ADAPTIVE FUZZY INFERENCE SYSTEM AND ITS APPLICATION IN MODELLING AND MODEL BASED CONTROL

János Abonyi , Lajos Nagy, Ferenc Szeifert
Department of Chemical Engineering Cybernetics
University of Veszprém,
Veszprém, H-8201, POB. 158, Hungary

This study presents an adaptation method for Sugeno fuzzy inference systems that maintain the readability and interpretability of the fuzzy model during and after the learning process. This approach can be used for modelling of dynamical systems and for building adaptive model-based control algorithms for chemical processes.

The gradient-descent based learning algorithm can be used on-line to form an adaptive fuzzy controller—this ability allows these controllers to be used in applications where the knowledge to control the process does not exist or the process is subject to changes in its dynamic characteristics. The proposed approach was applied in an internal model (IMC) fuzzy control structure based on the inversion of the fuzzy model. The adaptive fuzzy controller was applied in the control of a non-linear plant and is shown to be capable of providing good overall system performance.

Keywords: model identification; model based control; fuzzy modelling

INTRODUCTION

Chemical manufacturing processes present many challenging problems, including nonlinear dynamic behaviour, uncertain and time-varying parameters, unmeasured disturbances. Among several recent research efforts, fuzzy modelling and control has been widely applied to many chemical engineering problems[1,2,3].

During the past few years two principally different approaches to design of fuzzy controllers have emerged: heuristics-based design and model based design. The main motivation for the heuristics-based design is given by the fact that many industrial processes are still controlled in one of the following ways:

- The process is controlled manually by an experienced operator.

- The process is controlled by an automatic control system that needs manual, on-line "trimming" of their parameters by an experienced operator.

In spite of many practical successes, this type of fuzzy control still has some drawbacks. First, the fuzzy control rules are experience oriented and their parameters are obtained in many cases after time-consuming trial-and error experiments. Secondly, when some significant changes in the system occur and are outside the operator experience, re-tuning of the fuzzy controller is necessary.

These difficulties of heuristics-based approach explain the recent surge of interest in the model-based design of fuzzy controllers. This alternative approach uses fuzzy or inverse fuzzy models in process control, because it is much easier to obtain information on how a process responds to particular inputs than to record how, and why, an operator responds to particular situations[3].

The fuzzy models can be used as a means of capturing humans' expert knowledge about the process, in terms of fuzzy (if-then) rules like: **If** *the temperature is High* **Then** *close the stream valve a Large amount.*

The fuzzy inference system can initialise and learn linguistic and semi-linguistic (Sugeno) rules[4] – therefore it can be considered as direct transfer knowledge, which is the main advantage of fuzzy inference systems over classical learning systems and neural-networks.

This suggests that the fuzzy process models can be initialised by expert knowledge and can be adapted by the use of process data. Often the rules of the fuzzy system are designed a priori and the parameters of the membership functions are adapted in the learning process from input-output data sets[5,6]. In conventional adaptive fuzzy inference systems, the rules of the fuzzy model may lose their meanings that were initially assigned to them. Lotfi applied a constrained learning method for presenting the physical meaning of the rules[7].

This paper looks at a new method of fuzzy model adaptation, to maintain the interpretation of the adaptive fuzzy inference system during learning. The adapted fuzzy inference system can be applied in an internal model based controller (IMC)[8] structure. One case study based on simulation of a nonlinear process is used to illustrate this approach, and finally some conclusions are drawn.

THE SUGENO FUZZY MODEL

This paper deals with a Sugeno fuzzy model proposed by Takagi, Sugeno, and Kang[4] in an effort to develop a systematic approach to generating fuzzy rules from a given input-output data set. The fuzzy model can be formulated with a set of rules as follows:

$$r_{i_1,\ldots,i_N}: \text{ if } z_1 \text{ is } A_{1,i_1} \text{ and } \ldots \text{ and } z_N \text{ is } A_{N,i_N} \text{ then } y_m = f_{i_1,\ldots,i_N}(z_1,\ldots,z_N) \quad (1)$$

where $r_{i_1,\ldots,i_N}$ denotes the fuzzy rule, N is the number of inputs, $\mathbf{z} = [z_1,..,z_N]^T$ is a N vector containing all inputs of the fuzzy model. $A_{j,i_j}(z_j)$ is the $i_j = 1,2,\ldots,M_j$-th antecedent fuzzy set referring to the j - th input, whose membership functions are denoted by the same symbols as the fuzzy values, where M_j is the number of the fuzzy sets on the j-th input domain.

$f_{i_1,\ldots,i_N}$ is a consequent crisp function corresponding to the output of the fuzzy model, y_m. Usually $f_{i_1,\ldots,i_N}$ is a polynomial in the input variables, but it can be any function as long as it can appropriately describe the output of the system within the region specified by the antecedent of the rule. When $f_{i_1,\ldots,i_N}$ is a first order polynomial, the resulting fuzzy inference system is called first-order Takagi-Sugeno (T-S) fuzzy model. When $f_{i_1,\ldots,i_N}$ is a constant $f_{i_1,\ldots,i_N} = d_{i_1,\ldots,i_N}$ we have a zero-order Sugeno fuzzy model, which can be viewed as a special case of the linguistic fuzzy inference systems.

This paper deals with zero-order Takagi-Sugeno or so called product-sum crisp type fuzzy models.

Using fuzzy inference based upon product-sum-gravity[9] at a given input, $\mathbf{z}$, the final output of the fuzzy model, y_m, is inferred by taking the weighted average of the $d_{i_1,\ldots,i_N}$ ‘s:

$$y_m = \frac{\sum_{i_1=1,i_N=1}^{M_1,M_N} w_{i_1,\ldots,i_N} d_{i_1,\ldots,i_N}}{\sum_{i_1=1,i_N=1}^{M_1,M_N} w_{i_1,\ldots,i_N}} \quad (2)$$

where $\sum_{i_1=1,i_N=1}^{M_1,M_N} \equiv \sum_{i_1=1}^{M_1} \cdots \sum_{i_N=1}^{M_N}$ and $w_{i_1,\dots,i_N} > 0$ denotes the weights, imply the overall truth value of the $i_1,\dots,i_N$ –th rule calculated based on the degrees of membership values:

$$w_{i_1,\dots,i_N} = \prod_{j=1}^{N} A_{j,i_j}(z_j), \qquad j = 1,2,\dots,N, \quad i_j = 1,2,\dots,M_j \tag{3}$$

In this paper, we employ the triangular membership functions for each fuzzy linguistic value as shown in figure 1, where a_{j,i_j} denotes the cores of fuzzy set A_{j,i_j} :

$$a_{j,i_j} = core(A_{j,i_j}) = \left\{ \; z_j \,\middle|\, A_{j,i_j}(z_j) = 1 \; \right\} \tag{4}$$

The cores of the adjacent fuzzy sets determine the support of a set thus ensuring that the fuzzy sets on a universe of discourse always form a fuzzy partition, as well as keeping the sum of the membership degrees equal to 1. Constraints such as this are important in modelling or in adaptive fuzzy control since they help to obtain an interpretable and grid-type partitioning rule-base. Figure 2. illustrates a typical grid partition in a two-dimensional input space. The intersections of the grids divide the input domain mean the membership functions on the If-part of fuzzy rules. Thus, the membership functions are defined as follows:

$$\begin{aligned} A_{j,i_j}(z_j) &= \frac{z_j - a_{j,i_j-1}}{a_{j,i_j} - a_{j,i_j-1}}, \quad a_{j,i_j-1} < z_j < a_{j,i_j} \\ A_{j,i_j}(z_j) &= \frac{a_{j,i_j+1} - z_j}{a_{j,i_j+1} - a_{j,i_j}}, \quad a_{j,i_j} \le z_j < a_{j,i_j+1} \end{aligned} \tag{5}$$

Because the product operator is used for and- connective, the sum operator is used for aggregation and the fuzzy-mean method for deffuzzification, a complete rule base results:

$$\sum_{i_1=1,i_N=1}^{M_1,M_N} w_{i_1,\dots,i_N} = 1 \tag{6}$$

Therefore, at given observations, $z_j \in \left[a_{j,m_j}, a_{j,m_j+1}\right)$, (2) can be simplified:

$$y_m(k) = \sum_{i_1=m_1,i_N=m_N}^{m_1+1,m_N+1} \left[\left(\prod_{j=1}^{N} A_{j,i_j}(z_j) \right) \cdot d_{i_1,\dots,i_N} \right] \tag{7}$$

THE ADAPTATION METHOD

If a reliable set of process input-output data is available, it is possible to generate the fuzzy process model based on measured data. In this chapter, a learning rule determinates the cores of the fuzzy sets on the If-part and the consequent constants on the Then-part, is described. By using the proposed algorithm the interpretability and readability of the fuzzy system after (and during) the adaptation is maintained.

