Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 40 (2007) 2550–2562
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

A clustering procedure for exploratory mining of vector time series

T. Warren Liao*

Industrial and Manufacturing Systems Engineering Department, Louisiana State University, Baton Rouge, LA 70803, USA

Received 13 February 2006; received in revised form 24 October 2006; accepted 1 January 2007

Abstract

A two-step procedure is developed for the exploratory mining of real-valued vector (multivariate) time series using partition-based clustering methods. The proposed procedure was tested with model-generated data, multiple sensor-based process data, as well as simulation data. The test results indicate that the proposed procedure is quite effective in producing better clustering results than a hidden Markov model (HMM)-based clustering method if there is a priori knowledge about the number of clusters in the data. Two existing validity indices were tested and found ineffective in determining the actual number of clusters. Determining the appropriate number of clusters in the case that there is no a priori knowledge is a known unresolved research issue not only for our proposed procedure but also for the HMM-based clustering method and further development is necessary.

Keywords: Vector time series; Clustering; Clustering algorithms; Validity index

1. Introduction

Time series data is pervasive in science, engineering, business, medicine, and other areas. Mining of time series data for useful information is thus important to many disciplines. Due to its high dimension, analysis and modeling of time series data is always time consuming, especially for multivariate time series data. Multivariate time series data are also called vector time series data due to the common use of vector notation. In this paper, multivariate and vector time series are used interchangeably. To reduce the computational burden, it is desirable to preprocess the data before performing detailed data analysis. Data preprocessing could involve several different tasks.

In the case that the number of variables is high, it is desirable to reduce them by removing the redundant or irrelevant ones. To this end, Swift et al. [1] experimented with three methods (the random bag, genetic algorithm, and evolutionary programming) in finding the approximate correlation structure among variables and compared these with an exhaustive search method for verification. Later, Tucker et al. [2] followed up with a more comprehensive study to outline a framework for decomposing high-dimensional vector time series into low-dimensional vector time series so that the reduced number of variables are relatively independent of each other based on their correlation.

It is also advisable to group a given large set of time series so that similar time series are put in the same group and dissimilar time series in different groups. By doing so, it allows us to analyze and model only a selected number of representative time series in each group. This line of research is called clustering of time series data, which is pivotal in the exploratory phase of a data mining process. The time series data can be univariate or multivariate. The data value can be continuous (or real) or discrete. Various clustering procedures have been developed to deal with different time series data from different application domains. For more details, readers are referred to the recent survey on this topic by Liao [3]. Microarray gene expression data are often treated as univariate time series of continuous values. The clustering procedures developed for gene expression data are thus only applicable to univariate continuous data. Jiang et al. [4] classified cluster analysis studies of gene expression data into three categories: gene-based clustering, sample-based clustering, and subspace (or two-way) clustering.

This study focuses on the clustering of real-valued vector time series data. Previous studies on this subject are first reviewed below. Košmelj and Batagelj [5] modified the *relocation*

* Tel.: +1 225 578 5365; fax: +1 225 578 5109.
E-mail address: ieliao@lsu.edu.

0031-3203/$30.00
doi:10.1016/j.patcog.2007.01.005

clustering procedure that was originally developed for static features for the clustering of multivariate time varying data. To measure the dissimilarity between trajectories, they first introduced a general model that incorporated the time dimension based on the cross-sectional approach, and then developed a specific model based on the compound interest idea to determine the time-dependent linear weights. Their cross-sectional procedure ignores the correlations between the variables over time and is applicable only to vector time series with equal length. To form a pre-determined number of clusters, the best clustering is selected as the one with the minimum *generalized ward criterion function* among all the possible clusterings. Also taking the cross-sectional approach but without collapsing the time dimension, Liao et al. [6] applied several clustering algorithms including k-means, fuzzy c-means, and genetic clustering to real-valued vector time series data obtained in battle simulation runs, with the objective to visualize how the battle state changes with time based on a pre-specified numbers of battle states.

To group multivariate series of earthquakes and mining explosions, Kakizawa et al. [7] applied *hierarchical clustering* as well as *k-means clustering*. They measured the disparity between the spectral matrices, corresponding to the $p \times p$ matrices of auto-covariance functions, of two zero-mean vector stationary time series with two quasi-distances: the *J divergence* and the *symmetric Chernoff information divergence*. Owsley et al. [8] developed a method of automatic clustering of vector sequences by generalizing a common vector-clustering method, i.e., the generalized Lloyd algorithm (GLA). It uses hidden Markov models (HMMs) instead of codebooks to define clusters and attempts to find a set of models, which best describes the data distribution as a whole. They discussed the influence of initial conditions and illustrated its use in tool health monitoring based on the vibration signal patterns. Bicego et al. [9] proposed a scheme for HMM-based sequential data clustering, in which each object is described by the vector of its similarities with respect to a predetermined set of other objects with each modeled by a HMM. Each similarity is defined as the sequence-to-HMM log likelihood divided by the length of the sequence.

Ramoni et al. [10] introduced a Bayesian method for clustering univariate dynamic processes. The Bayesian method models dynamics as Markov chains and then applies an agglomerative clustering procedure to discover the most probable set of clusters capturing different dynamics. They also [11] presented a Bayesian clustering algorithm for discrete-valued vector time series, called MBCD. The first step of the MBCD algorithm replaces the original time series by Markov chains represented by transition probability matrices, with the assumption that process dynamics can be modeled as first-order Markov chains. The MBCD algorithm is agglomerative hierarchical. It starts by assigning each set of transition matrices to a separate cluster and iteratively merges them until a certain stopping criterion is met. Both the decision of merging sets and the stopping criterion are based on the posterior probability of the obtained clustering. The optimal partition is the one with maximum posterior probability.

This paper presents a new procedure for clustering real-valued vector time series based on two previous studies, i.e., Liao et al. [6] and Ramoni et al. [11]. Our procedure differs from that of Ramoni et al. [10] in the targeted data type: continuous rather than discrete. Our procedure works directly with raw data and thus does not need to extract features, as required by the procedure of Kakizawa et al. [7]. Our procedure is capable of handling time series data of unequal length, which was not considered by Košmelj and Batagelj [5]. The only comparable procedures to ours are those of Owsley et al. [8] and Bicego et al. [9]. It should be noted that our approaches differ from those procedures developed for clustering gene expressions because gene expressions normally are univariate with equal length.

More details of the proposed procedure are given in the next section. It is followed by the presentation of an illustrative example. Additional test results are presented in Section 4. The results of applying a HMM-based clustering method to the same test data sets are presented in Section 5, followed by the discussion section. The last section concludes the paper and identifies topics for future study.

2. A two-step procedure for clustering real-valued vector time series

The proposed method for clustering real-valued vector time series is a two-step procedure with each step implemented by an existing clustering algorithm. As mentioned before, it can handle multivariate time series of continuous values with unequal length. Given a set of N numbers of p-variate real-valued time series with unequal length, denoted as $\mathbf{x}^i(\mathbf{1}), \ldots, \mathbf{x}^i(\mathbf{t}-\mathbf{1}), \mathbf{x}^i(\mathbf{t}), \mathbf{x}^i(\mathbf{t}+\mathbf{1}), \ldots, \mathbf{x}^i(\mathbf{n}_i)$, where $i = 1, \ldots, N, n_i$ is the length of the ith multivariate time series, and $\mathbf{x}(\mathbf{t}) = \{x_1(t), \ldots, x_p(t)\}$, the proposed procedure organize them into groups (or clusters) according to their similarity or dissimilarity. Ideally, similar multivariate time series should be put in the same group and dissimilar ones should be in different groups.

