Supervised Fuzzy Clustering for Rule Extraction

Magne Setnes

***Abstract*—This paper is concerned with the application of orthogonal transforms and fuzzy clustering to extract fuzzy rules from data. It is proposed to use the orthogonal least squares method to supervise the progress of the fuzzy clustering algorithm and remove clusters of less importance with respect to describing the data. Clustering takes place in the product space of systems inputs and outputs and each cluster corresponds to a fuzzy IF-THEN rule. By initializing the clustering with an overestimated number of clusters and subsequently remove less important ones as the clustering progresses, it is sought to obtain a suitable partition of the data in an automated manner. The approach is generally applicable to the fuzzy c-means and related algorithms. It is studied in this paper for adaptive distance norm fuzzy clustering and applied to the identification of Takagi–Sugeno type rules. Both a synthetic example as well as a real-world modeling problem are considered to illustrate the working and the applicability of the algorithm.**

***Index Terms*—Clustering methods, fuzzy systems, identification, modeling, transforms.**

I. Introduction

FUZZY rule-based models are often used to model systems in an input/output sense by means of IF-THEN rules. It is desirable that the rule base covers all the states of the system that are of importance for the considered application. At the same time, the number of rules should be kept low to increase the generalizing ability of the model, and to ensure a compact and transparent model.

Fuzzy rules can sometimes be obtained from human experts. Knowledge acquisition, however, is a cumbersome task, and for (partially) unknown systems, human experts are not available. Therefore, data-driven construction of fuzzy rules from measured input/output data has received a lot of attention. Such modeling approaches typically seek to optimize some numerical objective function, while less attention is paid to the complexity of the resulting model in terms of the number of rules [1]. Various methods have been proposed to balance the tradeoff between model accuracy and complexity, like entropy [2], genetic algorithms [3], [4], orthogonal transformation methods [5], [6], similarity measures [7], [8], and statistical information criteria [9], to mention a few.

This paper is concerned with rule extraction from data by means of fuzzy clustering in the product space of inputs and outputs where each cluster corresponds to a fuzzy IF-THEN rule [10], [11]. It is proposed to use the orthogonal least squares (OLS) method [5], [12] to remove redundant or less important clusters during the clustering process in order to extract fuzzy rules that capture the important features of the systems input/output state space in a compact and transparent rule base. By initializing the clustering with an overestimated number of clusters, there is an increased possibility that all the important regions in the data are covered, and the result becomes less dependent on the initialization. In [13] the idea was briefly introduced for the fuzzy c-means algorithm, assuming a zero-order Takagi–Sugeno (TS) fuzzy model. In this paper, the Gustafson–Kessel (GK) algorithm with adaptive distance measure [14] is considered, together with the more general TS fuzzy model, with functional rule consequents.

In the following, Section II describes the assumed fuzzy model type and how to determine its parameters from data. The identification by product space clustering is explained in Section III, while the OLS-based cluster reduction method and the supervised clustering algorithm are described in Section IV. The proposed approach is demonstrated in Section V with two examples. First, the reconstruction from data of a known rule-based system [15] is considered. It is shown that the algorithm successfully can detect the structure (premise) of the data generating rule base. In the second example the algorithm is applied to a real-world problem of modeling the pressure dynamics in a fermenter. The result is favorably compared to that of a trial-and-error approach with the standard GK algorithm. Finally, some concluding remarks are given in Section VI.

II. Fuzzy Modeling

A. Takagi–Sugeno Fuzzy Model

A fuzzy rule-based model suitable for the approximation of many systems and functions is the TS fuzzy model [16]. In the TS fuzzy model, the rule consequents are typically taken to be either crisp numbers or linear functions of the inputs

$$R_i\text{: IF } \boldsymbol{x} \text{ is } A_i \text{ THEN } y_i = \boldsymbol{a}_i^T \boldsymbol{x} + b_i, \quad i = 1, 2, \ldots, M \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the input variable (antecedent) and $y_i \in \mathbb{R}$ is the output (consequent) of the ith rule R_i. The number of rules is denoted by M and A_i is the (multivariate) antecedent fuzzy set of the ith rule

$$A_i(\boldsymbol{x})\text{: } \mathbb{R}^n \rightarrow [0, 1]. \tag{2}$$

In the case of univariate membership functions $\mu_{ij}(x_j)$ the fuzzy antecedent in the TS model is typically defined as an and-conjunction by means of the product operator

$$A_i(\boldsymbol{x}) = \prod_{j=1}^{n} \mu_{ij}(x_j). \tag{3}$$

Manuscript received December 27, 1999; revised March 10, 2000. This work was supported by the Research Council of Norway.

The author is with the Department of Electrical Engineering, Control Laboratory, Delft University of Technology, 2600 GA Delft, The Netherlands (e-mail: magne@ieee.org).

Publisher Item Identifier S 1063-6706(00)06590-5.

For the kth input $\boldsymbol{x}_k$ the total output $y(k)$ of the model is computed by aggregating the individual rules contributions

$$y(k) = \sum_{i=1}^{M} u_{ki} y_i(k) \tag{4}$$

where u_{ki} is the normalized degree of fulfillment of the antecedent clause of rule R_i

$$u_{ki} = \frac{A_i(\boldsymbol{x}_k)}{\sum_{i'=1}^{M} A_{i'}(\boldsymbol{x}_k)}. \tag{5}$$

B. Data Driven Identification

The TS model is identified in two steps. First, the fuzzy antecedents A_i in the rules are determined. The next section describes how this can be done using fuzzy clustering. In the second step, the rule antecedents are kept fixed, and least squares (LS) estimation from data is applied to determine the consequent parameters, $\boldsymbol{a}_i^T$ and b_i, of the rules. There are two main LS approaches. One is to solve M independent or local weighted LS problems—one for each rule. The other is to solve a global LS problem following from the aggregated output equation (4). Local LS gives more reliable local models, while global LS gives a minimal prediction error estimate [17]. In the following the global LS approach is followed.

