An improved algorithm for online identification of evolving TS fuzzy models

ESMAEEL BANYSAEED, MOHAMMADREZA RAFIEI , and MOHAMMAD HADDAD
Department of Electrical & Robotics Engineering
Shahrood University of Technology, Shahrood
IRAN

***Abstract*—** In this paper an improved algorithm for online identification of Takagi-Sugeno fuzzy rule-based models from I/O data streams is proposed. The TS model has evolving structure i.e the fuzzy rules can be added, modified or deleted automatically. Both parts of identification algorithm (unsupervised fuzzy rule-base antecedent learning by a recursive, non-iterative clustering, and the supervised linear sub-model parameters learning by RLS estimation) are developed for the MIMO case. The radius of influence of each fuzzy rule is calculated as an adaptive vector instead of being fixed vector, allowing different areas of data space to be covered. The centers and widths of membership functions initially determined by online clustering are optimized continuously using a gradient descent method. This feature enables the identification algorithm to deal with time-varying systems and non-stationary data streams. Simulation studies (using two benchmark problems) and comparisons with some other online learning algorithms demonstrate that a more compact structure with higher performance is achieved by the proposed approach.

***Keywords*—**online system identification, evolving fuzzy model, parameter optimization

1 Introduction

System Modeling plays an important role in many engineering fields such as prediction, communication and control. TS fuzzy models are powerful practical engineering tools for modeling and control of complex systems. They have a quasi-linear nature and utilize the idea of approximation of a nonlinear system by a collection of fuzzily mixed local linear sub-models. This feature enables them to approximate severe nonlinearity, multiple operating modes, and significant parameter or structure variation [1].

The methods for learning TS models from data have two major parts: structure identification (estimation the number of required fuzzy rules and the centers and widths of their membership functions) and parameter identification (learning the free parameters of linear sub-models of the consequents). With fixed antecedent parameters, the TS model is transformed to a linear model with some parameters which are calculated by RLS algorithm.

Real-time implementation of offline identification methods is very difficult or even impossible due to their time-consuming re-training procedures. High-volume and non-stationary data streams (produced for example in large industrial processes) can not be analyzed in batch mode methods such as back-propagation or genetic algorithm. Real-world engineering problems such as intelligent systems require important features such as fast and online incremental adaptive learning, open structure organization, ability for memorizing information, knowledge acquisition and self-improvement [7]. These requirements lead to development of evolving fuzzy neural networks (EfuNN), evolving connectionist systems and evolving rule-based takagi-sugeno (eTS) models.

The eTS learning algorithm is based on a recursive evaluation of the informative potential of new data points and the focal points of the rules[1]. The algorithm updates the number of rules and their antecedent and consequent parameters continuously and dynamically. Outliers have no chance to become

rule centers. It is important to note that learning could start without a priori information and only one data sample. Therefore this approach is very useful in real-time control, adaptive control, robotic, diagnostic systems and data acquisition [2].

In [1], an approach for online learning of MISO (Multi-Input Single-Output)TS fuzzy models is proposed. the extension of this algorithm into MIMO (Multi-Input Multi-Output) case is demonstrated in [2]. In some situations the number of fuzzy rules grows dramatically because their criteria do not consider the conditions where some inefficient fuzzy rules need to be deleted. Reference [3] presents an adaptive fuzzy modeling and control scheme. Both [1] and [3] consider the radius of influence of each fuzzy rule as a constant vector that is determined initially by the user. To the authors' experience, the radius of each cluster should not kept constant, because the distribution of data points significantly changes in dynamic systems. This concept is considered in this paper. Even if the radius of influence of some clusters can be constant, due to the lack of priori information about the distribution of data points in clusters it is not easy to determine it at the start of algorithm without clustering.

Most of structure identification methods are based on data clustering such as fuzzy C-means clustering, mountain clustering, and subtractive clustering. These approaches require all input-output pairs of data to be ready before starting the identification process. A few online clustering methods can be found such as one presented in [6], but it is suitable for a special neural fuzzy system only (self-constructing neural fuzzy inference network).

In [4] a nonlinear transversal fuzzy filter with online clustering is proposed. The clustering method needs some constants to be predefined. These constants significantly affect the number of rules, but there is no straightforward way to determine them. In [5] and [6], dynamic fuzzy neural networks (DFNN) are presented. However, in these methods all past training data must be stored. Therefore, heavy memory and computation burden are unavoidable.