The first step of the proposed procedure converts a real-valued vector time series into a discrete-valued univariate time series. As the result, $\mathbf{x}^i(\mathbf{1}), \ldots, \mathbf{x}^i(\mathbf{t}-\mathbf{1}), \mathbf{x}^i(\mathbf{t}), \mathbf{x}^i(\mathbf{t}+\mathbf{1}), \ldots, \mathbf{x}^i(\mathbf{n}_i)$, become $y^i(1), \ldots, y^i(t-1), y^i(t), y^i(t+1), \ldots, y^i(n_i)$, with each y takes on a discrete value. This converted variable y is interpreted as the process state variable and there are a discrete numbers of possible process states with each state corresponding to a cluster. The first step works like vector quantization and can be viewed as a dimension reduction step. It is achieved by applying a clustering algorithm to time-striped multivariate data. By stripping the time information, a multivariate time series data $\mathbf{x}^i(\mathbf{1}), \ldots, \mathbf{x}^i(\mathbf{t}-\mathbf{1}), \mathbf{x}^i(\mathbf{t}), \mathbf{x}^i(t+1), \ldots, \mathbf{x}^i(\mathbf{n}_i)$ is converted into n_i numbers of multivariate feature data. To avoid the potential problem of inconsistent cluster labeling that is very likely to occur if processing each time series separately, all multivariate feature data converted from the corresponding multivariate time series data are concatenated into one single file. A set of N numbers of p-variate time series thus

generates $\sum_{i=1}^{N} n_i$ numbers of p-variate feature data after the time-stripping and vectorizing operation. The first step can be implemented with any existing clustering algorithm. Since there is no knowledge about the number of clusters in the data, it is recommended to attempt a range of different values (2–9 is used in this study) in order to determine its effect.

The second step of the proposed procedure clusters the N numbers of converted univariate discrete-valued time series into pre-determined numbers of clusters. Two different approaches can be taken to deal with time series of unequal length in this step. The first approach uses the dynamic time warping (DTW) distance that is known capable of computing distance between two sequences of unequal length. The DTW distance, however, cannot be directly used in partition-based clustering algorithms such as k-means due to the difficulty involved in determining the length of the cluster centers. Therefore, hierarchical or medoid-based clustering algorithms are needed if this approach is taken. The second approach first expresses each discrete-valued univariate time series as a transition probability matrix, following the work of Romani et al. [11], and then employs the symmetric version of Kullback–Liebler distance to measure the distance between two transition probability matrices. Consider a univariate time series $y(1), \ldots, y(t-1), y(t), y(t+1), \ldots, y(n_i)$, the corresponding transition probability matrix has a size of $s \times s$ with each entry taking a value of $n(S_i \rightarrow S_j)/(n_i - 1)$, where $n(S_i \rightarrow S_j)$ denotes the number of times that two adjacent discrete values in the series changing from state i, S_i, to state j, S_j. In the above, s is the total number of states corresponding to the number of clusters specified in the first step. For instance, the series 01201201201 is converted into $\begin{bmatrix} 0 & 0.4 & 0 \\ 0 & 0 & 0.3 \\ 0.3 & 0 & 0 \end{bmatrix}$. The second approach allows the use of virtually all existing clustering algorithms in the second step, including k-means, fuzzy c-means, genetic clustering, hierarchical clustering, etc. In the case that there is no knowledge about the number of clusters, then the same issue of identifying the number of clusters exists as in the first step.

In summary, there are many alternatives to implement the proposed two-step procedure. To limit the scope, this study will employ a partition-based algorithm in both steps; specifically fuzzy c-means (FCM). In the first step, the Euclidean distance is used in the partition-based clustering algorithms to measure the dissimilarity between two multivariate feature data. The resultant univariate time series take on discrete values with each value corresponds to a particular cluster. The discrete value is chosen to be the one having the highest membership value. In the second step, the symmetric version of Kullback–Liebler distance is used as mentioned above.

As mentioned above, in both steps there is an issue of determining the appropriate number of clusters using some measure often called validity index. This issue is a challenging research topic itself. Numerous validity indices have been proposed in the past and continuous efforts are being made. Two selected validity indices that have been reported to produce good performance are tested in the context of our study.

More details about the key components of the proposed procedure are described below.

2.1. Fuzzy c-means clustering algorithm

Given n patterns $\{\mathbf{x}_k | k = 1, \ldots, n\}$, c-means [12] determines c cluster centers $\{\mathbf{v}_i | i = 1, \ldots, c\}$, by minimizing the objective function given as

$$\text{Min} \quad J_1(\mathbf{U}, \mathbf{V}) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik} \|\mathbf{x}_k - \mathbf{v}_i\|^2$$

$$\text{s.t.} \quad (1)\ u_{ik} \in \{0, 1\}\ \ \forall i, k, \quad (2) \sum_{i=1,c} u_{ik} = 1\ \ \forall k. \tag{1}$$

$\|\cdot\|$ denotes a distance measure. Dunn [13] first extended the c-means algorithm to allow for fuzzy partition, rather than hard partition, by using the objective function given in

$$\text{Min} \quad J_2(\mathbf{U}, \mathbf{V}) = \sum_{i=1}^{c} \sum_{k=1}^{n} (\mu_{ik})^2 \|\mathbf{x}_k - \mathbf{v}_i\|^2. \tag{2}$$

Note that $\mathbf{U} = [\mu_{ik}]$ in this and in the following equations denotes the matrix of a fuzzy c-partition. The fuzzy c-partition constraints are: (1) $\mu_{ik} \in [0, 1]\ \forall i, k$, (2) $\sum_{i=1,c} \mu_{ik} = 1\ \forall k$, and (3) $0 < \sum_{k=1,n} \mu_{ik} < n\ \forall i$. In other words, each $\mathbf{x}_k$ could belong to more than one cluster with a fractional degree between 0 and 1. Bezdek [14] generalized $J_2(\mathbf{U}, \mathbf{V})$ to an infinite number of objective functions, i.e., $J_m(\mathbf{U}, \mathbf{V})$, where $1 \leqslant m \leqslant \infty$. The new objective function subject to the same fuzzy c-partition constraints is

$$\text{Min} \quad J_m(\mathbf{U}, \mathbf{V}) = \sum_{i=1}^{c} \sum_{k=1}^{n} (\mu_{ik})^m \|\mathbf{x}_k - \mathbf{v}_i\|^2. \tag{3}$$

The distance measure used in this study is Euclidean in the first step and the symmetric version of Kullback–Liebler distance in the second step. By differentiating the objective function with respect to $\mathbf{v}_i$ (for fixed $\mathbf{U}$) and to μ_{ik} (for fixed $\mathbf{V}$) subject to the conditions, one obtains the following two equations:

$$\mathbf{v}_i = \frac{\sum_{k=1}^{n} (\mu_{ik})^m \mathbf{x}_k}{\sum_{k=1}^{n} (\mu_{ik})^m}, \quad i = 1, \ldots, c, \tag{4}$$

$$\mu_{ik} = \frac{(1/\|\mathbf{x}_k - \mathbf{v}_i\|^2)^{1/(m-1)}}{\sum_{j=1}^{c} (1/\|\mathbf{x}_k - \mathbf{v}_j\|^2)^{1/(m-1)}}, \quad i = 1, \ldots, c;\ \ k = 1, \ldots, n. \tag{5}$$

To solve the fuzzy c-means model, an iterative alternative optimization procedure is required. To run the procedure the number of clusters, c, and the weighting coefficient, m, must be specified. The FCM algorithm has the following steps:

(1) Choose c $(2 \leqslant c \leqslant n)$, m $(1 < m < \infty)$, and ε (a small number for stopping the iterative procedure). Set the counter $l = 0$ and initialize the membership matrix, $\mathbf{U}^{(l)}$.
(2) Calculate the cluster center, $\mathbf{v}_i^{(l)}$ by using Eq. (4).
(3) Update the membership matrix $\mathbf{U}^{(l+1)}$ by using Eq. (5) if $\mathbf{x}_k \neq \mathbf{v}_i^{(l)}$. Otherwise, set $\mu_{jk} = 1(0)$ if $j = (\neq) i$.
(4) Compute $\Delta = \|\mathbf{U}^{(l+1)} - \mathbf{U}^{(l)}\|$. If $\Delta > \varepsilon$, increment l and go to Step 2. If $\Delta \leqslant \varepsilon$, stop.

Note that the fuzzy c-means algorithm requires the number of clusters be pre-specified.