The tuning of the parameters used in fuzzy inference systems can be considered as a numerical optimisation procedure. Among the methods that have been implemented so far the gradient-descent (GD) adaptation method permits accurate learning of all parameters of the fuzzy model.

The GD adaptation rules are probably the best known supervised learning rules, producing a technique ideally suited to on-line instantaneous learning[5,6] . It is simple, it has low memory requirements and low computational cost, therefore it allows to built real-time learning algorithms.

The fuzzy controller is parameterised by the following parameters at the *k*th time instant:

$$\Theta(k) = \left\{ a_{j,i_j}, d_{i_1,\ldots,i_N}; \quad j = 1,2,\ldots,N; i_j = 1,2,\ldots,M_j \right\} \tag{8}$$

where M_j is the number of the antecedent fuzzy sets at the *j*-th input domain.

The purpose of the adaptation algorithm is to tune the parameters of the fuzzy model in order to minimise the prediction error at each sample instant between the model and the plant outputs.

Therefore, the GD method seeks to decrease the value of the quadratic objective function based on the instantaneous model output error:

$$E(k) = \frac{1}{2}\left(y_m(k) - y(k)\right)^2 \tag{9}$$

where $y(k)$ is the reference for the modelled output $y_m(k)$.

The parameter set, $\theta(k)$, of the fuzzy model is changed via the following iterative learning rule:

$$\Theta(k+1) = \Theta(k) + \Delta\Theta(k) = \Theta(k) - \alpha \frac{\partial E(k)}{\partial \Theta(k)} \tag{10}$$

where α is learning parameter, which controls how much the parameters are altered at each iteration.

The determination of the partial derivative of the modelling error $E(k)$ with respect to the parameters of the fuzzy model, $\theta(k)$,is given in the appendix.

The resulted learning rules can be expressed in the following form:

$$\Delta a_{j,m_j} = -\alpha \cdot \left(y_m(k) - y(k)\right)$$

$$\cdot \frac{A_{j,m_j}\left(z_j\right)}{a_{j,m_j+1} - a_{j,m_j}} \cdot \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left(\prod_{\substack{l=1\\ l\neq j}}^{N} A_{l,i_l}\left(z_l\right)\right)\left(d_{i_1,\ldots,i_j=m_j,\ldots,i_N} - d_{i_1,\ldots,i_j=m_j+1,\ldots,i_N}\right)\right] \quad (11.a)$$

$$\Delta a_{j,m_j+1} = -\alpha \cdot \left(y_m(k) - y(k)\right)$$

$$\cdot \frac{A_{j,m_j+1}\left(z_j\right)}{a_{j,m_j+1} - a_{j,m_j}} \cdot \left\{ \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left(\prod_{\substack{l=1\\ l\neq j}}^{N} A_{l,i_l}\left(z_l\right)\right) \cdot \left(d_{i_1,\ldots,i_j=m_j,\ldots,i_N} - d_{i_1,\ldots,i_j=m_j+1,\ldots,i_N}\right)\right]\right\} \quad (11.b)$$

$$\Delta d_{i_1,\ldots,i_N} = -\alpha \cdot \left(y_m(k) - y(k)\right) \cdot w_{i_1,\ldots,i_N} \quad (11.c)$$

Before the application of the proposed adaptation method, the fuzzy model has to be initialised.

This can be based on the expert's knowledge or based on measured input-output data.

In the latest case, the least-squares error method (LSE) can efficiently used to find the initial consequent singleton fuzzy sets that minimise the mean square output error (MSE), while the antecedent membership functions are equidistantly distributed over the input universes.

Assuming the given training data set has n_s entries, the MSE error criterion is defined by:

$$MSE = \frac{1}{n_s} \sum_{s=1}^{n_s} \left(y^s - y_m\right)^2 \quad (12)$$

where s denotes the $s=1,\ldots,n_s$ training data, $\left\{y^s, z_1^s, \ldots, z_N^s\right\}$.

Introducing matrix notation, the problem can be written as

$$Y = W \cdot D + \varepsilon \quad (13)$$

where ε denotes the modelling error, Y denote the vector having y_k as its kth components,

$$Y = \left[y^1 \ \cdots \ y^{n_s}\right]^T \quad (14)$$

W denote a matrix having the membersip degrees of each rule for each training data sample as elements.

$$W = \begin{bmatrix} w_{1,\ldots 1,1}^{1} & \cdots & w_{M_1,\ldots,M_N}^{1} \\ \vdots & \ddots & \vdots \\ w_{1_1\ldots,1,1}^{n_s} & \cdots & w_{M_1,\ldots,M_N}^{n_s} \end{bmatrix} \quad (15)$$

$$w^s_{i_1,\ldots,i_N} = \prod_{j=1}^{N} A_{j,i_j}\left(z^s_j\right). \tag{16}$$

The consequent parameters of the rule are concentrated into the *D* vector.

$$D = \left[d_{1,\ldots 1,1}, d_{1,\ldots 1,2} \;\; \cdots \;\; d_{M_1,\ldots,M_{N-1},M_N}\right]^T \tag{17}$$

The equation (13) represents the standard linear least-squares estimation problem. The best solution for *D* is that minimises $\left\| W \cdot D - Y \right\|^2$.

The numerical handling of the $D = \left(W^T W\right)^{-1} W^T Y$ expression could be difficult, because *W* is a rare matrix. Therefore the sequential least-squares method was used, where *D* can calculated iteratively as follows:

$$D_{i+1} = D_i + S_{i+1} \cdot W^T_{i+1}\left(y^{i+1} - W_{i+1} D_i\right) \tag{18}$$

$$S_{i+1} = S_i - \frac{S_{i+1} \cdot W^T_{i+1} \cdot W_{i+1} \cdot S_{i+1}}{1 + W_{i+1} \cdot S_i \cdot W^T_{i+1}}, \quad i = 0,1,\ldots,n_s - 1 \tag{19}$$

where W_i be the *i*th row vector of the matrix *W*.

The initial values of D_0 and S_0 are defined as $D_0 = 0$ and $S_0 = \gamma \cdot I$, respectively, where γ is a positive large number; I is the identity matrix of dimension $\prod_{i=1}^{N} M_i \times \prod_{i=1}^{N} M_i$.

After the last iteration the least-squares estimate of *D* is obtained, $D = D_{n_s}$.

The approach that uses LSE for initialisation is called *gradient descent and one pass of LSE* learning method[6]. The reason for this hybrid learning rule is the fact that the adaptation of the fuzzy sets parameters on the IF-part is a nonlinear optimisation problem, therefore the LSE cannot be used for that. However, for achieving a prescribed performance level by the tuning only of the consequent parameters, the LSE is much faster.

FUZZY DYNAMIC MODELLING

The fuzzy process model

The Nonlinear AutoRegressive with eXogenous input (NARX) model is frequently used with many nonlinear identification methods for dynamic modelling of chemical process systems. The fuzzy NARX model establishes a relation between the past input-output and the predicted output.

$$y_m(k+1) = g\left(y(k),\ldots,y(k-n_a+1),u(k),\ldots,u(k-n_b+1)\right) \quad (20)$$

where $y(k),\ldots,y(k-n_a+1)$ and $u(k),\ldots,u(k-n_b+1)$ are lagged process outputs and inputs respectively.

In order to represent a nonlinear single input - single output (SISO) process with Sugeno fuzzy model, the input vector, ***z***, of the fuzzy model has defined as follows:

$$\mathbf{z} = \left[y(k),\ldots,y(k-n_a+1),u(k),\ldots,u(k-n_b+1)\right] \quad (21)$$

The output of the fuzzy model is the one-step-ahead model prediction, $y_m(k+1)$ of the process output.

Application 1 – Identification of a model from Box-Jenkins data

The proposed fuzzy modelling approach was tested on the well-known Box-Jenkins furnace data[10] in order to be able to compare the modelling ability of the proposed fuzzy model with other methods. The learning data consists of 296 pairs of data taken from a laboratory furnace with a sampling time of 9 seconds. Each sample of data includes methane flow rate (process input), and the percentage of CO_2 in off gas (process output). From a number of studies, the best model structure for this system is:

$$y_m(k+1) = g\left(y(k),u(k-n_d+1)\right) \quad (22)$$

where the discrete time delay is, $n_d=4$.

The fuzzy model of the process can be formulated as follows:

$$r_{i_1,i_2}: \text{ if } y(k) \text{ is } A_{1,i_1} \text{ and } u(k-n_d+1) \text{ is } A_{2,i_2} \text{ then } y_m(k+1) = d_{i_1,i_2} \quad (23)$$

Table 1. compares the modelling accuracy of the fuzzy models derived using different numbers of fuzzy partitions after the initialisation and 200 iterations using free and constrained (*) adaptation.

