Kernel-Density-Based Clustering of Time Series Subsequences Using a Continuous Random-Walk Noise Model

Anne Denton
Department of Computer Science
North Dakota State University
Fargo, North Dakota 58105-5164, USA
anne.denton@ndsu.edu

Abstract

Noise levels in time series subsequence data are typically very high, and properties of the noise differ from those of white noise. The proposed algorithm incorporates a continuous random-walk noise model into kernel-density-based clustering. Evaluation is done by testing to what extent the resulting clusters are predictive of the process that generated the time series. It is shown that the new algorithm not only outperforms partitioning techniques that lead to trivial and unsatisfactory results under the given quality measure, but also improves upon other density-based algorithms. The results suggest that the noise elimination properties of kernel-density-based clustering algorithms can be of significant value for the use of clustering in preprocessing of data.

1. Introduction

Finding patterns in time series subsequence data is a notoriously difficult problem. Standard clustering techniques, such as k-means and hierarchical clustering, result in clusters that are largely independent of the time series from which they originate [13]. Kernel-density-based clustering can lead to meaningful results, especially if an appropriate noise model is chosen [6]. Noise elimination in kernel-density-based clustering is based on the concept of a noise threshold, below which maxima in the density distribution are not considered as cluster centers [10]. Most time series data follow a noise distribution that differs from standard assumptions on randomness, and it can be beneficial to incorporate a more accurate noise distribution. While previous work [6] assumed a discrete random-walk model, the current paper is based on a continuous model that is much more realistic in most settings.

Time series clustering algorithms have been used directly as pattern extraction algorithms [15], and as preprocessing step for further data mining [5]. Neither application relies on assigning a cluster to all subsequences. It can be expected that noise elimination as part of a clustering process will increasingly become important in the preprocessing of data for classification. While attribute selection in classification has traditionally been performed on the basis of classification quality [14] this approach is vulnerable to the ”curse of dimensionality”, and does not scale well to a large number of attributes. It is, therefore, important to develop preprocessing techniques that are able to distinguish between noise and meaningful data without using the class label. In many current data mining problems, objects are characterized by diverse data, that can include time series data as well as other attributes. In this setting it is very important that attributes derived from a time series should contribute information that can be beneficial to a classification process, and noise elimination in clustering gains new importance.

The current paper examines the time series subsequence clustering problem from a classification perspective, where the class label is the correct identification of the entire clustering rather than any individual cluster. Quality of pattern extraction is evaluated by testing to what extent the assignment to clusters can be used to identify the type of time series from which the clusters were derived. K-means and other partitioning methods produce a trivial and very poor result under this measure because a subsequence is guaranteed to be assigned to some cluster even if the clustering was produced based on a different time series. In kernel-density-based clustering, subsequences may be identified as noise, and will then not be assigned to the respective time series. It will be shown that the ratio of correctly to incorrectly assigned subsequences can become very large for some models that discard a substantial number of subsequences as noise.

The paper is organized as follows. Section 2 introduces the concepts and techniques relevant to the problem, section 3 explains the concept underlying the algorithm; section 4 describes the implementation; section 5 discusses the experimental evaluation, and section 6 concludes the paper.

2. Background

The term "time series" is typically used for sequential real-valued data that are collected at regular time intervals. Many other types of sequential data can be distinguished, such as sequences of integer-valued and categorical data. Examples of time series data include ECG measurements, weather recordings, and stock market data.

Definition 1: A time series $T = t_1, \ldots, t_n$ is a sequence of real numbers, corresponding to values of an observed quantity, collected at equally spaced points in time.

Definition 2: A subsequence of time series $T = t_1, \ldots, t_n$, with length w, is a sequence $S = t_m, \ldots, t_{m+w-1}$ with $1 \leq m \leq n-w+1$. The process of extracting subsequences by incrementing m in steps of one is called application of a sliding window.

Subsequences are commonly represented as vectors in a w-dimensional vector space. Comparison of subsequences can be done through any measure defined on a vector space, such as an L_p norm. A typical choice is the L_2 norm, i.e., Euclidean distance. Some distance measures that are specific to time series data, such as dynamic time warping (DTW) [2] and longest common subsequence similarity (LCSS) [17] are not suitable for the short subsequences (w between 8 and 16) considered in this paper.