2.2. Kullback–Liebler distance

Let P_1 and P_2 be matrices of transition probabilities of two Markov chains (MCs) with s probability distributions each and p_{1ij} and p_{2ij} be the $i \to j$ transition probability in P_1 and P_2. The asymmetric Kullback–Liebler distance of two probability distributions is

$$d(p_{1i}, p_{2i}) = \sum_{j=1}^{s} p_{1ij} \log(p_{1ij}/p_{2ij}). \tag{6}$$

The symmetric version of Kullback–Liebler distance of two probability distributions is

$$D(p_{1i}, p_{2i}) = [d(p_{1i}, p_{2i}) + d(p_{2i}, p_{1i})]/2. \tag{7}$$

The average distance between P_1 and P_2, $D(P_1, P_2)$, is computed as $\sum_{i=1,s} D(p_{1i}, p_{2i})/s$.

2.3. Validity indices

Recently, Pakhira et al. [15] proposed a crisp and a fuzzy version of the PBM-index and showed that it is superior to three other well-known measures: the Davies–Bouldin index, Dunn's index, and the Xie–Beni index. Two of the above-mentioned five indices that can be used on fuzzy partition results are evaluated for their effectiveness in the context of this study. Let $\mathbf{x}_k,\ k=1,\ldots,n$ be the data vectors of the data set X with each vector having a size of P and $\mathbf{v}_i,\ i=1,\ldots,c$ be the centroid vector of cluster i, C_i. Let the membership value μ_{ik} indicate the degree of $\mathbf{x}_k$ belonging to $\mathbf{v}_i$ as a result of a fuzzy clustering operation using a fuzzy weight of m. V_{XB} stands for the XB index [16], which is defined as

$$V_{XB}(c) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n}[\mu_{ik}^{m} d_E(\mathbf{x}_k, \mathbf{v}_i)^2]}{n[\min_{i,j,i\neq j} d_E(\mathbf{v}_i, \mathbf{v}_j)^2]}. \tag{8}$$

In the above equation, $d_E(\bullet)$ denotes the Euclidean distance. The fuzzy version of the PBM index [14], $PBMf$, is defined as

$$PBMf(c) = \left(\frac{1}{c} \times \frac{E_1}{J_m(c)} \times D_{\max}(c)\right)^2,$$

$$D_{\max}(c) = \max_{i,j=1}^{c} d_E(\mathbf{v}_i, \mathbf{v}_j), \quad E_1 = \sum_{k=1}^{n} d_E(\mathbf{x}_k - \bar{\mathbf{x}}). \tag{9}$$

Throughout this study, the c value is set to vary from 2 to 9. This range includes the known number of clusters in the data sets tested, as described in the next section. The optimal number of clusters, c^*, is determined as the one with the minimum V_{XB} value and the maximum $PBMf$ value.

3. An illustrative example

As an illustration, this section presents the results of applying the proposed procedure to a set of model generated data. The *varmasim* procedure of SAS was employed for the generation of real-valued vector time series data sets with three different models. For the sake of easier visualization, only bivariate models are considered. Specifically, we generated a bivariate ($k=2$) stationary VARMA(1, 1) time series

$$y_t - \mu = \Phi(y_{t-1} - \mu) + \varepsilon_t - \Theta\varepsilon_{t-1},$$

where

$$\Phi = \begin{bmatrix} 1.2 & -0.5 \\ 0.6 & 0.3 \end{bmatrix}, \quad \Theta = \begin{bmatrix} -0.6 & 0.3 \\ 0.3 & 0.6 \end{bmatrix}, \quad \mu = \begin{bmatrix} 10 \\ 20 \end{bmatrix},$$

$$\Sigma = \begin{bmatrix} 1.0 & 0.5 \\ 0.5 & 1.25 \end{bmatrix}.$$

The stationary VMA(1), and VAR(1) series were generated by resetting the Φ and Θ of the stationary VMARA(1, 1) model to zero, respectively. Five series of 100 data points each from each of the above three models were generated as shown in Fig. 1.

Fig. 2(a) shows the distribution of 1500 time stripped data before clustering. The number of clusters is not known and unclear with visual inspection. Therefore, the number of clusters is varied from 2 to 9. After applying fuzzy c-means to the time-stripped data for each specified number of clusters from 2 to 9 in the first step, each real-valued vector time series is converted into a discrete-valued univariate time series with 2 to 9 possible values, respectively. Due to space limitation, only the results of six clusters are given below. Fig. 2(b) shows the six clusters formed by the fuzzy c-means algorithm. Taking six as the number of clusters, each bi-variate time series shown in Fig. 1 is converted into a discrete-valued univariate time series as shown in Fig. 3. It can be visually seen that Fig. 3 pretty much preserves the same three different patterns as displayed in Fig. 1.

Each discrete-valued univariate time series is converted into a transitional probability matrix. For our implementation, each matrix is expressed as a vector by concatenating all rows together into one row. Fig. 4 shows each vector of transitional probability values corresponding to each discrete-valued univariate time series shown in Fig. 3. Three different patterns again can be seen in Fig. 4 after this data transformation. Based on the KL distance, the fuzzy c-means algorithm partitions the vectors of state transition probability values into three clusters because it is known that the data set was generated from three different models.

Table 1 summarizes the clustering accuracy results of nine runs for the set of model-generated data. It can be easily seen that the best result comes from forming five clusters in Step 1 of the procedure, which consistently makes the same one error in all nine runs. The specific error is mistaking the 2arma(1, 1)-4 time series as one generated from the 2ar(1) model. For comparison the clustering accuracies when using the Euclidean distance in the second step in place of the KL distance are also given in Table 1. The best result now comes from forming six clusters in Step 1 of the procedure, which consistently makes the same one error in all nine runs. The specific error is mistaking the 2ar(1)-3 time series as one generated

