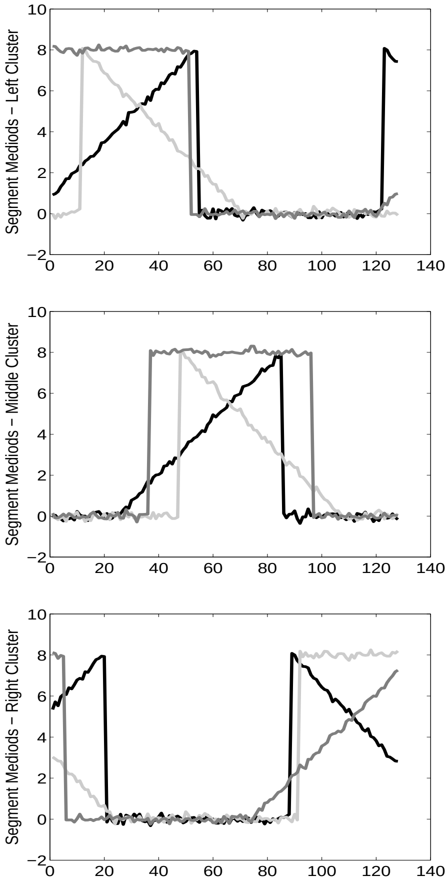


Fig. 5. The data point set for each cluster shown in Figure 2 includes representatives for all three Cylinder, Bell and Funnel features

the medium-coloured centre cluster representative is the (smoothed) cylinder feature, with the left and right humps being the (smoothed) funnel and bell features respectively. To investigate this possibility, we first present two definitions.

Definition 2.1. A *cyclic* data series is one where $|x_t - x_{t+\delta}| < \epsilon$ for some fixed δ for all $t = 1 \ldots (n - \delta)$, where ϵ is a small value compared to x_t and where δ is usually quite a bit smaller than n.

That is, a cyclic data series is one that, except for noise, repeats itself, where the value ϵ represents the level of noise in the data.

Definition 2.2. Let X be a data series and Z be the series of subsequences obtained by using the sliding windows technique on X. If we conduct a clustering on Z (ie. we are STS-clustering X) to obtain a set of clusters $C_j, j = 1 \ldots k$, a "segment" in C_j is a set of members of C_j that were originally contiguous in Z.

Then it is clear that our data series of concatenated Cylinder, Bell and Funnel features is cyclic, and that the cluster representatives shown in Figure 2 will each result from a set of cluster member data points made up of a number of distinct *segments*. We saw that we can avoid the smoothing effect by simply taking the medoid member of a cluster as its representative. However plotting the medoid of all points in a cluster will not expose to us whether points representing different features exist in the one cluster. What is required is to plot the medoid of each segment in a cluster, and this is what we could do. However, due to the cyclic nature of the Bell-Cylinder-Funnel data series, it turns out that only three significantly distinct segment mediods exist for each cluster. Hence, in Figure 5 we show only these three distinct segment mediods, for the left, middle and right clusters of Figure 2 in the top, middle and bottom plots of Figure 5 respectively. Clearly, there exist all three Cylinder, Funnel and Bell features in each plot in Figure 5. That is, data points representing each of these features exist in each of the clusters found in Figure 2, and so it is definitely not the case that the left cluster in Figure 2 is made up of only the Funnel feature, nor is the middle or right cluster made up of only the Cylinder and Bell features respectively. Clearly, the smoothing effect described above is not the only force at work in the non-intuitive results of Figure 2, and further investigation is required.

3. Clustering in Delay Space

Up to this point, our presentation has been caged in terms of the sequential time series framework that is popular in the data mining community. It now becomes useful to view this method in the wider context of the method of delays, as used in the research field of Dynamical Systems (H.Kantz and T.Schreiber, 1997). Forming a set of subsequences of length w from a time series X using the sliding windows method, and representing them in a w-dimensional space, can be viewed as just a special case of the method of delays, where Equation 2 generalises into