In this paper, a modified online subtractive clustering is used to determine the number of rules and the center and radius of influence of each new fuzzy rule. To reach a precise and efficient model, the modeling approach needs to be able to handle all the necessary operators as adding , removing , and modifying . But, most of the existing methods either do not consider all the aforementioned operators (especially the removing one) or suffer from lack of a good mechanism to handle the operators. In this paper , heuristic criterions are proposed to add, modify or delete the rules. The centers and widths of all membership functions are updated using the gradient descent (GD) method.

This paper is organized as follows. In section 2 we formulate the MIMO TS model. An online identification method for models with some new aspects is presented in section 3. Numerical examples are provided in section 4 to illustrate the performance of the presented method. Concluding remarks are given in section 5.

2 General description of MIMO TS Model – a review

TS models can be described as a set of fuzzy rules of the following form:

$$R_i : IF(x_1 is A_{i1}) AND ..... AND (x_n is A_{in}) \quad THEN \; y^i = x_e^T \pi^i \quad ; \quad i = \{1, R\} \tag{1}$$

R is the number of fuzzy rules, $x = [x_1 , x_2 , \dots , x_n]^T$ denotes the input vector, $x_e = [1;x]$ is extended input vector, to consider a free parameter of consequent part. A_{ij} denote the antecedent fuzzy sets, y^i is the multidimensional output vector of the i-th linear sub-model and π^i are its parameters(assuming m output variables):

$$y^i = [y_1^i, y_2^i, ..., y_m^i] \tag{2}$$

$$\pi^i = \begin{bmatrix} a_{01}^i & a_{02}^i & \cdots & a_{0m}^i \\ a_{11}^i & a_{12}^i & \cdots & a_{1m}^i \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1}^i & a_{n2}^i & \cdots & a_{nm}^i \end{bmatrix} \tag{3}$$

We use Gaussian antecedent fuzzy sets because it ensures greatest possible generalization of the description (it assumes normal distribution of data in a clusters):

$$\mu_{ij}(x_j) = \exp\left[-\left(\frac{xij - cij}{\sigma ij} \right)^2 \right]; \quad i = 1:R \; ; \quad j = 1:n \tag{4}$$

Where c_{ij} and σ_{ij} are the center and width of the Gaussian function that represents the degree of belongness of x_j to the i-th cluster.

Linearity of consequent part sub-models is a very useful characteristic , especially for controller design. For example, in the model predictive controllers

which are extensively being used in process control, an optimization problem should be solved in each sampling period. If we use the nonlinear model for the process, the optimization problem becomes nonlinear and complicated , which has no explicit solution and should be solved by time-consuming iterative procedures. By using linear model of the process, there are several explicit solutions in the form of program codes for a linear optimization problem which is encountered. Likewise, controller design for a linear system is easier than a nonlinear ones.

Activation level of the rules are defined as Cartesian product of a respective fuzzy sets for this rule:

$$\varphi_i(x) = \prod_{j=1}^{n} \mu_{ij}(x_i) \tag{5}$$

The normalized activation level of the i-th rule is calculated as follows:

$$\lambda_i = \frac{\varphi_i}{\sum_{k=1}^{R} \varphi_k} \tag{6}$$

The level of contribution of the i-th linear sub-model to the overall output of the TS model is proportional to λ_i The j-th output variable of TS model is calculated by weighted averaging of individual rules contribution :

$$y_j = \sum_{i=1}^{R} \lambda_i y_j^i \quad ; \quad j = 1:m \tag{7}$$

3 Improved online identification of eTS model

Essential parts of online learning of eTS models are recursive clustering, derivation of a gradually evolving rule-base and weighted RLS algorithm. Basic stages of the procedure are described.

3.1 Rule-base initialization

At the start of learning, the first data point is considered as the focal point of the first cluster. Its potential is assumed equal to one. The covariance matrix C is initialized with large values, Ω, in its main diagonal:

$$k = 1; \quad R = 1; \quad x_1^* = x_k; \quad p_1(c_1) = 1$$
$$\theta_1 = \pi_1 = 0; \quad C_1 = \Omega I \tag{8}$$

Where c_1 is the center of first cluster and x_1^* is the focal point of first rule(a projection of c_1 on the axis x)

3.2 Calculating potential of new data point (start the loop)

At the next time step $(k = k+1)$, the potential of new data point (z^k)is calculated by a cauchy type function of first order:

$$p_k(z_k) = \left(1 + \frac{1}{k-1} \sum_{l=1}^{k-1} \sum_{j=1}^{n+m} (d_{lk}^j)^2\right)^{-1}; k = 2,3,... \tag{9}$$

where $d_{lk}^j = z_j^l - z_j^k$.