By using constrained adaptation not only the interpretability and readability of the fuzzy inference system is maintained but also its monotonicity. This means that the modelled process is assumed to be invertible, therefore the fuzzy model has to make unique mapping between the inputs and the output. After the careful analyses of the data the following *regularisation rules* were founded: $d_{i_1+1,i_2} > d_{i_1,i_2}$ and $d_{i_1,i_2+1} < d_{i_1,i_2}$.

The development of regularization rules was the follow:

At given operating point, ***z***, the fuzzy model can be approximated by a local, linear first-order plus dead time ARX model:

$$y(k+1) = a \cdot y(k) + b \cdot u(k-n_d+1) + bias \quad (24)$$

The parameters of the local linear time invariant system (LTI) can be calculated by taking the partial derivatives of the fuzzy model in the given operating point, ***z***:

$$\frac{\partial y}{\partial z_t} = \sum_{i_1=m_1}^{m_1+1} \cdots \sum_{i_N=m_N}^{m_N+1} \left[\left(\frac{\Gamma_{i_t-1}(z_t)}{a_{t,i_t} - a_{t,i_t-1}} - \frac{\Gamma_{i_t}(z_t)}{a_{t,i_t+1} - a_{t,i_t}} \right) \cdot \prod_{\substack{j=1 \\ j \neq t}}^{N} A_{j,i_j}(z_j) \cdot d_{i_1,\ldots,i_N} \right] \tag{25}$$

$$\Gamma_{i_t} = \begin{cases} 1, & if \;\; z_t \in \left[a_{t,i_t}, a_{t,i_t+1}\right) \\ 0, & if \;\; z_t \notin \left[a_{t,i_t}, a_{t,i_t+1}\right) \end{cases} \tag{26}$$

By using the previous equations the parameters of the LTI models can be calculated as:

$$a = \frac{\partial f(\mathbf{x})}{\partial y(k)} = \left(\frac{d_{m_1+1,m_2} - d_{m_1,m_2}}{a_{1,m_1+1} - a_{1,m_1}} \right) \left(\frac{a_{2,m_2+1} - u(k-n_d+1)}{a_{2,m_2+1} - a_{2,m_2}} \right) + \left(\frac{d_{m_1+1,m_2+1} - d_{m_1,m_2+1}}{a_{1,m_1+1} - a_{1,m_1}} \right) \left(\frac{u(k-n_d+1) - a_{2,m_2}}{a_{2,m_2+1} - a_{2,m_2}} \right) \tag{27}$$

$$b = \frac{\partial f(\mathbf{x})}{\partial u(k-n_d+1)} = \left(\frac{a_{1,m_1+1} - y(k)}{a_{1,m_1+1} - a_{1,m_1}} \right) \left(\frac{d_{m_1,m_2+1} - d_{m_1,m_2}}{a_{2,m_2+1} - a_{2,m_2}} \right) + \left(\frac{y(k) - a_{1,m_1}}{a_{1,m_1+1} - a_{1,m_1}} \right) \left(\frac{d_{m_1+1,m_2+1} - d_{m_1+1,m_2}}{a_{2,m_2+1} - a_{2,m_2}} \right) \tag{28}$$

For first-order processes, the knowledge about the sign of the process gain, $K = \frac{b}{(1-a)}$, and time constant, $\tau = -\left(\frac{T_0}{\ln a}\right)$, can easily transformed into regularisation rules.

The analysis of the Box-Jenkins data showed, the process has a negative gain in the whole operating range. Therefore the regularisation rules become $d_{i_1+1,i_2} > d_{i_1,i_2}$ and $d_{i_1,i_2+1} < d_{i_1,i_2}$.

Constraints like this are important in modelling and model based control since they help to reduce the generalisation error of the fuzzy model and generate invertible process model, though they may cause loss of fitting performance.

If the model is validated in the same data set from which it was estimated the fit always improves as the model structure increases. There are several approaches to compensate for this automatic decrease of the fitting performance (MSE). In this study the selection of the best fuzzy model structure was carried out through the minimisation of Akaike's information Theoretic Criterion[11] (AIC).

The AIC is formed as

$$AIC = \ln\left[\left(1 + \frac{2 \cdot n_{par}}{n_s} \right) \cdot MSE \right] \tag{29}$$

where n_{par} and n_s is the number of the estimated parameters and the number of training data points respectively.

Applying AIC means compromise between the accuracy and complexity of the approximation.

Figure 3 and 4 shows the antecedent membership functions after adaptation and the prediction surface respectively in the case of the optimal fuzzy model (with 4 fuzzy sets).

Postlethwaite has proposed and compared fuzzy relational model identification algorithms using the examined Box-Jenkins furnace data[12] and achieved smallest MSE error (0.149) where 5 reference fuzzy sets are used by each variable. The main difference from this method that he used fixed fuzzy set (membership function) placements, while the proposed method adjusts both the fuzzy set positions and the rule consequents.

INDIRECT ADAPTIVE FUZZY MODEL BASED CONTROL

The vast majority of conventional control techniques have been devised for linear-time invariant systems. In most practical instances, however, the systems to be controlled are nonlinear and the basic physical processes in it are not completely known a priori. These types of model uncertainties and parameter changes are extremely difficult to manage even with the conventional adaptive techniques.

There are two distinct architectures, which have been formulated in the adaptive control field: direct and indirect schemes[14]. A direct learning control scheme learns the control law based on the performance measure, whereas an indirect scheme produces a model of the plant and synthesises the control law using a predefined optimisation/inversion calculation.

In this paper we present an indirect adaptive controller based on the inversion of the fuzzy model.

The Control Strategy

For the demonstration of the applicability of the proposed fuzzy inference systems and adaptation methods in a control application, internal model control (IMC) strategy was used.

The IMC structure consists of a controller, a process model and a process itself (Figure 5).

In internal model control arrangement, the process model is placed in parallel with a real system. The difference between the system and model output represents the modelling error and unmodelled process disturbances. This difference is then fed back into the controller where it is used to compensate disturbance and effects of the modelling error. The details of the IMC control strategy for linear systems can be found in the paper of Garcia and Morari[8]. The main conclusion of their paper is that for stable open loop processes one can find a perfect controller, which is essentially the inverse of the “minimum-phase” part of the plant. In addition it was proven that, under some assumptions, the overall system is stable and led to zero steady state offset. This approach, in the form of a single step predictor was extended to nonlinear systems by Economou et al[15], whose work was the basis of the application of feedforward neural networks in direct and indirect control strategy proposed by Psichogios and Ungar[16].

In the direct control approach the control action was explicitly calculated as the output of the controller (neural network or fuzzy model) that was trained to represent the inverse process dynamics. It was found, though, that the approximation error involved in learning the inverse dynamics significantly affects the controller’s performance.

An alternative way to calculate the control actions indirectly, based on the on-line inversion of the model describing the process dynamics. This poses certain problems since it is rather difficult to manipulate process models mathematically in neural-network or in relational fuzzy model form in order to obtain controller equation. Therefore, the further approaches are based on some numerical optimisation methods to solve this problem[3,16].

Many articles have recently been published about fruitful applications based on fuzzy model inversion. The first-order Takagi-Sugeno fuzzy models can be analytically inverted in the case when the output variable of the inverse model (controller) is not contained in the antecedent of the forward model (process model). For this type of models the fuzzy model inversion was applied in a speed control of a radial industrial cooling blast designed for air-conditioning of buildings[17]. The applied Sugeno fuzzy inference system can be mathematically inverted according to Babuska’s work[18] where the inversion algorithm was used to form a non-adaptive nonlinear controller for the pressure control of a simulated laboratory fermenter.

In this paper the proposed adaptation algorithm was used in order to form an indirect adaptive fuzzy model based controller.

