Information Sciences 367–368 (2016) 1–13

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Time series *k*-means: A new *k*-means type smooth subspace clustering for time series data

Xiaohui Huang[a,b,*], Yunming Ye[b], Liyan Xiong[a], Raymond Y.K. Lau[c], Nan Jiang[a], Shaokai Wang[b]

[a] School of Information Engineering Department, East China Jiaotong University, Nanchang, 330013, China
[b] Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, 518055, China
[c] Department of Information Systems, City University of Hong Kong, Hong Kong, China

ARTICLE INFO

Article history:
Received 9 October 2015
Revised 7 May 2016
Accepted 27 May 2016
Available online 2 June 2016

Keywords:
Time series
k-means clustering
Subspace clustering
Feature selection
Data mining

ABSTRACT

Existing clustering algorithms are weak in extracting smooth subspaces for clustering time series data. In this paper, we propose a new *k*-means type smooth subspace clustering algorithm named Time Series *k*-means (TSkmeans) for clustering time series data. The proposed TSkmeans algorithm can effectively exploit inherent subspace information of a time series data set to enhance clustering performance. More specifically, the smooth subspaces are represented by weighted time stamps which indicate the relative discriminative power of these time stamps for clustering objects. The main contributions of our work include the design of a new objective function to guide the clustering of time series data and the development of novel updating rules for iterative cluster searching with respect to smooth subspaces. Based on a synthetic data set and five real-life data sets, our experimental results confirm that the proposed TSkmeans algorithm outperforms other state-of-the-art time series clustering algorithms in terms of common performance metrics such as Accuracy, Fscore, RandIndex, and Normal Mutual Information.

© 2016 Elsevier Inc. All rights reserved.

1. Introduction

The last decade has witnessed growing interest in clustering techniques for times series data to facilitate advanced applications in various fields, such as bioinformatics [13], environmental monitoring [15], financial applications [32], to name just a few. The ubiquity of time series data has expedited substantial interests in indexing [28,40], classification [20,44], and clustering [34,43] of such data. Times series clustering aims at identifying structure in an unlabeled time series data set by objectively organizing data into homogeneous groups where the similarities among objects in the same group are maximized and the similarities among objects across groups are minimized [29]. The interesting patterns hidden in a time series data set can be identified by studying the clusters uncovered by the clustering algorithms.

A time series is a collection of observations pertaining to a chronological order. The nature of time series data can be characterized by the following features: large in data size, high dimensionality, and the necessity of a continuous updating. Most time series clustering techniques critically depend on the choice of a similarity measure, and these techniques often calculate the similarities among time series objects with respect to the whole sequence. However, it is more reasonable to

* Corresponding author. Tel.: +8613767990625.
E-mail address: hxh016@gmail.com (X. Huang).

http://dx.doi.org/10.1016/j.ins.2016.05.040
0020-0255/© 2016 Elsevier Inc. All rights reserved.

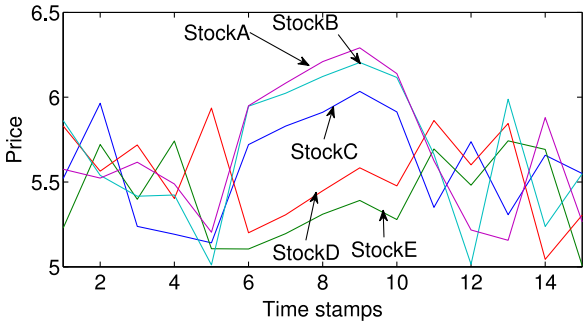

Fig. 1. The time-dependent patterns of stock price movements.

calculate the similarity between two time series objects with reference to a segment of a time interval instead of the whole sequence for high dimensional time series clustering in many applications. For example, the prices of some stocks may rise or slump with a similar pattern in a time span because of an incident related to all of these stocks as shown by the time span between time stamp 6 and time stamp 10 in Fig. 1. However, the prices of these stocks may change with a totally different pattern driven by the financial performance of the corresponding companies in another time span. Therefore, it is important to identify the relevant time spans (or subspaces) when similar patterns of objects are exhibited, and then cluster a data set according to the corresponding subspaces. As a matter of fact, subspace clustering has been widely used to solve the clustering problem of high-dimensional data by many researchers [21,23,26]. The features extracted from a subspace by conventional clustering algorithms do not have a sequential relationship. However, for time series data, a subspace is usually a part of a longer time span e.g., time stamp 6 to time stamp 10 in Fig. 1, and the features extracted from subspaces are interdependent instead of independent as in a conventional data set. In general, the subspaces produced by traditional clustering algorithms cannot be directly applied to cluster time series data.

Here, we introduce a new *k*-means type model for time series data analysis named Time Series *k*-means (TSkmeans) which is able to automatically weight the time stamps according to the importance of a time span in the clustering process. On the basis of the W-kmeans algorithm [21], we introduce a constraint to the weights of time stamps so as to induce a smooth subspace in the clustering process. The main contributions of our research work reported in this paper are threefold. First, we develop a new objective function for the proposed TSkmeans algorithm. Second, we analytically derive the novel updating rules to estimate the memberships, centroids, and weights of time stamps by optimizing the objective function. Meanwhile, the convergence of the proposed TSkmeans algorithm is guaranteed. Third, we conduct extensive experiments to evaluate the effectiveness of the TSkmeans algorithm by using both synthetic and real-life data sets. Our experimental results confirm that the TSkmeans algorithm outperforms several state-of-the-art clustering algorithms over various time series data. The subspaces produced by TSkmeans are able to capture the importance of the corresponding time spans. The ability of TSkmeans to allocate larger weights to certain adjacent time stamps of a time span suggests that the corresponding time span has larger discriminative power for clustering objects.

The remaining sections of this paper are organized as follows. A brief overview of related work about time series clustering is given in Section 2. Section 3 illustrates the computational details of the TSkmeans subspace clustering algorithm. Based on both synthetic and real data, the experiments for evaluating the effectiveness of the TSkmeans algorithm are presented in Section 4. Finally, we offer concluding remarks and highlight future directions of our research work in Section 5.

2. Related work

Tremendous research effort has been devoted to analyze time series data in the past decade [2,27,29,30,42], especially the development of new time series clustering methods [16,17,27]. According to the way how clustering is performed, we can broadly classify time series clustering methods to two approaches. The first approach is to employ a similarity function to put time series objects into different clusters with respect to the whole sequence. The second approach tries to identify different time spans (subspaces) of the whole sequence, and then clustering time series objects with respect to these time spans. In this section, we give a brief survey of time series clustering methods with reference to the two approaches. For a detailed review of time series clustering methods, readers may refer to [42] and [27].

2.1. Whole sequence clustering

For many real-world applications, objects can be represented by a time series such as the prices of a stock, electrocardiogram, medical and biological experimental observations, and many more. Whole sequence clustering aims to discover objects that demonstrate similar patterns with reference to the whole sequence. Clustering algorithms [19,29] based on the Euclidean distance or the Pearson's correlation coefficient are commonly used to tackle the clustering problem for various time series data. However, these algorithms cannot effectively handle the shifting and the scaling problems of patterns.