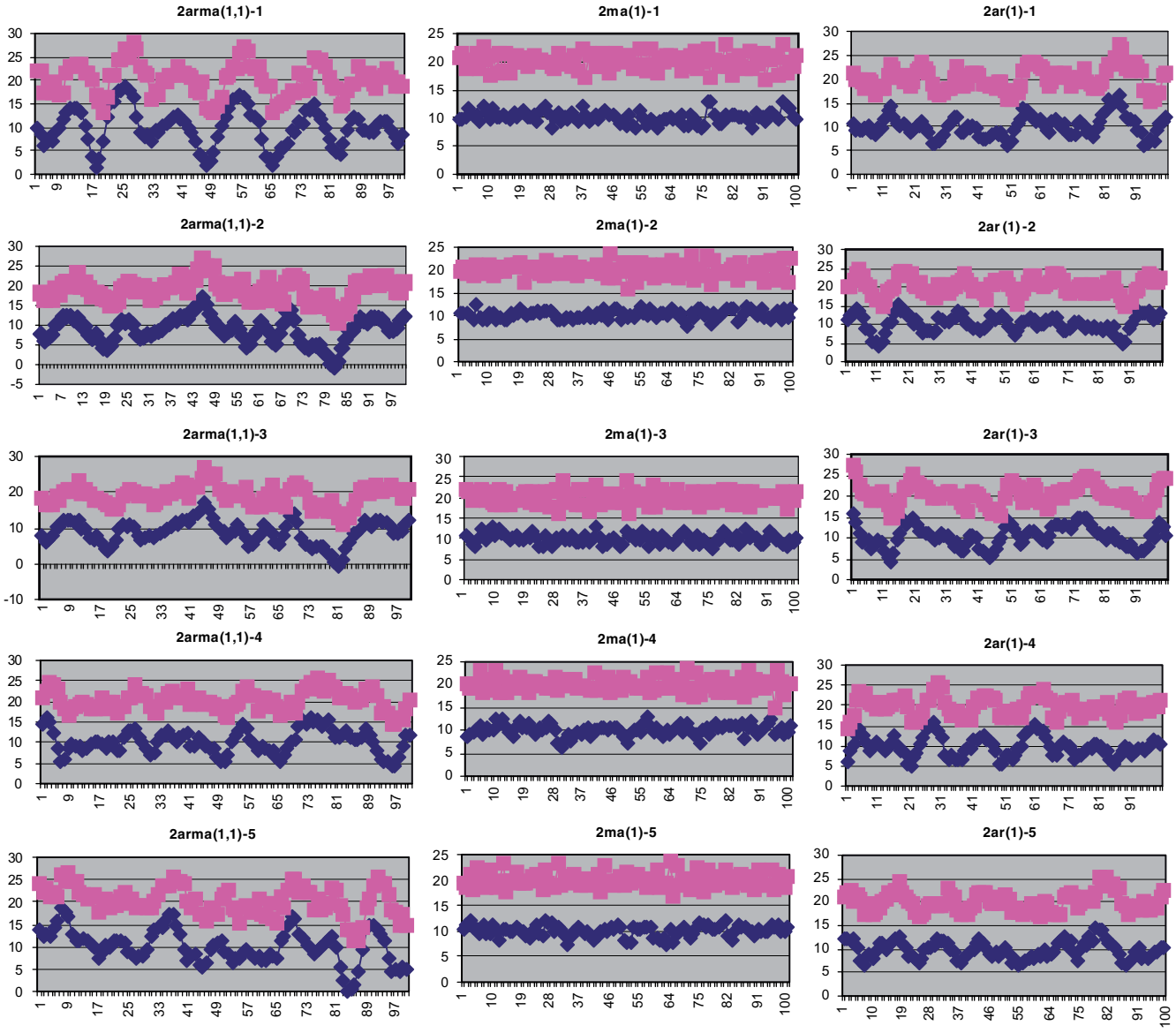

Fig. 1. Five series generated from three different models.

from the 2arma(1, 1) model. Depending upon the number of clusters used in Step 1, the Euclidean distance might increase the clustering accuracy as in the case of six clusters or decrease the clustering accuracy as in the case of four clusters, than that obtained by the KL distance.

To determine whether the two fuzzy validity indices can identify the actual number of clusters, i.e., 3, the number of clusters is set either 5 or 6 in Step 1 and varied from 2 to 9 in the second step. The clustering results are then used to compute the values of both validity indices. The reason for choosing only 5 and 6 in Step 1 is based on their clustering performance. Since the clustering accuracies for other cases are not good, it is not useful to study whether the indices can identify the number of clusters correctly for those cases. The results indicate that (i) the *PBMf* index tends to overestimate the number of clusters by identifying eight clusters one time and nine clusters four times in a total of five runs; (2) the *XB* index tends to underestimate the number of clusters by identifying two clusters in all five times. In other words, both indices are unable to correctly identify the actual number of clusters for the model-generated data.

4. Test results

To further confirm the results observed in the previous section, three more data sets from different application areas were tested.

4.1. Three-component force signals in grinding

Three-component force signals were collected with a dynamometer in grinding operations with two different grinding

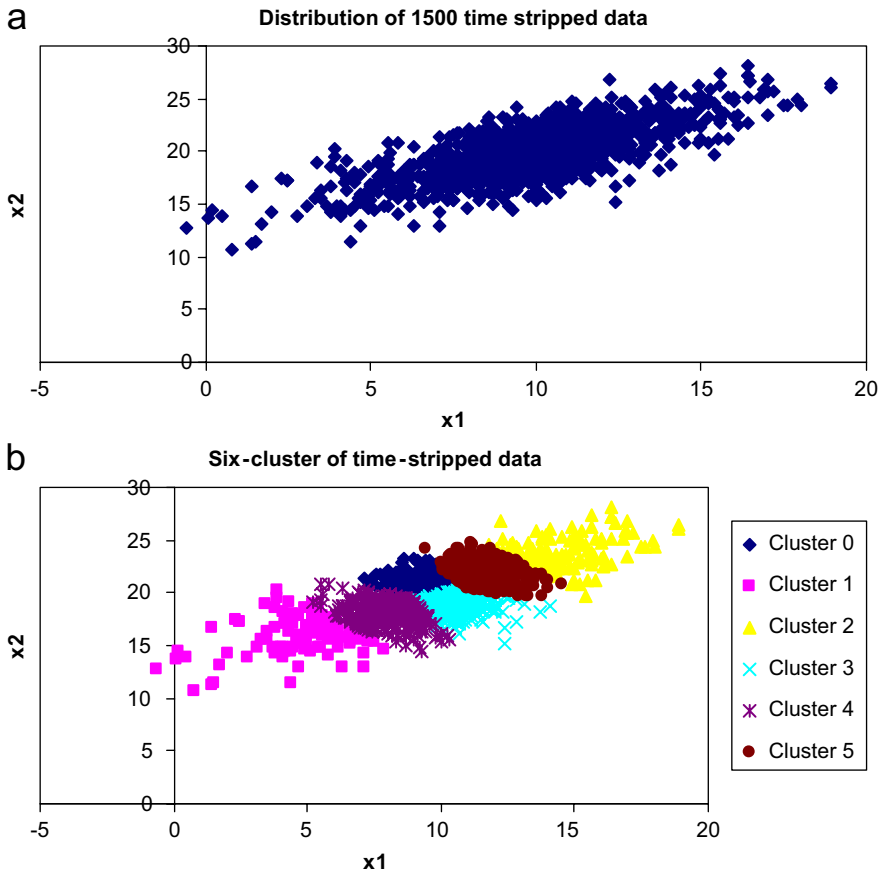

Fig. 2. Distribution of time stripped data: (a) before clustering, (b) forming six clusters.

conditions. A total of 35 signal segments with varying lengths from 901 to 2501 data points were obtained for this study. Since the 35 force signals come from two different grinding conditions, the task is to cluster them into two clusters with each cluster comprising the signals from the same grinding condition. More details about this data set can be found in Ref. [17], in which discrete wavelet analysis was first used to extract features from each resultant force signal computed from the corresponding three-component force signal, and an adaptive genetic clustering approach was then taken to cluster the data.

By varying the number of clusters from 2 to 9 in Step 1 and specifying the number of clusters as two in Step 2, the proposed procedure consistently gives 100% clustering accuracy. To make it a little bit more challenging, we redid the experiments by specifying the number of clusters as four in Step 2. Physically, for each grinding condition the signals generated in up-cut (the workpiece is fed into the wheel opposite to the wheel rotation direction) are different from those generated in down-cut (the workpiece moves in the same direction as the wheel rotation direction). Table 2 summarizes the clustering results of forming four clusters in Step 2 with varying number of clusters from 2 to 9 in Step 1. The highest accuracy is 80%, which is obtained when setting six as the number of clusters in Step 1. The results of specifying 2 and 3 clusters in Step 1 are not listed in the table because of their poor performance. Using such low number of clusters in Step 1 runs the risk of losing too much information, resulting in low clustering accuracy.

Fixing six clusters in Step 1 and varying the number of clusters from 2 to 9 in Step 2, both the two fuzzy validity indices identify 9 as the number of clusters in all five runs. In other words, both indices overestimate the number of clusters for this particular data set, failing to find 2 or 4 as the actual number of clusters.

4.2. Multi-sensor based signals in grinding

This data set comprises twenty 500-point segments of multiple signals extracted from the raw signals collected in grinding operations using a wheel in two different states: “sharp” and “dull”. The six dimensions are x-force, y-force, z-force, power, vibration, and root mean squared acoustic emission (RMS AE). The clustering task is to group signals generated when the wheel is sharp or dull together and separated them away from each other.