The Controller Formulation

The nonlinear SISO process can be formulated with fuzzy if-then rules as follows:

$$\begin{aligned} r_{i_1,\ldots,i_{n_a+n_b}} &: \text{if } y(k) \text{ is } A_{1,i_1} \text{ and } \ldots \text{ and } y(k-n_a+1) \text{ is } A_{n_a,i_{n_a}} \text{ and} \\ & u(k) \text{ is } A_{n_a+1,i_{n_a+1}} \text{ and } \ldots \text{ and } u(k-n_b+1) \text{ is } A_{n_a+n_b,i_{n_a+n_b}} \\ & \text{then } y_m(k+1) = d_{i_1,\ldots,i_{n_a+n_b}} \end{aligned} \tag{30}$$

The fuzzy model can be described using these rules as well:

$$\begin{aligned} r_{i_1,\ldots,i_{n_a+n_b}} &: \text{if } y(k) \text{ is } A_{1,i_1} \text{ and } \ldots \text{ and } y(k-n_a+1) \text{ is } A_{n_a,i_{n_a}} \text{ and} \\ & u(k-1) \text{ is } A_{n_a+2,i_{n_a+2}} \text{ and } \ldots \text{ and } u(k-n_b+1) \text{ is } A_{n_a+n_b,i_{n_a+n_b}} \\ & \text{then } \left(\text{if } u(k) \text{ is } A_{n_a+1,i_{n_a+1}} \text{ then } y_m(k+1) = d_{i_1,\ldots,i_{n_a+n_b}} \right) \end{aligned} \tag{31}$$

Based on the previous form, at given state, $\boldsymbol{x}(k) = [y(k),\ldots,y(k-n_a+1),u(k-1),\ldots,u(k-n_b+1)]$, the model can be simplified to a SISO fuzzy model with using a partial deffuzzification method:

$$\tilde{r}_{i_{n_a+1}}: \quad \text{if } u(k) \text{ is } A_{n_a+1,i_{n_a+1}} \text{ then } y_m(k+1) = \tilde{d}_{i_{n_a+1}} \tag{32}$$

where the consequent parameter of the resulted fuzzy model is:

$$\tilde{d}_{i_{n_a+1}} = \sum_{i_1=m_1,i_{n_a}=m_{n_a}}^{m_1+1,m_{n_a}+1} \ \sum_{i_{n_a+2}=m_{n_a+2},i_N=n_a+n_b}^{m_{n_a+2}+1,m_N+1} \left(\prod_{\substack{l=1 \\ l \neq n_a+1}}^{N} A_{l,i_l}(z_l) \right) d_{i_1,\ldots,i_{n_a+n_b}} \tag{33}$$

Based on this simplified form the rules of the inverse fuzzy model:

$$\tilde{r}^{-1}{}_{i_{n_a+1}}: \quad \text{if } y_m(k+1) \text{ is } \tilde{D}_{i_{n_a+1}} \text{ then } u(k) = a_{n_a+1,i_{n_a+1}} \tag{34}$$

where $\tilde{D}_{i_{n_a+1}}$ is the antecedent membership function on the domain of the output of the fuzzy model:

$$\begin{aligned} \tilde{D}_{i_{n_a+1}}(y_m(k+1)) &= \frac{y_m(k+1) - \tilde{d}_{i_{n_a+1}-1}}{\tilde{d}_{i_{n_a+1}} - \tilde{d}_{i_{n_a+1}-1}}, \quad \tilde{d}_{i_{n_a+1}-1} < y_m(k+1) < \tilde{d}_{i_{n_a+1}} \\ \tilde{D}_{i_{n_a+1}}(y_m(k+1)) &= \frac{\tilde{d}_{i_{n_a+1}+1} - y_m(k+1)}{\tilde{d}_{i_{n_a+1}+1} - \tilde{d}_{i_{n_a+1}}}, \quad \tilde{d}_{i_{n_a+1}} \leq y_m(k+1) < \tilde{d}_{i_{n_a+1}+1} \end{aligned} \tag{35}$$

The proposed inverse fuzzy model can be used as a one-step-ahead controller:

$$\tilde{r}^{-1}{}_{i_{n_a+1}}: \quad \text{if } (sp(k+1) - e_f(k)) \text{ is } \tilde{D}_{i_{n_a+1}} \text{ then } u(k) = a_{n_a+1,i_{n_a+1}} \tag{36}$$

where *sp(k+1)* denotes the set-point at the *k+1*th time instant and e_f is the filtered modelling error. Symbolically the feed-back model error handling procedure is:

$$e_f(k) = e_f(k-1) \cdot (1 - K_F) + K_F \cdot e_m(k) , \tag{37}$$

where

$$e_m(k)=y(k)-y_m(k). \tag{38}$$

The procedure for designing and implementing the controller can be described in four steps:

Initialisation – off-line identification of the fuzzy process model

Adaptation – using the previous process input *u(k-1)* and previous process state, ***x****(k-1)*, the fuzzy model calculates $y_m(k)$ which is used to calculate the error of the fuzzy model by equation (38). The modelling error is used to adapt the fuzzy model with on-line gradient-descent method and to produce the feedback signal, $e_f(k)$.

Control – Based on the plant state, ***x****(k)*, and the reference signal, *sp(k+1)*, the feed-forward fuzzy controller produces the control command *u(k)* by inversion of the fuzzy model.

Application 2 – Simulation of liquid level control

In this application study, the controller was configured to carry out liquid level control on a simple simulation study identical to that used by Graham and Newel[19], Posthlethwaite[3,12], and Linkens et al.[13], to allow the comparative assessment of different versions of fuzzy controllers. The simulation is of the level of liquid in a tank with manipulated inflow and outflow, which is dependent on the square root of the level in the tank. The simulation model is just a single, non-linear, differential equation:

$$A\frac{dh}{dt} = F - c \cdot \sqrt{h} \tag{39}$$

where A (10 cm^2) is the cross-sectional area in the tank, h (0-100 cm) is the liquid level in the tank, F is the inlet flowrate, c is a flow coefficient (equal to 1).

The simulator was built using MATLAB^{TM}/Simulink. The controller output, *u(k)=F(k)*, the value of the inlet flowrate, was limited between 0 and 15 flow units.

The problem is investigated in set-point changes over two ranges: the first between 10 and 15 cm, and the second between 90 and 95 cm.

Since the system can be modelled as a first-order system, the following Sugeno fuzzy model structure was used:

$$IF\ \ h(k)\ \ is\ \ A_{i_1}\ \ and\ \ F(k)\ \ is\ \ A_{i_2}\ \ THEN\ \ h_m(k+1) = d_{i_1,i_2} \tag{40}$$

The initial parameters of the fuzzy model were determined with off-line least-squares identification by using 2500 data points. The training database was developed by forcing *F* with a uniformly distributed signal in the range of 0-15, because the conventional use of PRBS signals may cause loss in identifiabilily of the nonlinear system[20].

The liquid level was predicted one step into the future. A time step Δt =*10* sec was used. In order to maintain the invertibility of the first order fuzzy model with positive gain on the whole operating range, the following *regularisation* rules were used: $d_{i_1,i_2} < d_{i_1+1,i_2}$ and $d_{i_1,i_2} < d_{i_1,i_2+1}$.

The modelling results are shown in Figure 6 in case of 7 fuzzy sets on each input domain.

The dashed line corresponds to the modelled output and the solid line to measured (simulated) output. One can see that the fuzzy model gives excellent predictions.

In order to compare the control algorithms developed the Integral of the Absolute Value of the Error (IAE) performance index was used:

$$IAE = \sum_{k=1}^{N_t} |e(k)| \tag{41}$$

Where $e(k) = sp(k) - y(k)$ is the plan output error in the *k*-th discrete time step, $N_t = \frac{t_{\max}}{\Delta t}$ the maximal number of time steps (1000/10) and Δt =10 sec is the sampling time.

The controller has two tuning parameters: the adaptation factor, $\alpha \in [0,1]$, and the adjustable filter parameter, $K_F \in [0,1]$.

The α learning rate has great influence the speed of the convergence. If α is small, the gradient method will closely approximate the gradient path, but convergence will be slow since the gradient must be calculated several times. On the other hand if α is large, convergence will initially be very fast, but the algorithm will oscillate around the optimum. Based on this observations, in this simulation experiments α was set 0.01.

The feedback filter is introduced in order to filter out the measurement noise and minimise the instability introduced by the modelling error feedback. The parameters K_F should be determined such that an optimal compromise between performance and robustness is reached. Therefore, if the modelling error is small and there are no significant disturbances the filter parameter could be large. In this study K_F was set 0.9 in order to be handle the uniformly distributed noise in the range –1 to +1 added to the output of the system.

Table 2. shows the control performance of the adaptive fuzzy controller using different fuzzy partitioning after initialisation and five runs and compares with that achieved in previous studies under noise free conditions.

Figure 7.a and 7.b show the performance of the proposed fuzzy controller at the bottom and the top of the tank.

We achieved performance values of 150-114 and 109-102 which is better than PI and Takagi-Sugeno fuzzy model based predictive control developed by Linkens and Kandiah and RSK identified fuzzy relational model based control. The result is slightly worse than that of the off- line least-square identified fuzzy relational model based algorithm. This relational fuzzy model was also applied in IMC architecture. Since the relational fuzzy model could not have been directly inverted the authors solved the inverse problem by Fibonacci search, which demands more calculation than the fuzzy model inversion shown here.