$$z_p = x_{p-(w-1)q}, x_{p-(w-2)q}, \ldots, x_{p-q}, x_p \tag{4}$$

where z_p is called a delay vector, and where q is a lag introduced so that the members in z_p need not be contiguous in the original time series X. The sequence of delay vectors Z is then formed as the set $Z = \{z_p \,|\, p = {\scriptstyle (w-1)q+1}, \ldots, n\}$. From the dynamical systems perspective, the idea is that a system will live in some phase space, and the aim is to reconstruct a space *equivalent* [2] to phase space, so that one can study the reconstructed space and know that any findings made will also hold for the original phase space. The dynamical systems literature has shown that, in general, such a space (called a Delay Space) can be formed as a space housing vectors constructed according to Equation 4, with certain restrictions on the values of q and w, and with the important restriction that data be sampled at equispaced time intervals (F.Takens, 1981). One of the central ideas in the approach is that the evolution of the dynamics of a system will form a data series that traces out some trajectory in delay space, and it is really this idea that we are interested in using to pursue the reasons for our strange results in Figure 5. The delay space for our Cylinder-Bell-Funnel experiment is of a dimension too great to represent graphically, so we take, as an example, a data series obtained from a small-swing, mathematical pendulum; having dynamics governed by the differential equation $\ddot{\theta} + \theta = 0$. The data series produced by this

[2] in the sense that most of phase spaces' properties are retained by the reconstructed space

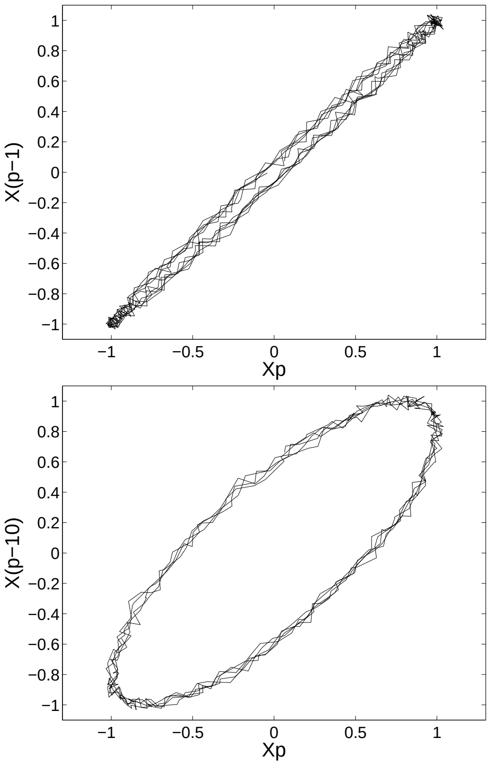


Fig. 6. Delay space representation of mathematical pendulum data series; with lag values $q = 1$ (top) and $q = 10$ (bottom)

system is a simple sinusoid $\theta = \theta_0 cos(\omega t + \alpha)$, and if we assume a realistic measurement process that includes noise, one can see in Figure 6 how this data series forms a trajectory in delay space, where the ellipse in the top plot corresponds to the trajectory in a delay space formed with $w = 2$ and $q = 1$, and the ellipse in the bottom plot to one formed with $w = 2$ and $q = 10$. An interesting aside is to note how taking $q = 1$ (i.e. the classical sliding windows approach) results in a "collapsed" trajectory lying along the bisectrix of delay space. This is almost never optimal from the clustering point of view, since dissimilar points will then lie close together. It is almost always more optimal to select $q > 1$, where methods for determining q can be found in H.Kantz and T.Schreiber (1997). It is for this reason that we focus on the $q = 10$ case in the following discourse.

With the concept in place of a data series as a trajectory in delay space, we now turn our attention back to Figure 5. We saw in that figure how each cluster was made up of a mixture of the Bell, Cylinder and Funnel features. A hint as to why this occurs can be seen by a 3-cluster clustering of the pendulum delay vectors shown in Figure 7, where the delay vectors in the same cluster have been marked with the same symbol; either a circle, a cross, or a triangle. Note how the cluster with the cross marker extends across two regions of very distinct types of delay vectors. That is, this cluster contains delay vectors representing the pendulum dynamics of: (i) the pendulum swinging left to right, and (ii) the pendulum swinging right to left. The reason this sort of clustering results is plain to see; we are using Euclidean distance as the measure of "similarity", and clearly these

[h]

