Available online at www.sciencedirect.com
SCIENCE DIRECT®
Fuzzy Sets and Systems 149 (2005) 39–56
FUZZY sets and systems
www.elsevier.com/locate/fss

Modified Gath–Geva clustering for fuzzy segmentation of multivariate time-series

Janos Abonyi*, Balazs Feil, Sandor Nemeth, Peter Arva

Department of Process Engineering, University of Veszprem, P.O. Box 158, Veszprem H-8201, Hungary

Available online 19 August 2004

Abstract

Partitioning a time-series into internally homogeneous segments is an important data-mining problem. The changes of the variables of a multivariate time-series are usually vague and do not focus on any particular time point. Therefore, it is not practical to define crisp bounds of the segments. Although fuzzy clustering algorithms are widely used to group overlapping and vague objects, they cannot be directly applied to time-series segmentation, because the clusters need to be contiguous in time. This paper proposes a clustering algorithm for the simultaneous identification of local probabilistic principal component analysis (PPCA) models used to measure the homogeneity of the segments and fuzzy sets used to represent the segments in time. The algorithm favors contiguous clusters in time and is able to detect changes in the hidden structure of multivariate time-series. A fuzzy decision making algorithm based on a compatibility criteria of the clusters has been worked out to determine the required number of segments, while the required number of principal components are determined by the screeplots of the eigenvalues of the fuzzy covariance matrices. The application example shows that this new technique is a useful tool for the analysis of historical process data.

Keywords: Fuzzy clustering; Time-series segmentation; Process monitoring

1. Introduction

Knowledge discovery in databases (KDD) and/or data mining refers to the overall process of discovering knowledge from data. KDD has evolved from the intersection of research fields such as

* Corresponding author.
E-mail address: abonyij@fmt.vein.hu (J. Abonyi).
URL: http://www.fmt.vein.hu/softcomp.

0165-0114/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.fss.2004.07.008

machine learning, pattern recognition, databases, statistics, artificial intelligence, and more recently it has received new inspiration from soft computing [20]. In this paper, a new soft computing-based algorithm is presented which is useful for the fuzzy segmentation of multivariate temporal databases.

A sequence of N-observed data, $\mathbf{x}_1, \ldots, \mathbf{x}_N$, ordered in time, is called time-series. Real-life time-series can be taken from business, physical, social and behavioral science, economics, engineering, etc. Time-series segmentation addresses the following data-mining problem: given a time-series, T, find a partitioning of T into c segments that are internally homogeneous [24]. Depending on the application, the goal of the segmentation is to locate stable periods of time, to identify change points, or to simply compress the original time-series into a more compact representation [18]. Although, in many real-life applications a lot of variables must be simultaneously tracked and monitored, most of the segmentation algorithms are used for the analysis of only one time-variant variable [16]. However, in some cases it is necessary to synchronously segment the time-series of the variables.

The segmentation of multivariate time-series is especially important in the data-based analysis and monitoring of modern production systems, where huge amount of historical process data are recorded with distributed control systems (DCS). These data definitely have the potential to provide information for product and process design, monitoring and control [27]. This is especially important in many practical applications, where first-principles modeling of complex "data-rich and knowledge-poor" systems are not possible [28]. Therefore, KDD methods have been successfully applied to the analysis of process systems, and the results have been used in process design, process improvement, operator training, and so on [25]. Hence, the data-mining algorithm proposed in this paper has been developed to the analysis of the historical process data of a medium and high-density polyethylene (MDPE, HDPE) plant. The operators of this polymerization process should simultaneously track many process variables. Of course, due to the hidden nature of the system the measured variables are correlated. Hence, it is useful to monitor only some principal components that is widely applied in advanced process monitoring. The main problem of this approach is that in some cases the hidden process, which can be observed as the correlation among the variables, varies in time. In our example, this phenomenon can occur when a different product is formed, and/or different catalyst is applied, or there are significant process faults, etc. The segmentation of only one measured variable is not able to detect such changes. Hence, the segmentation algorithm should be based on multivariate statistical tools.

To demonstrate this problem, let us consider the synthetic dataset shown in Fig. 1. The observed variables that can be seen on Fig. 1(b) are not independent, they were generated by the latent variables shown in Fig. 1(a). The correlation among the observed variables changes at the quarter of the time period, and the mean of the latent variables changes at the half of the time period. These changes are marked by vertical lines on Fig. 1(a).

As can be seen on Fig. 1(b), such information can be detected neither by application of univariate segmentation algorithms, nor by the visual inspection of the observed variables. Hence, the aim of this paper is to develop an algorithm that is able to handle time-varying characteristics of multivariate data: (i) changes in the mean; (ii) changes in the variance; and (iii) changes in the correlation structure among the variables.

To discover that type of changes of the hidden relationships of multivariate time-series, multivariate statistical tools should be applied by the segmentation algorithm. Among the wide range of possible tools, e.g., random projection, independent component analysis, the proposed algorithm utilizes principal component analysis (PCA). PCA maps the data points into a lower dimensional space, which is useful in the analysis and visualization of the correlated high-dimensional data [22]. Linear PCA models have