The control algorithm was capable of handling effects of the change in process dynamics. Figure 8.a and 8.b show the process dynamics when the flow coefficient, *c*, and the area of the tank, *A*, changes from 1 and 10 to 1.5 and 15 respectively at the 300th second. The adaptation algorithm based on the gradient-descent method compensates the changes in process dynamics. In this case study the performance value decreases from 249 to 241.2 in noise free conditions by the implementation of the proposed adaptation method when 3 fuzzy sets were used on each input domain. The initial and final parameters of the fuzzy model are listed in table 3. The presence of noise has large effect to the identification and the control performance. As figure 8.b shows, the conservative setting of the adaptation factor, $\alpha = 0.01$, can compensate this effect. Under noisy conditions the control performance reduced from 397.3633 to 385.1837 by using the proposed adaptation algorithm.

CONCLUSIONS

This study presented an adaptation method for Sugeno fuzzy inference systems that maintains the readability and interpretability of the fuzzy model during and after the learning process. The linear parameters of the fuzzy model was initialised by the least-squares method. The gradient-descent based learning algorithm was used for the adaptive modelling of chemical processes and for building adaptive model-based control algorithms.

The proposed method was applied in an internal model (IMC) fuzzy control structure based on fuzzy model inversion.

A simulation example using an adaptive fuzzy model of a non-linear process is given in order to demonstrate the applicability of the proposed fuzzy model inversion based adaptive internal fuzzy control algorithm and is shown to be capable of providing good overall system performance.

APPENDIX

The adaptation algorithm

The iterative algorithm seeks to decrease the value of the objective function (7) by changing $\theta(k)$ via the following:

$$\Theta(k+1) = \Theta(k) + \Delta\Theta(k) = \Theta(k) - \alpha \frac{\partial E(k)}{\partial \Theta(k)} \tag{A.1}$$

where α is a constant which controls how much the parameters are altered at each iteration.

$\dfrac{\partial E(k)}{\partial \Theta(k)}$ can be expressed in the following form:

$$\frac{\partial E(k)}{\partial \Theta(k)} = \frac{\partial E(k)}{\partial y_m} \frac{\partial y_m}{\partial \Theta(k)} = \left(y_m(k) - y(k)\right) \frac{\partial y_m(k)}{\partial \Theta(k)} \tag{A.2}$$

This results in the problem having to determinate the partial derivatives of y_m to the parameters, $a_{j,i_j}, d_{i_1,\ldots,i_N}$, that are to be adapted.

At given observations, $z_j \in \left[a_{j,m_j}, a_{j,m_j+1}\right)$, the partial derivative of $y_m(k)$ to a_{j,i_j} is determined by:

$$\frac{\partial y_m}{\partial a_{j,m_j}} = \frac{\partial y_m}{\partial A_{j,m_j}(z_j)} \frac{\partial A_{j,m_j}(z_j)}{\partial a_{j,m_j}} + \frac{\partial y_m}{\partial A_{j,m_j+1}(z_j)} \frac{\partial A_{j,m_j+1}(z_j)}{\partial a_{j,m_j}}$$

$$\frac{\partial y_m}{\partial a_{j,m_j+1}} = \frac{\partial y_m}{\partial A_{j,m_j}(z_j)} \frac{\partial A_{j,m_j}(z_j)}{\partial a_{j,m_j+1}} + \frac{\partial y_m}{\partial A_{j,m_j+1}(z_j)} \frac{\partial A_{j,m_j+1}(z_j)}{\partial a_{j,m_j+1}}$$

$$\frac{\partial y_m}{\partial a_{j,i_j}} = 0 \qquad \text{if } i_j \neq m_j \text{ or } m_j + 1 \tag{A.3}$$

where:

$$\frac{\partial A_{j,m_j}(z_j)}{\partial a_{j,m_j}} = \frac{A_{j,m_j}(z_j)}{a_{j,m_j+1} - a_{j,m_j}}, \quad \frac{\partial A_{j,m_j}(z_j)}{\partial a_{j,m_j+1}} = \frac{A_{j,m_j+1}(z_j)}{a_{j,m_j+1} - a_{j,m_j}}$$

$$\frac{\partial A_{j,m_j+1}(z_j)}{\partial a_{j,m_j}} = \frac{-A_{j,m_j}(z_j)}{a_{j,m_j+1} - a_{j,m_j}}, \quad \frac{\partial A_{j,m_j+1}(z_j)}{\partial a_{j,m_j+1}} = \frac{-A_{j,m_j+1}(z_j)}{a_{j,m_j+1} - a_{j,m_j}} \tag{A.4}$$

Therefore:

$$\frac{\partial y_m}{\partial a_{j,m_j}} = \frac{A_{j,m_j}(z_j)}{a_{j,m_j+1} - a_{j,m_j}} \left(\frac{\partial y_m}{\partial A_{j,m_j}(z_j)} - \frac{\partial y_m}{\partial A_{j,m_j+1}(z_j)} \right)$$

$$\frac{\partial y_m}{\partial a_{j,m_j+1}} = \frac{A_{j,m_j+1}(z_j)}{a_{j,m_j+1} - a_{j,m_j}} \left(\frac{\partial y_m}{\partial A_{j,m_j}(z_j)} - \frac{\partial y_m}{\partial A_{j,m_j+1}(z_j)} \right) \tag{A.5}$$

The partial derivative of y_m to $A_{j,i_j}(z_j)$ is determined by:

$$\frac{\partial y_m}{\partial A_{j,m_j}(z_j)} = \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left(\prod_{\substack{l=1 \\ l \neq j}}^{N} A_{l,i_l}(z_l) \right) \cdot d_{i_1,\dots,i_j=m_j,\dots,i_N} \right]$$

$$\frac{\partial y_m}{\partial A_{j,m_j+1}(z_j)} = \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left(\prod_{\substack{l=1 \\ l \neq j}}^{N} A_{l,i_l}(z_l) \right) \cdot d_{i_1,\dots,i_j=m_j+1,\dots,i_N} \right] \tag{A.6}$$

The partial derivative of $y_m(k)$ to a_{j,i_j} is determined using (A.3-A.6):

$$\frac{\partial y_m}{\partial a_{j,m_j}} = \frac{A_{j,m_j}(z_j)}{a_{j,m_j+1} - a_{j,m_j}} \cdot \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left(\prod_{\substack{l=1 \\ l \neq j}}^{N} A_{l,i_l}(z_l) \right) \left(d_{i_1,\dots,i_j=m_j,\dots,i_N} - d_{i_1,\dots,i_j=m_j+1,\dots,i_N} \right) \right]$$

$$\frac{\partial y_m}{\partial a_{j,m_j+1}} = \frac{A_{j,m_j+1}(z_j)}{a_{j,m_j+1} - a_{j,m_j}} \cdot \left\{ \sum_{i_1=m_1, i_{j-1}=m_{j-1}}^{m_1+1, m_{j-1}+1} \sum_{i_{j+1}=m_{j+1}, i_N=m_N}^{m_{j+1}+1, m_N+1} \left[\left[\left(\prod_{\substack{l=1 \\ l \neq j}}^{N} A_{l,i_l}(z_l) \right) \cdot \left(d_{i_1,\dots,i_j=m_j,\dots,i_N} - d_{i_1,\dots,i_j=m_j+1,\dots,i_N} \right) \right] \right] \right\} \tag{A.7}$$

The partial derivative of y_m to $d_{i_1,\dots,i_N}$:

$$\frac{\partial y_m(k)}{\partial d_{i_1,\dots,i_N}} = w_{i_1,\dots,i_N} \tag{A.8}$$

NOMENCLATURE

A	cross-sectional area of the tank (1×10^{-3} m^2)
$A_{j,i_j}\left(z_j\right)$	$i_j = 1,2,...,M_j$ -th antecedent fuzzy set referring to the j - th input
a_{j,i_j}	cores of fuzzy set A_{j,i_j}
c	flow coefficient (equal to 1).
$d_{i_1,...,i_N}$	consequent crisp constant (singleton) on the $i_1,...,i_N$ th rule.
$\tilde{D}$	antecedent membership function in the inverse fuzzy model
e_f	filtered modelling error
e_m	modelling error
$f_{i_1,...,i_N}$	consequent crisp function on the $i_1,...,i_N$ th rule.
F	inlet flowrate (0-15 cm^3 s^{-1})
h	liquid level in the tank (0-100 cm)
$h_m(k+1)$	one-step ahead prediction of the liquid level using fuzzy process model at time instant k
k	time instant
$K_{F,}$	adjustable modelling error filter gain
M_j	number of the fuzzy sets on the j -th input domain
n_a and n_b	NARX model orders
n_d	discrete time delay
n_s	entries of the training data set
n_{par}	number of the fuzzy model's parameters
N	number of inputs of the fuzzy model
N_t	number of fuzzy model's parameters
$r_{i_1,...,i_N}$	$i_1,...,i_N$ th fuzzy implication (rule)
$\tilde{r}^{-1}$	fuzzy implication (rule) in the inverse fuzzy model
s	denotes the index of the training data, $\left\{y^s, z_1^s,..., z_N^s\right\}$.
$sp(k+1)$	set-point of the fuzzy controller at the time instant k
$u(k)$	control command at the time instant k