Consider a collection of N input–output data pairs $\{\boldsymbol{x}_k, y_k\}$, $k = 1, 2, \cdots, N$ where $\boldsymbol{x}_k$ is the n dimensional input vector $[x_{1k}, x_{2k}, \cdots, x_{nk}]^T$ and y_k is to be approximated by the model given $\boldsymbol{x}_k$. Let X_e denote the matrix $[X, \mathbf{1}]$ with rows $[\boldsymbol{x}_k^T, 1]$. The activation of each rule R_i, $i = 1, 2, \cdots, M$ is gathered in Γ_i which is a diagonal matrix in $\mathbb{R}^{N\times N}$ having the normalized degree of fulfillment u_{ki} as its kth diagonal element. Further, denote X' the matrix in $\mathbb{R}^{N\times MN}$ composed from matrices obtained by multiplying the matrices Γ_i and X_e

$$X' = [\Gamma_1 X_e, \Gamma_2 X_e, \cdots, \Gamma_M X_e]. \tag{6}$$

Denote $\boldsymbol{\theta}'$ the vector in $\mathbb{R}^{M(n+1)}$ given by

$$\boldsymbol{\theta}' = \left[\boldsymbol{\theta}_1^T, \boldsymbol{\theta}_2^T, \cdots, \boldsymbol{\theta}_M^T\right]^T \tag{7}$$

where $\boldsymbol{\theta}_i^T = [\boldsymbol{a}_i^T, b_i]$ for $1 \le i \le M$. The model in (4) can now be written as a regression model

$$\boldsymbol{y} = X'\boldsymbol{\theta}' + \boldsymbol{e} \tag{8}$$

where $\boldsymbol{e}$ is the approximation error. From this, the least squares solution to the consequent parameter estimation problem can be written as

$$\boldsymbol{\theta}' = \left[(X')^T X'\right]^{-1} (X')^T \boldsymbol{y}. \tag{9}$$

III. Identification by Fuzzy Clustering

Fuzzy clustering methods partition a set of data into a number of overlapping clusters based on the distance in a metric space between the data points and the cluster prototypes. Various clustering algorithms can be used depending on the assumed structure of the identification data and the model type one wants to obtain [11]. A clustering method that has proven suitable for the identification of TS fuzzy models is the GK fuzzy clustering algorithm [14]. Unlike the popular fuzzy c-means algorithm [18], the GK algorithm employs an adaptive distance norm in order to detect clusters of different geometric shapes in the data set.

Each cluster in the product space of the input/output data represents a rule in the rule base. The goal is to establish the fuzzy antecedents A_i in the rules (2). These are defined by the fuzzy clusters found in the data. If desired, univariate membership functions μ_{ij} can be obtained by projections onto the various input variables x_j spanning the cluster space (for details, see, e.g., [10], [11]).

A. Fuzzy Partition

From the available input/output data pairs, the regression matrix X and the output vector $\boldsymbol{y}$ are constructed

$$X^T = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N], \quad \boldsymbol{y}^T = [y_1, \cdots, y_N] \tag{10}$$

where $N \gg n$ is the number of samples used for identification. The antecedent fuzzy sets A_i in (1) are determined by means of fuzzy clustering in the product space of the systems inputs and outputs. Hence, the data set $Z \in \mathbb{R}^{(n+1)\times N}$ to be clustered is represented as a $(n+1)\times N$ data matrix composed from X and $\boldsymbol{y}$:

$$Z^T = [X, \boldsymbol{y}] \tag{11}$$

where each column $\boldsymbol{z}_k$, $k = 1, 2, \cdots, N$ of Z contains an input/output data pair: $\boldsymbol{z}_k = [\boldsymbol{x}_k^T, y_k]^T$.

Given Z and an estimated number of clusters M, fuzzy clustering partitions Z into M fuzzy clusters. A fuzzy partition can be represented as an $N \times M$ matrix U, whose elements $u_{ki} \in [0, 1]$ represents the membership degree of $\boldsymbol{z}_k$ in cluster i. Hence, the ith column of U contains values of the ith *membership function* in the fuzzy partition, which is taken to be a pointwise representation of the antecedent fuzzy set A_i of the ith rule (1). The sum of each row of U is constrained to one, but the distribution of membership among the M fuzzy subsets is not constrained. Also, there can be no empty clusters and no cluster may contain all the objects. This means that the membership degrees in the partition matrix U are normalized, and for the given identification data, the membership values u_{ki} correspond to the normalized degree of fulfillment of the rule antecedents (5). Thus, the N membership values in the ith column $\boldsymbol{u}_i$ of the fuzzy partition matrix corresponds to those in the diagonal matrix Γ_i used in (6) to construct the regression matrix X' for the least-squares parameter estimation problem in (9). Thus, $\Gamma_i = \text{diag}(\boldsymbol{u}_i)$, where $\text{diag}(\boldsymbol{u}_i)$ denotes a diagonal matrix with the kth element u_{ki} of the vector $\boldsymbol{u}_i$ as the kth diagonal element.

B. Gustafson–Kessel Fuzzy Clustering

In adaptive distance norm clustering, each cluster has its own norm-inducing matrix D_i which is obtained from the covariance of the clusters (see Algorithm IV.2 for details). The distance of a data point $\boldsymbol{z}_k$ to a cluster center $\boldsymbol{v}_i$ is given by the inner-product norm

$$d_{ki}^2 = (\boldsymbol{z}_k - \boldsymbol{v}_i)^T D_i (\boldsymbol{z}_k - \boldsymbol{v}_i) \tag{12}$$

where $V = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_M]$ is a vector of *cluster prototypes* $\boldsymbol{v}_i \in \mathbb{R}^{n+1}$ which have to be determined. The GK fuzzy clus-

tering algorithm determines V based on the minimization of

$$J(X;\, U,\, V) = \sum_{i=1}^{M} \sum_{k=1}^{N} (u_{ki})^m\, d_{ki}^2 \tag{13}$$

where $m \in (1, \infty)$ is a weighting exponent which determines the fuzziness of the clusters. The minimization of (13) represents a nonlinear optimization problem, which is solved in an iterative manner [18]. The cluster algorithm stops when a predetermined stopping criterion is fulfilled (convergence).

IV. Cluster Reduction

An important issue in clustering is the determination of the relevant number of clusters in the data. Cluster validity techniques [19] have been proposed to assess the goodness of a given partition considering criteria like the compactness of the clusters and the distance between the clusters. A drawback of such methods is the need for repetitive clustering of the data using different number of clusters. Moreover, it can be difficult to choose a suitable validity measure from among the many measures proposed in the literature. Another approach is to use some kind of cluster merging method, like, e.g., the compatible cluster merging [20] or the extended fuzzy c-means method proposed in [21]. However, unlike the OLS-based cluster reduction described below, which considers the output contribution of each cluster, these methods only consider the structure of the partition. As such, they do not fully utilize the available output data for systems identification. In this section, first the OLS reduction algorithm is described. Then a stopping criterion for the algorithm is introduced and, finally, the supervised GK clustering algorithm with OLS-based reduction is summarized.

A. OLS Reduction Algorithm

An OLS rule reduction algorithm similar to the one proposed in [5], [12] is used to supervise the process of clustering. By initializing the clustering with an overestimated number of clusters, the selection of the most relevant number of clusters is automated by the OLS algorithm, which selects the clusters in decreasing order of importance in a forward regression manner by evaluating the contribution to the output energy by the corresponding rule.