The fuzzy c-means algorithm is first applied to the time stripped data (with size $10{,}000\times6$) to form varying number of

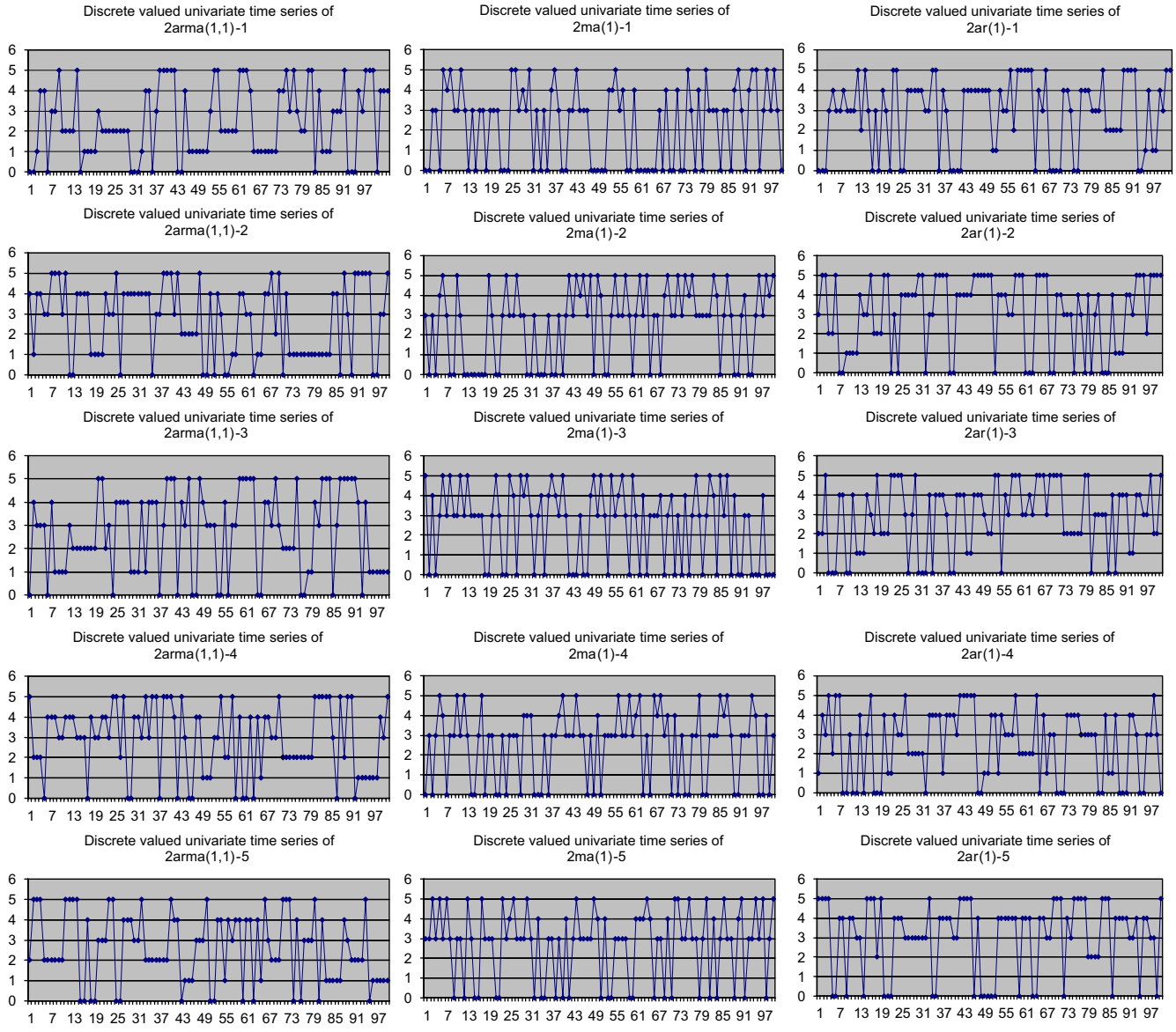

Fig. 3. Discrete valued univariate time series of model-generated data at the end of Step 1.

clusters from 2 to 9 using the Euclidean distance to measure the dissimilarity of two objects being compared. As a result, each six-dimensional time series is converted into a discrete-valued univariate time series with c possible discrete values if c clusters were formed, where c is varied from 2 to 9 as mentioned above. Each discrete-valued univariate time series in turn is converted into a transitional probability vector. The set of transitional probability vectors are then processed by the fuzzy c-means algorithm to form the desirable number of clusters, i.e., 2 in this case, using the KL distance.

The clustering accuracy results of nine runs for this set of multi-sensor based grinding signals is 100% regardless the number of clusters ranging from 2 to 9. To determine its effect, the Euclidean distance is used to replace the KL distance and the results are summarized in Table 3. Since the KL distance gives perfect results, the Euclidean distance could not improve the clustering accuracy for any case. In the meanwhile it degrades the performance of two cases.

To determine whether the two fuzzy validity indices can identify the actual number of clusters, i.e., 2, for this data set the number of clusters is varied from 2 to 9 in Step 2 of the proposed procedure. The clustering results are then used to compute the values of both validity indices. The results, as summarized in Table 4, indicate that (i) the *PBMf* index tends to overestimate the number of clusters; (2) the *XB* index correctly identifies the actual number of clusters in four cases and overestimates the number of clusters in three cases; (iii) the validity results obtained when forming two clusters in Step 1 are not reliable and omitted because the transitional probability vectors could not form high number of clusters in Step 2.

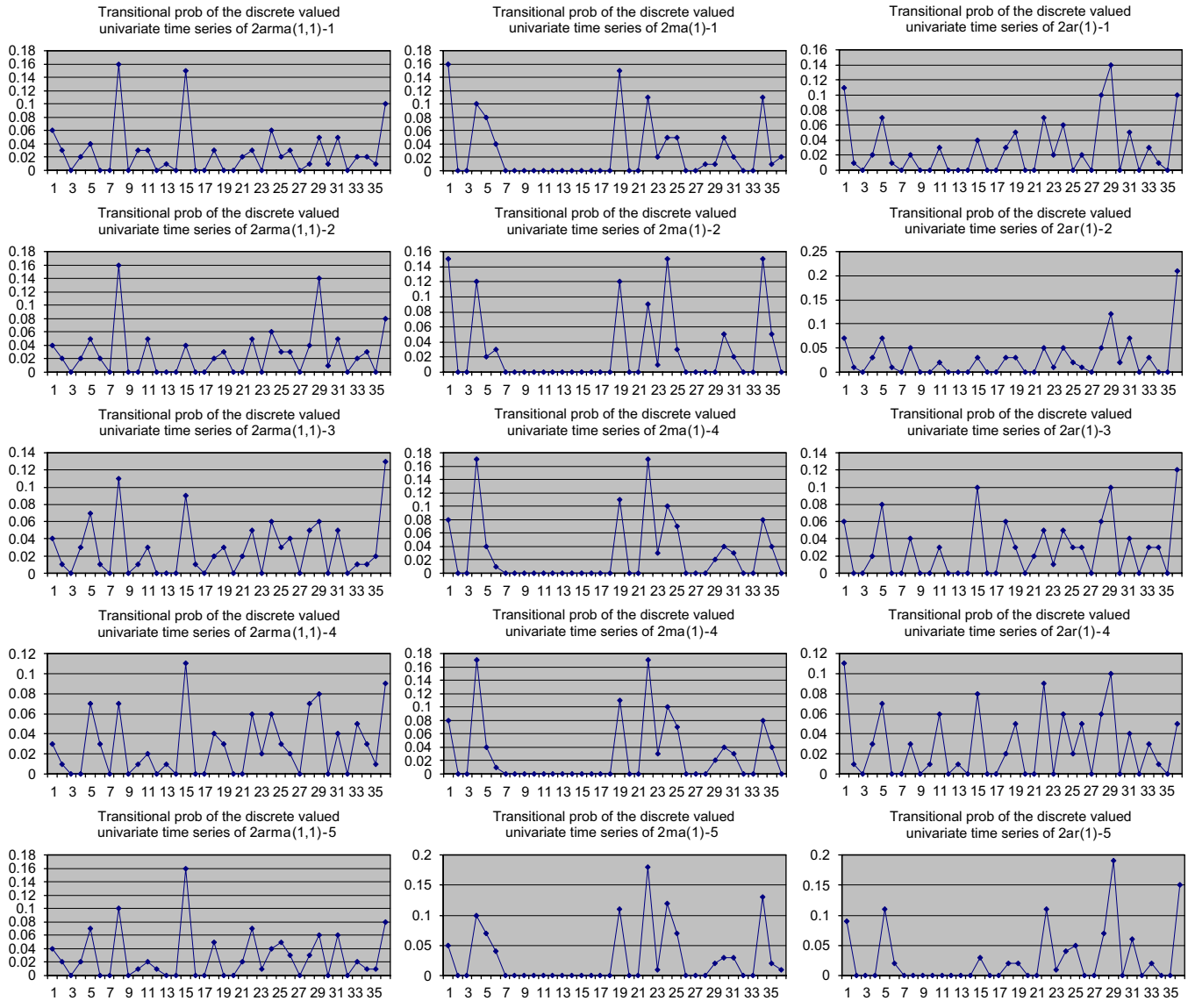

Fig. 4. Transitional probabilities of the discrete-valued univariate time series shown in Fig. 3.