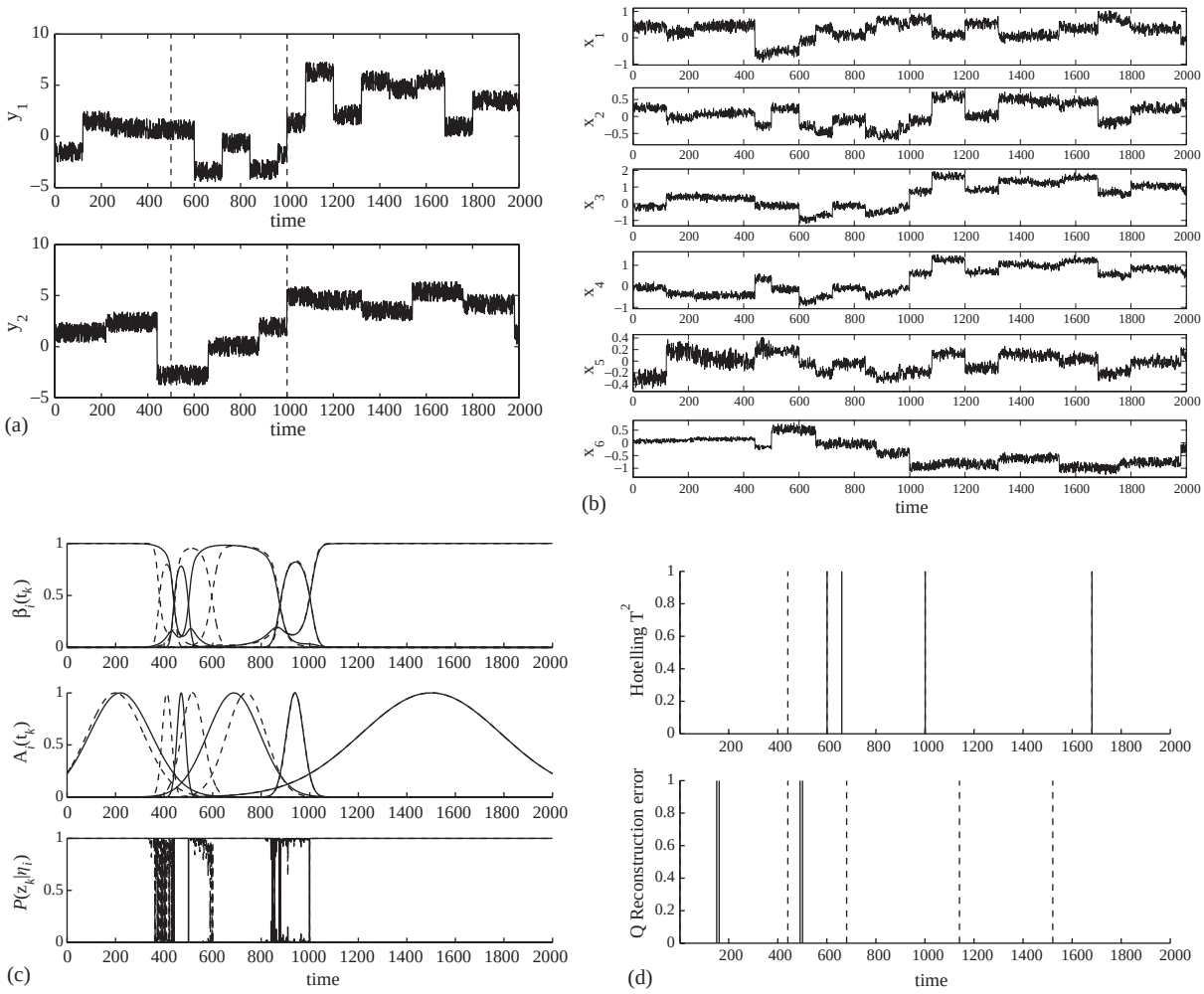

Fig. 1. The synthetic dataset and its segmentation by different algorithms based on two and five principal components: (a) Latent variables, (b) observed variables, (c) results obtained by fuzzy clustering, (–): $q = 2$, (- -): $q = 5$ and (d) results obtained by the bottom-up algorithm, (–): $q = 2$, (- -): $q = 5$.

two particularly desirable features: they can be understood in great detail and they are straightforward to implement. Linear PCA can give good prediction results for simple time-series, but can fail in the analysis of historical data having changes in regime or having nonlinear relations among the variables. The analysis of such data requires the detection of locally correlated clusters [8]. These algorithms do the clustering of the data to discover the local relationship among the variables similar to a mixture of principal component models [23].

Mixtures have been extensively used as models where the data can be considered as originating from several populations mixed in varying proportions, and expectation maximization (EM) is widely used to estimate the parameters of the components in a mixture [7]. The clusters obtained by Gath–Geva (GG) clustering, also referred to fuzzy maximum-likelihood clustering, are multivariate Gaussian functions. The

alternating optimization (AO) of these clusters is identical to the EM (maximum-likelihood estimation) identification of the mixture of these Gaussian models with fuzzy-weighting exponent $m = 2$ [6].

Time-series segmentation may be considered as clustering with a time-ordered structure. The contribution of this paper is the introduction of a new fuzzy clustering algorithm which can be effectively used to segment large, multivariate time-series. Since the points in a cluster must come from successive time points, the time-coordinate of the data has to be also considered during the clustering. One possibility to deal with time is to define a new cluster prototype that uses time as an additional variable. Hence, the clustering is based on a distance measure which consists of two terms: the first distance term is based on how the data are in the given segment defined by the Gaussian fuzzy sets defined in the time domain, while the second term measures how far the data are from the hyperplane of the PCA model of the segments.

The fuzzy segmentation of time-series is an adequate idea. The changes of the variables of the time-series are usually vague and are not focused on any particular time point. Therefore, it is not practical to define crisp bounds of the segments. For example, if humans visually analyze historical process data, they use expressions like "this point belongs to this operating point less and belongs to the other more". A good example of this kind of fuzzy segmentation is how fuzzily the start and end of *early morning* is defined. Fuzzy logic is widely used in various applications, where the grouping of overlapping and vague objects is necessary [1], and there are many fruitful examples in the literature for the combination of fuzzy logic with time-series analysis tools [5,9,18,26].

The key problem of the application of fuzzy clustering for time-series segmentation is the selection of the number of segments for the clustering process. Obviously, this is a standard problem also in the classical *c*-means clustering. In the context of time-series, however, it appears to be even more severe. For this purpose a bottom-up algorithm has been worked out where the clusters are merged during a recursive process. The cluster merging is coordinated by a fuzzy decision-making algorithm which utilizes a compatibility criterion of the clusters [12], where this criterion is calculated by the similarity of the principal component models of the clusters [17].

The rest of this paper is organized as follows. Section 2 defines the time-segmentation problem and presents the proposed clustering algorithm. Section 3 contains computational examples using a synthetic dataset and historical process data collected during the production of high-density polyethylene. Finally, some conclusions are given in Section 4.

2. Clustering-based time-series segmentation

2.1. Time-series segmentation

A time-series $T = \{\mathbf{x}_k | 1 \leqslant k \leqslant N\}$ is a finite set of N samples labelled by time points $t_1, \ldots, t_N$, where $\mathbf{x}_k = [x_{1,k}, x_{2,k}, \ldots, x_{n,k}]^{\mathrm{T}}$. A segment of T is a set of consecutive time points $S(a, b) = \{a \leqslant k \leqslant b\}$, $\mathbf{x}_a, \mathbf{x}_{a+1}, \ldots, \mathbf{x}_b$. The c-segmentation of time-series T is a partition of T to c non-overlapping segments $S_T^c = \{S_i(a_i, b_i) | 1 \leqslant i \leqslant c\}$, such that $a_1 = 1$, $b_c = N$, and $a_i = b_{i-1} + 1$. In other words, a c-segmentation splits T to c disjoint time intervals by segment boundaries $s_1 < s_2 < \cdots < s_c$, where $S_i(s_{i-1} + 1, s_i)$.

Usually, the goal is to find homogeneous segments from a given time-series. In such cases, the segmentation problem can be defined as constrained clustering: data points should be grouped based on their similarity, but with the constraint that all points in a cluster must come from successive time points. In order to formalize this goal, a $cost(S(a, b))$ cost function with the internal homogeneity of individual

segments should be defined. The cost function can be any arbitrary function. For example, in [10,24] the sum of variances of the variables in the segment was defined as $cost(S(a, b))$. Usually, the $cost(S(a, b))$ cost function is defined based on the distances between the actual values of the time-series and the values given by a simple function (constant or linear function, or a polynomial of a higher but limited degree) fitted to the data of each segment. Hence, the optimal c-segmentation simultaneously determines the a_i, b_i borders of the segments and the θ_i parameter vectors of the models of the segments by minimizing the cost of c-segmentation which is usually the sum of the costs of the individual segments:

$$cost(S_T^c) = \sum_{i=1}^{c} cost(S_i)\,. \tag{1}$$

This cost function can be minimized by dynamic programming, which is computationally intractable for many real datasets [10]. Consequently, heuristic optimization techniques such as greedy top-down or bottom-up techniques are frequently used to find good but suboptimal c-segmentations [14,22]:

Sliding window: A segment is grown until it exceeds some error bound. The process repeats with the next data point not included in the newly approximated segment. For example, a linear model is fitted on the observed period and the modeling error is analyzed.

Top-down method: The time-series is recursively partitioned until some stopping criterion is met.

Bottom-up method: Starting from the finest possible approximation, segments are merged until some stopping criterion is met.

Search for inflection points: Searching for primitive episodes located between two inflection points.

In the following subsection a different approach will be presented based on the clustering of the data.

2.2. Fuzzy cluster-based fuzzy segmentation

When the variance of the segments are minimized during the segmentation, Eq. (1) results in the following equation:

$$cost(S_T^c) = \sum_{i=1}^{c} \sum_{k=s_{i-1}+1}^{s_i} \| \mathbf{x}_k - \mathbf{v}_i^x \|^2 = \sum_{i=1}^{c} \sum_{k=1}^{N} \beta_i(t_k) D^2(\mathbf{x}_k, \mathbf{v}_i^x), \tag{2}$$

where $D^2(\mathbf{x}_k, \mathbf{v}_i^x)$ represents the distance between the $\mathbf{v}_i^x$ mean of the variables in the ith segment (center of the ith cluster) and the $\mathbf{x}_k$ data point; and $\beta_i(t_k) \in \{0, 1\}$ stands for the crisp membership of the kth data point in the ith segment, and

$$\beta_i(t_k) = \begin{cases} 1 & \text{if } s_{i-1} < k \leqslant s_i\,, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

This equation is well comparable to the typical error measure of standard k-means clustering but in this case the clusters are limited to contiguous segments of the time-series instead of the Voronoi regions in R^n.