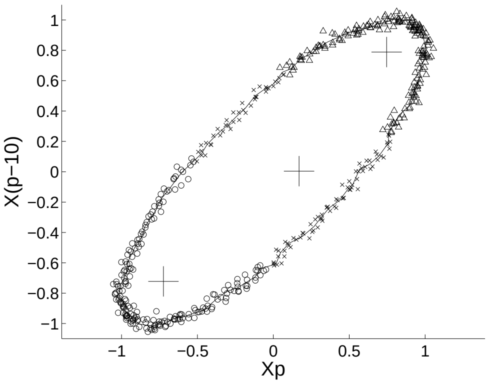


Fig. 7. Clusters formed using k-means based on the Euclidean distance measure

two regions are close by this measure. This phenomenon also seems the likely reason for the clustering outcome of the Bell-Cylinder-Funnel data in Figure 5; where delay vectors representing very different dynamics were also clustered together. Also of note in Figure 7 is the plot of the cluster representative for each cluster (as cross-hairs), determined as per Equation (3). The surprising result is that none of the representatives actually lie among the members they represent. This fact explains the smoothing effect identified in Section 2. More specifically, we identify two reasons for this phenomenon: (a) because the spectrum of dynamics represented by the delay vectors in the cluster are non-contiguous (such as in the cross-marker cluster in Figure 7), or (b) because (where the spectrum of dynamics in the cluster is contiguous) the trajectory of vectors in the cluster exhibits curvature (such as in the circle and triangle marker clusters in Figure 7). In both cases (a) and (b), calculating the mean according to Equation 3 will cause the cluster centre to not lie among the cluster members, resulting in the selection of a smoothed cluster representative.

4. A Solution

In the previous section we identified two problems with the STS-clustering approach used to date: it incorrectly clusters data that represents very different dynamics into the same cluster, and it incorrectly selects the cluster representative. Both problems stemmed from the use of Euclidean distance as the measure of (dis)similarity in delay space. Clearly the use of Euclidean distance is incorrect, but the obvious question then is what sort of distance metric should be used? In this section we propose a solution for the quite general class of time series produced by time-invariant, deterministic dynamical systems.

At the heart of finding an appropriate distance metric for delay vector space is the issue of defining precisely when two delay vectors are similar/dissimilar. We identify two types of similarity between delay vectors, (i) temporal similarity, and (ii) similarity of form, which we will call *formal* similarity. Figure 8 shows pictorially what we mean by temporal and formal similarity. Each plot represents a distinct delay vector, labelled z_a to z_e respectively. We propose, for example, that z_a, z_b and z_c exhibit temporal similarity and that z_a and z_d exhibit for-

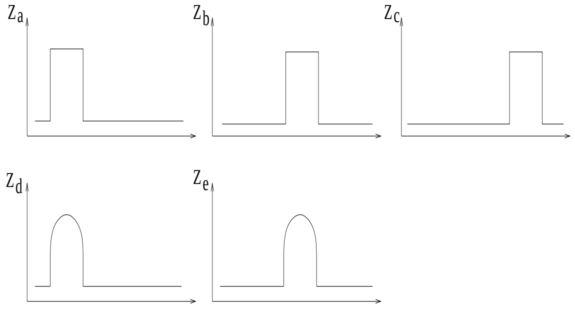


Fig. 8. Delay vectors showing temporal similarity, formal similarity, and a combination of both

mal similarity, with z_a and z_e exhibiting both temporal and formal similarity. Temporal similarity is a measure of how long it will take for one delay vector to evolve into another (eg z_b or z_c into z_a), or into something that has purely formal similarity to another (eg z_e into z_d, which has purely formal similarity to z_a; we will precisely define formal similarity in a moment). Clearly z_b will require less time (less additional signal) than z_c to evolve into z_a, so we should measure z_a as being closer to z_b than z_c. This motivates the following definition.

Definition 4.1. Temporal similarity between two delay vectors z_1 and z_2 is measured as the distance along the trajectory on which z_2 lies between z_2 and either (a) z_1 (if z_1 lies on the same trajectory as z_2), or (b) the vector on the trajectory through z_2 which has purely formal similarity to z_1.

We described formal similarity as a measure of similarity of form, or shape, between two delay vectors. However, given the subjective nature of the notion of form or shape, what is required is a definition. First define the idea of a flow.

Definition 4.2. Let $\mathcal{S}$ define a local neighbourhood to z_p in delay space. Define a vector field $\mathcal{V}$ on $\mathcal{S}$ such that $\mathcal{V}$ represents the delay space evolution of the system producing our data series for each point in $\mathcal{S}$ (i.e. that $\mathcal{V}(z_p^{\mathcal{S}})$, $z_p^{\mathcal{S}} \in \mathcal{S}$ is the tangent vector at $z_p^{\mathcal{S}}$ of the trajectory in delay space through $z_p^{\mathcal{S}}$). Then we denote $\mathcal{V}$ as defining the flow in $\mathcal{S}$.

Next, define precisely what we mean by formal similarity.