4.3. Auslan sign data

We took 20 files from the John4 holder of Auslan signs (short for Australian sign language) in the UCI KDD Archive for testing. The selected data involves four signs: ‘alive’, ‘all’, ‘boy’, and ‘cold’ with each repeated five times by the same signer. Each data file is a real-valued vector time series with 15 variables; only the first 11 numeric variables are used in this study. Note that each file in this data set varies in length from 45 to 60 data points.

The fuzzy c-means algorithm is first applied to the time stripped data (with size 1072×11) to form varying number of clusters from 2 to 9 using the Euclidean distance to measure the dissimilarity of two objects being compared. As a result, each eleven-dimensional time series is converted into a discrete-valued univariate time series with c possible discrete values if c clusters were formed. Each discrete-valued univariate time series in turn is converted into a transitional probability vector. The set of transitional probability vectors are then processed by the fuzzy c-means algorithm to form the desirable number of clusters, i.e., 4 in this case, using the KL distance.

Table 5 summarizes the clustering accuracy results of nine runs for this set of Auslan sign signals. Based on the results, it can be easily seen that forming nine clusters in Step 1 of the procedure produces the highest accuracy with one miss only consistently in eight of nine runs. The specific error is mistaking one of the ‘boy’ signs as an ‘all’ sign. To determine the effect of using the Euclidean distance instead, the results are also summarized in Table 5. Again, forming nine clusters in Step 1 of the procedure produces the highest accuracy with the same mistake as using the KL distance; but the best result occurs in only six of nine runs.

Table 1
Summary of clustering results of model-generated data

Distance	Run	2-cluster	3	4	5	6	7	8	9
KL	1	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	2	0.6000	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	3	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	4	0.6000	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	5	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	6	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	7	0.6000	0.6000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	8	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	9	0.6000	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	Best	0.7337	0.8000	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
	Avg	0.6741	0.7778	0.8667	0.9333	0.8667	0.8667	0.8667	0.8667
Euclidean	1	0.7337	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	2	0.6000	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8000
	3	0.7337	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	4	0.6000	0.8667	0.8000	0.9333	0.9333	0.6000	0.8667	0.8667
	5	0.7337	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	6	0.7337	0.6000	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	7	0.6000	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	8	0.7337	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	9	0.6000	0.8667	0.8000	0.6000	0.9333	0.9333	0.8667	0.8667
	Best	0.7337	0.8667	0.8000	0.9333	0.9333	0.9333	0.8667	0.8667
	Avg	0.6741	0.8370	0.8000	0.8963	0.9333	0.8963	0.8667	0.8593

Table 2
Summary of 4-cluster clustering results of three-component grinding signals

Distance	Run	4-cluster	5	6	7	8	9
KL	1	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	2	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	3	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	4	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	5	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	6	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	7	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	8	0.6571	0.6000	0.6571	0.7714	0.7429	0.7714
	9	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	Best	0.6571	0.6000	0.8000	0.7714	0.7429	0.7714
	Avg	0.6571	0.6000	0.7841	0.7714	0.7429	0.7714

Table 3
Summary of clustering results of multi-sensor based grinding signals

Distance	Run	2-cluster	3	4	5	6	7	8	9
KL	1–9	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Euclidean	1	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	2	0.50	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	3	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	4	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	5	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	6	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	7	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	8	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	9	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	Best	1.00	1.00	0.80	1.00	1.00	1.00	1.00	1.00
	Avg	0.94	1.00	0.80	1.00	1.00	1.00	1.00	1.00

Table 4
Number of clusters identified by two fuzzy validity indices for multi-sensor based grinding signals

Index	Run	3-cluster	4	5	6	7	8	9
PBMf	1	8	9	9	9	9	7	9
	2	8	6	9	9	9	9	9
	3	8	9	9	8	7	9	9
	4	8	7	9	9	9	9	9
	5	8	8	9	9	9	9	9
XB	1	2	2	2	2	4	4	4
	2	2	2	2	2	4	4	4
	3	2	2	9	2	4	4	4
	4	2	2	9	2	4	4	4
	5	2	2	2	2	2	2	3

To determine whether the two fuzzy validity indices can identify the actual number of clusters, i.e., 2, for this data set the number of clusters is set at 9 in Step 1 and varied from 2 to 9 in Step 2 of the proposed procedure. The clustering results are then used to compute the values of both validity indices. The results indicate that both indices overestimate the actual number of clusters with (i) the PBM-f index identifies eight clusters one time and nine clusters four times; (2) the *XB* index identifies six, eight, and nine clusters one time each and seven clusters twice.

5. Comparison with a HMM-based clustering method

HMM is well known for sequence modeling. However, HMM-based clustering methods are developed only recently.

Table 5
Summary of clustering results of Auslan sign data

Distance	Run	2-cluster	3	4	5	6	7	8	9
KL	1	0.55	0.65	0.70	0.65	0.75	0.85	0.80	0.95
	2	0.55	0.65	0.70	0.65	0.75	0.85	0.80	0.95
	3	0.55	0.65	0.90	0.65	0.75	0.70	0.70	0.95
	4	0.55	0.65	0.70	0.65	0.75	0.70	0.70	0.70
	5	0.55	0.65	0.65	0.65	0.75	0.70	0.70	0.95
	6	0.55	0.65	0.70	0.85	0.75	0.85	0.80	0.95
	7	0.55	0.65	0.65	0.65	0.75	0.70	0.70	0.95
	8	0.55	0.65	0.70	0.85	0.75	0.85	0.75	0.95
	9	0.55	0.60	0.90	0.65	0.75	0.70	0.70	0.95
	Best	0.55	0.65	0.90	0.85	0.75	0.85	0.80	0.95
	Avg	0.55	0.64	0.73	0.69	0.75	0.77	0.74	0.92
Euclidean	1	0.65	0.60	0.65	0.65	0.75	0.75	0.70	0.95
	2	0.70	0.60	0.65	0.65	0.75	0.75	0.70	0.95
	3	0.65	0.60	0.90	0.80	0.75	0.75	0.70	0.95
	4	0.70	0.60	0.65	0.65	0.75	0.75	0.70	0.95
	5	0.65	0.60	0.65	0.65	0.75	0.75	0.70	0.40
	6	0.65	0.60	0.65	0.80	0.75	0.75	0.70	0.95
	7	0.65	0.60	0.65	0.65	0.75	0.65	0.65	0.60
	8	0.65	0.60	0.65	0.65	0.75	0.75	0.70	0.95
	9	0.65	0.60	0.65	0.65	0.65	0.75	0.70	0.60
	Best	0.70	0.60	0.90	0.80	0.75	0.75	0.70	0.95
	Avg	0.66	0.60	0.68	0.68	0.74	0.74	0.69	0.84

The HMM-based sequential data clustering method proposed by Becego et al. [9] is chosen for comparison with the proposed procedure. Their method kind of also has two steps: to build a new representation space using the similarity values between sequences (in the form of vector time series in this study) obtained via the HMMs, and to perform the clustering in that space. Given a set of sequences $T = \{\mathbf{X}^1, \ldots, \mathbf{X}^N\}$ to be clustered, their method can be described as follows:

(1) Let R be a set of "reference" or "representative" objects; Set $R = T$ in this study.
(2) Train one HMM λ_r for each sequence in R.
(3) Represent each sequence $\mathbf{X}^i$ of T by the set of similarities $S_R(\mathbf{X}^i)$ to the elements of the representative set R, computed with the HMMs $\lambda_1 \ldots \lambda_R$ as

$$S_R(\mathbf{X}^i) = \frac{1}{L_i}\begin{bmatrix} \log P(\mathbf{X}^i|\lambda_1) \\ \vdots \\ \log P(\mathbf{X}^i|\lambda_R) \end{bmatrix}, \quad \text{where } L_i \text{ is the length of } \mathbf{X}^i. \tag{10}$$

(4) Perform clustering using any general technique appropriate for clustering points in an Euclidean space; Fuzzy c-means is chosen here to be consistent with our proposed procedure.