The changes of the variables of the time-series are usually vague and are not focused on any particular time point. As it is not practical to define crisp bounds of the segments, in this paper Gaussian membership

functions, $A_i(t_k)$, are used to represent the $\beta_i(t_k) \in [0, 1]$ fuzzy segments of a time-series:

$$A_i(t_k) = \exp\left(-\frac{1}{2}\frac{\left(t_k - v_i^t\right)^2}{\sigma_{i,t}^2}\right), \quad \beta_i(t_k) = \frac{A_i(t_k)}{\sum_{j=1}^{c} A_j(t_k)}. \tag{4}$$

For the identification of the v_i^t centers and $\sigma_{i,t}^2$ variances of the membership functions, a fuzzy clustering algorithm is introduced. The algorithm, which is similar to the modified GG clustering [3], assumes that the data can be effectively modeled as a mixture of multivariate (including time as a variable) Gaussian distribution, so it minimizes the sum of the weighted squared distances between the $\mathbf{z}_k = [t_k, \mathbf{x}_k^{\mathrm{T}}]^{\mathrm{T}}$ data points and the η_i cluster prototypes

$$J = \sum_{i=1}^{c}\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m D^2(\mathbf{z}_k, \eta_i), \tag{5}$$

where $\mu_{i,k}$ represents the degree of membership of the observation, $\mathbf{z}_k = [t_k, \mathbf{x}_k^{\mathrm{T}}]^{\mathrm{T}}$ is in the ith cluster ($i = 1, \ldots, c$) and $m \in [1, \infty)$ is a weighting exponent that determines the fuzziness of the resulting clusters (usually chosen as $m = 2$).

The GG clustering algorithm can be interpreted in a probabilistic framework, since the $D^2(\mathbf{z}_k, \eta_i)$ distance is inversely proportional to the probability that the $\mathbf{z}_k$ data point belongs to the ith cluster, $p(\mathbf{z}_k|\eta_i)$. The data are assumed to be normally distributed random variables with expected value $\mathbf{v}_i$ and covariance matrix $\mathbf{F}_i$. The GG clustering algorithm is equivalent to the identification of a mixture of Gaussians that represents the $p(\mathbf{z}_k|\eta)$ probability density function expanded in a sum over the c clusters

$$p(\mathbf{z}_k|\eta) = \sum_{i=1}^{c} p(\mathbf{z}_k|\eta_i)p(\eta_i), \tag{6}$$

where the $p(\mathbf{z}|\eta_i)$ distribution generated by the ith cluster is represented by the Gaussian function

$$p(\mathbf{z}|\eta_i) = \frac{1}{(2\pi)^{(n+1)/2}\sqrt{\det(\mathbf{F}_i)}} \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{v}_i)^{\mathrm{T}}(\mathbf{F}_i)^{-1}(\mathbf{z} - \mathbf{v}_i)\right) \tag{7}$$

and $p(\eta_i)$ is the unconditional cluster probability (normalized such that $\sum_{i=1}^{c} p(\eta_i) = 1$ holds), where η_i represents the parameters of the ith cluster, $\eta_i = \{p(\eta_i), \mathbf{v}_i, \mathbf{F}_i | i = 1, \ldots, c\}$.

Since the time variable is independent from the $\mathbf{x}_k$ variables, the proposed clustering algorithm is based on the following $D^2(\mathbf{z}_k, \eta_i)$ distance measure

$$\begin{aligned} p(\mathbf{z}_k|\eta_i) = \frac{1}{D^2(\mathbf{z}_k, \eta_i)} = \underbrace{\alpha_i}_{p(\eta_i)} \underbrace{\frac{1}{\sqrt{2\pi\sigma_{i,t}^2}} \exp\left(-\frac{1}{2}\frac{(t_k - v_i^t)^2}{\sigma_{i,t}^2}\right)}_{p(t_k|\eta_i)} \\ \times \underbrace{\frac{1}{(2\pi)^{\frac{r}{2}}\sqrt{\det(\mathbf{A}_i)}} \exp\left(-\frac{1}{2}(\mathbf{x}_k - \mathbf{v}_i^x)^{\mathrm{T}}\mathbf{A}_i^{-1}(\mathbf{x}_k - \mathbf{v}_i^x)\right)}_{p(\mathbf{x}_k|\eta_i)}, \end{aligned} \tag{8}$$

which consists of three terms. The first α_i term represents the a priori probability of the cluster, while the second represents the distance between the kth data point and the v_i^t center of the ith segment in time. The

third term represents the distance between the cluster prototype and the data in the feature space where $\mathbf{v}_i^x$ means the coordinate of the ith cluster center in the feature space and r is the rank of $\mathbf{A}_i$ distance norm corresponding to the ith cluster.

The proposed cluster prototype formulated by (8) is similar to that used by the GG clustering algorithm. However, it utilizes a different distance norm, $\mathbf{A}_i$. In the following section, it will be demonstrated how this norm can be based on the principal component analysis of the cluster.

2.3. PCA-based distance measure

The $\mathbf{A}_i$ distance norm can be defined in many ways. It is wise to select this norm to scale the variables so that those with greater variability do not dominate the clustering. One can scale by dividing by standard deviations, but a better procedure is to use statistical (Mahalanobis) distance, which also adjusts for the correlations among the variables. In this case, $\mathbf{A}_i$ is the fuzzy covariance matrix $\mathbf{A}_i = \mathbf{F}_i$, where

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m \left(\mathbf{x}_k - \mathbf{v}_i^x\right)\left(\mathbf{x}_k - \mathbf{v}_i^x\right)^{\mathrm{T}}}{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m}. \tag{9}$$

When the variables are highly correlated, the $\mathbf{F}_i$ covariance matrix can be ill-conditioned and cannot be inverted. Recently, two methods have been worked out to handle this problem [4]. The first method is based on fixing the ratio between the maximal and minimal eigenvalues of the covariance matrix. The second method is based on adding a scaled unity matrix to the calculated covariance matrix. Both methods result in invertible matrices, but neither of them extracts the potential information about the hidden structure of the data.

PCA is based on the projection of correlated high-dimensional data onto a hyperplane. This mapping uses only the first few q nonzero eigenvalues and the corresponding eigenvectors of the $\mathbf{F}_i = \mathbf{U}_i \Lambda_i \mathbf{U}_i^{\mathrm{T}}$, covariance matrix, decomposed to the Λ_i matrix that includes the eigenvalues $\lambda_{i,j}$ of $\mathbf{F}_i$ in its diagonal, in decreasing order, and to the $\mathbf{U}_i$ matrix that includes the eigenvectors corresponding to the eigenvalues in its columns. The vector $\mathbf{y}_{i,k} = \mathbf{W}_i^{-1}(\mathbf{x}_k) = \mathbf{W}_i^{\mathrm{T}}(\mathbf{x}_k)$ is a q-dimensional reduced representation of the observed vector $\mathbf{x}_k$, where the $\mathbf{W}_i$ weight matrix contains the q principal orthonormal axes in its column $\mathbf{W}_i = \mathbf{U}_{i,q} \Lambda_{i,q}^{1/2}$.