Definition 4.3. Let λ be a surface (in general a hyper-surface) defined within our local neighbourhood $\mathcal{S}$, where $z_p \in \lambda$. Form λ such that it is at all points orthogonal to the flow in $\mathcal{S}$ (i.e. if we denote $T^{\lambda}_{z_p^{\lambda}}$ as the tangent subspace of λ at z_p^{λ}, then for all $v \in T^{\lambda}_{z_p^{\lambda}}$, $\mathcal{V}(z_p^{\lambda})$ will be orthogonal to v, where $\mathcal{V}(z_p^{\lambda})$ denotes the tangent vector to the flow at z_p^{λ}). Then any vector z_p^{λ} lying on λ is defined to have purely formal similarity with z_p, where the dissimilarity of z_p and z_p^{λ} can be measured as the length of the shortest path lying on λ between z_p and z_p^{λ}.

Thus, the surface λ will define points with purely formal similarity to z_p on trajectories neighbouring the trajectory through z_p. Note that our definition of formal similarity is based on Definition 4.2, and that in this definition we have implicitly restricted the class of time series under discussion. The assumption is that, each time we visit a particular point in delay space, there will exist a *unique* tangent vector that describes the direction of the evolution of our dynamical system. Let the dynamical system from which our time series emanates be described by the following equations,

$$X_{n+1} = s(S_{n+1}) + \eta_1 \tag{5}$$

$$S_{n+1} = F(S_n + \eta_2; \mu) \tag{6}$$

where S_n is the current true state of the system, S_{n+1} the next true state, F is a nonlinear map from S_n to S_{n+1}, X_{n+1} is the measured value of S_{n+1} (i.e. an element in our time series) through the measurement function s, η_2 is a noise source perturbing the system by a random amount each time step; generally termed dynamical noise, η_1 is also a noise source; this time associated with the measurement process, and μ is a vector of system parameters. Generally the dynamical system described by Equations 5 and 6 is denoted as time-invariant if μ does not change over time; it is denoted as stochastic if η_2 results in a significant component of the final time series data, and as deterministic otherwise. Clearly, a time-invariant, deterministic system will produce a unique S_{n+1} for each S_n, resulting in a unique tangent vector at each z_p in delay space. It may be envisaged that this will only be the case for zero measurement noise η_1, but there are methods (e.g. H.Kantz and T.Schreiber (1997, §10.3)) for removing the effects of η_1 from the measurement process. In many situations, a time-invariant, deterministic dynamic system with measurement noise is an appropriate model for a time series source, however, be it, or be it not the case for a specific application, it is important to note this restriction on the framework we are proposing here. Note that deterministic dynamical systems can produce time series of high complexity, i.e. they are capable of exhibiting chaotic behaviour.

With this clarification made, we continue on with our discussion of temporal and formal similarity. Note that the idea of temporal similarity comes naturally from the problem at hand; it is clear how a z_p will evolve into some z_{p+1} depending on the dynamics of the system. This is in contrast to our presentation of formal similarity, where we have chosen to define this quantity in one particular way. We now try to provide some more motivation for the definition proposed. Given some delay vector z_p, we can imagine the process of creating a delay vector similar to z_p by perturbing each $x_k \in z_p, k = 1 \ldots w$ by a small amount. For a particular z_p, only one combination of the many possible perturbations of each x_k will result in the next delay vector z_{p+1}; where the particular combination of perturbations for this case is defined in delay space in the direction of the tangent vector to z_p. There are, of course, a multitude of possible perturbations that could be applied to the $x_k \in z_p$ which is not this one, and we have chosen to define a delay vector with purely formal similarity to z_p as one where each x_k is perturbed in a way that the relative proportion of perturbations applied to each x_k causes the selection of points (delay vectors) lying in the direction orthogonal to the flow at z_p. While we do not prove this is the best proportion of perturbations to apply, it is intuitively seems the right one since it results in a type of similarity most "distinct" to temporal similarity.

Our preceding discussion on similarity of delay vectors clearly suggests an alternative measure of distance to Euclidean distance in delay space; the measure should be some function of two distances: one measured always parallel to the flow, and the other measured always orthogonal to the flow. That is, we should choose to measure the similarity between two delay vectors as the combination of their temporal and formal similarities. The practical implementation of such an algorithm is not trivial, and is the focus of ongoing work. For the purposes of this paper, we conduct a STS-clustering experiment that uses the type of metric just proposed, however, for the sub-class of time-invariant, deterministic dynamical systems that produce time series that are cyclic (see Definition 2.1). Our aim is not propose an algorithm for cyclic data series per se, but rather to show on a real data series that the abstract concepts of temporal and formal similarity defined above do in fact lead to a valid clustering outcome. Specifically, we propose the following algorithm for calculating the distance between any two delay vectors formed from a cyclic data series.