Given a fuzzy partition matrix U obtained from clustering, the Gram–Schmidt OLS algorithm performs an orthogonal decomposition $U = WA$, where W is an orthogonal matrix and A is an upper-triangular matrix with unity diagonal elements. W is called the orthogonal basis of U. Following the approach in [5], [12], we substitute this orthogonal basis for U in order to determine the individual contributions of the rules. By using the ith column of W to construct a diagonal matrix W_i (i.e., $W_i = \text{diag}(\boldsymbol{w}_i)$) that replaces Γ_i in (6) we obtain $X^* = [(W_1 X_e), (W_2 X_e), \cdots, (W_M X_e)]$, and the corresponding regression problem in (8) becomes $\boldsymbol{y} = X^* \boldsymbol{\theta}^* + \boldsymbol{e}$, where $\boldsymbol{\theta}^* = [\boldsymbol{g}_1^T, \boldsymbol{g}_2^T, \cdots, \boldsymbol{g}_M^T]^T$ is the OLS equivalent of the solution vector in (7). The elements $\boldsymbol{g}_i$ of $\boldsymbol{\theta}^*$ can be determined one-by-one in the orthogonal space in order to calculate the output energy contribution of the corresponding rule. The OLS algorithm below is based on the one in [5], [12]. It does not decompose the complete matrix U, but selects the $M_S < M$ most dominant columns of U, corresponding to the most influential clusters (rules) in the model according to an estimated error reduction ratio.

Algorithm IV.1: OLS reduction algorithm

- Step 1: *Select the first vector w_1 of the orthogonal basis W*

  For $1 \le i \le M$, set $\boldsymbol{w}_1^{(i)} = \boldsymbol{u}_i$, where $\boldsymbol{u}_i = [u_{ki}, \cdots, u_{Ni}]^T$ is the ith column of the fuzzy partition matrix U, and construct the regression matrix $X_1^{*(i)} = [W_1^{(i)} X_e]$, where $W_1^{(i)} = \text{diag}(\boldsymbol{w}_1^{(i)})$.

  Calculate the corresponding element of the OLS solution vector

$$\boldsymbol{g}_1^{(i)} = \frac{\left(X_1^{*(i)}\right)^T \boldsymbol{y}}{\left(X_1^{*(i)}\right)^T X_1^{*(i)}}$$

  and the error-reduction ratio

$$[err]_1^{(i)} = \frac{\left(X_1^{*(i)} \boldsymbol{g}_1^{(i)}\right)^T \left(X_1^{*(i)} \boldsymbol{g}_1^{(i)}\right)}{\boldsymbol{y}^T \boldsymbol{y}}.$$

  Find the rule with the largest error reduction ratio

$$[err]_1^{(i_1)} = \max_{1 \le i \le M} \left([err]_1^{(i)}\right)$$

  and select the first basis vector $\boldsymbol{w}_1$ and the first elements $\boldsymbol{g}_1$ of the OLS solution vector

$$\begin{aligned} \boldsymbol{w}_1 &= \boldsymbol{w}_1^{(i_1)} = \boldsymbol{u}_{i_1}, \\ \boldsymbol{g}_1 &= \boldsymbol{g}_1^{(i_1)}. \end{aligned}$$

- Step 2: *Select the next basis vectors $\boldsymbol{w}_k$*

  Repeat for $2 \le k \le M_S$:

  For $1 \le i \le M$, $i \ne i_1, \cdots, i \ne i_{k-1}$, calculate

$$\begin{aligned} \boldsymbol{w}_k^{(i)} &= \boldsymbol{u}_i - \sum_{j=1}^{k-1} \frac{\boldsymbol{w}_j^T \boldsymbol{u}_i}{\boldsymbol{w}_j^T \boldsymbol{w}_j} \boldsymbol{w}_j,\ 1 \le j < k \\ W_k^{(i)} &= \text{diag}\left(\boldsymbol{w}_k^{(i)}\right) \\ X_k^{*(i)} &= \left[W_k^{(i)} X_e\right] \\ \boldsymbol{g}_k^{(i)} &= \frac{\left(X_k^{*(i)}\right)^T \boldsymbol{y}}{\left(X_k^{*(i)}\right)^T X_k^{*(i)}} \\ [err]_k^{(i)} &= \frac{\left(X_k^{*(i)} \boldsymbol{g}_k^{(i)}\right)^T \left(X_k^{*(i)} \boldsymbol{g}_k^{(i)}\right)}{\boldsymbol{y}^T \boldsymbol{y}}. \end{aligned}$$

Find the remaining rule with the largest error reduction ratio

$$[err]_k^{(i_k)} = \max_{1\le i\le M, i\neq i_1,\cdots,i\neq i_{k-1}} \left([err]_k^{(i)}\right)$$

and select the kth basis vector $\boldsymbol{w}_k$ and the kth element $\boldsymbol{g}_k$ of the OLS solution vector.

$$\boldsymbol{w}_k = \boldsymbol{w}_k^{(i_k)},$$
$$\boldsymbol{g}_k = \boldsymbol{g}_k^{(i_k)}.$$

- Step 3: *Remove rules (clusters)* Keep only the parameters of the M_S most significant rules. For the cluster algorithm, this means that only the corresponding M_S columns of U are retained in the remaining optimization.

B. Stopping Criterion for Cluster Selection

In each repetition k of step 2 the algorithm selects one of the remaining rules (clusters) based on the maximum $[err]_k^{(i)}$ value. This value represents the error-reduction ratio due to $\boldsymbol{w}_k^{(i)}$ [12] and is the part of the systems output variance that is explained by the corresponding rule. Hence, the most significant of the remaining rules is selected. The user must either pre-determine the number of steps M_S or define a stopping criterion for the selection. In [12] it was proposed to terminate at an unspecified M_Sth step when $1 - \sum_{j=1}^{M_S}[err]_j < \lambda$, where $\lambda \in [0, 1]$ is a chosen tolerance. This approximation accuracy criterion was also adopted in [5]. However, it is not always straight forward to determine *a priori* how well the fuzzy model is going to approximate the given data. Sometimes a high approximation accuracy is possible, but other times this need not be the case. To overcome this, we propose to use a criterion concerning the relative contribution of the rules. In this case, the algorithm is ended at an unspecified M_Sth step when the least important of the selected rules has a contribution to the error reduction less than $\rho\%$ compared to the previously selected rules

$$\frac{100\left(\sum_{j=1}^{M_S}[err]_j - \sum_{j=1}^{M_S-1}[err]_j\right)}{\sum_{j=1}^{M_S-1}[err]_j} < \rho\%, \quad \rho \in [0, 100]. \tag{14}$$

This criterion is more comprehensive as it is related to the relative contribution that each rule has to the approximation capabilities of the rule base containing the rules selected so far. There is thus no need to specify an approximation accuracy. Using this criterion, the algorithm will pick the rules needed to approximate the data, with the constraint that each rule has to have a certain relative contribution to the error reduction ratio of the rule base. This contribution is determined by the user and reflects the willingness to include detailed rules with a high level of specificity and little generality in the rule base. For a high value of ρ, fewer and more general and well separated rules are constructed (less clusters) than with a low value of ρ for the same data set. Threshold values in the range 10–20% have been found to give good results in many cases.