One limiting disadvantage of these definitions of PCA is the absence of an associated probability density or generative model which is required to compute $p(\mathbf{x}_k|\eta_i)$. Tipping and Bishop [23] developed a method called probabilistic principal component analysis (PPCA). In the PPCA the log-likelihood of the observing the data under this model is

$$\mathcal{L} = \sum_{k=1}^{N} \ln(p(\mathbf{x}_k|\eta_i)) = -\frac{N}{2}\left\{n \ln(2\pi) + \ln\left(\det(\mathbf{A}_i)\right) + tr(\mathbf{A}_i^{-1}\mathbf{F}_i)\right\}, \tag{10}$$

where $\mathbf{A}_i = \sigma_{i,x}^2 \mathbf{I} + \mathbf{W}_i \mathbf{W}_i^{\mathrm{T}}$ is the modified covariance matrix of the ith cluster which can be used to compute the $p(\mathbf{x}_k|\eta_i)$ probability. The log-likelihood is maximized when the columns of $\mathbf{W}_i$ span the principal subspace of the data. Tipping and Bishop proved that the only nonzero stationary points of the derivative of (10) with respect to $\mathbf{W}_i$ occur for

$$\mathbf{W}_i = \mathbf{U}_{i,q}\left(\Lambda_{i,q} - \sigma_{i,x}^2 \mathbf{I}\right)^{1/2} \mathbf{R}_i, \tag{11}$$

where $\mathbf{R}_i$ is an arbitrary $q \times q$ orthogonal rotation matrix and $\sigma^2_{i,x}$ is given by

$$\sigma^2_{i,x} = \frac{1}{n-q} \sum_{j=q+1}^{n} \lambda_{i,j}. \tag{12}$$

The algorithmic description of the EM approach to PPCA model is given in [23], but it can also be found in the following section, where the estimation of this model is incorporated into the clustering procedure.

2.4. Modified GG clustering for time-series segmentation

One of the most important advantages of PPCA models is that it allows their combination into mixture of models. Mixtures have been extensively used as models where data can be viewed as arising from several populations mixed in varying proportions, and EM is widely used to estimate the parameters of the components in a mixture [7]. The clusters obtained by GG clustering, also referred to fuzzy maximum-likelihood clustering, are multivariate Gaussian functions. The alternating optimization (AO) of these clusters is identical to the EM (maximum-likelihood estimation) identification of the mixture of these Gaussian models when the fuzzy weighting exponent $m = 2$ [6].

Similarly to GG clustering, in the proposed algorithm the optimal parameters of the $\eta_i = \{\mathbf{v}_i^x, \mathbf{A}_i, v_i^t, \sigma^2_{i,x}, \alpha_i\}$ cluster prototypes are determined by the minimization of the (5) functional subjected to the following constraints to get a fuzzy partitioning space,

$$\mathbf{U} = [\mu_{i,k}]_{c\times N} \,|\, \mu_{i,k} \in [0,1], \quad \forall i,k; \sum_{i=1}^{c} \mu_{i,k} = 1, \quad \forall k;\ 0 < \sum_{k=1}^{N} \mu_{i,k} < N \quad \forall i. \tag{13}$$

The most popular method to solve this problem is the AO, which consists of the application of Picard iteration through the first-order conditions for the stationary points of (5), which can be found by adjoining the constraints (13) to J by means of Lagrange multipliers [11]

$$\overline{J}(\mathbf{Z}, \mathbf{U}, \eta, \boldsymbol{\lambda}) = \sum_{i=1}^{c}\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m D^2(\mathbf{z}_k, \eta_i) + \sum_{k=1}^{N} \lambda_i \left(\sum_{i=1}^{c} \mu_{i,k} - 1\right) \tag{14}$$

and by setting the gradients of $\overline{J}$ with respect to $\mathbf{Z}$, $\mathbf{U}$, η and $\boldsymbol{\lambda}$ to zero.

This results in the following easily implementable algorithm.

Initialization. Given a time-series T specify c and q, choose a termination tolerance $\varepsilon > 0$, and initialize the values of $\mathbf{W}_i$, $\mathbf{v}_i^x$, $\sigma^2_{i,x}$, $\mu_{i,k}$.

Repeat for $l = 1, 2, \ldots$

Step 1: Calculate the η_i parameters of the clusters

- a priori probability of the cluster

$$\alpha_i = \frac{1}{N} \sum_{k=1}^{N} \mu_{i,k}, \tag{15}$$

- the cluster centers

$$\mathbf{v}_i^x = \frac{\sum_{k=1}^{N} (\mu_{i,k})^m \left(\mathbf{x}_k - \mathbf{W}_i \langle \mathbf{y}_{i,k} \rangle\right)}{\sum_{k=1}^{N} (\mu_{i,k})^m}, \tag{16}$$

where the expectation of the latent variables is $\langle \mathbf{y}_{i,k} \rangle = \mathbf{M}_i^{-1} \mathbf{W}_i^{\mathrm{T}} \left(\mathbf{x}_k - \mathbf{v}_i^x\right)$ and the $q \times q$ matrix $\mathbf{M}_i = \sigma_{i,x}^2 \mathbf{I} + \mathbf{W}_i^{\mathrm{T}} \mathbf{W}_i$.

- the new values of $\mathbf{W}_i$

$$\widetilde{\mathbf{W}}_i = \mathbf{F}_i \mathbf{W}_i \left(\sigma_{i,x}^2 \mathbf{I} + \mathbf{M}_i^{-1} \mathbf{W}_i^{\mathrm{T}} \mathbf{F}_i \mathbf{W}_i\right)^{-1}, \tag{17}$$

where $\mathbf{F}_i$ computed by (9).

- the new value of $\sigma_{i,x}^2$

$$\sigma_{i,x}^2 = \frac{1}{q} tr(\mathbf{F}_i - \mathbf{F}_i \mathbf{W}_i \mathbf{M}_i^{-1} \widetilde{\mathbf{W}}_i^{\mathrm{T}}), \tag{18}$$

- the distance norm ($n \times n$ matrix)

$$\mathbf{A}_i = \sigma_{i,x}^2 \mathbf{I} + \widetilde{\mathbf{W}}_i \widetilde{\mathbf{W}}_i^{\mathrm{T}}, \tag{19}$$

- the model parameters in time: the center and the standard deviation

$$v_i^t = \frac{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m t_k}{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m}, \quad \sigma_{i,t}^2 = \frac{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m \left(t_k - v_i^t\right)^2}{\sum_{k=1}^{N} \left(\mu_{i,k}\right)^m}. \tag{20}$$

Step 2: Compute the $D^2(\mathbf{z}_k, \eta_i)$ distance measures by (8).

Step 3: Update the partition matrix

$$\mu_{i,k}^{(l)} = \frac{1}{\sum_{j=1}^{c} \left(D(\mathbf{z}_k, \eta_i)/D(\mathbf{z}_k, \eta_j)\right)^{2/(m-1)}} \quad 1 \leqslant i \leqslant c,\ 1 \leqslant k \leqslant N. \tag{21}$$

until $||\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}|| < \varepsilon$.

The usefulness and accuracy of the algorithm depends on the right choice of the q number of principal components (PCs) and the c number of the segments. Hence, the crucial question of the usefulness of the proposed cluster algorithm is how these parameters can be determined in an automatic manner. This will be presented in the following two subsections.

2.5. Automatic determination of the number of the segments

In data mining, the bottom-up segmentation algorithm has been extensively used to support a variety of time-series data-mining tasks [14]. The algorithm starts by creating a fine approximation of the time-series, and iteratively merges the lowest cost pair of segments until a stopping criteria is met. For the automatic selection of the number of segments, a similar approach is proposed in this section. The proposed recursive cluster-merging technique evaluates the adjacent clusters for their compatibility (similarity) and merges the clusters that are found to be compatible. Then, after a proper initialization of the parameters

of the new cluster the clustering is performed again. During this merging and re-clustering procedure the number of clusters is gradually reduced, until an appropriate number of clusters is found. This procedure is controlled by a fuzzy decision-making algorithm based on the similarity between the PCA models.