Starting from (9) and expressing the projection of distances in an explicit form for the time k , recursive formula of the potential is derived as follows, which is very important for online implementation of the learning algorithm:

$$p_k(z_k) = \frac{(k-1)}{(k-1)(\omega_k + 1) + \tau_k - 2\nu_k} \tag{10}$$

Where

$$\omega_k = \sum_{j=1}^{n+m} \left(z_j^k\right)^2; \tau_k = \sum_{l=1}^{k-1} \sum_{j=1}^{n+m} \left(z_j^l\right)^2; \nu_k = \sum_{j=1}^{n+m} z_j^k \beta_j^k \; ; \beta_j^k = \sum_{l=1}^{k-1} z_j^l$$

It is seen that β_j^k and τ_k are related to the previous data samples. in order to reduce the computational load, these variables are calculated recursively:

$$\tau_k = \tau_{k-1} + \sum_{j=1}^{n+m} (z_j^{k-1})^2 \quad ; \quad \beta_j^k = \beta_j^{k-1} + z_j^k \tag{11}$$

3.3 Updating the potentials of the centers

Definition of potential depends on the distance to all data points, including new data point. Therefore the new data point z^k affects the potential of the centers of existing clusters. Using (9), the potentials of the focal points of the existing clusters are updated recursively:

$$p_k(c_l) = \frac{(k-1)p_{k-1}(c_l)}{[k-2 + p_{k-1}(c_l) + p_{k-1}(c_l) * \left\| c_i - z^k \right\|}] \tag{12}$$

3.4 Rule base evolution

The potential of z^k is compared to the potential of the centers of existing clusters and a decision whether to add, modify or delete a rule is made as follows:

- **To add or modify(replace) the rule:**

IF the potential of the new data point is higher than the potential of existing cluster centers:

$$p_k(z^k) > p_k(c_i) \quad ; \quad i = 1:R \tag{13}$$

AND z^k is close to the $_j$th cluster center:

$$\frac{\delta_{\min}}{\|c_j\|} < 0.5 \tag{14}$$

Where $\delta_{\min} = \min_{i=1}^{R} \|x^k - x_i^*\|$ is the distance from z^k to the closest existing rule center ,**THEN** z^k replaces this center:

$$x_j^* = x^k; \quad p_k(c_j) = p_k(z^k) \tag{15}$$

By this method, we replace a less informative rule with a more informative one. Note that the definition of δ_{min} is done around x instead of z. Therefore, it will be impossible for a rule with similar antecedents to exist in rule-base. Such rules could exist according to the definition of δ_{min} used in [1-2], if their consequents are different. such rules lead to contradiction in their linguistic interpretation.

IF only condition(13) is satisfied, **THEN** z^k is added to the rule-base as a new center and a new rule is formed with a focal point based on a projection of this center on the axis x:

$$R = R+1 \; ; \; x_R^* = x^k \; ; \; p_k(c_R) = P_k(z^k) \tag{16}$$

- **To delete the redundant rule**

At this stage, suppose that we have R'rule. let $d_{\min}$ as :

$$d_{\min} = \min\left\{\exp\left(-\frac{2\|c_i - c_l\|}{\|\sigma_i\| + \|\sigma_l\|}\right)\right\} \; ; i = 1:R'-1; l = 2:R' \tag{17}$$

and $p_{\max} = \max_{i=1}^{R} p_k(c_i)$. Suppose c_a and c_b are the two centers with the closest distance and $p_k(c_a) < p_k(c_b)$.

IF $\frac{d_{\min}}{\|\sigma_a\|} + \frac{p_k(c_a)}{p_{\max}} < 1$ **THEN** the a-th rule is deleted from the rule-base.

3.5 Adjusting consequent parameters

Structure identification needs an initial set of parameters for model verification. Therefore, structure and parameter identification can not be completely separated. It is possible to determine the initial consequent parameters using offline training or chose them arbitrarily. for a fixed rule-base and antecedent parameters, the rule consequent form a set of linear equations leading to a linear regression problem, which can be solved by recursive least square estimation.