Using Euclidean distance on the raw time series data leads to clusterings that are dominated by the mean of the subsequence. Hence, it is common to normalize the data. A common type of normalization is Z-normalization in which the mean is subtracted and the subsequence is divided by its standard deviation [9]. In this paper a different approach, which follows more naturally from the random-walk model, is used. The mean is subtracted as in Z-normalization, but the time series is then normalized based on the root-mean-square of differences between adjacent data points. Note that the differences are only used for the normalization process and not as the actual data to be compared as in [7].

A very simple model of a time series is "strict white noise" as defined in [16].

Definition 3: A Strict white noise time series is a normally distributed sequence of values $\{e_t\}$ corresponding to time t. Mean μ and variance σ are assumed to be the same for all time points.

$$\begin{aligned} E\{e_t\} &= \mu, \quad \mathrm{var}\{e_t\} = \sigma^2, \forall t, \\ \mathrm{cov}\{e_t, e_s\} &= 0, \forall t \neq s. \end{aligned} \tag{1}$$

Subsequences of such a time series, when represented as vectors in a w-dimensional vector space, correspond to data points, for which each attribute follows a Gaussian noise distribution. The assumption of Gaussian noise is commonly made for data from a wide variety of sources and data mining algorithms are typically able to handle this type of noise.

Strict white noise behavior is, however, not often observed in time series data. It is much more common that successive points in a time series are correlated. For example, a weather report that consistently predicts tomorrow's average temperature to be the yearly average would typically perform much more poorly than a report that claims that tomorrow's average temperature will be the same as today's. It is, therefore, important to define a noise model for which the jumps between successive points are assumed to be randomly distributed rather than the values themselves. A time series that follows this model is called a random-walk time series.

Definition 4: A Gaussian random-walk time series is a normally distributed sequence that satisfies

$$\begin{aligned} &X_t - X_{t-1} = e_t \\ &E\{e_t\} = 0, \quad \mathrm{var}\{e_t\} = \sigma^2, \ \forall t, \\ &\mathrm{cov}\{e_t, e_s\} = 0, \ \forall t \neq s. \end{aligned} \tag{2}$$

An example of such a time series is provided among the data sets from the UCR time series repository [12]. The values of the Gaussian random-walk time series vary continuously. Previous work [6] on clustering of time series data used a limited representation in which successive points were assumed to differ by unit steps. Such discrete random-walk time series are typically of less practical relevance and will, therefore, not be considered in this paper.

2.1. Kernel-Density-Based Clustering

Clustering based on kernel-density estimation is an established technique in many contexts [3, 4, 10]. The kernel density estimator for n data points $x_i, i = 1, \ldots, n$ in a d-dimensional space is given by

$$\hat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) \tag{3}$$

where the kernel function $K(\mathbf{x})$ is normalized

$$\int_{\mathbf{R}^d} K(\mathbf{x}) d\mathbf{x} = 1. \tag{4}$$

One of the most common choices as kernel function is a Gaussian kernel. All calculations in this paper use this kernel function

$$K^{(G)}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{|\mathbf{x}|^2}{2}\right) \tag{5}$$

The Gaussian kernel is radially symmetric. A profile can, therefore, be defined through

$$K^{(G)}(\mathbf{x}) = c_{k,d} k(|\mathbf{x}|^2) \tag{6}$$

where $c_{k,d}$ is a normalization constant that guarantees the normalization in equation (4). Cluster centers are defined as local maxima in the density landscape. In [10] these local maxima are called density attractors and in [4] modes. While [10] uses a feature space grid to assist in the search for maxima, [4] parses the table of data points for each hill climbing step. The latter approach was chosen in this paper because it avoids representing the high-dimensional feature space. To identify modes, all data points are taken as starting points and their location is updated through a sequence of hill climbing step. Updates, i.e. differences between tentative cluster center locations for successive steps, are computed as

$$\mathbf{m}_h(\mathbf{x}) = \frac{\sum_{i=1}^{n} \mathbf{x_i} g\left(\left|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right|^2\right)}{\sum_{i=1}^{n} g\left(\left|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right|^2\right)} - \mathbf{x} \tag{7}$$

where $g(x) = -k'(x)$ is the negative derivative of the kernel profile. Data points are associated with the cluster center to which they are attracted. Only modes above a threshold t are considered cluster centers. Data points that are attracted to modes below t are outliers or noise. When testing data points for cluster membership only those points were considered for which the density at the original location already exceeded t. In a standard application of density-based clustering t is considered a parameter that has to be selected by the user. The number of free parameters is often quoted as drawback of density-based clustering. In this paper a comparison with random-walk data is used to determine t, effectively eliminating this parameter.