2.5.1. Similarity of PCA models

The similarity of two PCA models (i.e. hyperplanes) can be calculated by the PCA similarity factor, S_{PCA}, developed by Krzanowski [17,21]. Consider two segments, S_i and S_j, of a dataset having the same n variables. Let the PCA models for S_i and S_j consist of q PCs each. The similarity between these subspaces is defined based on the sum of the squares of the cosines of the angles between each principal component of $\mathbf{U}_{i,q}$ and $\mathbf{U}_{j,q}$:

$$S_{PCA}^{i,j} = \frac{1}{q}\sum_{i=1}^{q}\sum_{j=1}^{q}\cos^2\,\theta_{i,j} = \frac{1}{q}tr\left(\mathbf{U}_{i,q}^{\mathrm{T}}\mathbf{U}_{j,q}\mathbf{U}_{j,q}^{\mathrm{T}}\mathbf{U}_{i,q}\right). \tag{22}$$

Because the $\mathbf{U}_{i,q}$ and $\mathbf{U}_{j,q}$ subspaces contain the q most important principal components that account for most of the variance in their corresponding datasets, $S_{PCA}^{i,j}$ is also a measure of similarity between the segments S_i and S_j.

Since the purpose of the segmentation is also to detect changes in the mean of the variables, it is not sufficient to compute only the $S_{PCA}^{i,j}$ similarity factor, but the distance among the cluster centers also has to be taken into account

$$D(\mathbf{v}_i^x, \mathbf{v}_j^x) = \|\mathbf{v}_i^x - \mathbf{v}_j^x\|. \tag{23}$$

Hence, the compatibility criterion has to consider the $c_{i,j}^1 = S_{PCA}^{i,j}$ and $c_{i,j}^2 = D(\mathbf{v}_i^x, \mathbf{v}_j^x)$ factors.

2.5.2. The decision-making algorithm

Because the compatibility criterion quantifies various aspects of the similarity of the clusters, the overall cluster compatibility should be obtained through an aggregation procedure. A fuzzy decision-making algorithm can be used for this purpose [12].

Compatibility criteria (22) and (23) are evaluated for each pair of clusters. The resulting compatibility matrix is descriptive concerning the structure of the whole time-series, e.g., repetitive motifs can be also detected from the analysis of the compatibility of the non-adjacent clusters. Following a fuzzy decision-making approach, the decision goals for each criterion have to be defined using a fuzzy set. Fig. 2 shows the triangular membership functions defined for the two criteria. The important parameters of the membership functions are the limits of their support, characterized by the v^1 knot point for parallelism and v^2 for closeness. The values of v^1 and v^2 are given by averaging compatibilities according to

$$v^1 = \frac{1}{c(c-1)}\sum_{i=1}^{c}\sum_{\substack{j=1\\ j\neq i}}^{c} c_{i,j}^1, \tag{24}$$

$$v^2 = \frac{1}{c(c-1)}\sum_{i=1}^{c}\sum_{\substack{j=1\\ j\neq i}}^{c} c_{i,j}^2. \tag{25}$$

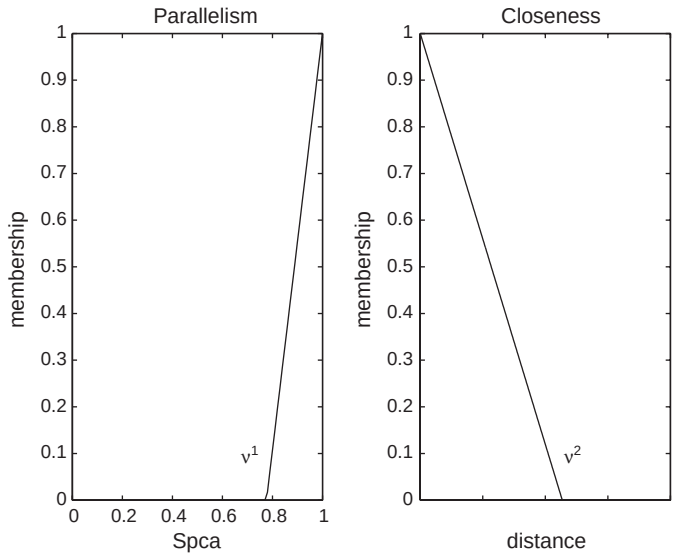

Fig. 2. Membership functions for parallelism and closeness of clusters.

Evaluating the membership functions with the values $c^1_{i,j}$ and $c^2_{i,j}$, one obtains the $\mu^1_{i,j}$ degree of parallelism and $\mu^2_{i,j}$ of closeness. The overall cluster compatibility is determined by the aggregation of the two criteria. A fuzzy aggregation operator is used for this purpose. The outcome of the decision procedure is the **O** overall compatibility matrix whose $O_{i,j}$ elements are given by

$$O_{i,j} = \left[\frac{(\mu^1_{i,j})^2 + (\mu^2_{i,j})^2}{2}\right]^{1/2}. \tag{26}$$

Given the **O** compatibility matrix, the clusters that will be merged must be identified and combined. Clusters can be merged in several ways. Our method merges the most similar pair of adjacent clusters as long as the value of the corresponding $O_{i,i+1}$ is above a threshold γ.

2.5.3. Cluster merging

The applied bottom-up strategy merges two adjacent clusters in each iteration. To preserve the information represented by the clusters, there is a need for a merging method that can directly compute the new initial parameters of the cluster from the merged clusters. This can be done by several ways (e.g., by Stroupe and Durrant–Whyte methods [19]). In this paper, the method developed by Kelly is applied [13].

The $\mathbf{v}^x_{i*}$ mean of the resulting cluster is computed from the individual cluster means

$$\mathbf{v}^x_{i*} = \frac{N_i}{N_{i*}}\mathbf{v}^x_i + \frac{N_j}{N_{i*}}\mathbf{v}^x_{i+1}, \tag{27}$$

where $N_i = \sum_{k=1}^{N} \mu_{i,k}$, $N_{i+1} = \sum_{k=1}^{N} \mu_{i+1,k}$ and $N_{i*} = N_i + N_{i+1}$, while the $\mathbf{F}_{i*}$ new covariance matrix is calculated from the $\mathbf{F}_i$ and $\mathbf{F}_{i+1}$ old covariance matrices as

$$\mathbf{F}_{i*} = \frac{N_i - 1}{N_{i*} - 1}\mathbf{F}_i + \frac{N_{i+1} - 1}{N_{i*} - 1}\mathbf{F}_{i+1} + \frac{N_i N_{i+1}}{N_{i*}(N_{i*} - 1)}[(\mathbf{v}_i^x - \mathbf{v}_{i+1}^x)(\mathbf{v}_i^x - \mathbf{v}_{i+1}^x)^{\mathrm{T}}]. \tag{28}$$

The new values of $\mathbf{W}_{i*}$ and $\sigma_{i*,x}^2$ can be computed by (11) and (12) based on the eigenvector–eigenvalue decomposition of $\mathbf{F}_{i*}$. Based on the obtained results, the $p(\mathbf{x}_k|\eta_{i*})$ probabilities can be easily computed (see the algorithm in Section 2.4).

This method can also be applied to merge the membership functions characterized by the parameters v_i^t, $\sigma_{i,t}^2$ and v_{i+1}^t, $\sigma_{i+1,t}^2$, and the $p(t_k|\eta_{i*})$ probabilities can be determined from the new values of v_{i*}^t and $\sigma_{i*,t}^2$ by (8). The unconditional probability of the $i*$th cluster is equal to the sum of the probabilities of the previous clusters $p(\eta_{i*}) = p(\eta_i) + p(\eta_{i+1})$. After the previously presented initialization of the new cluster prototype, the new $\mu_{i*,k}$ membership values can be computed by (8) and (21).