During continuous learning, normalized activation levels of rules change, which affects past and new data. Therefore, straightforward application of RLS and WRLS is not correct. To solve this problem, Covariance matrices and parameters are reset once an existing rule is modified/deleted or a new rule is added.

Vector form of the output is presented as

$$\hat{y}_{k+1} = \psi_k^T \theta_k \tag{18}$$

Where ψ is a vector of inputs that are weighted by a normalized activation levels of the rules and θ is a vector composed of the linear sub-model parameters:

$$\theta_k = [\pi_1^T, \pi_2^T, ..., \pi_R^T]^T \tag{19}$$

$$\psi_k = [\lambda_1 x_e^T, \lambda_2 x_e^T, ..., \lambda_R x_e^T]^T \tag{20}$$

For global optimization, we have:

$$J_G = \sum_{k=1}^{N} \left(y_k - \psi_k^T \theta\right)^2 \tag{21}$$

Consequent parameters vector θ that minimize J_G can be estimated by the following RLS algorithm:

$$\hat{\theta}_k = \hat{\theta}_{k-1} + C_k \psi_k (y_k - \psi_k^T \hat{\theta}_{k-1}) \tag{22}$$

$$C_k = C_{k-1} - \frac{C_{k-1}\psi_k \psi_k^T C_{k-1}}{1 + \psi_k^T C_{k-1} \psi_k}; k = [1, N] \tag{23}$$

Where $\hat{\theta}_1 = 0$ and $C_1 = \Omega I$..

When a new rule is added, the RLS algorithm is reset in the following form:

1. Consequent parameters are calculated as:

$$\hat{\theta}_k = [\hat{\pi}_{1(k-1)}^T, \hat{\pi}_{2(k-1)}^T, ..., \hat{\pi}_{R(k-1)}^T, \hat{\pi}_{(R+1)k}^T]^T \tag{24}$$

where $\hat{\pi}_{(R+1)k} = \sum_{i}^{R} \lambda_i \hat{\pi}_{i(k-1)}$.

2. Covariance matrices reset as follows:

$$C_k = \begin{bmatrix} \rho\alpha_{11} & \cdots & \rho\alpha_{1R(n+m)} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots\cdots & \cdots & \cdots & \cdots \\ \rho\alpha_{R(n+m)1} & \cdots & \rho\alpha_{R(n+m)R(n+m)} & 0 & \cdots & 0 \\ 0 & 0 & 0 & \Omega & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & \Omega \end{bmatrix} \tag{25}$$

α_{ij} is an element of previous covariance matrix and $\rho = \frac{R^2+1}{R^2}$.

When a rule is replaced with another rule, the covariance matrix and consequent parameters of the new rule are taken from previous rule without change. It is quite straightforward how to reset C and θ when a rule is deleted.

In the case of local optimization, θ should minimize the following cost function:

$$J_L = \sum_{i=1}^{R} (Y - X^T \pi_i)^T \Lambda_i (Y - X^T \pi_i) \quad (26)$$

The local parameter estimation is based on WRLS:

$$\hat{\pi}_{ik} = \hat{\pi}_{i(k-1)} + C_{ik} x_{ek-1} \lambda_i(x_{k-1})(y_k - x_{ek-1}^T \hat{\pi}_{ik-1}) \quad (27)$$

$$C_{ik} = C_{i(k-1)} - \frac{\lambda_i(x_{k-1}) C_{i(k-1)} x_{ek-1} x_{ek-1}^T C_{i(k-1)}}{1 + \lambda_i(x_{k-1}) x_{ek-1}^T C_{ik-1} x_{ek-1}} \quad (28)$$

Initial conditions are $\hat{\pi}_1 = 0$; $C_{i1}=\Omega I$. the covariance matrices are separate for each rule in this case:

$$C_{ik} \in R^{(n+m)\times(n+m)}; \quad i = [1, R] \quad (29)$$

When a new rule is added, its consequent parameters are determined by (24). Also we have: $C_{(R+1)k}=\Omega I$. Parameters of the previous rules are inherited ($\pi_{ik}=\pi_{i(k-1)}$, $C_{ik}=C_{i(k-1)}$; i=[1,R]) . Parameters of all rules remain unchanged if a rule is replaced. When a rule is deleted, its parameters are deleted too.