Dynamic Time Warping (DTW) [35,45] has been proposed to automatically deal with time deformations and different speeds (e.g., speech recognition) associated with time-dependent data. However, clustering time series data by using the DTW distance is a computationally expensive task. Begum et al. proposed an accelerating DTW clustering method with an admissible pruning strategy [4]. To cluster a short time series data set, Möller-Levet et al. proposed a computational method of calculating the distance between two short time series as the sum of the squared differences of their corresponding slopes [33].

Another group of time series clustering algorithms has been developed based on edit distance which mainly includes longest common subsequence (LCSS) model [40,41] and edit sequence on real sequence (EDR) model [10]. Chen et al. proposed a Spatial Assembling Distance (SpADe) method [12] which was able to deal with shifting and scaling in both temporal and amplitude dimensions. The main idea of SpADe is to discover matching time segments named patterns, within an entire time series by shifting and scaling in both the temporal and amplitude dimensions. Bahadori et al. proposed a Functional Subspace Clustering (FSC) algorithm [3] which extended the power and flexibility of subspace clustering to time series data by permitting the deformations that underlie many popular functional similarity measures. To overcome the issues of high dimensionality, contextual constraints, and temporal smoothness, Cai et al. proposed a comprehensive method named FACETS [6] to simultaneously capture all these aspects by using tensor factorization and performing careful popularizations to tackle both contextual and temporal issues. Ferreira and Zhao [17] transformed a set of time series objects into a network by using different distance functions. More specifically, every time series object is represented by a vertex and the most similar vertexes are connected to uncover clusters. A fast clustering method for large-scale time series data named YADING was developed [16]. In particular, time series objects were allocated to clusters that were initially induced based on sampled subsets of the input data. As a whole, these existing algorithms do not take into account the possible subspaces of a time series data set.

2.2. Subsequence clustering

Prior to 2003, subsequence time series clustering was generally accepted as a valid technique for time series analysis [14,18,25]. For example, Fu et al. discovered patterns from stock data by using subsequence time series analysis technique [18]. However, Keogh et al. claimed that subsequence time series clustering was meaningless in 2003 because the centroids produced by subsequence time series clustering became sinusoidal pseudo-pattern for almost all kinds of time series data [30]. Moreover, further work [9,24,38] was published to explain the problem; for example, Idé theoretically explained why the centroids of subsequence time series data produced by k-means clustering method naturally formed sinusoidal patterns due to the Fourier state [24].

Nevertheless, some researchers continued to research new methods that produce meaningful subsequence clustering results [7,8]. Chen claimed that subsequence clustering could indeed be meaningful if distances were correctly measured in a delay space [9]. However, the sliding windows technique adopted for constructing the delay space is generally suboptimal as it causes the delay space that represents the data series dynamics to be aligned closely along the bisectrix of the delay space. Since similarity computation for time series is the bottleneck for clustering large-scale time series data, Rakthanmanon proposed the UCR-DTW method [36] which is capable of searching and mining trillions of time series subsequences under dynamic time warping. Most of the existing subsequence time series clustering approaches can only solve the kinds of problems which are characterized by some predefined parameters and the range of width variability is small. However, such an assumption turns out to be unrealistic for many real-world applications. Therefore, Madicar et al. proposed an enhanced parameter-free subsequence time series clustering algorithm for high-variability-width data [31]. Zolhavarieh et al. offered a solution to perform online pattern recognition for subsequence time series clustering [46]. Agarwal et al. discovered that high quality subsequence clusters could be uncovered from vehicular sensor data by using bounded spherical clustering, and the proposed method characterized by linear time complexity [1]. Due to the encouraging results of this work [1], authors found that the problem of generating meaningless subsequence clusters [30] could be resolved by using bounded spherical clustering.

2.3. Characteristics of the proposed TSkmeans algorithm

Similar to the traditional clustering methods that operate on the whole sequence, our proposed TSkmeans algorithm can iteratively discover the subspaces of the whole sequence, and then clusters objects based on the uncovered subspaces instead of the whole sequence. Following such an idea, we propose a new k-means type clustering framework that can smoothly assign weights to different time stamps for clustering time series data.

Although some weighted k-means type algorithms have been proposed by using different weighting methods [21,23,26,39], these algorithms aim to cluster data whose features do not have a chronological order. To effectively explore the temporal sequence information associated with time series data, the proposed TSkmeans algorithm tries to smooth the weights of adjacent time stamps, and hence the uncovered subspaces become more meaningful for clustering time series data.

3. The *k*-means type model for clustering time series data

In this section, we illustrate a *k*-means type smooth subspace approach for clustering time series data. First, we develop a new objective function for the time series clustering model. Second, the corresponding iterative clustering rules are analytically derived based on the objective function. Third, we describe the procedure of the TSkmeans algorithm according to the corresponding iterative rules. Finally, the computational complexity of the algorithm is analyzed.

3.1. Optimization model

Let $X = \{X_1, X_2, \ldots, X_n\}$ be a set of n time series objects. Each object $X_i = \{x_{i1}, x_{i2}, \ldots, x_{im}\}$ is characterized by m values with respect to m time stamps. The membership matrix U is a $n \times k$ binary matrix, where the element $u_{ip} = 1$ indicates that a time series object i is assigned to cluster p; otherwise, it is not assigned to cluster p. The centroids of k clusters are represented by a set of k vectors $Z = \{Z_1, Z_2, \ldots, Z_k\}$. $W = \{W_1, W_2, \ldots, W_k\}$ is a set of k vectors representing the weights of the time stamps on every cluster. The value of element w_{pj} indicates the weight of the j^{th} time stamp for the p^{th} cluster. The objective function of TSkmeans is formulated as follows:

$$P(U, Z, W) = \sum_{p=1}^{k}\sum_{i=1}^{n}\sum_{j=1}^{m} u_{ip} w_{pj} (x_{ij} - z_{pj})^2 + \frac{1}{2}\alpha \sum_{p=1}^{k}\sum_{j=1}^{m-1} (w_{pj} - w_{pj+1})^2, \tag{1}$$

subject to

$$\begin{cases} \sum_{p=1}^{k} u_{ip} = 1, \ u_{ip} \in \{0, 1\}, \\ \sum_{j=1}^{m} w_{pj} = 1, \ 0 \le w_{pj} \le 1, \end{cases} \tag{2}$$

where α is a parameter which is used to balance the effects between the scatters of objects within clusters and the smoothness of the weights of time stamps. The smoothness of the weights among adjacent time stamps increases with the increment of α's value. The first item of the objective function Eq. (1) aims at minimizing the sum of scatters of all the clusters. The second item of the objective function Eq. (1) is to smooth the weights of the adjacent time stamps. In the clustering process, this objective function simultaneously minimizes the within cluster scatter and smooths the weights of the adjacent time stamps.