2.6. The number of principal components

Beside the selection of the right number of clusters, the second bottleneck of the successful application of the proposed algorithm is the selection of the right number of principal components. This can be done by the analysis of the eigenvalues of the covariance matrices of the initial segments. For this purpose a so-called screeplot can be drawn that plots the ordered eigenvalues according to their contribution to the variance of data. Another possibility is to define q based on the desired accuracy (loss of variance) of the PPCA models:

$$\sum_{j=1}^{q-1} \lambda_{i,j} / \sum_{j=1}^{n} \lambda_{i,j} < accuracy \leqslant \sum_{j=1}^{q} \lambda_{i,j} / \sum_{j=1}^{n} \lambda_{i,j}. \tag{29}$$

The synthetic dataset shown in Fig. 1 was initially partitioned into ten segments. As Fig. 3(a) illustrates, the magnitude of the eigenvalues and their cumulative rate to their sum show that two PCs are sufficient to approximate the distribution of the data with 98% accuracy. Obviously, this analysis can be fully automatized.

It is interesting to note that the analysis of the fuzzy hypervolume cluster validity measure is similar to the approach of the analysis of the eigenvalues, because the $V_i = (\det(\mathbf{F}_i))^{1/2}$ hypervolume of a cluster is proportional to the product of the eigenvalues. Hence, the right number of principal components can be also determined based on the product of the q largest eigenvalues.

2.7. The segmentation algorithm

Based on the previously presented building blocks, the proposed clustering-based segmentation algorithm is formulated as follows:

Step 1: Uniformly segment the data by a large number of segments (In the examples given in this paper 10 segments were used as starting point). Determine the q number of the principal components based on the analysis of the eigenvalues of these segments. For this purpose screeplot or cluster validity measure can be used (see Section 2.6 for more details).

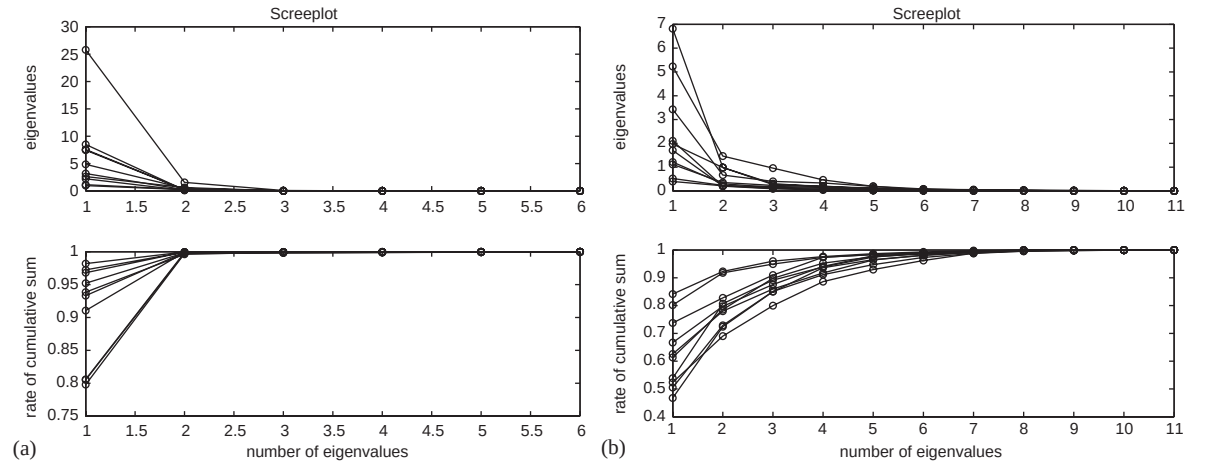

Fig. 3. Screeplots of the syntectic and the industrial data shown in Figs. 1 and 4, respectively: (a) Synthetic dataset and (b) industrial dataset.

Step 2: The values of m fuzziness parameter, the γ threshold for the $\mathbf{O}$ compatibility matrix and the ε termination tolerance must be chosen. In the case studies $m = 2, \varepsilon = 10^{-4}$, the value of γ is usually between 0.3 and 0.75 depending of the homogeneity of the time-series.

Step 3: Execute the clustering algorithm (see Section 2.4). The cluster merging must be evaluated after predefined number of iteration steps (see Section 2.5). In all of our applications this number was 100. The algorithm stops if the termination tolerance is reached *and* cluster merging is not necessary.

As this formulation of the algorithm shows, although there is a need to define some parameters of the algorithm before its application (γ, ε, and the initial number of clusters), it is possible to apply the method for high-dimensional time-series even if almost nothing is known about the structure of these series in advance. Hence, the method is useful for knowledge discovery. Of course, data mining is an iterative procedure. The results of the segmentation should be evaluated by human experts or by the performances of other modeling and data-mining tools based on the segmented data, and if it is needed, the "knowledge worker" should return to the segmentation task with a new set of these parameters.

3. Case studies

In this section the effectiveness of the developed algorithm will be illustrated by two examples: the synthetic dataset introduced in Section 1 and an industrial dataset taken from an industrial polymerization reactor, and the obtained results will be compared to the results given by the multivariate extension of the bottom-up segmentation algorithm of Keogh [14] (see the appendix).

3.1. Synthetic dataset

The synthetic dataset given in Fig. 1 is designed to illustrate how a multivariate segmentation algorithm should detect the changes of the latent process behind the high-dimensional data.

In Fig. 3 it has been shown that the screeplot of the eigenvalues suggests that the clustering algorithm should take into account two principal components. From Fig. 1(c)—which shows the β_i normalized and the $A_i(t) = p(t_k|\eta_i)$ Gaussian membership functions, and the $p(\mathbf{z}_k|\eta_i)$ probabilities—it can be seen that with this parameter the proposed method found five segments and it is able to detect the changes in the correlation structure and in the mean of the data. The $S_{PCA}^{i(i+1)}$ similarity measures of the adjacent clusters are 0.99, 0.17, 0.99, and 0.99812, which suggest that the correlation among the variables has significantly changed between the second and the third segments, while the other segments differ mainly in their mean. These results agree with Fig. 1(a) and justify the accuracy and the usefulness of the proposed method.

To illustrate the importance of the selection of the right number of PCs the same segmentation has been performed with only one PC. In this case, the algorithm found 10 nearly symmetric segments, hence it was not able to explore the hidden information behind the data. As it is depicted in Fig. 1(c), in case of five PCs the algorithm gave reasonable, but not so characteristic result.

These results were compared with the results of the bottom-up method based on the Hotelling T^2 (top) and the reconstruction error Q (bottom) shown in Fig. 1(d). The bottom-up algorithm based on the reconstruction error Q is sensitive to the change in the correlation structure but it was not able to find the change in the mean. The method based on the Hotelling T^2 measure is on the contrary. The method based on the Q measure is very sensitive to the number of PCs. As can be seen in Fig. 1(d) when $q = 2$ the result is very different from that obtained by $q = 5$, but in both cases the algorithm finds the change in the correlation structure.

This comparison showed that contrary to the multivariate extensions of the classical bottom-up segmentation algorithm, the developed cluster analysis-based segmentation algorithm can simultaneously handle the problems of the detection of the change of the latent process and the change of the mean of the variables and it is more robust with respect to the number of principal components.

3.2. Application to process monitoring

Manual process supervision relies heavily on visual monitoring of characteristic shapes of changes in process variables, especially their trends. Although humans are very good at visually detecting such patterns, it is a difficult problem for a control system software. Researchers with different background, for example from pattern recognition, digital signal processing and data mining, have contributed to the process trend analysis development [16,22,26].