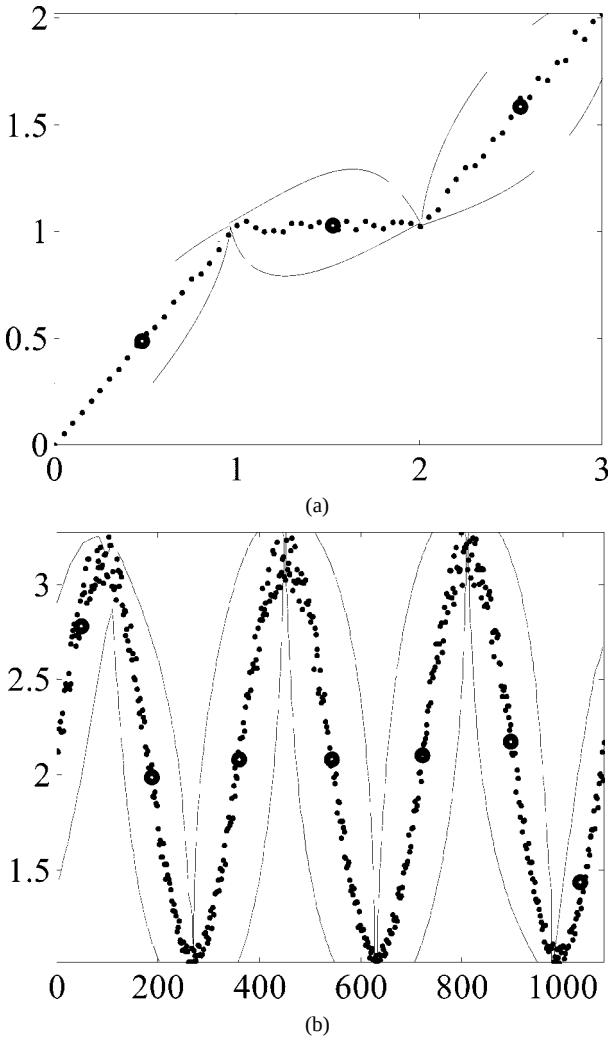

Fig. 1. Clustering results for two data sets of different complexity obtained using the proposed supervised cluster algorithm with $\rho = 10\%$ and fuzziness $m = 2$. Both cases were initialized with 20 random clusters. The data in the cluster space (dots), the cluster centers (thick circles), and the main contour lines of the partitions are shown. (a) Simple data with three lines. (b) Data from sinusoid function.

Using a given threshold ρ, for difficult to approximate data sets, requiring a high number of rules, the algorithm will retain a higher number of clusters than for data sets of lower complexity. This is illustrated in Fig. 1, which shows the result from the clustering of two data sets of different complexity. In both cases, the threshold value was $\rho = 10\%$, and the clustering was initialized with 20 random clusters. The results were obtained by the supervised cluster algorithm with reduction given in Algorithm IV.2 in the next section.

C. Cluster Algorithm with Reduction

The GK fuzzy clustering algorithm with OLS-based cluster reduction can now be written down. In this algorithm the progress is supervised by the OLS reduction algorithm presented above. When the clustering approaches convergence, the reduction algorithm evaluates the contribution of the various clusters and

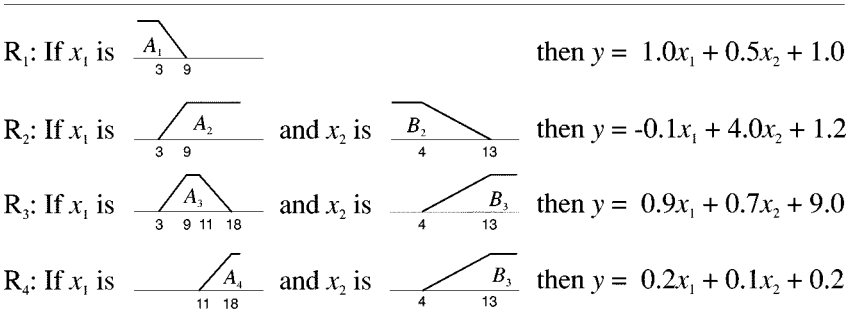

Fig. 2. Sugeno's rule-based system.

selects the most important ones for further optimization by the cluster algorithm such that the criterion in (14) is met.

Algorithm IV.2: Supervised cluster algorithm
Given the data Z, an initially overestimated number of clusters $1 < M < N$, the fuzziness parameter $m > 1$, the rule contribution threshold $\rho\%$, and the termination tolerance $\epsilon > 0$. Initialize $U^{(0)}$ randomly.
Repeat for $l = 1, 2, \cdots$
•Step 1: *Compute cluster prototypes:*

$$\boldsymbol{v}_i^{(l)} = \frac{\sum_{k=1}^{N} \left(u_{ki}^{(l-1)}\right)^m \boldsymbol{z}_k}{\sum_{k=1}^{N} \left(u_{ki}^{(l-1)}\right)^m}, \; 1 \le i \le M.$$

•Step 2: *Compute covariance matrices:*

$$F_i = \frac{\sum_{k=1}^{N} \left(u_{ki}^{(l-1)}\right)^m \left(\boldsymbol{z}_k - \boldsymbol{v}_i^{(l)}\right)\left(\boldsymbol{z}_k - \boldsymbol{v}_i^{(l)}\right)^T}{\sum_{k=1}^{N} \left(u_{ki}^{(l-1)}\right)^m}, \; 1 \le i \le M.$$

•Step 3: *Compute distances to cluster prototypes:*

$$d_{ki}^2 = \left(\boldsymbol{z}_k - \boldsymbol{v}_i^{(l)}\right)^T D_i \left(\boldsymbol{z}_k - \boldsymbol{v}_i^{(l)}\right), \; 1 \le i \le M, \; 1 \le k \le N$$

where the $D_i = [det(F_i)^{1/(n+1)} F_i^{-1}]$.
•Step 4: *Update the partition matrix:*
for $1 \le i \le M, 1 \le k \le N$
if $d_{ki} > 0$

$$u_{ki}^{(l)} = \frac{1}{\sum_{j=1}^{M} (d_{ki}/d_{kj})^{2/(m-1)}}$$

else if $d_{ki} = 0$

$$u_{ki}^{(l)} = 1.$$

•Step 5: *Run OLS reduction algorithm:*
if $\|U^{(l)} - U^{(l-1)}\| < 2\epsilon$ run OLS algorithm and keep only the the selected M_S clusters

$$M := M_s$$
$$U^{(l)} := [\boldsymbol{u}_i], \quad i = i_1, i_2, \cdots, i_{M_S}$$

normalize $U^{(l)}$
until $\|U^{(l)} - U^{(l-1)}\| < \epsilon$.