3.2. TSkmeans algorithm

In this subsection, we minimize the objective function Eq. (1) under the constraints of Eq. (2) to obtain the updating rules for the membership U, centroid Z, and weight W pertaining to the fundamental constructs of our algorithm. We establish membership U, centroid Z, and weight W by using the common way of optimization e.g., fixing two variables and minimizing the objective function $P(U, W, Z)$ to obtain the values of other variables. Therefore, TSkmeans can solve the optimization problem by the following three steps.

Step 1 Given weight $\widehat{W}$ and centroid $\widehat{Z}$ are fixed, $P(U, \widehat{W}, \widehat{Z})$ is minimized if

$$u_{ip} = \begin{cases} 1, \ \textit{if } D_{pj} \ge D_{p'j}, \ p' \ne p, \ 1 \le p' \le k, \\ 0, \ \textit{otherwise}, \end{cases} \tag{3}$$

where

$$D_{pj} = \sum_{j=1}^{m} w_{pj} (x_{ij} - z_{pj})^2. \tag{4}$$

The proof process of step 1 can be found in references [5,37]. Intuitively, the time series object is allocated to the cluster where the weighting distance between the centroid of the cluster and the object is minimized.

Step 2 Given weight $\widehat{W}$ and centroid $\widehat{U}$ are fixed, $P(\widehat{U}, \widehat{W}, Z)$ is minimized if

$$z_{pj} = \frac{\sum_{i=1}^{n} u_{ip} x_{ij}}{\sum_{i=1}^{n} u_{ip}}. \tag{5}$$

Eq. (5) can be achieved by setting the derivative of $P(\widehat{U}, \widehat{W}, Z)$ with respect to Z to zero.

Proof. We calculate the derivative of $P(\widehat{U}, \widehat{W}, Z)$ with respect to Z as follow:

$$\frac{\partial P(\widehat{U}, \widehat{W}, Z)}{\partial z_{pj}} = 2 \sum_{i=1}^{n} u_{ip} w_{pj} (x_{ij} - z_{pj}). \tag{6}$$

We solve z_{pj} by setting the derivative Eq. (6) to zero. Then, Eq. (5) is able to be obtained. □

Step 3 Given membership matrix $\widehat{U}$ and centroid $\widehat{Z}$ are fixed, $P(\widehat{U}, W, \widehat{Z})$ is a quadratic programming problem. We can solve $P(\widehat{U}, W, \widehat{Z})$ with a quadratic programming tool.

Thus, the overall procedure of TSkmeans can be described as Algorithm 1.

Algorithm 1 TSkmeans

Input: $X = \{X_1, X_2, \ldots, X_n\}$, k, α.
Output: U, Z, W.
Initialize: Randomly choose an initial $Z^0 = Z_1, Z_2, \ldots, Z_k$ and weight $W = \{w_{p,j}\}$.
repeat
  Fixed Z, W, solve the membership matrix U with (3);
  Fixed U, W, solve the centroids Z with (5);
  Fixed U, Z, solve the weight W with quadratic programming;
until Convergence.

3.3. Feature weighting

During the clustering process, the weights of time stamps are used to characterize the subspaces in the proposed algorithm. As the intrinsic smoothness of time series, we try to extract the smooth subspaces for each cluster i.e., the weights of adjacent time stamps must be similar in a cluster. On one hand, the weights can help to identify the important time stamps which have high discriminative power for improving clustering performance. On the other hand, the weights that are produced by the algorithm can help identifying the time span where the objects of a cluster exhibit a similar pattern. So, our approach facilitates the analysis of time series data.

Given a data partition, the principal for feature weighting is to assign larger weights to adjacent time stamps that have smaller within cluster scatters. On the other hand, smaller weights are assigned to the time stamps that have larger within cluster scatters. The smoothness of the weights is controlled by the parameter α. Based on the above principle, we can analyze the impact of choosing different values of the parameter α as follows.

When $\alpha = 0$ is established, the second item of the objective function Eq. (1) is zero. Accordingly, we cannot obtain smooth feature subspaces. During the clustering process, the weight of the time stamp which has minimal within cluster distance will be assigned one and the weights of the other time stamps will be assigned zero. This approach guarantees to obtain the minimal value of the objective function Eq. (1). However, such an assumption is unreasonable for clustering time series data.

When $\alpha < 0$, the second item of the objective function Eq. (1) is negative, which encourages the oscillation of weights among adjacent time stamps. It does not fulfill the requirement of smooth subspace for clustering time series data either.

When $\alpha > 0$, the value of the second item in the objective function Eq. (1) increases with the increment of the α's value, which stimulates the smoothness of weights among adjacent time stamps. And, according to the first item of the objective function Eq. (1), smaller weights of adjacent time stamps will be assigned larger weights. It can fulfill all the requirements of clustering time series data.

3.4. Computational complexity

Similar to traditional k-means and W-kmeans [21] algorithms, TSkmeans is also an iterative algorithm which includes three iterative steps: updating membership matrix U, updating centroids Z, and updating weights W. The computational complexity of updating membership matrix U is $O(knm)$, where k, n, and m represent the number of clusters, the number of time series objects, and the number of time stamps, respectively. Moreover, the computational cost of updating centroids is also $O(knm)$. We must solve a quadratic programming problem when the weights are updated. The computational cost of updating weights is dependent on the time complexity of the quadratic programming tool. Therefore, the overall time cost of the first two steps is $O(tknm)$, where t is the number of iterations of the algorithm. Moreover, TSkmeans is involved in solving t quadratic programming problems for estimating the weights of time stamps in the clustering process.

4. Experimental results

In this section, we first introduce the experimental setup and performance metrics applied to evaluate the proposed algorithm. Then, we report the experimental results of the proposed algorithm based on an evaluation data set that consists of a synthetic data set and five real-life data sets.

4.1. Experimental setup

For our experiments, the benchmark clustering algorithms: Euclidean distance based k-means (Euclid) [29], Pearson coefficient based k-means (Pearson) [29], Short Time Series distance based k-means (STS) [33], Dynamic Time Wrap distance

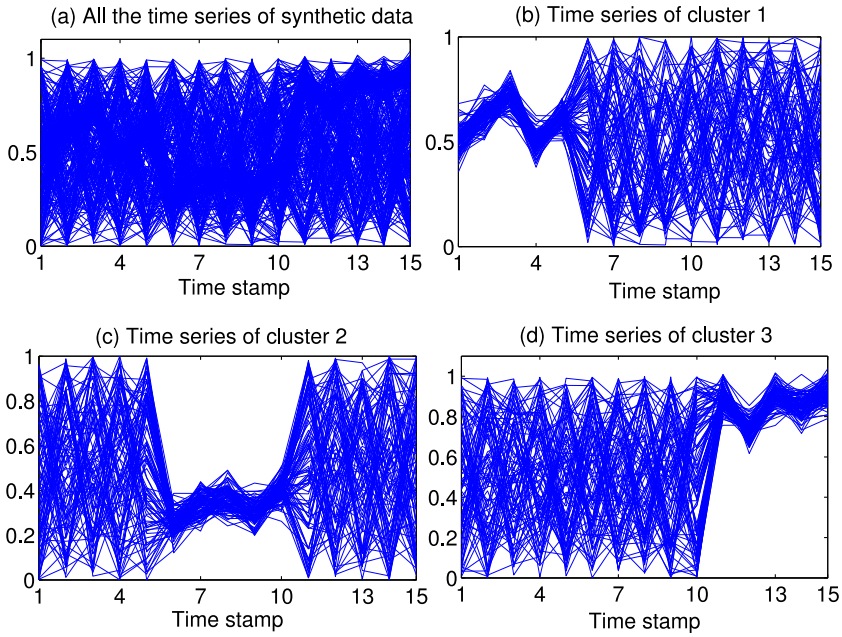

Fig. 2. The time series of the synthetic data set.

based *k*-means (DTW) [35], SpADe [12] as well as Functional Subspace Clustering (FSC) [3] were chosen for a comparative evaluation of the proposed algorithm. We implemented all the algorithms except SpADe and FSC which were provided by their authors.