3.6 Calculating radius of new rule

The width of fuzzy model membership functions is significant for its generalization. if the width is less than the distance between the adjacent inputs which means underlapping , the neuron does not generalize well. However, if the width is too large, the output of the sub-model may always be large (near 1) irrespective of inputs and the partition which makes no sense in this case. Therefore, the width must be carefully selected so as to ensure proper and sufficient degree of overlapping [6]. In our method, The width of the j-th membership function of newly generated rule is computed as follows :

$$\sigma_{(R+1)j} = \frac{\max\left\{\left|z_j - c_{aj}\right|, \left|z_j - c_{bj}\right|\right\}}{S} * n \quad (30)$$

Where c_a and c_b are the two nearest neighboring centers of the clusters adjacent to the receptive field, where the newly arrived pattern is located,. Also S is an orbitary constant $(1.5 < S < 2.5)$ and n is the input dimension.

G. Optimization of membership function parameters

Contrary to the conventional fuzzy-neural network based approaches, consequent parameters of the eTS model can be computed without the BP based algorithms [4]. Because the eTS model is linear after the corresponding centers and the widths are allocated. The scheme of parameters determination consist of two phases, during the forward pass, parameters of membership functions are assumed to be fixed and the free parameters of linear sub-models are determined. during the backward pass , the centers and widths will be optimized by a gradient descent method.[13]

To optimize the centers and widths, we adopt the following cost function

$$E = \sum_{k=1}^{N} e^2(k) = \sum_{k=1}^{N} \left(y(k) - \hat{y}(k)\right)^2 \quad (31)$$

Where $y(k)$ and $\hat{y}(k)$ are the real and estimated outputs.The centers will be updated as follows:

$$c_{ij} = c_{ij} + \Delta c_{ij} \quad (32)$$

$$\Delta c_{ij} = -lr_c \frac{\partial E}{\partial c_{ij}} = -lr_c \frac{\partial E}{\partial y} \frac{\partial y}{\partial \lambda_i} \frac{\partial \lambda_i}{\partial c_{ij}} = 2lr_c \lambda_i \left(\sum_{r=1}^{m} \left(y_r - \hat{y}_r\right)\left(\hat{y}_r^i - \hat{y}_r\right) \right) \left(\frac{x_i - c_{ij}}{\sigma_{ij}^2} \right) \quad (33)$$

And the widths are updated as follows:

$$\sigma_{ij} = \sigma_{ij} + \Delta \sigma_{ij} \quad (34)$$

$$\Delta \sigma_{ij} = -lr_\sigma \frac{\partial E}{\partial \sigma_{ij}} = -lr_\sigma \frac{\partial E}{\partial y} \frac{\partial y}{\partial \lambda_i} \frac{\partial \lambda_i}{\partial \sigma_{ij}} = 2lr_\sigma \lambda_i \left(\sum_{r=1}^{m} \left(y_r - \hat{y}_r\right)\left(\hat{y}_r^i - \hat{y}_r\right) \right) \left(\frac{\left(x_j - c_{ij}\right)^2}{\sigma_{ij}^3} \right) \quad (35)$$

Where l_{rc} and $l_{r\sigma}$ are the learning rates of centers and widths respectively.

For the linear regression model in the consequent parts, the second order statistics of the input signals in (18) is not only decided by the model input , but also the nonlinear mapping which depends on the membership functions. In other words, the input of the linear regression models are non-stationary which is made possible by adjusting of the centers and widths of membership functions. To optimize the model, the recursive algorithm has to seek the optimal weights θ and keep track of changing position of the optimal point as well. To prevent deteriorating the

performance, the parameters of the membership functions are updated in each p sample period, not in each sampling period. p is a constant which depends on the number of free parameters in the consequent part. Normally p is set to 2μ(n+1) because the RLS algorithm is convergent in the mean value for every time-step p greater than 2μ(n+1) [4].Therefore, the centers and widths are updated as follows:

$$\Delta c_{ij} = -\frac{\partial E_p}{\partial c_{ij}} \quad ; \quad \Delta \sigma_{ij} = -\frac{\partial E_p}{\partial \sigma_{ij}} \tag{36}$$

where $E_p = \sum_{i=k}^{k+p} \left(y(i) - \hat{y}(i)\right)^2$.