The proposed clustering algorithm differs from the standard GK algorithm [14] by the additional step 5. This step performs the cluster reduction by means of the OLS algorithm. The introduction of this additional step has no influence on the *convergence properties* of the clustering algorithm. If the OLS algorithm decides to remove one or more clusters, i.e., $M_S < M$ in step 5, the clustering will simply proceed without these clusters in the following iterations. This can be seen as a re-initialization of the cluster algorithm with the remaining M_S cluster centers.

The *convergence result* of fuzzy clustering is determined by the initialization [18]. In the proposed approach the probability that the most important regions in the data are covered by different clusters is increased by initializing with an overestimated number of clusters. It can be expected that as the less influential clusters are removed, the remaining M_S clusters are more likely to converge to a more suitable optimum than in the case of a random initialization of the standard GK algorithm with M_S clusters. This is illustrated in the next section.

V. Systems Identification Examples

Two problems are studied to illustrate the working of the proposed method and its applicability. The first example considers the reconstruction of a known rule base from data. The purpose is to identify a partition in the data that can be used to construct a rule-base premise that is confirm the premise partition of the data generating rule base. The second example deals with a real-world problem of modeling the pressure dynamics in a fed-batch bioreactor. Sampled data is used, and the goal is to obtain a rule-based model by clustering. The use of the supervised clustering algorithm is compared to a trial-and-error approach applying the standard GK clustering algorithm.

In all the experiments reported in this section, the fuzziness parameter $m = 2$ and the termination tolerance $\epsilon = 0.001$ were

TABLE I
FREQUENCY OF TERMINATION WITH THREE, FOUR OR FIVE CLUSTERS DEPENDING ON THE NUMBER OF INITIAL CLUSTERS. RESULTS ARE BASED ON 1000 RANDOMLY INITIALIZED TRIALS FOR EACH CASE

Initial:	4	5	6	7	8	9	10	11	12
Final=3	0%	26.4%	0%	9.9%	0.3%	13.6%	17.8%	0%	0.8%
Final=4	100%	73.6%	100%	83.1%	99.7%	79.5%	70.4%	100%	98.7%
Final=5	n.a.	0%	0%	7%	0%	6.9%	11.8%	0%	0.5%

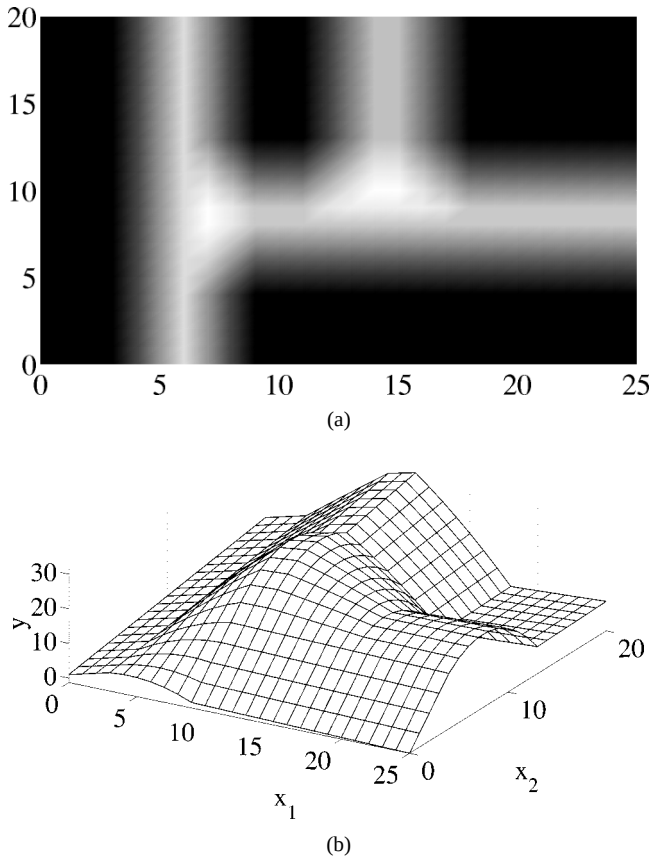

Fig. 3. (a) The partitioning of the premise space and (b) the output surface of the data generating rule-based system.

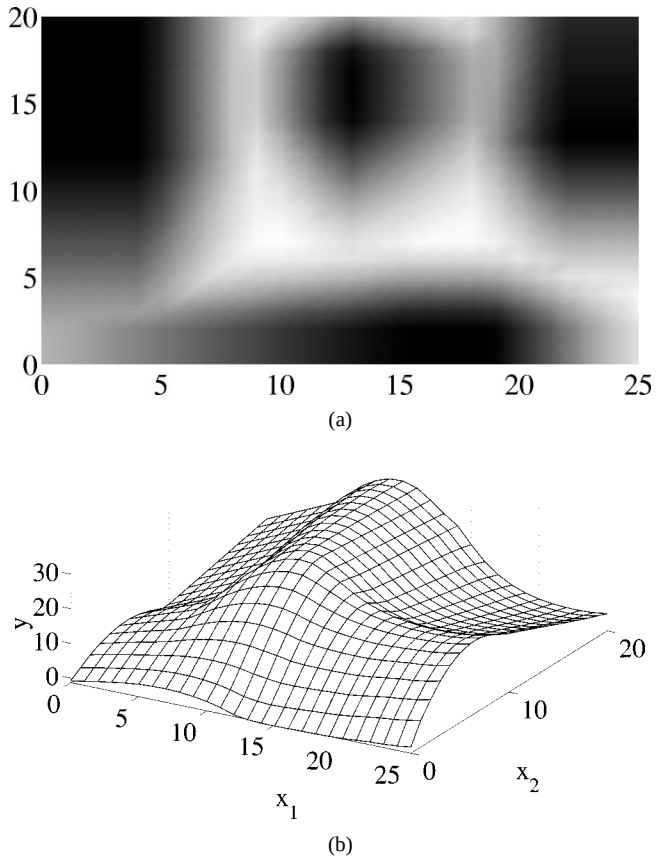

Fig. 4. (a) Premise of a TS rule-based model obtained by fitting trapezoid membership functions to the projections of the clusters and (b) the corresponding model surface.

used. For supervised clustering, the cluster selection threshold was kept at $\rho = 10\%$.