The next section addresses differences between standard clustering problems and time series subsequence clustering and their impact on kernel-density-based clustering.

3. Scaling the Coordinate System

Kernel-density-based clustering is robust against noise, provided the noise leads to an approximately constant density surface. Constant contributions to the density distribution do not affect the position of maxima. This section will demonstrate that, by default, the density landscape corresponding to a random-walk time series cannot be considered constant. Using a standard implementation of density-based clustering would result in cluster centers that are due to random-walk behavior, i.e., noise. This section also demonstrates how to scale the coordinate system to avoid this problem.

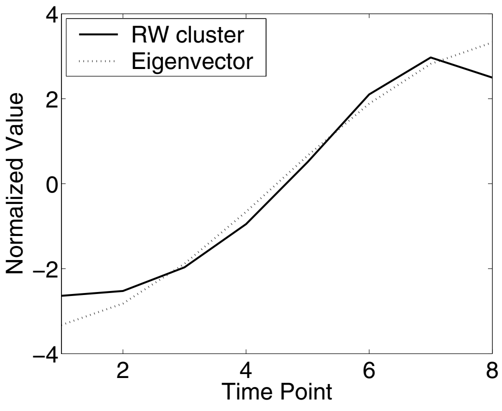

Figure 1: Most relevant eigenvector for $w = 8$ together with the only cluster of random-walk time series for $h = 3$, both normalized according to equation (16).

Consider the density distribution of a white noise time series. The kernel-density function has the shape

$$\rho(\mathbf{x}') \sim \int \mathrm{d}x_1 \cdots \mathrm{d}x_w \exp\left(-\sum_{i=1}^{n} \frac{(x_i' - x_i)^2}{2h^2}\right) \times \exp\left(-\sum_{i=1}^{w} \frac{x_i^2}{2\sigma^2}\right) \tag{8}$$

where w is the window size, and thereby dimensionality, of the subsequences, σ is defined in 2 and h is the width of the Gaussian that is used as kernel function.

The distribution in equation (8) is invariant under rotations of the coordinate system, implying that the distribution $\rho(\mathbf{x}')$ depends on only the absolute value $r = |\mathbf{x}'|$. For data points of interest, r is kept fixed, and hence the distribution is constant for the space that is explored. Note that, strictly, a normalization process that results in fixed r cannot preserve a Gaussian distribution in the individual coordinates: The normalization limits the maximal value in any one dimension to $\sqrt{w}$ for $\sum_1^w x_i^2 = w$ and, thereby, causes the tail of the Gaussian distribution to be cut off. For large w this effect is negligible and it will be ignored in the following.

As mentioned earlier, noise in time series data are more likely to be described well by a random-walk series than by white noise. A random-walk time series has the following density distribution

$$\rho(\mathbf{x}') \sim \int \mathrm{d}x_1 \cdots \mathrm{d}x_w \exp\left(-\sum_{i=1}^{n} \frac{(x_i' - x_i)^2}{2r^2}\right) \times \exp\left(-\sum_{i=2}^{w} \frac{(x_i - x_{i-1})^2}{2\sigma^2}\right) \tag{9}$$

This distribution is not rotationally invariant. The following steps have the goal of tranforming the distribution such that it becomes rotationally invariant. As a first step, the argument of the second exponential function ($S(\mathbf{x})$) is expanded

$$\rho(\mathbf{x}') \sim \int \mathrm{d}x_1 \cdots \mathrm{d}x_w \exp\left(-\sum_{i=1}^{n} \frac{(x'_i - x_i)^2}{2r^2}\right) \exp\left(-\frac{S(\mathbf{x})}{2\sigma^2}\right) \tag{10}$$

with

$$S(\mathbf{x}) = x_1^2 + x_w^2 + 2\sum_{i=2}^{w-1} x_i^2 - 2\sum_{i=2}^{w} x_{i-1}x_i \tag{11}$$

$S(\mathbf{x})$ can be written in matrix form $S = \mathbf{x}^T\mathbf{A}\mathbf{x}$

$$S(\mathbf{x}) = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_w \end{pmatrix}^T \begin{pmatrix} 1 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ 0 & -1 & 2 & \dots & 0 \\ \dots & & & & \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_w \end{pmatrix} \tag{12}$$

Matrix $\mathbf{A}$ is then diagonalized. In practice, this is done numerically. For $w = 4$ the diagonalization can be done analytically, so this case will be used as an example. The diagonalized matrix for $w = 4$ is

$$\mathbf{D} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & (2-\sqrt{2}) & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & (2+\sqrt{2}) \end{pmatrix} \tag{13}$$