Δt	sampling time (10 sec)
y or y^s	reference output fuzzy model
y_m,	output of the fuzzy model
$z = [z_1,.., z_N]^T$	vector containing all inputs of the fuzzy controller
$w_{i_1,...,i_N}$	truth value of the $i_1,...,i_N$ –th rule

Greek letters

α	learning-rate
$\theta(k)$	the parameter vector of the fuzzy controller at time instant k

REFERENCES

1. Kraslawski, A. and Nyström, L., 1993, Fuzzy Sets. Introduction to Theory and Application in Process Engineering, *Chem. Eng. Technol.*,16: 291-295.
2. Babuska, R. and Verbruggen, H.B., 1996, An Overview of Fuzzy Modelling and Control, *Control Eng. Practice*, 4:44-51.
3. Posthlethwaite, B.,1994, A Model-based Fuzzy Controller, *Trans IChemE*, 72(A1): 38-46.
4. Takagi, T. and Sugeno, M., 1985, Fuzzy Identification of Systems and its Application to Modelling and Control, *IEEE Trans. on Systems, Man, and Cybernetics*, 15:116-132.
5. Guely, F. and Siarry, P., 1993, Gradient Descent Method for Optimising Various Fuzzy Rule Bases, *In Proceedings of the IEEE-Fuz-93*, 1241-1246.
6. J.- S. R. Jang, 1993, ANFIS: Adaptive-Network-based Fuzzy Inference Systems, *IEEE Transactions on Systems, Man and Cybernetics*, 23(3)**:** 665-685.
7. Lotfi, A., Andersen, H.C., and Tsoi, A.C., 1996, Interpretation Preservation of Adaptive Fuzzy Inference Systems, *International Journal Approximate Reasoning*, 15: 379-396.
8. Garcia, C. E. and Morari, M., 1982, Internal Model Control. 1 A Unifying Review and Some New Results, *Ind. Eng. Chem. Process Des. Dev.*, 21:308-323.

9. Driankov, D., Helerndoorn H., and Reinfrank M., 1993, *An Introduction to Fuzzy Control,* Springer-Verlag, Berlin.

10. Box, G.E.P., and Jenkins, G.M., 1970, *Time Series Analysis, Forecasting and Control,* Holden Gay, San Francisco

11. Ljung, L., 1987, *System Identification - Theory for the User,* Prentice-Hall, Englewood Cliffs, N.J.

12. Posthlethwaite, B. M., 1997, Brown. Sing C. H., A New Identification Algorithm for Fuzzy Relational Models and its Applications in Model-based Control, *Trans IChemE,* 75(A1):453-458.

13. Linkens, D.A. and Kandiah, S., 1996, Long-Range Predictive Control Using Fuzzy Process Models, *Trans IChemE,* 74(A1):77-88

14. Aström, K. J. and Wittenmark, B., 1989, *Adaptive Control,* Addison-Wesley, Reading, MA.

15. Economou, C.G.; et al., 1986, Internal Model Control. 5. Extension to Nonlinear Systems, *Ind. Eng. Chem. Process Des. Dev.,* 25:403-411.

16. Psichogios D.C. and Ungar, L.H., 1991, Direct and Indirect Model Based Control Using Artificial Neural Networks, *Ind. Eng. Chem. Process Des. Dev.,* 30:2564-2573.

17. Fisher, M. and Isermann R., 1998, Inverse Fuzzy Process Models for Robust Hybrid Control, In *Advances in Fuzzy Control*, Eds. (Driankov D. and Palm R.), 103-128, Physica-Verlag, Heilderberg, New York

18. Babuska R., 1997, *Fuzzy Modelling and Identification,* Ph.D. Thesis, Delft University of Technology, Delft.

19. Graham, P. and Newel, R. B., 1989, Fuzzy Adaptive Control of a First Order Process. *Fuzzy Sets and Systems,* 31:47-65.

20. Leontaritis, I. J., and Billings, S. A., 1987, Experimental Design and Identifiably for Non-linear Systems , *Int. J. Syst. Sci.* 18:189-202.

ACKNOWLEDGEMENTS

The financial support of the Hungarian Science Foundation (OTKA T023157) and the Hungarian Ministry of Culture and Education PFP-VE-MK-3011-1997 is greatly appreciated.

ADDRESS

Correspondence concerning this paper should be addressed to Dr. F. Szeifert, Department of Chemical Engineering Cybernetics, University of Veszprém, 10 Egyetem Street, Veszprém, P.O.Box 158, Hungary.

The manuscript was received …

LIST OF FIGURES AND TABLES

FIGURES

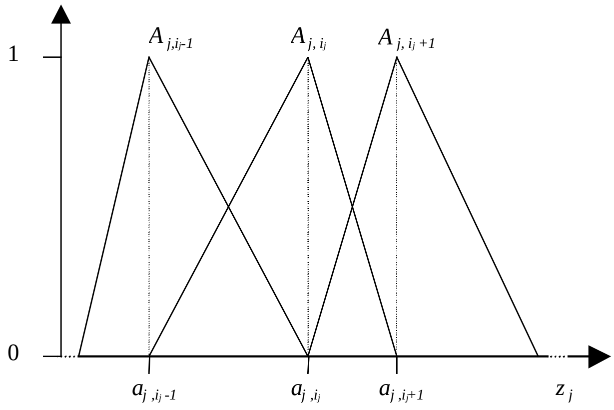


Figure 1. Membership functions used for the fuzzy model

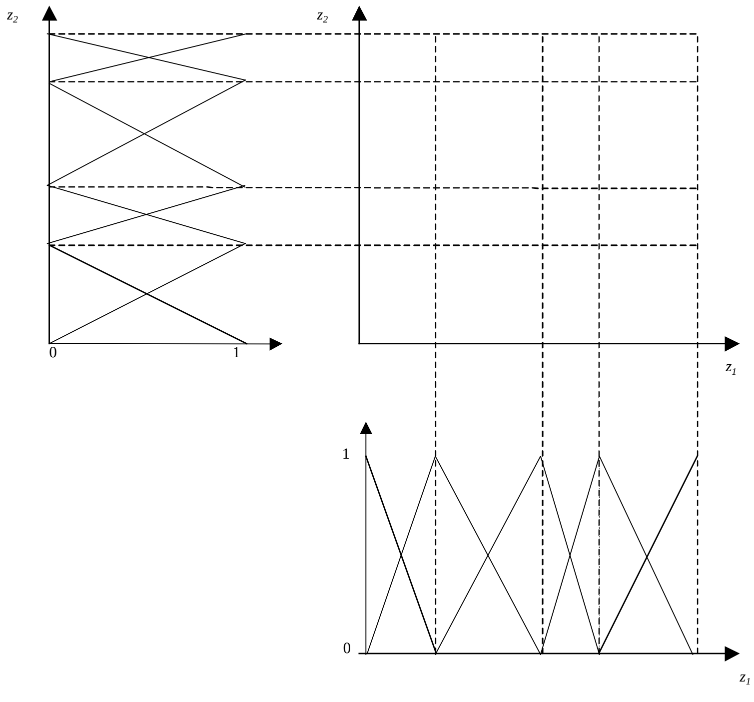
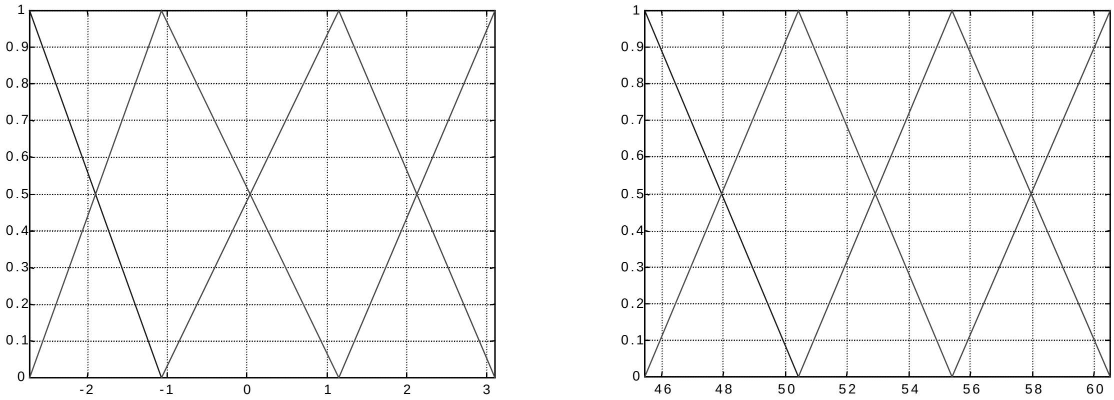


Figure 2. Partition of the input domain (when $N=2$, $M_1=5$, $M_2=5$)