A. Modeling a Known Rule-Base

We consider the identification of the two input one output TS type rule-based system studied in [15]. The rule base consist of four rules as shown in Fig. 2. The premise partition and the input/output mapping of the rule base are shown in Fig. 3.

The systems surface was uniformly sampled with a small white noise disturbance. A total of $N = 546$ input–output observations $\boldsymbol{z}_k = [x_1, x_2, y]^T$ were gathered in a 3×546 pattern matrix Z. We know that the data generating system consist of four rules, but it is not straightforward to say how many clusters are actually present in the data Z. Both cluster validity measures and trial-and-error modeling indicate that from three to five clusters is a suitable choice. With more than five clusters, little improvement is gained in fitting the data and, with less than three clusters, the model output becomes unacceptable.

The supervised clustering algorithm was applied to the data several times with various number of clusters in its random initialization. It always converged with either three, four, or five clusters. The results obtained when initializing with various number of clusters are reported in Table I. For each case, the algorithm was run 1000 times with random initialization, and the table reports how frequent the algorithm converged with three, four, or five clusters.

From Table I we see that for most trials, the supervised clustering algorithm determines that there are four clusters in the data set corresponding to the four rules in the data generating rule base. The premise and the output of the reconstructed rule base with four rules are shown in Fig. 4. Here, the antecedent fuzzy sets were obtained by fitting trapezoid membership functions to the projected clusters [10], [11] and the parameters of the rule consequents were determined by least squares estimation as in (9).

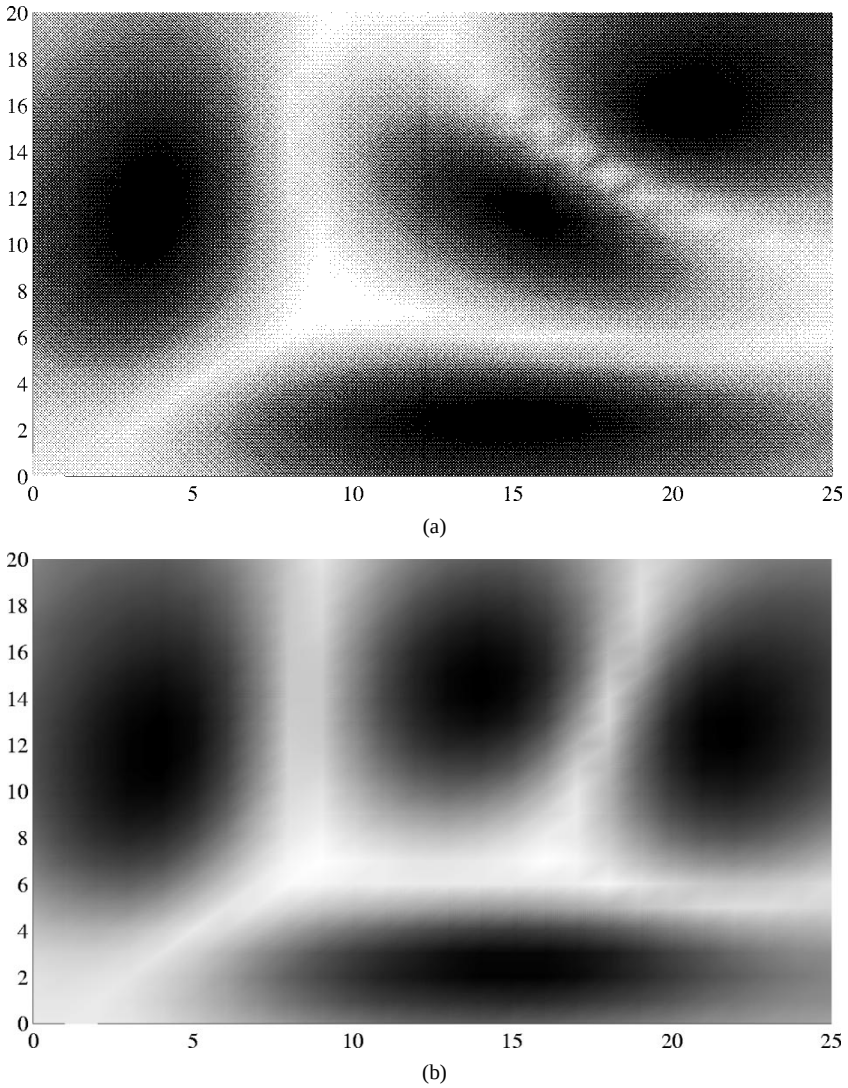

Fig. 5. Two typical results from GK clustering ($m = 2$ and $\epsilon = 0.001$). The random initialized GK algorithm with four clusters produced a suboptimal result in 45 out of 100 trials, while this happened only 14 times with the overinitialized supervised clustering algorithm.(a) Suboptimal clustering result. (b) Correct clustering result.

Determining a suitable number of clusters is not the only problem when applying clustering. The convergence result of a clustering algorithm is dependent on its initialization. For the studied data, when the standard GK clustering algorithm is applied with four clusters, the two most common results are the partitions shown in Fig. 5. The result in Fig. 5(a) is clearly less representative for the data generating rule base than the result in Fig. 5(b). When the standard GK algorithm is applied 100 times with four clusters, it converges 55 times to the correct solution and 45 times to the suboptimal solution. As discussed in the previous section, by initializing the supervised clustering algorithm with an overestimated number of clusters, the convergence result is expected to improve. This is verified by experiment and, out of 100 trials with 12 randomly initialized clusters, the supervised clustering algorithm converges 86 times to the correct solution and only 14 times to the suboptimal solution.

B. Modeling Pressure Dynamics

One of the variables that must be carefully controlled during a fermentation process is the pressure in the fermenter tank. We consider the fermenter illustrated in Fig. 6, where air is fed into the water at a constant flow-rate u_2 during fermentation. The head-space pressure y is controlled by the outlet valve u_1. A process model can be used to control the outlet valve, but also to enable detection of, e.g., valve failures or clogged filters.

The goal is to identify a fuzzy model of the pressure dynamics

$$y(k+1) = f(y(k), u_1(k)) \tag{15}$$

where $f(\cdot)$ is a fuzzy model of the TS type constituted by rules with a two-dimensional premise:

$$R_i\text{: IF } \boldsymbol{x}(k) \text{ is } A_i \text{ THEN } y(k+1) = \boldsymbol{a}_i^T \boldsymbol{x}(k) + b_i \tag{16}$$

where $i = 1, \cdots, M$ and $\boldsymbol{x}(k) = [y(k), u_1(k)]^T$. Both the number of rules M and the rules them selves have to be determined. Two approaches are considered for this purpose; first a trial-and-error approach is followed, then the proposed supervised clustering method is applied.