The algorithm then returns to 3.2

4 Simulation results

Example 1: *Nonlinear dynamic system identification*: the nonlinear dynamic plant is described as:

$$y(t+1) = \frac{y(t)y(t-1)[y(t)+2.5]}{1+y^2(t)+y^2(t+1)} + u(t) \tag{37}$$

The input/output data of this plant are widely used to verify the performance of system identification procedures. The corresponding fuzzy model can be represented by

$$\hat{y}(t+1) = f(y(t), y(t-1), u(t)) \tag{38}$$

Where $f(.)$ is unknown. Our objective is to find an evolving TS fuzzy model of f(.) by using the algorithm proposed in section 3. The input signal is chosen as $u(k) = 0.5\sin(2\pi k/5) + 0.5\sin(2\pi k)$ .the results and the comparison with some other methods are shown in figure 1 and table 1 respectively.

Table 1 confirms that our model is of lower error (RMSE), while its number of parameters is much lower than the number of parameters of the other online models.

Example 2: *Mackey-glass time series prediction:* This series is a well-known benchmark problem for testing system identification algorithms. It is generated by the following equation

$$\dot{x}(t) = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{39}$$

We assume $x(0) = 1.2$, $\tau = 17$. The aim is to identify the following prediction model:

$$x(t+85) = f[x(t), x(t-6), x(t-12), x(t-18)] \tag{40}$$

Figure 2 shows the Rule-base evolution and estimation error. In this section, non-dimensional error index (NDEI) is defined as a ratio of RMSE over the standard deviation of target data. The comparison of the proposed model with other models based on different online identification methods is shown in table 2.

It is clearly seen in table 2 that our model is more accurate with least number of parameters. Although the NDEI of some other models (such as Denfis with 883 fuzzy rules) is relatively close to the NDEI of our model, but the number of parameters of these models are much higher.

In order to test the robustness of the eTS model a 5% random noise has been added to the standard Mackey-glass time-series. Our model has evolved toonly 4 rules and NDEI=0.5189 which means that the noise

Table 1
comparison of structure and performance

Algorithm/authors	neurons / rules	RMSE	Para-meters
DFNN[5]	6	0.0283	48
GDFNN[6]	6	0.017	-
OLS (s.chu,.,1991)	65	0.0288	326
RBF-AFS (cho,..,1996)	35	0.1384	280
Adaptive model[3]	8	0.0849	-
Our method	**2**	**0.0092**	**8**

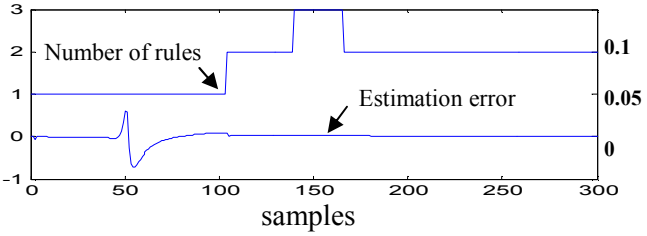

Fig. 1. Number of rules & Estimation error

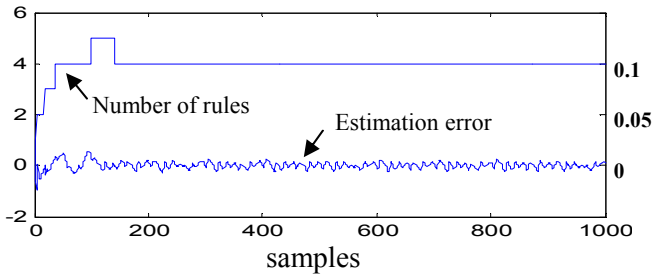

Fig. 2. Number of rules & Estimation error

Table2
comparison of structure and performance

Algorithm/authors	Rules/ units	NDEI
RAN(kasabov&song,2002)	113	0.0373
ESOM(kasabov&song,2002)	114	0.32
ESOM(kasabov&song,2002)	1000	0.044
EfuNN (kasabov&song,2002)	193	0.301
EfuNN (kasabov&song,2002)	1125	0.0904
Denfis (kasabov&song,2002)	883	0.033
Denfis (kasabov&song,2002)	58	0.276
eTS[1]	113	0.095
eTS (victor&durado , 2003)	9	0.38
Fuzzy Transversal filter[4]	-	0.0597
SONFIN [c.juang & c.lin ,1998]	9	0.0796
Neural gas	1000	0.062
DFNN[5]	5	0.0584
OLS	13	0.0698
RBF-AFS	16	0.0473
Our method	**4**	**0.0117**

has no effect on the number of rules (clusters). The ability of proposed algorithm to reject the noise and finding the appropriate number of clusters despite the noise is an important characteristic especially in classification problems and pattern recognition.