Membership functions on *u(k-n_d+1)*   Membership functions on *y(k)*

Figure 3. Fuzzy partitioning of the input space used for deriving fuzzy model of the Box-Jenkins furnace

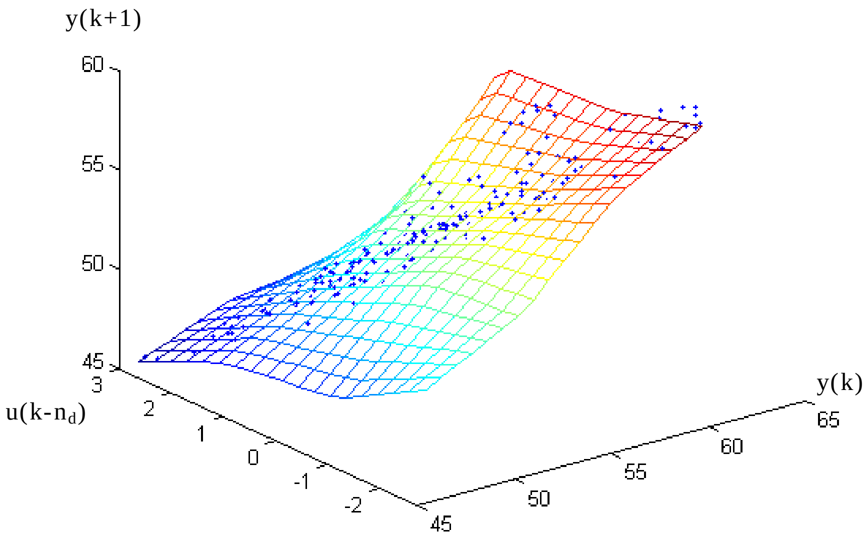


Figure 4. Prediction surface and training data on the Box-Jenkins furnace

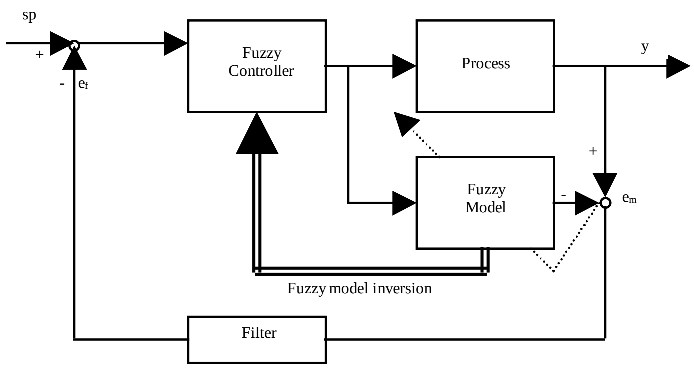


Figure 5. The adaptive IMC architecture

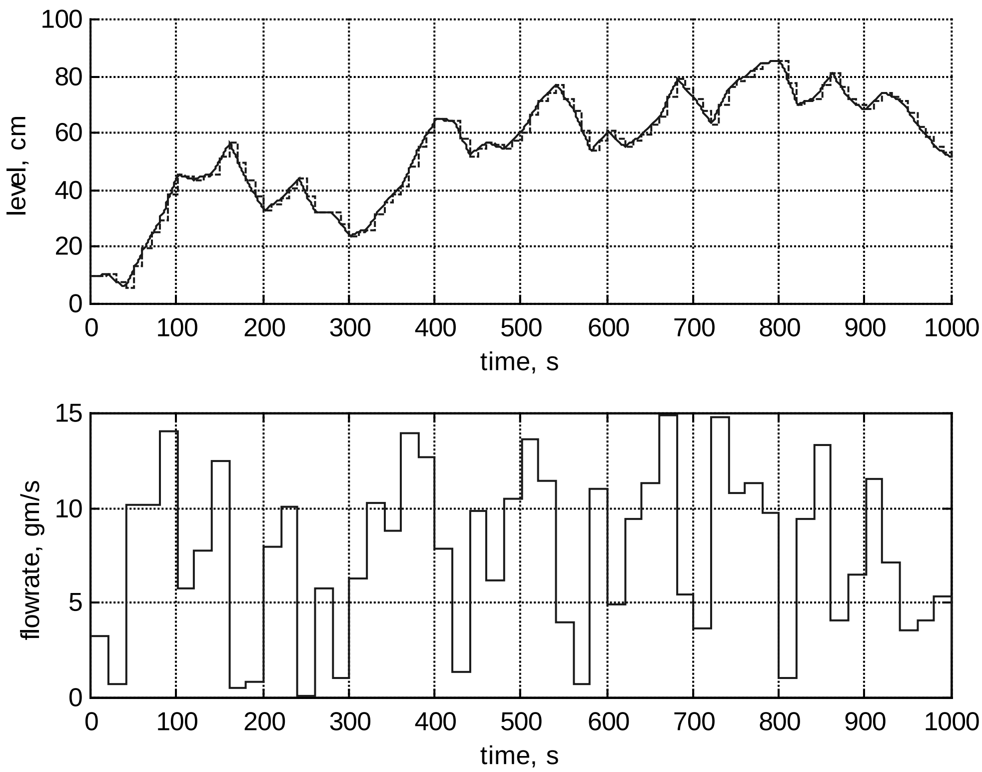


Figure 6 The modelling performance of the fuzzy model
(— process data, - - - the output of the fuzzy model)

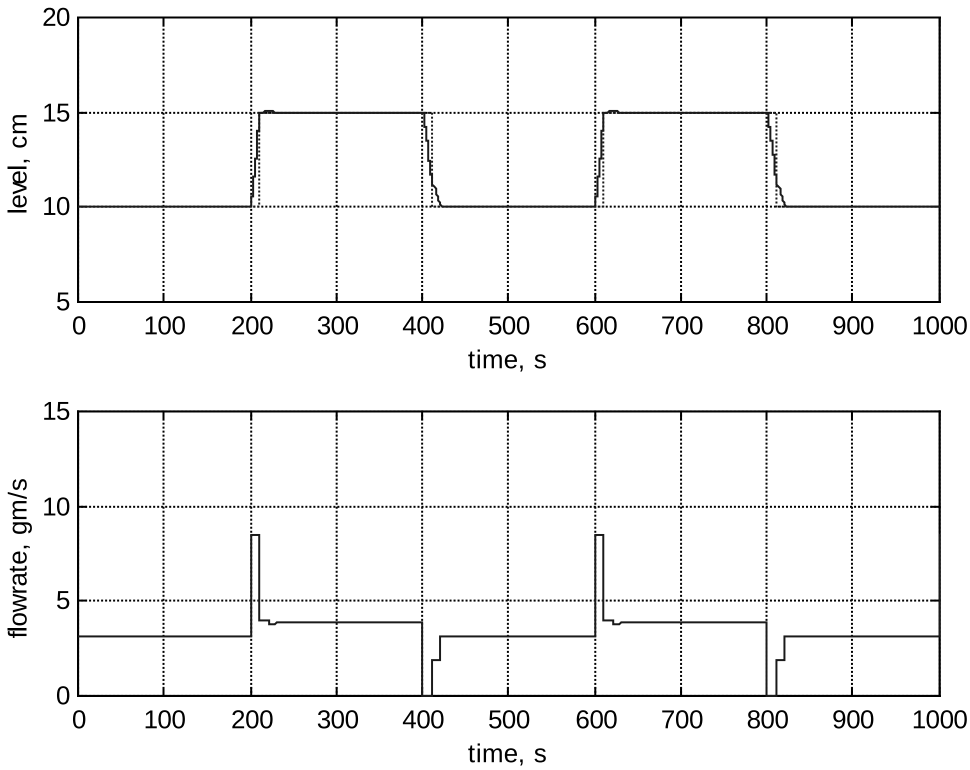


Fig. 7.a Simulated level control using adapted fuzzy model at the bottom of the tank
(— process data, ⋯⋯ reference signal)