Applying the above method (with both the number of hidden states and the number of mixed Gaussian distributions set at two) to the model-generated data, three-component grinding forces, multi-sensor grinding signals, and Auslan sign data yields clustering accuracies of 73.3%, 100% for two clusters and 64.3% for four clusters, 100%, and 65%, respectively. Liao and Hua recently proposed a new similarity measure for the HMM-based clustering method, which computes $S_j(X^i)$ as

$$S_j(X^i) = \frac{(\log P(X^i|\lambda_i) - \log P(X^i|\lambda_j)) + (\log P(X^j|\lambda_j) - \log P(X^j|\lambda_i))}{2}. \tag{11}$$

Ordinarily, $\log P(X^i|\lambda_j) < \log P(X^i|\lambda_i)$ since the ith model is trained to maximize the probability of the ith time series being generated by the ith model; so $S_j(X^i) > 0$ for $i \neq j$. In practice, there might be cases where the opposite is true, resulting in $S_j(X^i) \leqslant 0$. To handle this situation, a very small positive value is assigned as $S_j(X^i)$. Replacing Eqs. (10) with (11) and reapplying the same HMM-based clustering method to the same four data sets, the clustering accuracies are 93.3%, 100% for two clusters and 78.6% for four clusters, 100%, and 65%, respectively. Note that the new similarity measure improves the clustering accuracies for the first three data sets.

6. Discussion

Several issues related to the proposed clustering procedure are discussed in this section. First of all, we reiterate that the proposed procedure provides a general framework in which literally any existing or new clustering algorithm can be used in either one of the two steps of the proposed procedure. The algorithm can be anyone in the five major categories: partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods, as classified by Han and Kamber [18]. Even though this study elects to employ

the fuzzy *c*-means algorithm, a well-known partitioning method, other algorithms in other categories are definitely applicable. Depending upon the data one is dealing with, one clustering algorithm might work better than the others. To guarantee better results, good amount of knowledge about the data and which clustering algorithm would work better is helpful.

The second issue is related to specifying the number of clusters in the first step of the proposed procedure. Determining the number of clusters is a validity issue. Many validity indices have been proposed and several comparative studies have been carried out to investigate their performance in the past. Nevertheless, no index has been shown always identifying the actual number of clusters correctly. As a matter of fact, new indices are continuously being developed and shown that they work better than others based on a number of selected data sets. But one is often disappointed after such indices are applied to other data sets not included in the original study. A recent comparative study of us confirms that this indeed is the case. So many indices even fail to correctly identify the actual number of clusters in a data set with non-overlapping clusters; not to mention using them in highly overlapping data as often the case here, as shown in Fig. 2. Instead of searching for a golden index, some researchers suggest applying several indices and take the majority vote of all. No one has shown whether this idea actually works or not. We are very doubtful that it would work. The reason is that some indices tend to overestimate whereas others tend to underestimate; the majority vote of a combination of these indices does not seem to be able to produce any meaningful result at all. Therefore, in this study we opt for the trial and error process with a selected range of values (2–9). The trial and error process is tedious and time consuming but is essential to increase the chance of getting better results, as shown for the model-generated data, the 4-cluster clustering of three-component grinding force data and the Auslan sign data. For the easy data such as the multi-sensor based grinding signals, it does not seem to matter how many clusters are specified in Step 1.

Assuming that there is a priori knowledge about the number of clusters, the proposed procedure was shown to be quite effective in generating sensible clustering, even better than the ever more popular HMM-based clustering method. In the case that there is no knowledge then we are faced with the same problem as we discuss in the previous paragraph. In other words, the same issue of determining the number of clusters exists in the second step of the proposed procedure. Trying not to sound too pessimistic, we are quite sure that there is simply no one index that can help out here as we learned from our experience in conducting the comparative study mentioned above.

It is not our intention here to carry out a comprehensive evaluation of all existing indices to prove our point. The performances of the two indices evaluated in this study do provide some important clue. In summary, the *PBMf* index did not correctly find the actual number of clusters for all data sets tested. The *PBMf* index was only tested with well-separated static feature data in the original paper of Pakhira et al. Our results imply that the *PBMf* index does not work well when used on data with high degree of overlapping and/or high-dimensional dynamic time series data. Our results indicate that the *PBMf* index seems to have a tendency to overestimate the actual number of clusters. As far as the *XB*-index is concerned, it correctly identifies the actual number of clusters for the multi-sensor based grinding signals for some cases, but failed for other data sets as well. Unlike the *PBMf* index, it does not always err on one side. It underestimates the actual number of clusters for the model-generated data and overestimates for other data sets.

In the case that there is no ground truth about the actual number of clusters and no validity index is reliable enough to tell, one is advised to resort to visual inspection. As an example, we take 15 runs of battle simulation data and subject them to the proposed procedure. Assume that the intended application is to show three different situations whether a battle is a 'tie', 'win', or 'loss.' We thus specify six and three as the number of clusters in Step 1 and Step 2, respectively, and show the clustering results in Fig. 5. Each plot in Fig. 5 shows a five-dimensional time series denoted as "o" for battle field ownership, "s" for combat strength, "a" for ammunition support, "f" for fuel support, and "i" for combat intensity. For the battle scenario simulated, the "a" and "f" indicators do not differ much among all series compared with the other three indicators: "o", "s", and "i". The "o" and "s" indicators portray better the trend of the battle. They are highly correlated because it requires strength to overpower the opponent in order to better control the battle field and to take the lion share of the battle field. The "i" indicator, which shows the ratio of fire exchanges between the blue force and the red force, is mostly irregular, but not necessary telling. Based on visual inspection, the partition is generally sensible with one group showing the "tie" situation, another group the "win" situation, and the other the "loss" situation. Of course, the results are not 100% accurate and one could very well argue that a few series might fit better in other clusters. For example, b5 might fit better in the "tie" cluster than the "loss" cluster.

The KL distance used in the second step for comparing discrete-valued univariate profiles was evaluated against the Euclidean distance for most data sets tested. The overall performance of the KL distance seems to be slightly better than the Euclidean distance, but insignificant statistically. Other distance measures such as correlation could also be used but further study is needed to evaluate their performance.

7. Conclusions

This paper presented a general 2-step procedure developed for clustering real-valued vector time series data. Vector time series data exist in many domains in large volume. The proposed procedure is intended to help preprocess such large volume of data so that focused modeling and analysis efforts can be devoted to selected series only. In other words, the proposed procedure can be used for pattern recognition as well as exploratory data mining.