In the trial-and-error approach, the standard GK clustering algorithm is applied to the measured inputs and outputs $\boldsymbol{z}_k = [y(k), u_1(k), y(k+1)]^T$. Each cluster determines a rule in the

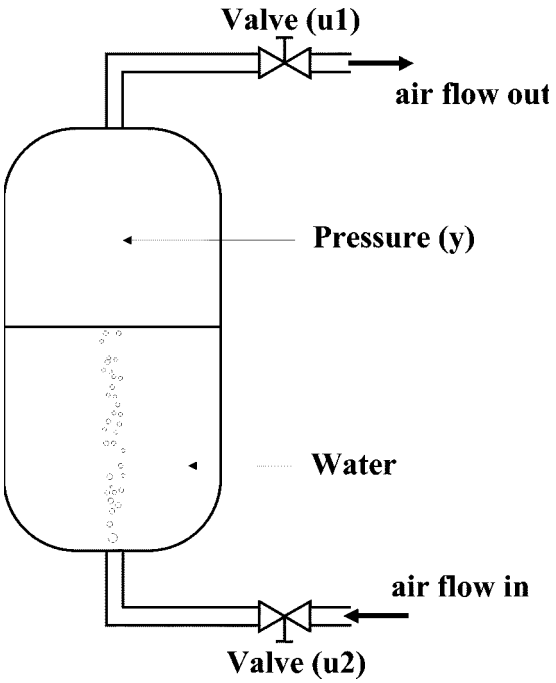

Fig. 6. Experimental setup of a laboratory fermenter tank.

rule base, and the parameters of the linear consequent functions are determined by a global least squares estimate as in (9). The used identification data is shown in Fig. 7(a). The modeling is repeated with the number of clusters varying from 2 to 12. For each case, clustering is repeated three times with random initialization in order to account for the dependency of the result on the initialization. Of the three resulting models, the one which best models the training data is recorded. This model is validated in a recursive simulation using the validation data shown in Fig. 7(b).

The results from the trial-and-error approach are summarized in Fig. 8(a). Typically, the MSE, in fitting the identification data, decreases as the number of clusters increases. The "learning-curve" starts to flatten around six clusters. The performance of the various models in simulating the validation data shows a decreasing trend until six clusters. With more than six clusters, the performance does not improve. Judging from the trial-and-error results, six clusters seems to be a suitable choice.

Now the supervised GK clustering algorithm proposed in Algorithm IV.2 is applied to the identification data. Since the algorithm is less dependent on the initialization, it is applied only once, starting with 12 randomly initialized clusters. The algorithm converges with six clusters whose cluster centers are similar to the ones found with trial-and-error. The validation of the resulting model is shown in Fig. 8(b).

The total computational costs of the trial-and-error approach; that is, the cost of clustering, consequent parameter estimation, evaluation on identification data, and validation in simulation of the selected models for 2–12 clusters, were about $708 \cdot 10^6$ FLOPS. For the approach using supervised clustering, the computational costs were about $66 \cdot 10^6$ FLOPS, including clustering, consequent parameter estimation, evaluation on identification data, and validation in simulation of the resulting model. For the considered example, the computational costs of the trial-and-error approach is thus more than ten times that of the approach based on supervised clustering.

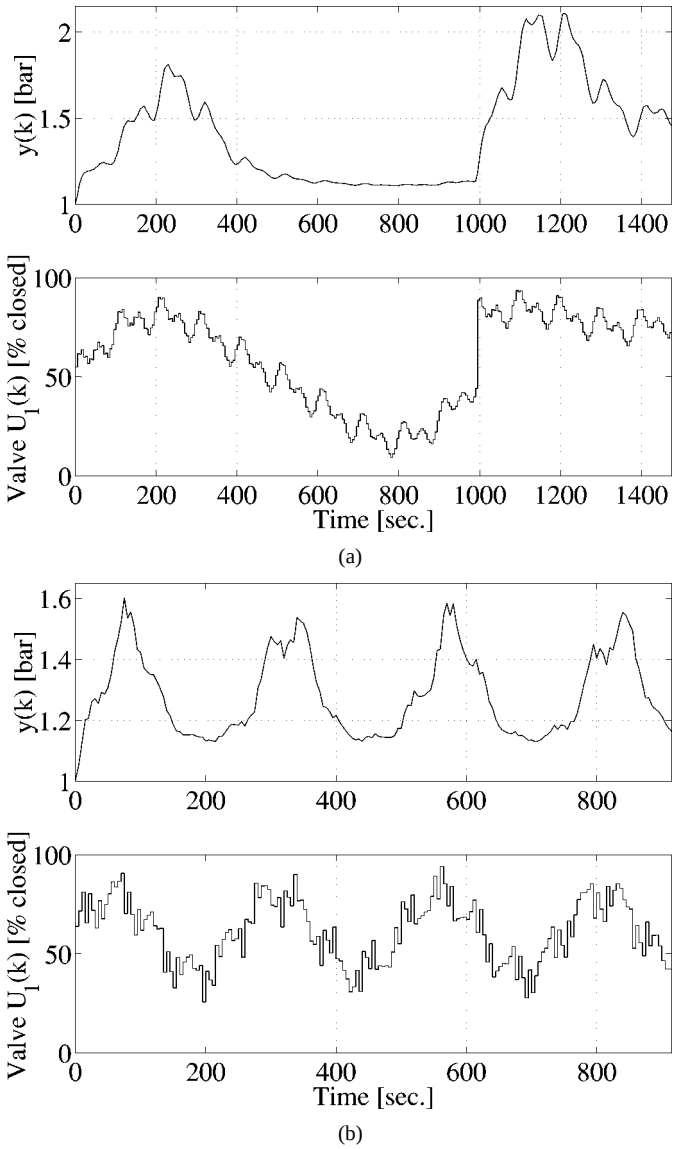

Fig. 7. Measured data used to (a) train and (b) validate the pressure dynamics model.

VI. Conclusion

A method to supervise the process of fuzzy clustering for rule extraction in order to detect and remove less important clusters has been presented. The reduction is based on the orthogonal least squares approach to subset selection presented in [12] and adopted for fuzzy clustering in this paper. The method is applicable for obtaining fuzzy rules from data for function approximation and systems modeling purposes. It helps the user in the difficult task of selecting an appropriate number of clusters when applying fuzzy clustering. The user is required to determine a relative lower threshold for the contribution of the rules that goes into the rule base. This threshold is used by the algorithm for selecting the appropriate number of clusters (rules) for the considered data. This parameter is transparent to the user and can easily be combined with other criteria, e.g., a minimum number of rules or an accuracy criterion.