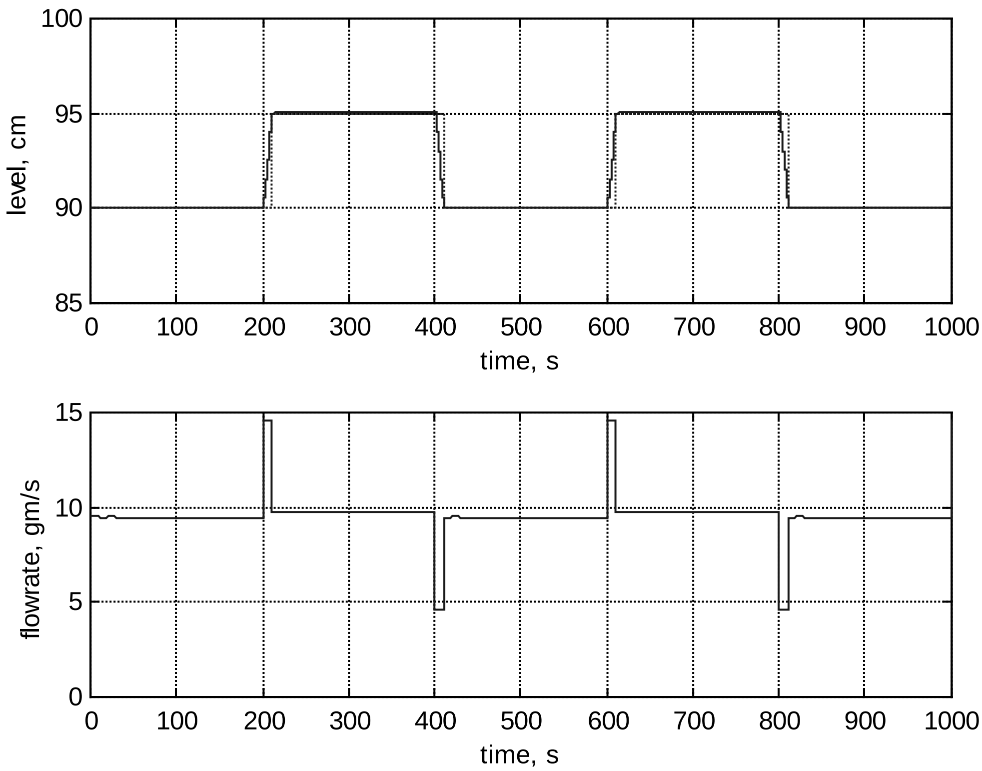


Fig. 7.b Simulated level control using adapted fuzzy model at the top of the tank
(— process data, ⋯⋯ reference signal)

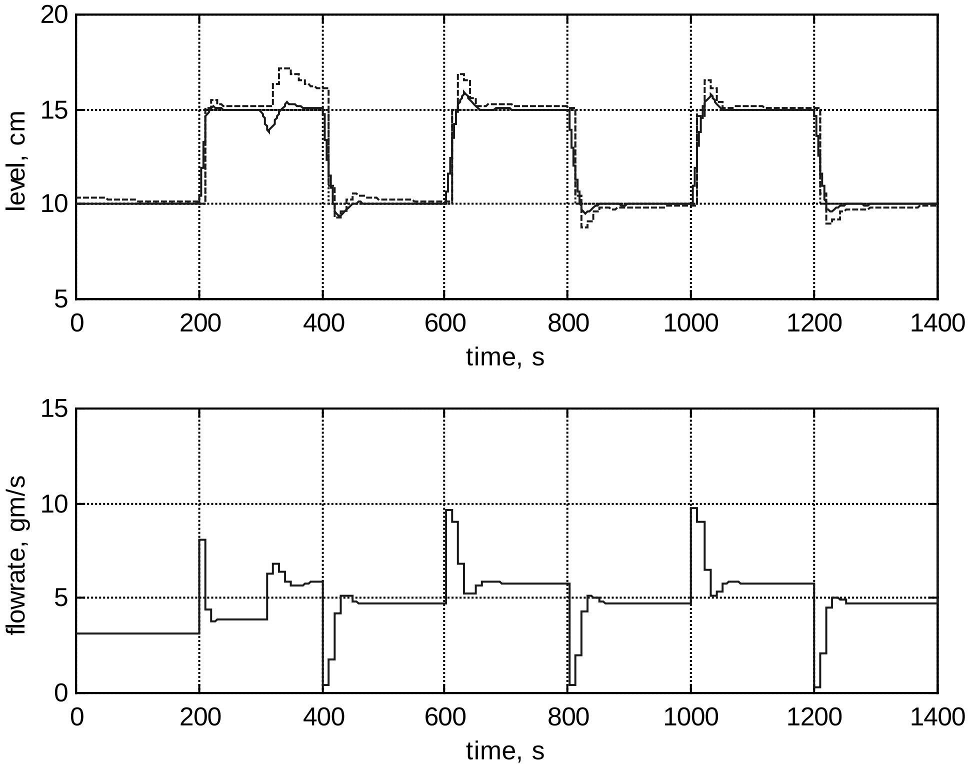


Fig. 8.a Simulated level control when the process dynamic changes at the 300th sec., without noise, (— process data, - - - the output of the fuzzy model, ···· reference signal)

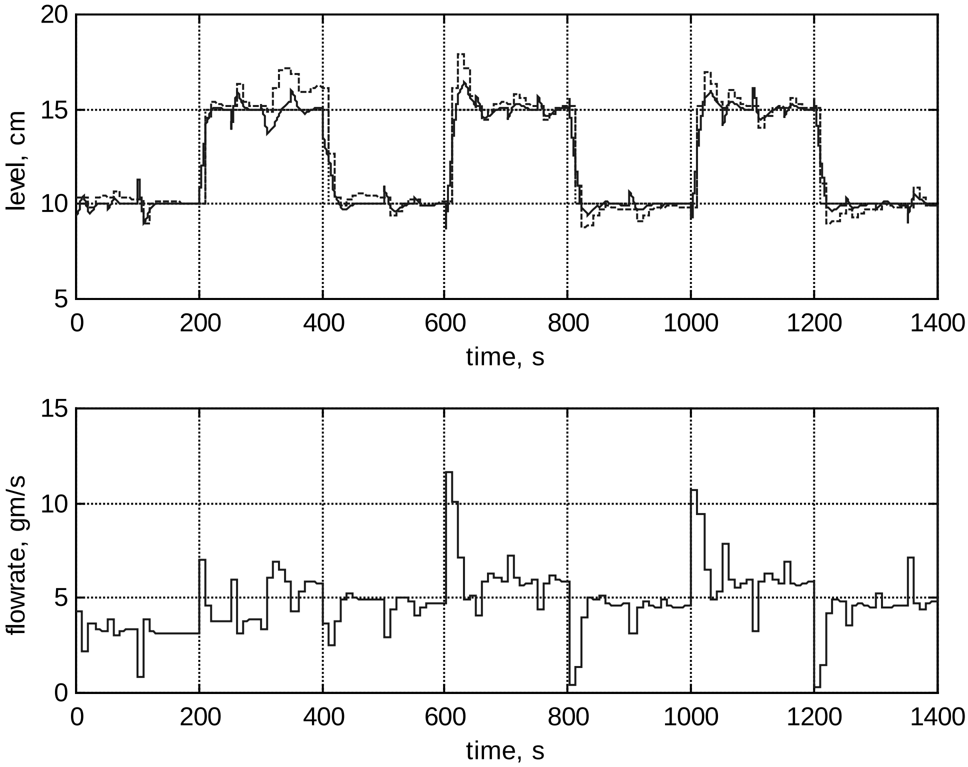


Fig. 8.b Simulated level control when the process dynamic changes at the 300th sec., under noisy conditions, (— process data, - - - the output of the fuzzy model, ···· reference signal)

TABLES

Table 1 Model accuracy and complexity in the identification of Box-Jenkins data

Number of MF	*Number of parameters*	*MSE*	*AIC*	*MSE**	*AIC**
2	8	0.2079	-1.5173	0.2078	-1.5576
3	15	0.1806	-1.6136	0.1819	-1.6065
4	24	0.1540	***-1.7186***	0.1663	***-1.6418***
5	35	0.1511	-1.6749	0.1578	-1.6315
6	48	0.1498	-1.614	0.1507	-1.6082
7	63	0.1430	-1.5861	0.1432	-1.5848
8	80	0.1333	-1.5782	0.1403	-1.5270

Table 2 A comparison of IAE values for the level control problem

	IAE between 10 and 15 cm	*IAE between 90 and 95 cm*
PI controller	426	300
Posthlethwaite, RSK model	524	349
Linkens, Takagi-Sugeno Model	194	194
Postlethwaite, Least-squares identified model	132	99
Adaptive IMC (proposed)		
3-partition	150	109
5-partition	136	107
7-partition	114	102

Table 3 The initial and final parameters

of the fuzzy model in the control task shown in figure 8.a

(**bold:** the core values of the membership functions, $a_{1,i}$: flow sets, $a_{2,i}$: level sets,

italic: the rule consequent parameters)

Initial parameter set

$a_{2,i}\backslash a_{1,i}$	**0**	**7.5**	**15.0**
5.0	*2.74*	*45.25*	*90.00*
52.50	*10.22*	*52.74*	*97.48*
100.00	*17.76*	*60.25*	*105.01*

Final parameter set

$a_{2,i}\backslash a_{1,I}$	**0**	**8.64**	**15.06**
5.99	*2.62*	*45.21*	*90.00*
52.71	*9.35*	*52.56*	*97.48*
100.00	*17.76*	*60.25*	*105.01*