The aim of this example is to show how the proposed algorithm is able to detect meaningful temporal shapes from multivariate historical process data. The monitoring of a medium and high-density polyethylene (MDPE, HDPE) plant is considered. The plant is operated by TVK Ltd. (http://www.tvk.hu), which is the largest Hungarian polymer production company in Hungary and produces raw materials for versatile plastics used for household goods, packaging, car parts, and pipe. An interesting problem with the process is that it requires the production about ten product grades according to the market demand. Hence, there is a clear need to minimize the time of changeover between the different products because off-specification product may be produced during the process transition. The difficulty of the analysis of the production and the transitions comes from the fact that there are more than 10 process variables that are needed to simultaneously monitor. Measurements are available in every 15 s on process variables $\mathbf{x}_k$, which are the polymer production intensity (PE), the inlet flowrates of hexene ($C6_{in}$), ethylene ($C2_{in}$), hydrogen ($H2_{in}$), the isobutane solvent (IB_{in}) and the catalyzator (Kat), the concentrations of ethylene

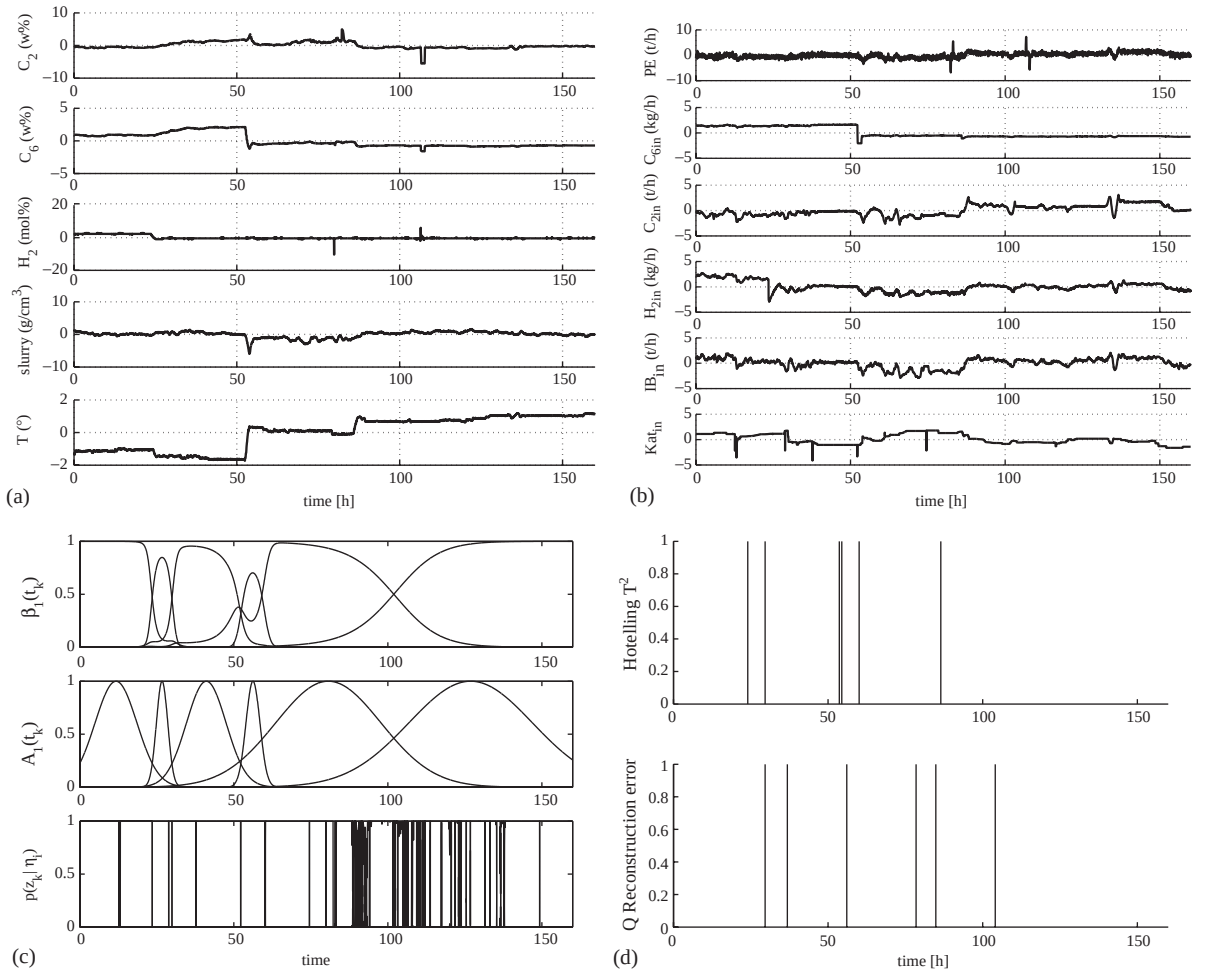

Fig. 4. Segmentation of the industrial dataset: (a) States of the reactor, (b) input variables of the reactor, (c) fuzzy segmentation results $q = 5$, $\gamma = 0.4$ and (d) bottom-up segmentation results.

(C_2), hexene (C_6), and hydrogen (H_2) and the slurry in the reactor (*slurry*), and the temperature of the reactor (T).

The dataset used in this example represents 160 h of operation and includes three product transitions around the 24, 54, and 86th hours. The initial number of the segments was ten and the γ threshold was chosen to $\gamma = 0.4$. In the Fig. 3 it can be seen that $q = 5$ principal components must be considered for 95% accuracy.

As Fig. 4 shows, both the bottom-up and the clustering-based algorithm are able to detect the product transitions, and all the three methods gave similar results. This reflects that in this realistic the mean and the covariance of the data were not independently changed. This is also confirmed by the analysis of the compatibilities of the adjacent clusters. As it can be seen, the product transitions are represented by two independent clusters, while the third transition was not so characteristic that it would require an

independent segment. This smooth transition between the third and the fourth products is also reflected by how the $p(\mathbf{z}_k|\eta_i)$ probabilities overlap between the 75–125th hour of operations. The changes of the $p(\mathbf{z}_k|\eta_i)$ probabilities around the 135th hour of operation are also informative, as the period of lower or drastically changing probabilities reflect some erroneous operation of the process. The results are similar if more than five principal components are taken into account.

This example illustrated that the proposed tool can be applied for the segmentation of a historical database and with the application of this tool useful information can be extracted concerning the changes of the operation regimes of the process and process faults. In the current state of our project we use this tool to compare the production of different products and extract homogenous segments of operation that can be used by a Kalman-filter-based state estimation algorithm for the identification of useful kinetic parameters and models which are able to predict the quality of the products [2].

4. Conclusions

This paper presented a new clustering algorithm for the fuzzy segmentation of large multivariate time-series. The algorithm is based on the simultaneous identification of fuzzy sets which represent the segments in time and the hyperplanes of local PCA models used to measure the homogeneity of the segments. The algorithm favors contiguous clusters in time and able to detect changes in the hidden structure of multivariate time-series. A fuzzy decision-making algorithm based on a compatibility criterion of the clusters have been worked out to determine the required number of segments, while the required number of principal components are determined by the screeplots of the eigenvalues of the fuzzy covariance matrices.

The results suggest that the proposed tool can be applied to extract useful information from temporal databases, e.g. the detected segments can be used to classify typical operational conditions and analyze product grade transitions.

Beside the industrial application example a synthetic dataset was analyzed to convince the readers about the usefulness of the method. Furthermore, the MATLAB code of the algorithm is available from our website (http://www.fmt.vein.hu/softcomp/segment), so the readers can easily test the proposed method on their own datasets.

The application of the identified fuzzy segments in intelligent query system designed for multivariate historical process databases is an interesting and useful idea for future research.