Algorithm 4.1. Let X be a cyclic data series $X = x_t | t = 1 \dots n$ where $|x_t - x_{t+\delta}| < \epsilon$ for some fixed δ, and ϵ small. Assume we have a delay vector sequence Z created from X with lag q and vector length w, i.e. $Z = z_p | p = {\scriptstyle (w-1)q+1} \dots n$ where z_p is as per Equation 4. Then, create the single-cycle, mean data series (i.e. a sequence of length δ) as $\hat{Z} = \hat{z_r} | r = {\scriptstyle (w-1)q+1} \dots {\scriptstyle (w-1)q+\delta}$, where $\hat{z_r} = 1/\alpha \sum_{i=0}^{\alpha} z_{r+i\delta}$ and where α is the quotient (i.e. integral part only) of $(n-r)/\delta$. Each z_p will contribute to the value of one, and only one, $\hat{z_r}$. Denote as z_p^* this $\hat{z_r}$ for z_p. Then, the distance between any two delay vectors z_a and z_b can be calculated as $d(z_a, z_a^*) + d(z_b, z_b^*) + min(\sum_{i=a}^{b-1} d(z_i^*, z_{i+1}^*), \sum_{i=b}^{a-1} d(z_i^*, z_{i+1}^*))$ [3].

In essence, the algorithm finds the mean trajectory of the cycle, and then calculates distance between any two points as the sum of three distances: (a) the two distances given by the distance between each of the points and its nearest point on the mean trajectory, and (b) a third distance as the distance along the flow between the two "nearest" points just mentioned. One can see that this approach is in the spirit of our discussion above, since step (a) measures orthogonal to the flow, and (b) measures along it. K-means clustering, using Algorithm 4.1 as the distance measure, can then occur in the usual way.

Algorithm 4.2. (k-means) : (1) Randomly select initial cluster centres $\bar{z}_j, j = 1 \dots k$, (2) Associate each $z_p \in Z$ to its closest $\bar{z}_j$ using distance measured according to Algorithm 4.1, (3) Find the new cluster centres $\bar{z}_j$ as the points which minimise the sum of distances between $\bar{z}_j$ and each $z_p \in C_j$ [4], (4) return to step 2, or stop if cluster centres have not moved.

Figure 9 shows the clustering outcome for the pendulum data from Figure 6 using Algorithm 4.2. In contrast to the clustering in Figure 7, where k-means based on Euclidean distance was used, the clusters in Figure 9 are well formed, in that each represents a contiguous spectrum of pendulum dynamics. Even though the two regions in the cross-marker cluster in Figure 7 may be close in terms of Euclidean (i.e."straight-line") distance, in delay space, their distance along the flow (i.e. temporal distance) is large, and hence they are not clustered together in Figure 9. Figure 9 also shows, as cross-hairs, the cluster centres

[3] the minimum is required since one can pass from z_a to z_b in two distinct directions around the cycle

[4] this is an optimisation problem which can now not simply be solved by finding the mean of all cluster members, as is the case when using Euclidean distance. We note that the point we seek must lie on the mean trajectory $\hat{Z}$ (since a $\bar{z}_j$ not on $\hat{Z}$ would have the distance from itself to $\hat{Z}$ added to every $d(z_p, \bar{z}_j)$) so, for the purposes of this experiment, we find this point by naively searching though each point on $\hat{Z}$ "in" C_j (more precisely, by searching for those $\hat{z_r}$ that correspond to a z_p^* whose z_p is in C_j)

[b]

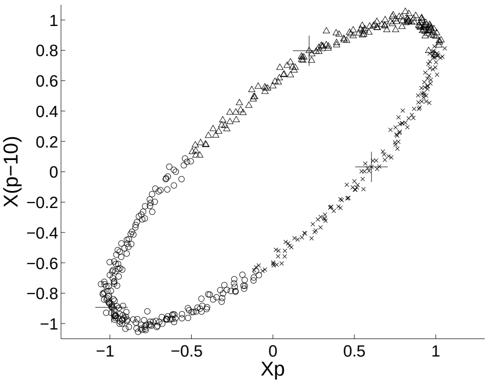


Fig. 9. Correct clustering: clusters contain regions of delay vectors representing similar pendulum dynamics