It can be seen that the system has four distinct eigenvalues. The corresponding eigenvectors are

$$u_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} u_2 = \frac{1}{2\sqrt{2-\sqrt{2}}} \begin{pmatrix} -1 \\ 1-\sqrt{2} \\ -1+\sqrt{2} \\ 1 \end{pmatrix} \tag{14}$$

$$u_3 = \frac{1}{2}\begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} u_4 = \frac{1}{2\sqrt{2+\sqrt{2}}} \begin{pmatrix} -1 \\ 1+\sqrt{2} \\ -1-\sqrt{2} \\ 1 \end{pmatrix}$$

The first eigenvector, u_1, corresponding to eigenvalue 0, does not contribute in the practical implementation, because the mean of each subsequence is subtracted as a normalization step. It can be shown, through approximate evaluation of the density integral in equation (8) that the eigenvector corresponding to the smallest non-zero eigenvalue, namely u_2, is responsible for the density maximum in the untransformed system. The calculation is omitted due to lack of space. Figure 1 shows the corresponding eigenvalue for $w = 8$ together with a result of density-based clustering without scaling.

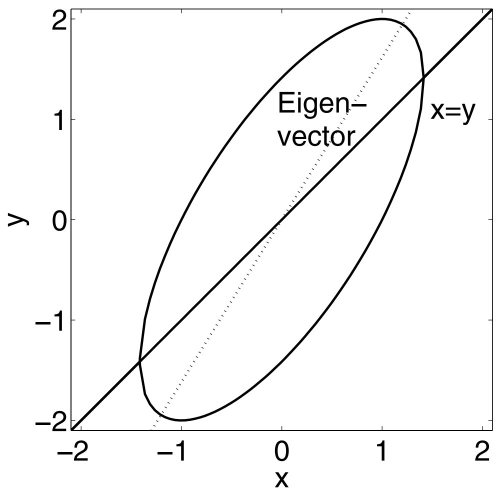

Figure 2: Simplified 2-dimensional system.

The remainder of the transformation is straightforward. The eigenvectors $u_1 \dots u_4$ define a rotation matrix V that can be used to transform the coordinate system

$$\mathbf{x}_{rot} = V^T\mathbf{x} \tag{15}$$

Transformation of $\mathbf{A}$ results in the diagonal matrix from equation (13) $\mathbf{D} = \mathbf{V^T A V}$. Note that the normalization in equation (14) was chosen such that $\mathbf{T}$ is orthogonal, which is different from the normalization applied in figure 1. A natural normalization condition for time series subsequences is

$$\sum_{i=2}^{w} (x_i - x_{i-1})^2 = (w-1) \tag{16}$$

In the limit of $w \to \infty$, a random-walk time series with $\sigma = 1$ satisfies this normalization. In the new coordinate system, equation (12) can be written as

$$\frac{1}{2\sigma^2} S(\mathbf{x}) = \frac{1}{2\sigma^2} \mathbf{x}_{rot}^T \mathbf{D} \mathbf{x}_{rot} \tag{17}$$

Note that the system can be interpreted as effectively having different values of σ for each dimension

$$\sigma_i^{eff} = \frac{\sigma}{\sqrt{d_i}} \quad \text{for} \quad d_i \neq 0 \tag{18}$$

where d_i is the i^{th} eigenvalue. Note that $d_1 = 0$ has to be excluded. This is consistent with the normalization procedure of $\mathbf{x}$ that involves subtracting the mean, or in other words, the component corresponding to u_1. Different values for σ in different dimension clearly violate rotational symmetry. To achieve rotational symmetry the dimensions are scaled by $\sqrt{\mathbf{D}}$, resulting in an overall transformation

$$\mathbf{x}_{scaled} = \sqrt{\mathbf{D}}\mathbf{V}^T\mathbf{x} \tag{19}$$

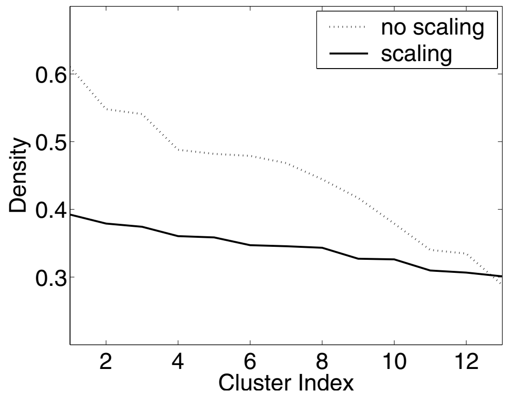

Figure 3: Comparison of densities of the first 13 clusters from random-walk data for the algorithm with and withouth scaling (h = 1, nodes = 4).