It is a well-known fact that *k*-means clustering algorithms tend produce local optimal solutions. The final results depend on the locations of the initial cluster centroids. To compare the performance of TSkmeans and that of existing algorithms, the same set of centroids which was randomly generated was applied to initialize the baseline algorithms. Finally, we calculated the average Accuracy, Fscore, RandIndex, and Normal Mutual Information (NMI) achieved by all algorithms after 100 rounds of runs.

In this paper, four performance metrics including Accuracy, Fscore, RandIndex, and Normal Mutual Information (NMI) were applied to evaluate the results of all algorithms. Accuracy is the percentage of the objects that are correctly uncovered in the resulting clusters, and RandIndex measures the percentage of the pairs of objects which are correctly clustered. Fscore is a weighted combination of precision and recall. NMI is a more reliable metric than other metrics for evaluating clustering results of imbalanced data. Readers can refer to the formal definitions of these four metrics in [22].

4.2. Synthetic data set

In this subsection, we describe a synthetic data set was generated to evaluate the performance of our proposed algorithm. In order to verify whether TSkmeans is able to extract smooth subspaces for clustering time series data, we design a continuous subspace for every cluster of the synthetic data set. The synthetic data is shown in Fig. 2 where Fig. 2(a) is the whole data set which includes three clusters: cluster 1 as shown in Fig. 2(b), cluster 2 as shown in Fig. 2(c), and cluster 3 as shown in Fig. 2(d). Each curve of Fig. 2 represents a time series object.

Every cluster of the synthetic data set includes 100 time series objects, and each object is composed of 15 time stamps where the objects in the same cluster exhibit a similar pattern with respect to 5 continuous time stamps. This synthetic data set was generated through two steps: (1) generating the sections of the objects which have similar patterns, i.e., the time stamp 1 to time stamp 5 of cluster 1, time stamp 6 to time stamp 10 of cluster 2 and time stamp 11 to time stamp 15 of cluster 3; (2) randomly generating the other sections of the objects.

4.2.1. Parametric study

Parameter α is an important mechanism that controls the smoothness of the weights of time stamps. In this subsection, we describe an empirical study that evaluates the impact of α on clustering results as shown in Fig. 3. From the observation of the objective function Eq. (1), we can find that the same value of α may yield different levels of smoothness pertaining to the weights of time stamps due to the different scales of different data sets. To minimize the impact of different scales of data sets on the smoothness of the weights, we set α's value according to the changing global scatters of different data

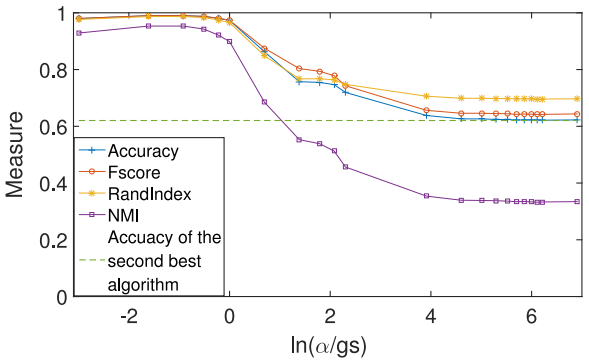

Fig. 3. The clustering performance of TSkmeans with various α for the synthetic data.

Table 1
The results based on synthetic data set.

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	0.6204	0.6408	0.6952	0.3312
Pearson	0.6027	0.6321	0.6801	0.3018
STS	0.4548	0.4714	0.5860	0.0912
DTW	0.5436	0.5368	0.6253	0.2097
SpADe	0.3359	0.5008	0.3337	0.0040
FSC	0.4876	0.4917	0.5955	0.0984
TSkmeans	**0.9733**	**0.9736**	**0.9654**	**0.8985**

sets. The global scatter is estimated as follows:

$$gs = \sum_{i=1}^{n}\sum_{j=1}^{m}(x_{ij} - z_{oj})^2 \tag{7}$$

where $z_{oj} = \left(\sum_{i=1}^{n} x_{ij}\right)/n$. Fig. 3 shows the changing trends of the four performance metrics with the values of $ln(\alpha/gs)$ from -2.9957 (i.e., $\alpha/gs = 0.05$) to 6.9078 (i.e., $\alpha/gs = 10^3$). As a matter of fact, we use the following values of α/gs in our experiments: 0.05, 0.2 to 1 with the step value 0.2, 2 to 10 with the step value 2, 50 to 500 with step value 50, and 10^3.

Fig. 3 shows the changing trends of the average performance achieved by the proposed TSkmeans algorithm after running 100 times based on the synthetic data set. According to the experimental results, we can observe that relatively good performance is achieved when the value of $ln(\alpha/gs)$ is less than 1. For the remaining experiments, we set $ln(\alpha/gs) = 0$ (i.e., $\alpha/gs = 1$). Moreover, Fig. 3 shows that the TSkmeans algorithm is able to achieve consistently better Accuracy than the second best algorithm based on the synthetic data set. The detailed clustering results of the experimental and the baseline algorithms will be given in the following subsection.

4.2.2. Results and analysis

The average Accuracy, Fscore, RandIndex and NMI achieved by all algorithms after 100 runs of these algorithms are summarized in Table 1 for the synthetic data set by using $ln(\alpha/gs) = 0$. The numbers in boldface indicate the best of clustering performance achieved by the respective algorithms. According to the overall experimental results, the clustering performance of TSkmeans is significantly better than the baseline algorithms with respect to all the performance metrics.