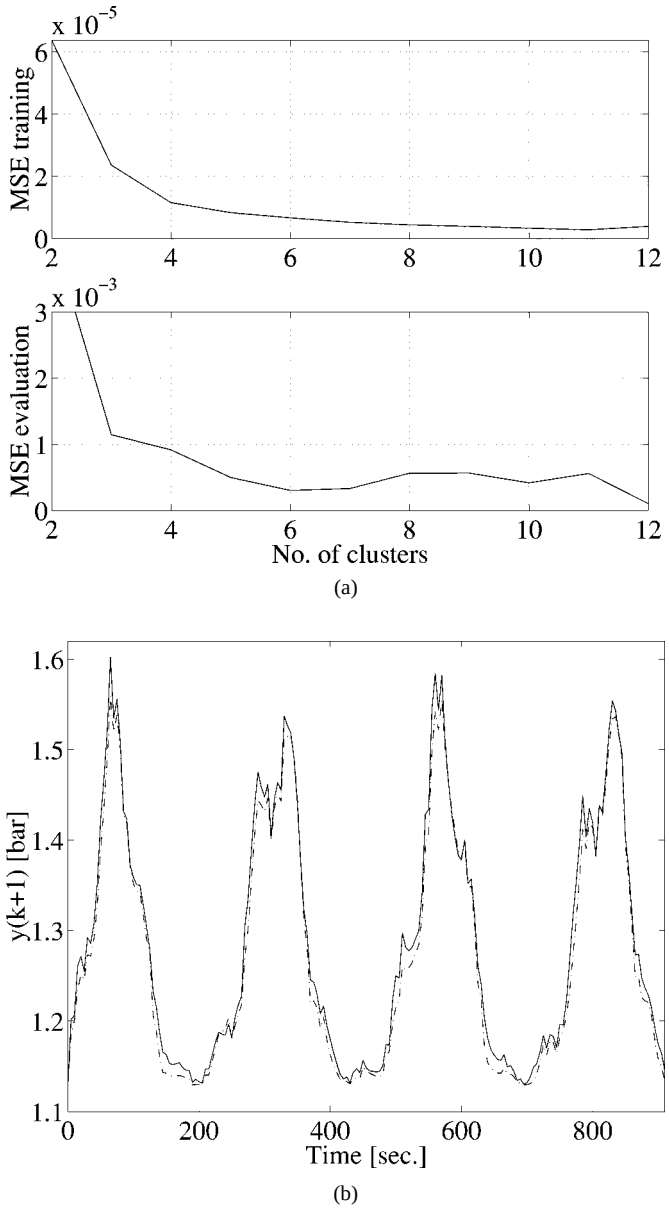

Fig. 8. (a) Results from modeling by trial-and-error and (b) the validation in a recursive simulation of a model (dash-dot line) based on six clusters determined by the supervised clustering algorithm. In (a), both the MSE in approximating the identification data and the error in validation by simulation is reported for various number of clusters.

When initialized with an overestimated number of clusters, the algorithm determines and keeps only the most important clusters. This overestimated initialization increases the possibility for the cluster algorithm to detect all the important regions of the data, thereby decreasing the dependency of the result on the (random) initialization.

The considered synthetic and real-world examples demonstrated the improved convergence properties due to the overestimated initialization and the algorithms capability of determining a suitable number of clusters in the data. In the real-world process modeling example, the proposed supervised algorithm proved more efficient than a trial-and-error approach.

ACKNOWLEDGMENT

The author would like to thank the anonymous referees for their constructive and detailed comments that helped improve the quality of this paper.

REFERENCES

[1] M. Setnes, R. Babuška, and H. B. Verbruggen, "Rule-based modeling: Precision and transparency," *IEEE Trans. Syst., Man, Cybern.*, pt. C, vol. 28, pp. 165–169, Feb. 1998.
[2] R. R. Yager and D. P. Filev, "Unified structure and parameter identification of fuzzy models," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 1198–1205, 1993.
[3] M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," in *Proc. FUZZ-IEEE/IFES'93*, San Francisco, CA, Mar. 1993, pp. 612–617.
[4] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy if–then rules for classification problems using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 260–270, Aug. 1995.
[5] L. X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Trans. Neural Networks*, vol. 3, pp. 807–813, Sept. 1992.
[6] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation methods," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 29, pp. 13–24, Feb. 1999.
[7] C. T. Chao, Y. J. Chen, and T. T. Teng, "Simplification of fuzzy-neural systems using similarity analysis," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 26, pp. 344–354, Apr. 1996.
[8] M. Setnes, R. Babuška, U. Kaymak, and H. R. van Nauta Lemke, "Similarity measures in fuzzy rule base simplification," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 28, pp. 376–386, June 1998.
[9] J. Yen and L. Wang, "Application of statistical information criteria for optimal fuzzy model construction," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 362–372, Aug. 1998.
[10] R. Babuška, *Fuzzy Modeling for Control*. Boston, MA: Kluwer , 1998.
[11] R. Kruse F, R. Höppner, F. Klawonn, and T. Runkler, *Fuzzy Cluster Analysis*. New York: Wiley, 1999.
[12] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 302–309, Mar. 1991.
[13] M. Setnes, "Supervised fuzzy clustering for rule extraction," in *Proc. FUZZ-IEEE'99*, Seoul, Korea, Aug. 1999, pp. 1270–1274.
[14] D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Proc. IEEE CDC*, San Diego, CA, 1979, pp. 761–766.
[15] M. Sugeno and G. T. Kang, "Structure identification of fuzzy model," *Fuzzy Sets Syst.*, vol. 28, pp. 15–33, 1988.
[16] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 116–132, Jan./Feb. 1985.
[17] J. Yen, L. Wang, and C. W. Gillespie, "Improving the interpretability of TSK fuzzy models by combining global learning and local learning," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 530–537, Nov. 1998.
[18] J. C. Bezdek, *Pattern Recognition With Fuzzy Objective Functions*. New York: Plenum, 1981.
[19] X. L. Xie and G. A. Beni, "Validity measure for fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 3, pp. 841–846, Aug. 1991.
[20] U. Kaymak and R. Babuška, "Compatible cluster merging for fuzzy modeling," in *Proc. FUZZ-IEEE/IFES'95*, Yokohama, Japan, Mar. 1995, pp. 897–904.
[21] M. Setnes and U. Kaymak, "Extended fuzzy c-means with volume prototypes and cluster merging," in *Proc. EUFIT'98*, Aachen, Germany, Sept. 1998, pp. 1360–1364.