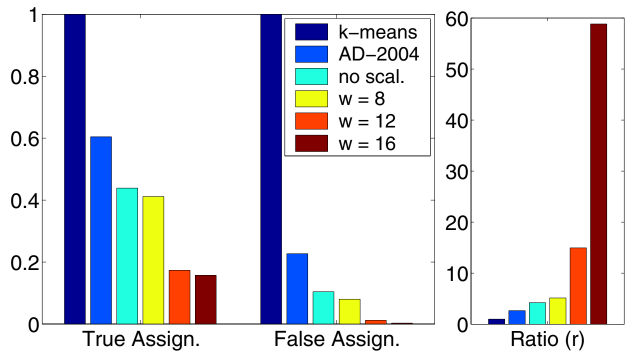

Figure 4: Relative number of correct (T) and incorrect (F) assignments, added over all data sets, and their ratio (r=T/F).

In the scaled coordinate system rotational invariance is satisfied, and the original σ describes the distribution for all dimensions

$$\frac{1}{2\sigma^2}S(\mathbf{x}) = \frac{1}{2\sigma^2}\mathbf{x}^T_{scaled}\mathbf{x}_{scaled} \tag{20}$$

3.1. Intuitive Interpretation

It may initially seem counter-intuitive that one random-walk time series should correspond to a higher kernel-density than all others, and thereby dominate the system unless scaling is applied. This section will discuss a system that is defined in a slightly different way, and captures the essential behavior already in two dimensions.

A 2-dimensional system can be defined by looking at time series subsequences with $w = 3$ for which the first time point is assumed to be fixed at $x_0 = 0$. The density distribution for the remaining two time points is then

$$\begin{aligned}\rho(\mathbf{x}') \quad &\sim \quad \int \mathrm{d}x_1 \mathrm{d}x_2 \exp\left(-\sum_{i=1}^{2}\frac{(x'_i - x_i)^2}{2r^2}\right) \\ &\exp\left(-\frac{x_1^2 + (x_2 - x_1)^2}{2\sigma^2}\right)\end{aligned} \tag{21}$$

The resulting equation for $S(\mathbf{x})$ analogous to equation (12) is

$$\begin{aligned}S(\mathbf{x}) &= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}^T \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= \mathbf{x}^T \mathbf{A}_s \mathbf{x}\end{aligned} \tag{22}$$

This system can be visualized by plotting the line of $S(\mathbf{x}) = 2$ which corresponds to the normalization condition equivalent to equation (16). Figure 2 shows the resulting ellipse. In this simple two-dimensional case it is easy to see why an ellipse should be expected: If variable x_1 and x_2 were each normally distributed with normalization condition $x_1^2 + x_2^2 = 2$ we would see the circle with radius $\sqrt{2}$. In the random-walk system x_2 is replaced by $u_2 = x_2 - x_1$. Hence, the ellipse is the result of adding $x_2 = x_1$ to the functions representing the circle $x_2 = \pm\sqrt{2 - x_1^2}$. Figure 2 also depicts the direction of the relevant eigenvector. It now becomes clear what causes the non-trivial direction of the most relevant eigenvector.

4. Implementation

The implementation was done in MATLAB as an extension of [11], a kd-tree-based [1] MATLAB toolbox for kernel-density estimation. Modes are evaluated using a hill-climbing algorithm. As in [6] sequences are not only compared in their original position but also in shifted locations. This shifting is necessary to prevent a characteristic pattern that extends over only part of the time interval from appearing as multiple cluster centers. The comparison process involves taking the first m time points of one subsequence and the last m time points of the other, normalizing both, calculating the distance, and applying a penalty as in [6].

The algorithm has the following components. Subsequences are normalized by subtracting the mean. For practical reasons a normalization was chosen such that $\sum_{i=2}^{w}(x_i - x_{i-1})^2 = (w-1)^2$ rather than the normalization from equation 16. A transformation matrix is then constructed as described in section 3 and all subsequences are scaled according to equation (19). Note that a back transformation has to be applied every time a distance is calculated because the shifting of locations described in the first paragraph of this section has to be applied in the original space. The back tranformation, which is also applied to the final cluster centers, is given by

$$\mathbf{x} = \mathbf{V}\sqrt{\mathbf{D}}^{-1}\mathbf{x}_{scaled} \tag{23}$$

Note that the inversion of $\sqrt{\mathbf{D}}$ is easy to achieve since the matrix is diagonal. Strictly speaking, the matrix D from equation (13) cannot be inverted because of its eigenvalue 0. However, only the $(w-1)$ eigenvectors corresponding to the non-zero eigenvalues have to be considered as basis vectors for the transformed space, due to the normalization. Transformation from the $(w-1)$-dimensional space back into the w-dimensional original space is straight forward, and unaffected by the eigenvalue 0.