5 Conclusion

An improved approach to online identification of evolving TS fuzzy models is proposed. Heuristic criterions are used in online potential-based subtractive clustering to evolve the rule-base. The centers and widths of membership functions are updated using identification error and GD method. The identification algorithm can work without a priori information of a process and/or any predefined constants. Furthermore, the model is obtained using I/O data of normal operation modes and there is no need to excite the system with persistent exciting signals (such as PRBS), which can be practically difficult, expensive or dangerous. Its reason is that the new operating modes (areas) are easily and rapidly represented by some new rules. Our evolving fuzzy model is a promising candidate for many real-time control , adaptive & model predictive control, as well as prediction, signal processing, diagnostic systems, data acquisition, and artificial intelligence due to its higher precision, compact structure (with low number of needed parameters) and adaptive nature**.**

REFERENCES

[1] P. Angelov and D. Filev, "An Approach to Online Identification of Takagi-Sugeno Fuzzy Models", IEEE Trans.Syst.,Man, and Cybern., Vol.34 , No. 1, pp 484-498 , Februray 2004.

[2] P. Angelov, C. Xydeas, and D.Filev, "On-line Identification of MIMO Evolving Takagi-Sugeno Fuzzy Models", IEEE Conf. pp 55-60 , July 2004 , Budapest. Hun.

[3] R .Qi and M. Brdys, "Adaptive Fuzzy Modeling and Control for Discrete-Time Nonlinear Uncertain Systems" , American Control Conf. , June 8-10 , 2005 . Portland, OR, USA.

[4] Z. Li and M.J. Er , "A Nonlinear Transversal Fuzzy Filter With Online Clustering", pp 1585-1593 , 5th Asian Control Conference ,2004.

[5] S. Wu & M.J.Er , "Dynamic Fuzzy Neural Networks-A Novel Approach to Function Approximation",IEEE Trans. on Syst. ,Man, and Cybern. ,Vol. 30, No. 2, pp 358-498 , Februray 2004.

[6] S. Wu, M.J. Er and Y. Gao, "A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks", IEEE Trans. On fuzzy systems, Vol. 9 , No. 2 , pp 578-594 , 2001.

[7] N. Kasabov and Q. Song , " DENFIS: dynamic evolving Neural-Fuzzy Inference System and its Application for Time-Series Prediction ", IEEE Trans. On fuzzy systems, Vol. 10, No. 2 , pp 144-154 , 2002

[8] W.J. Hao, W.Y. Qiang , Q.Chai , J.L.Tang , " Online Data-Driven Fuzzy Modeling for Nonlinear Dynamic Systems ", 4th international conference on machine learining, Guangzhou, 18-20 August 2005

[9] K.H. Quah and C. Quek , "Online Local Learning With Generic Fuzzy Input Takagi-sugeno-kang Fuzzy Framework for Nonlinear System Estimation", IEEE Trans.Syst.,Man Cybern.,vol.36 , no. 1, Februray 2006

[10] M. Sugneo and M.Y asukawa, " A fuzzy logic based approach to qualitive modeling " IEEE Trans. Fuzzy Syst., Vol. 1 , pp 7-31 ,1993

[11] C.F. Juang and X.T. Lin , " A Recurrent Self-Organizing Neural Fuzzy Inference Network " , IEEE Trans. On Neural Networks, Vol. 10 , pp 828-845 , 1999

[12] M.F.Azeem and M.Hanmandlu , " Structure Identification of Generalized Adaptive Neuro-Fuzzy Inference Systems " , IEEE Trans. Fuzzy Syst., Vol. 11 , No.5 , pp 666-680 , October 2003

[13] S. Wu, M.J. Er and Y.Gao , " A Fast Approach for Automatic Generation of Fuzzy Rules by Generalized Dynamic Fuzzy Neural Networks", IEEE Trans. Fuzzy Syst., Vol. 9, No . 4, pp 578-594 , August 2001.