According to Table 1, we can observe that the algorithms based on Euclidean distance and Pearson coefficient outperform STS, DTW, SpADe, and FSC based on the synthetic data set. A closer examination of the clustering results of STS, SpADe, and FSC algorithms, we find that most of the time series objects are assigned to one cluster, and the other clusters contain only a few objects as reflected by the lower NMI achieved by the respective algorithms. As the name of the STS (Short Time Series clustering) algorithm suggested, it favors measuring the similarity between two short time series. On the other hand, the SpADe algorithm is developed for handling shifting and scaling in both temporal and amplitude dimensions in the process of clustering time series data. FSC assumes that time series objects lie in deformed linear subspaces and it formulates the subspace learning problem as a sparse regression over neighborhood objects. FSC also calculates the similarities between an object and its neighborhoods with the whole time series. However, the objects of a cluster tend to have large similarities in a section of a sequence, and the other sections are relatively noisy in the synthetic data set. Hence, FSC cannot effectively extract the smooth subspaces hidden in the data, which causes the performance degradation of the clustering process. As a result, the performance of FSC is inferior to that of the TSkmeans algorithm. Furthermore, STS, SpADe, and FSC algorithms are all incapable of extracting smooth subspaces of time series data as shown in Fig. 2.

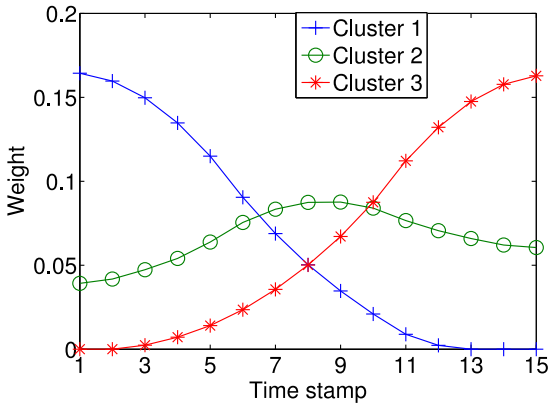

Fig. 4. The weights of different time stamps for every cluster of the synthetic data set.

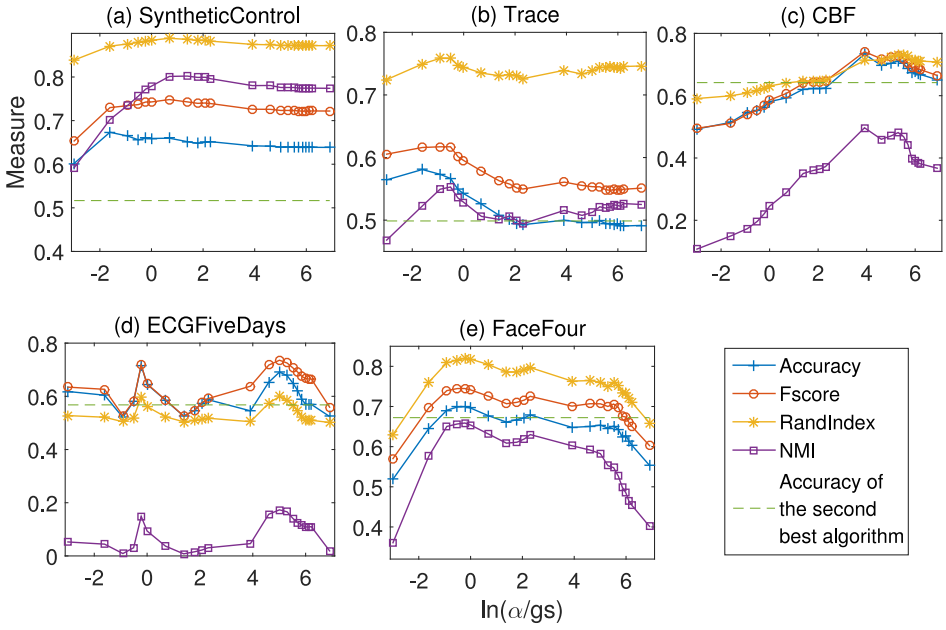

Fig. 5. The clustering performance of TSkmeans with various α for the real-life data set.

In comparison to Euclidean distance based algorithm, TSkmeans is able to achieve over 35% improvement in terms of Accuracy, 33% improvement in terms of Fscore, 17% improvement in terms of RandIndex, and 56% improvement in terms of NMI. These results suggest that TSkmeans can more effectively cluster time series data by exploiting the smooth subspaces. In particular, the TSkmeans algorithm significantly outperforms other baseline algorithms in terms of NMI. The probable reason is that the qualities of some clusters produced by the baseline algorithms are very poor.

4.2.3. Feature weighting

In this subsection, we study the feature weights produced by TSkmeans based on the synthetic data set. Fig. 4 shows the weights of time stamps for every cluster of the synthetic data set. From the original data shown in Fig. 2(b)–(d), we can observe that the smooth subspaces of cluster 1, cluster 2 and cluster 3 are between time stamp 1 and time stamp 5, between time stamp 6 and time stamp 10, between time stamp 11 and time stamp 15, respectively. In Fig. 4, we can find that the corresponding time span in each cluster is assigned larger weights, which suggest that TSkmeans can correctly discover the smooth subspaces. On one hand, the weighting curves in Fig. 4 suggest that the smooth subspace between time stamp 1 and time stamp 5 has high discriminative power for cluster 1, the smooth subspace between time stamp 6 and time stamp 10 has high discriminative power for cluster 2, and the smooth subspace between time stamp 11 and time stamp 15 has high discriminative power for cluster 3. On the other hand, the larger weights between time stamp 1 and time stamp 5 suggest that time series objects with higher similarities with respect to this time span are allocated to cluster 1. Likewise, Fig. 4 also demonstrates that objects with high similarities with respect to the time span between time stamp 6 and time

Table 2
The properties of real-life data sets.

Data Set	No. of time stamps	No. of clusters	No. of objects
SyntheticControl	60	6	600
Trace	275	4	200
CBF	128	3	930
ECGFiveDays	136	2	884
FaceFour	112	4	350

Table 3
The results based on real-life data set "SyntheticControl".

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	0.6378	0.7211	0.8717	0.7734
Pearson	0.6376	0.7023	0.8685	0.7143
STS	0.2393	0.2823	0.6597	0.0458
DTW	0.6376	0.7023	0.8685	0.7143
SpADe	0.4760	0.5832	0.7753	0.5751
FSC	0.4042	0.4793	0.7055	0.3096
TSkmeans	**0.6591**	**0.7428**	**0.8841**	**0.7790**

stamp 10 are put in cluster 2, and objects with high similarities with respect to the time span between time stamp 11 and time stamp 15 are allocated to cluster 3.

4.3. Real-life data sets

To further evaluate the performance of TSkmeans, we applied TSkmeans to five real-life data sets discussed in reference [11]. The properties of these data sets are described in Table 2. For the rest of this subsection, we will first examine the effect of choosing different values of the parameter α. Based on the parametric study, we will further analyze the clustering results of the proposed algorithm based on real-life data sets.