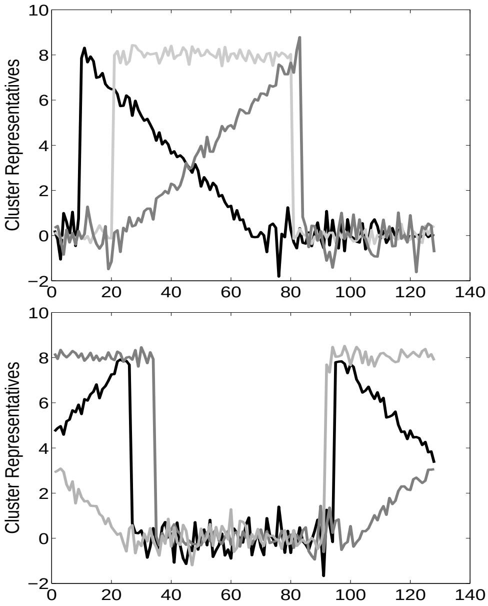


Fig. 10. Final, correct clustering of the Bell-Cylinder-Funnel data series

found by Algorithm 4.2. Unlike the representatives found in Figure 7, these lie among cluster representatives, and in each case look to give a good balanced representation of all members in their cluster.

With a successful clustering of the (cyclic) pendulum data, we now turn back to the original problem of clustering our (cyclic) Bell-Cylinder-Funnel data. Again, we use Algorithm 4.2 for clustering and Algorithm 4.1 for measuring distance. Figure 10 shows the results. The top window in the figure shows that we have successfully recovered the Bell, Cylinder and Funnel features, as desired. The bottom window shows an alternate set of features that were found using a different set of initial seeds for the k-means algorithm. Both sets of features are

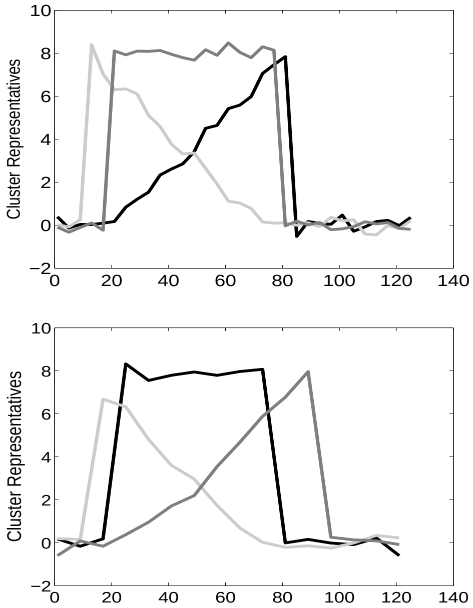


Fig. 11. Clustering outcome for the Bell-Cylinder-Funnel data when lag q set to 4 (upper) and 8 (lower)

a valid result of clustering the Bell-Cylinder-Funnel data series. One can see this fact more clearly by noting that the series can be formed by the concatenation of either of these sets of features. In fact, there will be whole spectrum of feature sets that will see the k-means algorithm return very similar total sum-of-distance values; corresponding to the different ways that one cycle can be broken into three equal length sequences.

Experience in the dynamical systems field tells us that we should be able to obtain essentially the same result as shown in Figure 10, but without requiring that delay vectors be formed with elements contiguous in the original time series, i.e. formed with a lag value $q = 1$. Recall how we saw with the pendulum example that it is generally not optimal to select a lag value $q = 1$, since it results in data points lying along the bisectrix of delay space. For Algorithm 4.1 presented here, we have required that distance be measured between two points in a way that must include the along-the-flow component, so we can guarantee that points in delay vector space from distinct region of the flow cannot accidently be measured as close. However there is another reason why we may want to set the value of q to be greater than one; namely for computational reasons. A q value greater than one means a delay vector of reduced length for the same “window” size, i.e. we will be clustering in a space of reduced dimension, leading to a much reduced requirement in computation. To investigate this possibility, we perform two experiments, where, in the first we set $q = 4$, $w = 32$, and in the second we set $q = 8$, $w = 16$ (all other facets of the original $q = 1$, $w = 128$ experiment remained the same). That is, in both these experiments, and in the original experiment, the size of the window across which we were looking for features is 128. Figure 11 shows the results. It can be seen that, for both the $q = 4$, $w = 32$, and $q = 8$, $w = 16$ experiments, the Cylinder, Bell and Funnel features were