A density landscape is constructed based on the transformed data points. Comparisons between cluster centers are done using the same algorithm that was described for individual sequences, and matches between cluster centers may involve a time shift. This significantly reduces apparent redundancies between cluster centers. The procedure requires an additional heuristic cut-off distance. In the current paper, only those modes of the density distribution are considered as separate cluster centers that do not have a cluster center with higher density within a range of $0.1\sqrt{w}$. A threshold has to be identified below which cluster centers are considered as noise. For this purpose a clustering is performed on the first 500 points of the random-walk time series from [12]. The maximum density of cluster centers in this clustering is chosen as noise threshold. The distribution of densities of cluster centers was also used to test the algorithm. Figure 3 shows that the density for the first 13 cluster centers shows a much slower decrease for the algorithm that uses scaling than for the same algorithm without scaling. A threshold that does not result in cluster centers due to noise in the scaled algorithm can still return noise-generated cluster centers in the unscaled algorithm. If the noise-threshold is chosen higher, the number of clusters that can be identified is reduced.

The testing is done as follows. Normalization and scaling are done in the same way as when constructing clusters. Assignment to a cluster center is achieved using hill-climbing on the same density landscape. Only those data points that have a density exceeding the noise threshold before beginning the hill-climbing are assigned to a cluster center. Due to the shifting procedure a subsequence can be assigned to multiple clusters. In the final evaluation a subsequence is considered as predicting a particular time series if it is assigned to at least one of the clusters of that time series. Note that evaluation is done on a different part of the time series for all experiments. A correct assignment corresponds to an instance of a subsequence being assigned to at least one cluster derived from a time series that has the same source. An incorrect assignment signifies that a subsequence matches at least one cluster of a time series of a different source.

Table 1: Results of evaluation

data set		AD-2004	no sc.	w = 8	w = 12	w = 16
total	T	0.60	0.44	0.41	0.17	0.16
	F	0.23	0.10	0.08	0.011	2.7e-3
	r	2.66	4.22	5.15	15.0	58.8
ecg	T	0.92	0.95	0.91	0.87	0.91
	F	0.13	0.14	0.085	0.033	0.016
	r	7.26	7.02	10.79	26.2	54.9
buoy	T	0.042	0.016	0	0	0
	F	0.026	6e-3	0	0	0
	r	1.62	2.67	NaN	NaN	NaN
balloon	T	0.46	0.5	0.40	0.30	0.29
	F	0.09	0.061	0.027	0.003	1.3e-3
	r	5.16	8.30	15.0	99.3	232
glass	T	0.52	2e-3	0.18	0.10	8e-3
	F	0.073	1.5e-3	0.047	0.052	4.3e-3
	r	7.11	1.33	3.89	1.88	1.88
steam	T	0.088	0.14	0.048	8e-3	8e-3
	F	0.053	9.1e-3	0.025	3e-3	3e-4
	r	1.65	1.59	1.96	2.67	32.0
speech	T	0.89	0.64	0.76	0.19	0.17
	F	0.48	0.22	0.23	8e-3	1e-3
	r	1.85	2.89	3.36	23.8	168
quake	T	0.87	0.55	0.76	0.05	4e-3
	F	0.49	0.18	0.18	4.3e-3	3e-4
	r	1.76	2.99	4.24	11.8	16
ocean	T	0.68	0.4	0.31	0.04	0.028
	F	0.26	0.057	0.05	0	0
	r	2.63	6.98	6.15	Inf	Inf
darwin	T	0.97	0.74	0.33	2e-3	0
	F	0.44	0.18	0.08	5e-4	5e-4
	r	2.18	4.14	4.08	4.0	0

5. Experimental Evaluation

The algorithm was evaluated on several standard data sets from the UCR Time Series Data Mining Archive [12] as well as on an ECG series (MIT-BIH Arrhythmia Database: mitdb100) from PhysioBank [8] (ecg). Descriptions of the data sets from [12] are distributed with the data. The series ecg was compressed by averaging over 20 consecutive values, the buoy series from the UCR Archive by averaging the buoy_sensor series over 4 values. Table 1 lists experimental results. Note that buoy stands for buoy_sensor, glass for the first series of glassfurnace, steam for the second series of steamgen, quake for earthquake, and shear for ocean_shear.