4.3.1. Parametric study

The average Accuracy, Fscore, RandIndex and NMI achieved by TSkmeans after running 100 times with respect to the range of parameter values from $ln(\alpha/gs) = -2.9957$ (i.e., $\alpha/gs = 0.05$) to $ln(\alpha/gs) = 6.9078$ (i.e., $\alpha/gs = 10^3$) against five real-life data sets are shown in Fig. 5. From this figure, we can observe that the clustering performance improves with the increment of α's value, and then the performance declines with further increment of α's value for most of the cases. However, the optimal clustering performance does not coincide with a specified value of α with respect to different data sets. However, we can still observe a general pattern that clustering performance tends to reach an optimum when the value of $ln(\alpha/gs)$ falls in the range between 0 and 5. Accordingly, we applied $ln(\alpha/gs) = 0$ (i.e. $\alpha/gs = 1$) to the rest of this series of experiments. Fig. 5 shows that the average Accuracy achieved by TSkmeans is consistently better than that of the second best algorithm with respect to all the values of α for the "SyntheticControl" data set. Moreover, TSkmeans outperforms the second best algorithm when the value of $ln(\alpha/gs)$ is close to 0 for the "Trace" and the "FaceFour" data sets. TSkmeans also outperforms the second best algorithm when the value of $ln(\alpha/gs)$ is close to 5 for the "CBF" data set. For the "ECGFiveDays" data set, TSkmeans performs better than the second best algorithm when $ln(\alpha/gs) = 0$ and $ln(\alpha/gs = 5)$ are set. These experimental results reveal that TSkmeans is sensitive to the value of α. A probable explanation for these results is that some data sets have obvious smooth subspaces, while it is more difficult to uncover the smooth subspaces for the other data sets.

4.3.2. Results and analysis

The average Accuracy, RandIndex, Fscore and NMI achieved by the experimental and the baseline algorithms after running 100 times for the five real-life data sets are tabulated in Tables 3, 4, 5, 6, 7. In these experiments, we chose the parameter $ln(\alpha/gs) = 0$ according to the parametric study depicted in Section 4.3.1. From these tables, we can observe that TSkmeans achieves the best performance with respect to all performance metrics in most cases.

From these tables, we can also observe that the algorithms based on Euclidean, Pearson coefficient, and DTW distance-based algorithm achieve similar clustering performance with respect to all the performance metrics in most cases. However, the STS distance based algorithm and the SpADe algorithm usually are inferior to other algorithms. The probable reason of the poor performance achieved by the STS distance based algorithm is that it can only effectively calculate the distance between two short time series. Moreover, the SpADe algorithm is effective to conduct shifting and scaling in both temporal and amplitude dimensions instead of discovering the subspaces hidden in a time series data set. When compared to the Euclidean based algorithm, TSkmeans is able to achieve improvement on average Accuracy by 2%, 5%, 3%, and 10% for the

Table 4
The results based on real-life data set "Trace".

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	0.4971	0.5411	**0.7500**	0.5170
Pearson	0.4987	0.5413	**0.7500**	0.5172
STS	0.3311	0.4351	0.4085	0.1699
DTW	0.4876	0.5371	0.7499	0.5172
SpADe	0.2500	0.4000	0.2462	0.0000
FSC	0.4939	**0.6326**	0.7208	**0.5481**
TSkmeans	**0.5423**	0.5953	0.7444	0.5285

Table 5
The results based on real-life data set "CBF".

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	**0.6420**	**0.6523**	**0.7021**	**0.3531**
Pearson	0.6399	0.6504	0.7001	0.3479
STS	0.3592	0.3690	0.5489	0.0031
DTW	0.6280	0.6604	0.6927	0.3532
SpADe	0.4079	0.5372	0.4324	0.0710
FSC	0.4140	0.4534	0.5733	0.0936
TSkmeans[a]	0.5810	0.5859	0.6313	0.2452

[a] Note: The results of TSkmeans in the table are produced by using $ln(\alpha/gs) = 1$. However, TSkmeans achieves the best performance achieved when $ln(\alpha/gs) = 4.6052$ (i.e., $\alpha/gs = 10^2$) is set for this data set according to Fig. 5(c).

Table 6
The results based on real-life data set "ECGFiveDays".

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	0.5166	0.5166	0.5000	0.0008
Pearson	0.5154	0.5154	0.4999	0.0007
STS	0.5244	0.5246	0.5008	0.0019
DTW	0.5678	0.5684	0.5087	0.0135
SpADe	0.5000	**0.6667**	0.4994	0.0000
FSC	0.5012	0.6640	0.4994	0.0096
TSkmeans	**0.6450**	0.6466	**0.5612**	**0.0938**

Table 7
The results based on real-life data set "FaceFour".

Algorithm	Accuracy	Fscore	RandIndex	NMI
Euclid	0.5897	0.6204	0.7400	0.4367
Pearson	0.5878	0.6178	0.7395	0.4448
STS	0.5291	0.5692	0.6619	0.3326
DTW	0.6275	0.6694	0.7650	0.5261
SpADe	0.4787	0.5570	0.6087	0.2479
FSC	0.6722	0.7235	0.7885	0.5474
TSkmeans	**0.6961**	**0.7414**	**0.8181**	**0.6539**

"SyntheticControl", "Trace", "ECGFiveDays", and "FaceFour" data sets, respectively. These experimental results confirm that TSkmeans is able to better cluster time series objects by extracting smooth subspaces from the whole sequence when compared to the traditional time series clustering algorithms. In Table 5, it is worth noting that the performance achieved by TSkmeans is not as promising as those achieved by the Euclid, the STS, and the DTW algorithms for the "CBF" data set. The reason is that the clustering results of TSkmeans are generated when the parameter $ln(\alpha/gs) = 1$ is set. However, TSkmeans should achieve the best performance when $ln(\alpha/gs) = 4.6052$ (i.e., $\alpha/gs = 10^2$) is set for this data as shown in Fig. 5(c).

The clustering results and the weights of time stamps produced by TSkmeans for the data set "SyntheticControl" are plotted in Fig. 6 and Fig. 7, respectively. According to these figures, we can observe that TSkmeans can effectively identify the smooth subspaces for different clusters. For example, the smooth subspace of cluster 6 falls between time stamp 1 and time stamp 20 as shown in Fig. 6(f). The subspace is correctly identified by TSkmeans as the weights between time stamp 1 and time stamp 20 of cluster 6 are considerably larger than the weights assigned to other time stamps as shown in Fig. 7. On one hand, the identified subspace suggests that the time span between time stamp 1 and time stamp 20 has more discriminative power than the other time spans for grouping objects in cluster 6. On the other hand, the subspace

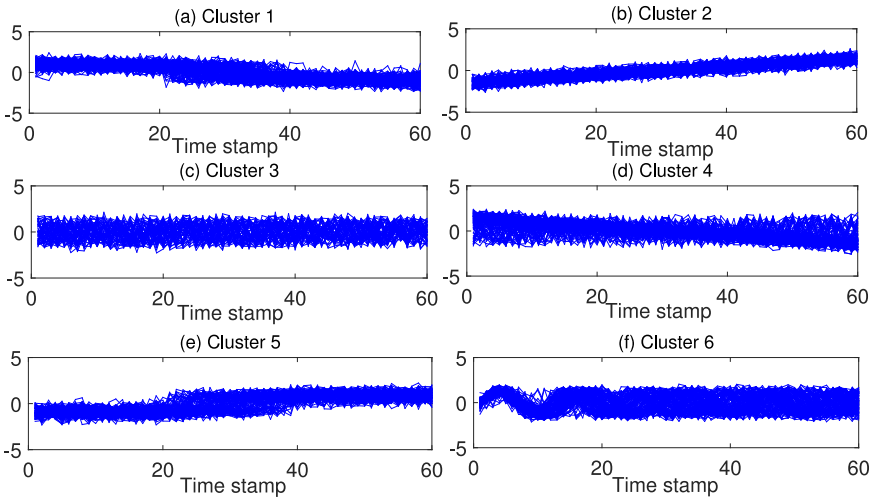

Fig. 6. The clustering results based on data set "SyntheticControl".