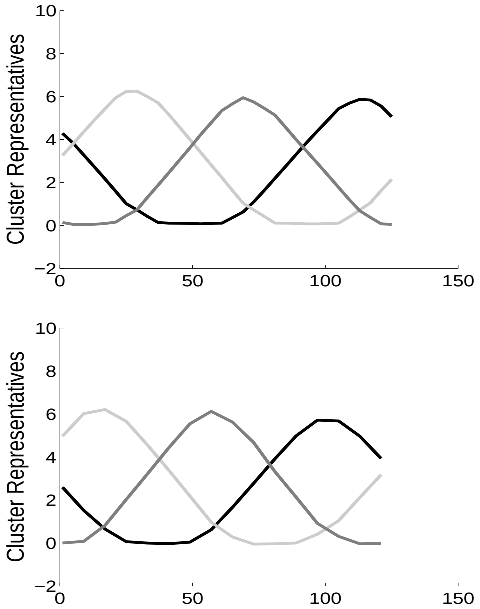


Fig. 12. Clustering outcome for the Bell-Cylinder-Funnel data when lag q set to 4 (upper) and 8 (lower) using the Euclidean distance metric

successfully recovered, and this was achieved with a much reduced computational requirement.

We saw in Figure 5 that each cluster in Figure 2 was made up of all three Bell, Cylinder and Funnel features. On inspection of the pendulum phase portrait in Figure 6, we postulated for the Bell-Cylinder-Funnel experiment that this result was accentuated by the selection of a lag $q = 1$, since the data points then concentrate along the bisectrix of delay space, resulting in points that represent very different signal dynamics lying close together. It is then interesting to confirm that we do not achieve a valid clustering outcome if we cluster the Bell-Cylinder-Funnel data using the Euclidean distance metric, but with a more sensible lag value, i.e. one that will “spread” data points in our delay space more widely. We choose two lag values, $q = 4$ (with $w = 32$) and $q = 8$ (with $w = 16$), and the results of the clustering for each are shown in upper and lower plots of Figure 12 respectively. Both plots show sine type waves, so clearly tuning the lag value has not helped, and our new distance metric approach would seem to be the correct approach for achieving a valid clustering outcome.

5. Conclusion

There are a number of conclusions to this work. First and foremost, that sequential time series clustering can indeed be meaningful, and that it is not, as recommended by E.Keogh et al. (2003), intrinsically flawed by definition. Second, that the key step in making it meaningful is by measuring distances in delay space correctly. We showed that the Euclidean distance measure that has been adopted by work in the area to date is flawed, and we introduced the idea of temporal and formal similarity in delay space for the class of time series produced by time-invariant, deterministic dynamical systems. We saw that the use of Euclidean distance gave rise to a number of problems, including the clustering together of regions in delay space that are close by this measure, but that really represent very distinct dynamics of the underlying time-series-producing system. In addition we saw that, even when the clustering outcome was correct, taking the cluster representative as the point with minimum average Euclidean Distance to all members in the cluster did not lead to a sensible outcome because of the generally "curved" nature of clusters in delay space. These results were in contrast to those obtained by our approach based on temporal and formal similarity, where experiments showed that sensible outcomes are achieved in both the areas of forming the clusters, and choosing the cluster representative. Finally, we saw that the sliding windows technique (i.e. lag $q = 1$) is generally suboptimal, and should be avoided, since it causes the delay space representation of the data series dynamics to be aligned closely along the bisectrix of delay space. However, we found that trying to obtain a valid clustering outcome using Euclidean Distance by tuning the lag parameter was not successful, although selecting a lag value greater than one was found to be beneficial from a computational point of view. Specifically, using the formal/temporal similarity metric and lag values greater than one, we produced correct clustering outcomes with reduced computational effort. The obvious further work to that presented in this paper involves determining a distance measuring algorithm for our temporal and formal similarity paradigm for the more general class of data series (i.e. other than cyclic) that can be produced by time-invariant, deterministic dynamical systems. This is the focus of our present work, and we hope to report on our findings soon.

References

Babcock, B., Datar, M., Motwani, R. and O'Callaghan, L. (2003). Maintaining variance and k-medians over data stream windows, *Proceedings of the 22nd Symposium on Principles of Database Systems (PODS)*.