The evaluation was done as follows. For each time series clusters were constructed based on the first 500 data points. As a first step, subsequences were extracted through application of a sliding window. The window size was $w = 8$, unless otherwise stated. Evaluation was done on the next

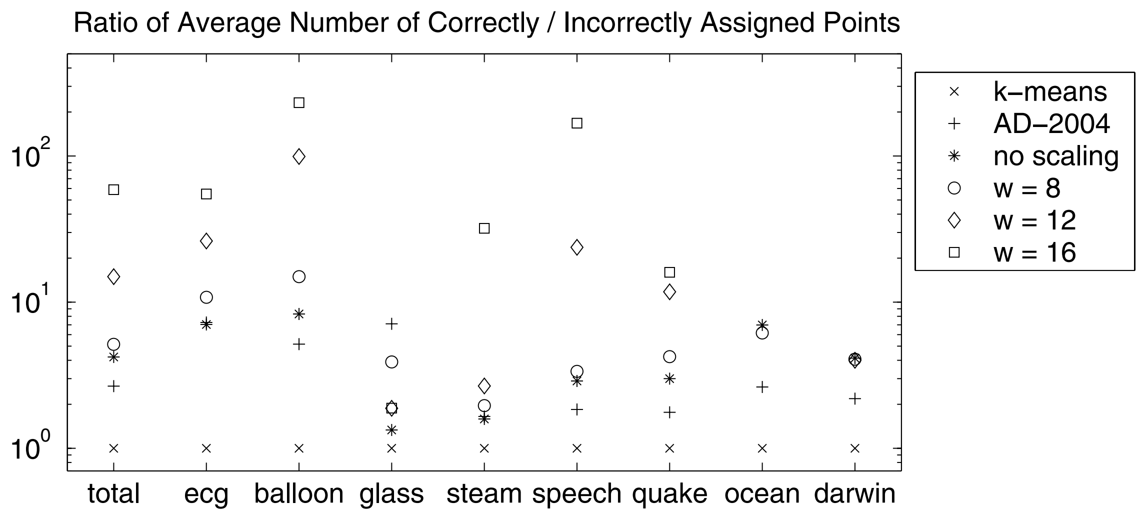

Figure 5: Comparison of results (for parameters see text).

500 data points. Results list the fraction of correctly / incorrectly assigned sequences. A subsequence can be assigned to a cluster from the correct time series, and can possibly also be assigned to a cluster of any of the other time series. In fact, k-means and other partitioning algorithms assign a subsequence to one cluster of every time series against which it is tested. The relative number of correct assignments is therefore guaranteed to be one, and the relative number of incorrectly assigned subsequences is one as well. This result is trivial and does not require an implementation.

Five algorithms were compared besides k-means. Density based clustering with compensation for disrete random-walk noise was performed using the parameters from [6]. Parameters for the density-based clustering with $w = 8$ with and without scaling were $h = 2$ and threshold $t = 5e - 5$. For $w = 12$ the choice was $h = 2$ and $t = 6e - 7$. For $w = 16$ the width of the kernel function was chosen to be $h = 3$ and the threshold $t = 5e - 11$. Note that the choice of threshold was determined from the density distribution of a random-walk time series that was clustered under the same conditions.

Figure 4 compares the average results over all data sets. It can be seen that the average number of correct assignments is relatively large for traditional algorithms. K-means and other partitioning algorithms trivially have the largest possible relative number of correct and incorrect assignments, namely one. Density-based algorithms, in general, allow outliers and both types of assignments are likely to be smaller than their maximum value. The implementation from [6] leads to the largest number of correct and incorrect assignments and the poorest ratio among the density-based algorithms. To understand this it is important to note that the normalization in the current paper is based on differences between adjacent points whereas the normalization in [6] used the standard deviation of the time series values. Figure 4 shows that the new algorithm performs better even without scaling, which suggests that the normalization based on differences between adjacent points is more successful.