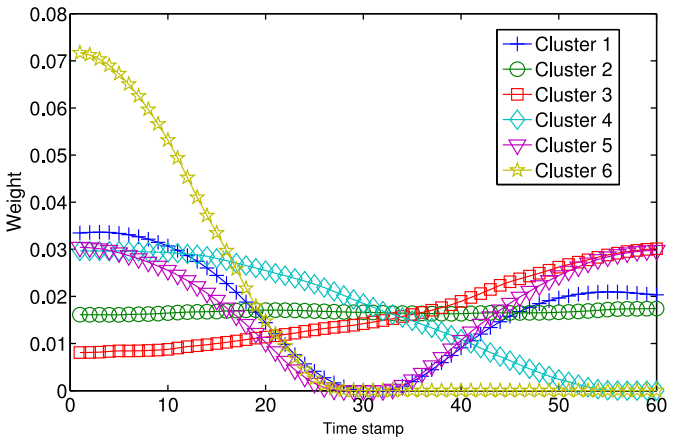

Fig. 7. The weights of time stamps for every cluster of the data set "SyntheticControl".

also demonstrates that the time series objects of cluster 6 have higher similarities with respect to the time span between time stamp 1 and time stamp 20 in the "SyntheticControl" data set.

Similar results can be observed for cluster 1 (Fig. 6(a)) and cluster 5 (Fig. 6(e)). The within cluster distances of the beginning and the ending section of these two clusters are relatively smaller. Fig. 7 shows that TSkmeans can produce two smooth subspaces for both clusters. Different from cluster 1, cluster 5, and cluster 6, all the time stamps of cluster 2 (Fig. 6(b)), cluster 3 (Fig. 6(c)) and cluster 4 (Fig. 6(d)) have similar within cluster distances. Therefore, TSkmeans assigns similar weights to these time stamps, and considers all of these time stamps as a smooth subspace.

4.4. Discussions

According to the experimental results reported in Section 4.2 and Section 4.3, we can see that TSkmeans outperforms the Euclidean, the Pearson coefficient, the STS, the DTW, the SpADe, and the FSC algorithms in terms of Accuracy, RandIndex, Fscore and NMI in most cases. These results confirm that TSkmeans is effective and it can improve the clustering performance for time series data by extracting smooth subspaces.

In comparison to existing clustering methods which cluster the time series objects with equally weighted time stamps, TSkmeans attempts to find smooth subspaces for each cluster, and then clusters the time series objects with respect to these subspaces. The smooth subspaces are represented by the weights of time stamps. The time stamps with larger weights have more discriminative power than the other time stamps in a subspace. Moreover, the extracted subspaces also indicate the time spans when the time series objects demonstrate similar trends. By clustering with respect to the subspaces, TSkmeans is able to achieve better clustering results for most time series data sets. Among the baseline algorithms, the clustering performance of STS-based algorithm is the poorest. The probable reason is that STS performs better for a data set with short time series objects as implied by the name of this method. The clustering performance achieved by the SpADe and

the FSC algorithms is inferior to that of the proposed TSkmeans algorithm. The probable reason is that the SpADe and the FSC algorithms aim to solve the problems of shifting and amplification of a time series data set instead of discovering the subspaces hidden in the data set.

According to the experimental results, the parameter α that is applied to balance the within cluster distance and the smoothness of weights in a subspace cannot be automatically derived in a straightforward manner. From our experimental results, we find that the feasible range of the values of the parameter α is relatively large, and the value of α is related to the scale of a data set to some extent. Accordingly, we plan to further extend the proposed algorithm by designing a new objective function such that the parameter α can be established automatically in our future work.

5. Conclusions

We have developed a new k-means type algorithm for clustering time series data by extracting smooth subspaces hidden in the data set. The main contributions of our research work are threefold. First, a new objective function is developed for clustering time series data. Second, the iterative updating rules for cluster searching are analytically derived by optimizing the objective function. Third, extensive experiments are conducted to evaluate the effectiveness of the proposed TSkmeans clustering algorithm based on synthetic and real-life data sets. Our experimental results confirm that the proposed algorithm is more effective than other state-of-the-art clustering algorithms.

Acknowledgement

The authors are very grateful to the editor and anonymous referees for their helpful comments. Huang's work was supported was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61562027, Social science planning project of Jiangxi Province under Grant No. 15XW12 and Education Department of Jiangxi Province under Grant No.GJJ150494. Ye's work was supported by NSFC under Grant No. 61572158, Shenzhen Science and Technology Program under Grant No. JCYJ20140417172417128 and JSGG20141017150830428. Xiong's work was supported was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61363072. Lau's work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project: CityU 11502115) and the Shenzhen Science and Technology Program under Grant No. JCYJ20140419115614350. Jiang's work was supported was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61462028 and No. 41402290.