We showed the effectiveness of the proposed procedure and compared it to a HMM-based clustering method using four data sets: one set of model generated data, two sets of sensory signals, and one set of simulation data, in which the number of clusters is known. Two fuzzy validity indices were tested

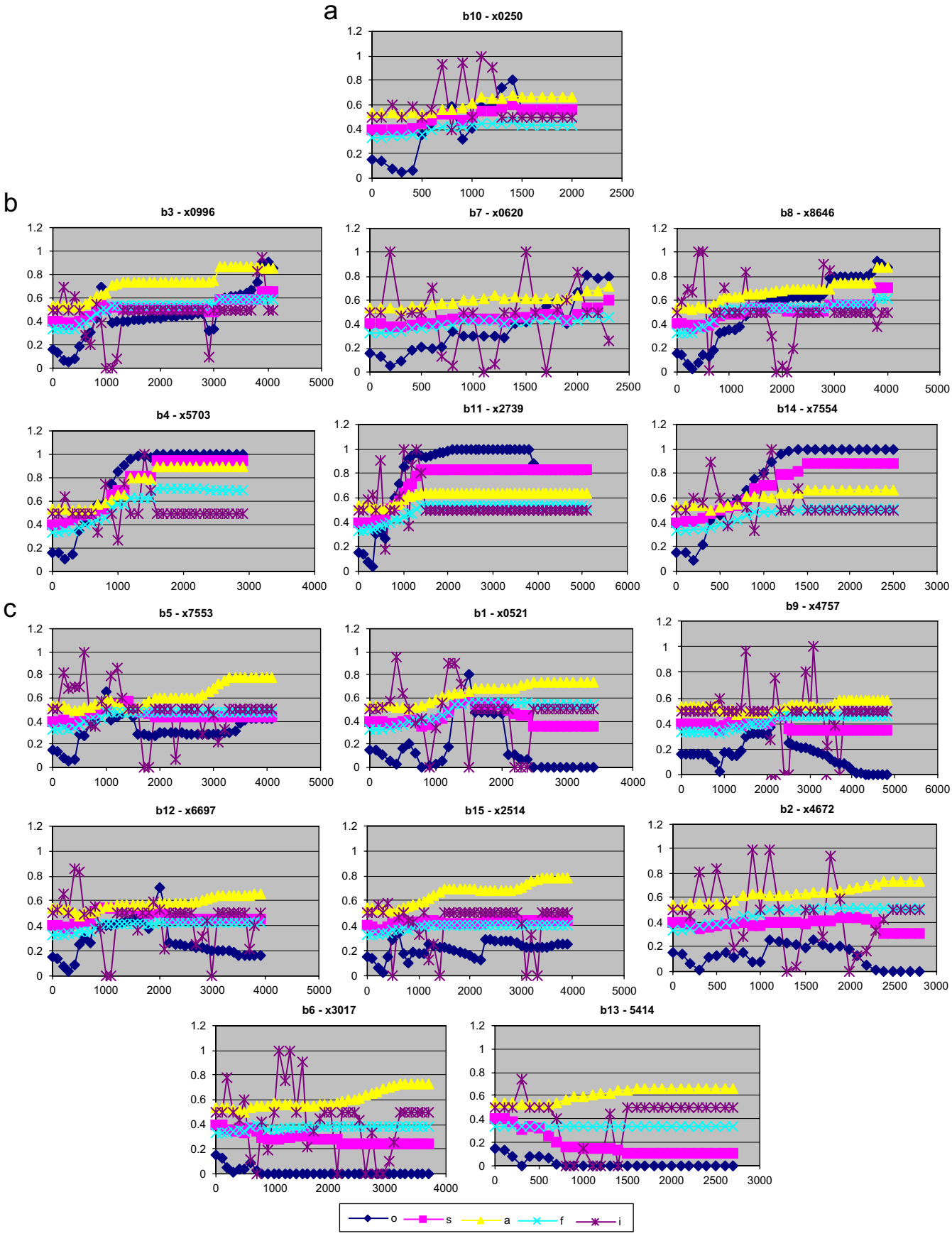

Fig. 5. Three clusters of battle simulation data: (a) the "tie" cluster, (b) the "win" cluster, (c) the "loss" cluster.

for their ability to identify the actual number of clusters and found unreliable for the subject application. Continuous effort to develop a better index for high dimensional data with highly overlapped clusters is thus necessary. Before such a powerful index can be developed, the trial and error procedure for the first step and visual inspection for the second step if the number of clusters is not known are recommended.

This study can be extended in two ways. The first is feature selection and weighting. All features are assumed equally weighted in the fuzzy c-means algorithm. To explore whether feature selection and weighting would help, a genetic algorithm based feature weighting method could be used. The second is scaling up the procedure for exploratory mining of large volume of time series data. For this, we are currently investigating the use of grid computing for distributed data mining. This might call for some modification of the proposed procedure here. We hope to report the results of such extended studies in the near future.

Acknowledgments

The author acknowledges the support of Army Research Laboratory for his sabbatical leave and the support of High Temperature Material Laboratory, Oak Ridge National Laboratory through their User Program for some of the data used in this study.

References

[1] S. Swift, A. Tucker, X. Liu, Evolutionary computation to search for strongly correlated variables in high-dimensional time-series, in: D.J. Hand, J.N. Kok, M.R. Berthold (Eds.), IDA'99, Lecture Notes in Computer Science, vol. 1642, Springer, Berlin, 1999, pp. 51–62.

[2] A. Tucker, S. Swift, X. Liu, Variable grouping in multivariate time series via correlation, IEEE Trans. Systems, Man, Cybern.—Part B: Cybernetics 31 (2) (2001) 235–245.

[3] T.W. Liao, Clustering of time series data—a survey, Pattern Recognition 38 (11) (2005) 1857–1874.

[4] D. Jiang, C. Tang, A. Zhang, Cluster analysis for gene expression data: a survey, IEEE Trans. Knowledge Data Eng. 16 (11) (2004) 1370–1386.

[5] K. Košmelj, V. Batagelj, Cross-sectional approach for clustering time varying data, J. Classification 7 (1990) 99–109.

[6] T.W. Liao, B. Bolt, J. Forester, E. Hailman, C. Hansen, R.C. Kaste, J. O'May, Understanding and projecting the battle state, in: Proceedings of the 23rd Army Science Conference, Orlando, FL, December 2–5, 2002.

[7] Y. Kakizawa, R.H. Shumway, M. Taniguchi, Discrimination and clustering for multivariate time series, J. Amer. Statist. Assoc. 93 (1998) 328–340.

[8] L.M.D. Owsley, L.E. Atlas, G.D. Bernard, Automatic clustering of vector time series for manufacturing machine monitoring, in: Proceedings of ICASSP '97, Munich, Germany, 1997, pp. 21–24.

[9] M. Bicego, V. Murino, M.A.T. Figueiredo, Similarity-based clustering of sequences using hidden Markov models, in: P. Perner, A. Rosenfeld (Eds.), MLDM 2003, Lecture Notes in Artificial Intelligence, vol. 2734, 2003, pp. 86–95.

[10] M. Romoni, P. Sebastiani, P. Cohen, Multivariate clustering by dynamics, in: Proceedings of the 2000 National Conference on Artificial Intelligence, San Francisco, CA, 2000, pp. 633–638.

[11] M. Romoni, P. Sebastiani, P. Cohen, Bayesian clustering by dynamics, Mach. Learn. 47 (1) (2002) 91–121.

[12] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: L.M. LeCam, J. Neyman (Eds.), Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, University of California Press, Berkeley, 1967, pp. 281–297.

[13] J.C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, J. Cybernet. 3 (1974) 32–57.

[14] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, London, 1987.

[15] M.K. Pakhira, S. Bandyopadhyay, U. Maulik, Validity index for crisp and fuzzy clusters, Pattern Recognition 37 (2004) 487–501.

[16] X.L. Xie, G. Beni, A validity measure for fuzzy clustering, IEEE Trans. Pattern Anal. Mach. Intell. 13 (8) (1991) 841–847.

[17] T.W. Liao, C.-F. Ting, P.-C. Chang, An adaptive genetic clustering method for exploratory mining of feature vector and time series data, Int. J. Production Res. 44 (14) (2006) 2731–2748.

[18] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, San Francisco, 2001.

About the Author—T. WARREN LIAO received his Ph.D. in Industrial Engineering from Lehigh University in 1990 and is currently a Professor with Industrial Engineering Department, Louisiana State University. His research interests include soft computing, pattern recognition, data mining, and their applications in manufacturing. He has more than 60 refereed journal publications and was the guest editor for several journals including *Journal of Intelligent Manufacturing*, *Computers & Industrial Engineering*, *Applied Soft Computing*, and *International Journal of Industrial Engineering*.