Acknowledgements

The authors would like to acknowledge the support of the Cooperative Research Center (VIKKK) (Project 2001-II-1), the Hungarian Ministry of Education (FKFP-0073/2001), and the Hungarian Science Foundation (T037600). Janos Abonyi is grateful for the financial support of the Janos Bolyai Research Fellowship of the Hungarian Academy of Science. The support of our industrial partners at TVK Ltd., especially Miklos Nemeth, Lorant Balint and dr. Gabor Nagy is gratefully acknowledged.

Table A.1
Bottom-up segmentation algorithm

- Create initial fine approximation.
- Find the cost of merging for each pair of segments:
  $mergecost(i) = cost(S(a_i, b_{i+1}))$
- while $min(mergecost) < maxerror$
  - Find the cheapest pair to merge: $i = argmin_i(mergecost(i))$.
  - Merge the two segments, update the a_i, b_i boundary indices, and recalculate the merge costs.
    $mergecost(i) = cost(S(a_i, b_{i+1}))$
    $mergecost(i-1) = cost(S(a_{i-1}, b_i))$

end

Appendix A. Appendix: bottom-up segmentation algorithm

In data mining, the bottom-up algorithm has been used extensively to support a variety of time-series data-mining tasks [14]. The algorithm begins creating a fine approximation of the time-series, and iteratively merge the lowest cost pair of segments until a stopping criteria is met. When the S_i and S_{i+1} pair of adjacent segments are merged, the cost of merging the new segment with its right neighbor and the cost of merging the S_{i-1} segment with its new larger neighbor must be calculated. The pseudocode of the algorithm is shown in Table A.1.

This algorithm is quite powerful since the merging cost evaluations requires simple identifications of PCA models which is easy to implement and computationally cheap to calculate. Because of this simplicity and because of the PCA defines linear hyperplane, the proposed approach can be considered as the multivariate extension of the piecewise linear approximation (PLA)-based time-series segmentation and analysis tools developed by Keogh [14,15].

Based on the PCA model the $cost(S_i)$ can be calculated in two ways. The cost can be equal to the reconstruction error of this segment

$$cost(S_i) = \frac{1}{b_i - a_i + 1} \sum_{k=a_i}^{b_i} Q_{i,k}, \tag{30}$$

where $Q_{i,k} = (\mathbf{x}_k - \hat{\mathbf{x}}_k)^{\mathrm{T}}(\mathbf{x}_k - \hat{\mathbf{x}}_k) = \mathbf{x}_k^{\mathrm{T}}(\mathbf{I} - \mathbf{U}_{i,p}\mathbf{U}_{i,p}^{\mathrm{T}})\mathbf{x}_k$ or the Hotelling T^2 measure:

$$cost(S_i) = \frac{1}{b_i - a_i + 1} \sum_{k=a_i}^{b_i} T_{i,k}^2 = \frac{1}{b_i - a_i + 1} \sum_{k=a_i}^{b_i} \mathbf{y}_{i,k}^{\mathrm{T}}\mathbf{y}_{i,k}. \tag{31}$$

References

[1] J. Abonyi, Fuzzy Model Identification for Control, Birkhauser, Boston, 2003.
[2] J. Abonyi, P. Arva, S. Nemeth, Cs. Vincze, B. Bodolai, Zs. Horvath, G. Nagy, M. Nemeth, Operator support system for multi product processes—application to polyethylene production, in: European Symp. Computer Aided Process Engineering,Lappeenranta, Finland, 2003, pp. 347–352.
[3] J. Abonyi, R. Babuska, F. Szeifert, Modified Gath–Geva fuzzy clustering for identification of Takagi–Sugeno fuzzy models, IEEE Trans. Systems Man Cybernet. 32 (5) (2002) 612–621.
[4] R. Babuška, P.J. van der Veen, U. Kaymak, Improved covariance estimation for Gustafson–Kessel clustering, IEEE Internat. Conf. Fuzzy Systems (2002) 1081–1085.

[5] J.F Baldwin, T.P. Martin, J.M. Rossiter, Time series modelling and prediction using fuzzy trend information, in: Proc. 5th Internat. Conf. on Soft Computing and Information Intelligent Systems,1998, pp. 499–502.
[6] J.C. Bezdek, J.C. Dunn, Optimal fuzzy partitions: a heuristic for estimating the parameters in a mixture of normal distributions, IEEE Trans. Comput. (1975) 835–838.
[7] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.
[8] K. Chakrabarti, S. Mehrotra, Local dimensionality reduction: a new approach to indexing high dimensional spaces, in: Proc. 26th VLDB Conf.,Cairo, Egypt, 1998, p. P089.
[9] A.B. Geva, Hierarchical-fuzzy clustering of temporal-patterns and its application for time-series prediction, Pattern Recognition Lett. 20 (1999) 1519–1532.
[10] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, H.T. Toivonen, Time-series segmentation for context recognition in mobile devices, in: IEEE Internat. Conf. on Data Mining (ICDM01),San Jose, CA, 2001, pp. 203–210.
[11] F. Hoppner, F. Klawonn, R. Kruse, T. Runkler, Fuzzy Cluster Analysis, Wiley, Chichester, 1999.
[12] U. Kaymak, R. Babuska, Compatible cluster merging for fuzzy modeling, in: Proc. IEEE Internat. Conf. on Fuzzy Systems,Yokohama, Japan, 1995, pp. 897–904.
[13] P.M. Kelly, An algorithm for merging hyperellipsoidal clusters, Technical Report LA-UR-94-3306, Los Alamos National Laboratory, Los Alamos, NM, 1994.
[14] E. Keogh, S. Chu, D. Hart, M. Pazzani, An online algorithm for segmenting time series. IEEE Internat. Conf. on Data Mining, http://citeseer.nj.nec.com/keogh01online.html, 2001.
[15] E. Keogh, M.J. Pazzani, An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback, in: 4th Int. Conf. on KDD,1998, pp. 239–243.
[16] S. Kivikunnas, Overview of process trend analysis methods and applications, ERUDIT Workshop on Applications in Pulp and Paper Industry, CD ROM, 1998.
[17] W.J. Krzanowsky, Between group comparison of principal components, J. Amer. Statist. Assoc. (1979) 703–707.
[18] M. Last, Y. Klein, A. Kandel, Knowledge discovery in time series databases, IEEE Trans. Systems Man Cybernet. 31 (1) (2000) 160–169.
[19] P. Marcelino, P. Nunes, P. Lima, M. I. Ribeiro, Improving object localization through sensor fusion applied to soccer robots, Actas do Encontro Cientifico do Robotica, 2003.
[20] S. Mitra, S.K. Pal, P. Mitra, Data mining in soft computing framework, IEEE Trans. Neural Networks 13 (1) (2002) 3–14.
[21] A. Singhal, E.E. Seborg, Matching patterns from historical data using PCA and distance similarity factors, in: Proc. American Control Conf.,2001, pp. 1759–1764.
[22] G. Stephanopoulos, C. Han, Intelligent systems in process engineering: a review, Comput. Chem. Eng. 20 (1996) 743–791.
[23] M.E. Tipping, C.M. Bishop, Mixtures of probabilistic principal component analyzers, Neural Comput. 11 (2) (1999) 443–482.
[24] K. Vasko, H.T.T. Toivonen, Estimating the number of segments in time series data using permutation tests, IEEE Internat. Conf. on Data Mining, 2002, pp. 466–473.
[25] X.Z. Wang, Data Mining and Knowledge Discovery for Process Monitoring and Control, Springer, Berlin, 1999.
[26] J.C. Wong, K. McDonald, A. Palazoglu, Classification of process trends based on fuzzified symbolic representation and hidden markov models, J. Process Control 8 (1998) 395–408.
[27] Y. Yamashita, Supervised learning for the analysis of the process operational data, Comput. Chem. Eng. 24 (2000) 471–474.
[28] J. Zhang, E.B. Martin, A.J. Morris, Process monitoring using non-linear statistical techniques, Chem. Eng. J. 67 (1997) 181–189.