Berkhin, P. (2002). Survey of clustering data mining techniques, *Technical report*, Accrue Software, San Jose, CA.
*http://citeseer.nj.nec.com/berkhin02survey.html

C.Li, P.S.Yu and V.Castelli (1998). Malm: A framework for mining sequence database at multiple abstraction levels, *Proceedings of the 7th ACM CIKM International Conference on Information and Knowledge Management*, Bethesda, MD.

C.Schittenkopf, P.Tino and G.Dorffner (2000). The benefit of information reduction for trading strategies, *Report Series for Adaptive Information Systems and Management in Economics and Management Science*. Report No. 45.

E.Keogh, J.Lin and W.Truppel (2003). Clustering of time series subsequences is meaningless: Implications for previous and future research, *Proceedings of the International Conference of Data Mining*.

Feng, X. and Huang, H. (2005). A fuzzy-set-based reconstructed phase space method for identification of temporal patterns in complex time series, *Transactions on Knowledge and Data Engineering* **17**(5): 601–612.

F.Takens (1981). Detecting strange attractors in turbulence, *Lecture Notes in Math.*, Vol. 898, Springer, New York.

G.Das, K.Lin, H.Mannila, G.Renganathan and P.Smyth (1998). Rule discovery from time series, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, New York, NY.

H.Kantz and T.Schreiber (1997). *Nonlinear Time Series Analysis*, Cambridge Univ. Press.

J.F.Roddick and M.Spiliopoulou (2002). A survey of temporal knowledge discovery paradigms and methods, *Transactions on Data Engineering* **14**(4): 750–767.

M.L.Hetland and P.Saetrom (2002). Temporal rules discovery using genetic programming and specialized hardware, *Proceedings of the 4th International Conference on Recent Advances in Soft Computing*, Nottingham, UK.

N.Radhakrishnan, J.D.Wilson and P.C.Loizou (2000). An alternate partitioning technique to quantify the regularity of complex time series, *International Journal of of Birfurcation and Chaos* **10**(7): 1773–1779.

P.Cotofrei (2002). Statistical temporal rules, *Proceedings of the 15th Conference on Computational Statistics*, Berlin, Germany.

P.Cotofrei and K.Stoffel (2002). Classification rules + time = temporal rules, *Proceedings of the 2002 International Conference on Computational Science*, Amsterdam.

P.Tino, C.Schittenkopf and G.Dorffner (2000). Temporal pattern recognition in noisy non-stationary time series based on quantization into symbolic streams: Lessons learned from financial volatility trading, *Report Series for Adaptive Information Systems and Management in Economics and Management Science.* Report No. 45.

R.Osaki, M.Shimada and K.Uehara (n.d.). A motion recognition mehtod by using primitive motions, *in* H.Arisawa and T.Catarici (eds), *Advances in Visual Information Management, Visual Database Systems*, Kluwer, pp. 117–127.

S.K.Harms, J.Deogun and T.Tadesse (2002). Discovering sequential association rules with constraints and time lags in multiple sequences, *Proceedings of the 13th International Symposium on Methodologies for Intelligent Systems*, Lyon, France.

S.K.Harms, S.Reichenbach, S.E.Goddard, T.Tadesse and W.J.Waltman (2002). Drought mining in a geospatial decision support system for drought risk management, *Proceedings of the 1st National Conference on Digital Government*, Los Angeles, CA.

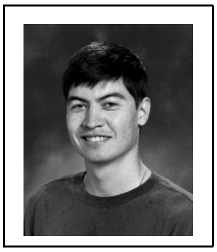
T.Mori and K.Uehara (2001). Extraction of primitive motion and discovery of association rules from human motion, *Proceedings of the 10th IEEE International Workshop on Human and Robot Communication*, Bordeaux-Paris, France.

T.Oates (1999). Identifying distinctive subsequences in multivariate time series by clustering, *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, USA, pp. 322–326.

T.Yairi, Y.Kato and K.Hori (2003). Fault detection by mining association rules in housekeeping data, *Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Montreal, Canada.

X.Jin, L.Wang, Y.Lu and C.Shi (2002). Indexing and mining of the local patterns in sequence database, *Proceedings of the 3rd International Conference on Intelligent Data Engineering and Automated Learning*, Manchester, UK.

Author Biography



Jason R. Chen received a B.E. degree from Sydney University, Australia, in 1991 and then worked mainly in the Banking and Finance industry until 1997. From 1997 to 2001, he completed his PhD at the Australian National University, Canberra, Australia, in the area of Robotics. From 2001 to the present he been a Research Engineer in the Research School of Information Science and Engineering, at the Australian National University. His research interests broadly include Robotics, Data mining, and AI.