The relative number of correct assignments is smaller for the implementation with scaling than for the implementation without scaling and the implementation from [6]. More importantly, the number of incorrect assignments follows the same trend but the decrease occurs much faster. The ratio between correct and incorrect assignments is largest for the implementation described in this paper. This result supports the claim that the algorithm with scaling is better able to separate noise from meaningful data. For larger window sizes the reduction of recognized clusters and the increase in the ratio between correct and incorrect assignments becomes even more significant. This is to be expected, since a long subsequence is less likely to match a cluster over its entire length and more likely to be considered noise. However, if a sequence does match a cluster, this match is expected to be reliable.

Details are listed in table 4 and the ratio of correct to incorrect assignments is plotted in figure 5. Correct assignments (T) were evaluated on data from the same time series, albeit a different section of it. Incorrect assignments were determined from all other data sets in the table and the result was divided by the total number of subsequences that were tested. The ratio r is calculated as r=T/F. It can be noted that for one data set — the ocean_shear data set from [12] — the number of incorrect assignments is 0 for $w = 12$ and $w = 16$ and the ratio, therefore, is infinite and cannot be plotted. All data sets from [6] are listed as well as several others. Results for the random-walk time series are not listed since the recognition threshold was picked based on the random data sets, and the number of matches was 0 for all runs of the new algorithm. Another data set, the buoy data set also leads to no assignments to clusters for the algorithm presented in this paper. This is an indication that the cluster result from [6] was still due to noise despite the compensation for discrete random-walk noise and that compensating for continuous rather than discrete random-walk noise is important. It can, furthermore, be seen that the subsequences are assigned much more reliably for some data sets than for others. Time series with a deterministic physical origin lead to far better results than measurements from less deterministic sources. The best results were achieved for speech data, the balloon data set, and ecg data. For these data sets, correct assignments were more likely than incorrect ones by about two orders of magnitude for $w = 16$.

6. Conclusions

A clustering algorithm for time series subsequences was introduced that considers continuous random-walks as its noise model. The algorithm makes use of a coordinate transformation on the feature space that results in a uniform noise threshold for all valid input sequences. Evaluation was based on a new measure that specifically tests the success of distinguishing cluster members from noise. According to the new evaluation measure, the quality of results is improved by more then two orders of magnitude on some data sets compared with k-means. These results suggest that the algorithm can be productively used in time series subsequence clustering and may also hold concepts that can be generalized beyond this particular application. The concept of using kernel-density-based clustering with a specific noise-model for improved noise elimination has the potential of being applied to many other settings.

7. Acknowlegements

Thanks to Alan Denton for proofreading the paper.

References

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), 1975.

[2] D. Berndt and J. Clifford. *Advances in knowledge discovery and data mining*, chapter Finding patterns in time series: a dynamic programming approach, pages 229–248. AAAI Press, Menlo Park, CA, 1996.

[3] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[4] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[5] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. In *Proceedings of the IEEE Int. Conf. on Data Mining*, Rio de Janeiro, Brazil, 1998.

[6] A. Denton. Density-based clustering of time series subsequences. In *In Proceedings of The Third Workshop on Mining Temporal and Sequential Data (TDM 04) in conjunction with The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, 2004.

[7] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani. Mining the stock market (extended abstract): which measure is best? In *Sixth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 487–496, Boston, MA, 2000.

[8] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000 (June 13).

[9] D. Goldin and P. Kanellakis. On similarity queries for time-series data: Constraint specification and implementation. In *1st Int'l Conf. on the Principles and Practice of Constraint Programming, LNCS 976*, pages 137–153. Springer, Sep. 1995.

[10] A. Hinneburg and D. Keim. A general approach to clustering in large databases with noise. *Knowl. Inf. Syst.*, 5(4):387–415, 2003.

[11] A. Ihler. Kernel density estimation toolbox for matlab (r13), accessed 04/2003.

[12] E. Keogh and T. Folias. The ucr time series data mining archive, 2002.

[13] E. Keogh, J. Lin, and W. Truppel. Clustering of time series subsequences is meaningless: implications for previous and future research. In *Proceedings of the IEEE Int. Conf. on Data Mining*, pages 115–122, Melbourne, FL, 2003.

[14] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 1–2:273–324, 1997.

[15] P. Patel, E. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. In *Proceedings of the IEEE Int. Conf. on Data Mining*, Maebashi City, Japan, 2002.

[16] M. Priestley. *Non-linear and non-stationary time series analysis*. Academic Press, 1988.

[17] M. Vlachos, D. Gunopoulos, and G. Kollios. Discovering similar multidimensional trajectories. In *Proceedings 18th International Conference on Data Engineering (ICDE'02)*, San Jose, CA, 2002.