References

[1] P. Agarwal, G. Shroff, S. Saikia, Z. Khan, Efficiently discovering frequent motifs in large-scale sensor data, in: Proceedings of the Second ACM IKDD Conference on Data Sciences, in: CoDS '15, ACM, New York, NY, USA, 2015, pp. 98–103.
[2] S. Aghabozorgi, A. Seyed Shirkhorshidi, T. Ying Wah, Time-series clustering - a decade review, Inform. Syst. 53 (MAY) (2015) 16–38.
[3] M.T. Bahadori, D. Kale, Y. Fan, Y. Liu, Functional subspace clustering with application to time series, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 228–237.
[4] N. Begum, L. Ulanova, J. Wang, E. Keogh, Accelerating dynamic time warping clustering with a novel admissible pruning strategy, in: Proceedings of the 21th ACM SIGKDD international conference on Knowledge discovery and data mining, 2015.
[5] J. Bezdek, A convergence theorem for the fuzzy isodata clustering algorithms, IEEE Trans. Pattern Anal. Mach. Intell. (1) (1980) 1–8.
[6] Y. Cai, H. Tong, W. Fan, P. Ji, Q. He, FACETS: Fast comprehensive mining of coevolving high-order time series, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 79–88.
[7] J. Chen, Subsequence time series clustering, in: The Encyclopedia of Data Warehousing and Mining (2nd Edition), IGI Global, 2009, pp. 1871–1876.
[8] J.R. Chen, Making subsequence time series clustering meaningful, in: Proceedings of the 5th IEEE International Conference on Data Mining, IEEE, 2005, pp. 1–8.
[9] J.R. Chen, Useful clustering outcomes from meaningful time series clustering, in: Proceedings of the 6th Australasian conference on Data mining and analytics, Gold Coast, Australia, 70, 2007.
[10] L. Chen, M.T. Ozsu, V. Oria, Robust and fast similarity search for moving object trajectories, Proc. 21th ACM SIGMOD Int. Conf. Manage. Data (2005) 491–502.
[11] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, The UCR time series classification archive, 2015. www.cs.ucr.edu/~amonn/time_series_data/.
[12] Y. Chen, M. Nascimento, B.C. Ooi, A.K. Tung, et al., SpADe: On shape-based pattern detection in streaming time series, in: Proceedings of the 23rd IEEE International Conference on Data Engineering, IEEE, 2007, pp. 786–795.
[13] T.-Y. Chiu, T.-C. Hsu, C.-C. Yen, J.-S. Wang, Interpolation based consensus clustering for gene expression time series, BMC Bioinform. 16 (117) (2015) 1–17.
[14] P. Cotofrei, K. Stoffel, Classification rules + time = temporal rules, Lecture Notes Comput. Sci. 23 (2002) 572–581.
[15] B. DeVries, J. Verbesselt, L. Kooistra, M. Herold, Robust monitoring of small-scale forest disturbances in a tropical montane forest using landsat time series, Remote Sensing Environ. 161 (2015) 107–121.
[16] R. Ding, Q. Wang, Y. Dang, Q. Fu, H. Zhang, D. Zhang, Yading: Fast clustering of large-scale time series data, Proc. VLDB Endowment 8 (5) (2015) 473–484.
[17] L.N. Ferreira, L. Zhao, Time series clustering via community detection in networks, Information Sciences 326 (2016) 227–242.
[18] T.C. Fu, F.L. Chung, V. Ng, R. Luk, Pattern discovery from stock time series using selforganizing maps, Workshop Notes of the Workshop on Temporal Data Mining at ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2001) 1–8.
[19] X. Golay, S. Kollias, G. Stoll, D. Meier, A. Valavanis, P. Boesiger, A new correlation-based fuzzy logic clustering algorithm for FMRI, Magnetic Resonance Med. 40 (2) (1998) 249–260.
[20] B. Hu, Y. Chen, E.J. Keogh, Time series classification under more realistic assumptions., in: Proceedings of the 2013 SIAM International Conference on Data Mining, 2013, pp. 578–586.
[21] J. Huang, M. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (5) (2005) 657–668.
[22] X. Huang, Y. Ye, H. Guo, Y. Cai, H. Zhang, Y. Li, DSKmeans: a new kmeans-type approach to discriminative subspace clustering, Knowl.-Based Syst. 70 (2014) 293–300.

[23] X. Huang, Y. Ye, H. Zhang, Extensions of kmeans-type algorithms: a new clustering framework by integrating intracluster compactness and intercluster separation, IEEE Trans. Neural Netw. Learn. Syst. 25 (8) (2014) 1433–1446.
[24] T. Idé, Why does subsequence time-series clustering produce sine waves, in: Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases, 2006, pp. 211–222.
[25] X. Jin, Y. Lu, C. Shi, Distribution discovery: Local analysis of temporal rules, in: Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, 2002, pp. 469–480.
[26] L. Jing, M.K. Ng, J.Z. Huang, An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data, IEEE Trans. Knowl. Data Eng. 19 (8) (2007) 1026–1041.
[27] A. Kattan, S. Fatima, M. Arif, Time-series event-based prediction: an unsupervised learning framework based on genetic programming, Inform. Sci. 301 (2015) 99–123.
[28] E. Keogh, A decade of progress in indexing and mining large time series databases, in: Proceedings of the 32nd international conference on Very large data bases, VLDB Endowment, 2006, p. 1268.
[29] T.W. Liao, Clustering of time series data!“a survey, Pattern Recog. 38 (11) (2005) 1857–1874.
[30] J. Lin, E. Keogh, W. Truppel, Clustering of streaming time series is meaningless, in: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, ACM, 2003, pp. 56–65.
[31] N. Madicar, H. Sivaraks, S. Rodpongpun, C.A. Ratanamahatana, An enhanced parameter-free subsequence time series clustering for high-variability-width data, in: Recent Advances on Soft Computing and Data Mining, Springer, 2014, pp. 419–429.
[32] G. Marti, S. Andler, F. Nielsen, P. Donnat, Clustering financial time series: How long is enough? arXiv preprint arXiv:1603.04017 (2016) 1–7.
[33] C.S. Möller-Levet, F. Klawonn, K.H. Cho, O. Wolkenhauer, Fuzzy clustering of short time-series and unevenly distributed sampling points, in: Proceedings of the 5th International Symposium on Intelligent Data Analysis, Berlin, Germany, 2003, pp. 330–340.
[34] F. Petitjean, A. Ketterlin, P. Gançarski, A global averaging method for dynamic time warping, with applications to clustering, Pattern Recog. 44 (3) (2011) 678–693.
[35] L. Rabiner, B.-H. Juang, Fundamentals of speech recognition, Prentice hall, 1993.
[36] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh, Addressing big data time series: mining trillions of time series subsequences under dynamic time warping, ACM Trans. Knowl. Discovery Data 7 (3) (2013) 10.
[37] S. Selim, M. Ismail, K-means-type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Trans. Pattern Anal. Mach. Intell. (1) (1984) 81–87.
[38] G. Simon, J.A. Lee, M. Verleysen, Unfolding preprocessing for meaningful time series clustering, Neural Networks 19 (6–7) (2006) 877–C888.
[39] S. Soheily-Khah, A. Douzal-Chouakria, E. Gaussier, Generalized k-means-based clustering for temporal data under weighted and kernel time warp, Pattern Recog. Lett. 75 (2016) 63–69.
[40] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, E. Keogh, Indexing multidimensional time-series, VLDB J. Int. J. Very Large Data Bases 15 (1) (2006) 1–20.
[41] M. Vlachos, G. Kollios, D. Gunopulos, Discovering similar multidimensional trajectories, in: Proceedings of the 18th International Conference on Data Engineering, 2002, pp. 673–684.
[42] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, E. Keogh, Experimental comparison of representation methods and distance measures for time series data, Data Mining Knowl. Discovery 26 (2) (2013) 275–309.
[43] J. Yang, J. Leskovec, Patterns of temporal variation in online media, in: Proceedings of the fourth ACM international conference on Web search and data mining, ACM, 2011, pp. 177–186.
[44] L. Ye, E. Keogh, Time series shapelets: a new primitive for data mining, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2009, pp. 947–956.
[45] Z. Zhang, P. Tang, R. Duan, Dynamic time warping under pointwise shape context, Inform. Sci. 315 (2015) 88–101.
[46] S. Zolhavarieh, S. Aghabozorgi, T.Y. Wah, Online pattern recognition in subsequence time series clustering, in: Proceedings of the 3rd International Conference on Computer Engineering and Mathematical Sciences, 2014, pp. 177–182.