The Clustering Algorithm for Nonlinear System Identification

JOSÉ DE JESÚS RUBIO AVILA, ANDRÉS FERREYRA RAMÍREZ, CARLOS AVILÉS-CRUZ, IVAN VAZQUEZ-ALVAREZ
Departamento de Electrónica, Area de Instrumentación
Universidad Autónoma Metropolitana, Unidad Azcapotzalco.
Av. San. Pablo 180 Col. Reynosa Tamaulipas. Azcapotzalco, 02200 México D. F.
MÉXICO
jrubio@correo.azc.uam.mx, fra@correo.azc.uam.mx, caviles@correo.azc.uam.mx , iva@correo.azc.uam.mx

Abstract: - A new on-line clustering fuzzy neural network is proposed. In the algorithm, structure and parameter learning are updated at the same time. There is not difference between structure learning and parameter learning. It generates groups with a given radius. The center is updated in order to get that the center is near to the incoming data in each iteration, in this way, It does not need to generate a new rule in each iteration, i.e., it does not generate many rules and it does not need to prune the rules.

Key-Words: Clustering algorithm, Fuzzy systems, Modeling, Identification.

1 Introduction

Both neural networks and fuzzy logic are universal estimators, they can approximate any nonlinear function to any prescribed accuracy, provided that sufficient hidden neurons or fuzzy rules are available. Resent results show that the fusion procedure of these two different technologies seems to be very effective for nonlinear system identification [2]. In the last few years, the application of fuzzy neural networks to nonlinear system identification is very active area [9], [10]. Fuzzy modeling involves structure and parameters identification. The second one is usually (and easily) addressed by some gradient descent variant, e.g., the least square algorithm and backpropagation.

Structure identification is to select fuzzy rules, it often lies on a substantial amount of heuristic observation to express proper strategy's knowledge. It often tackled by off-line, trial and error approaches, like the unbiasedness criterion [11]. Several approaches generate fuzzy rules from numerical data. One of the most common methods for structure initialization is uniform partitioning of each input variable into fuzzy sets, resulting to a fuzzy grid. This approach is followed in ANFIS [5]. In [1] the TKS was used for designing various neurofuzzy identifiers. This approach consists of two learning phases, structure learning which involves to find the main input variables of all the possible, specifying the membership functions, the partition of the input space and determining the number of fuzzy rules. Parameter learning involves the unknown parameters determination and the optimization of the ready existing ones in the model, using some optimization method based on the linguistic information from the human expert and form the numeric data obtained from the actual system to be modeled. These two learning phases are interrelated, and none of them can be independent from the other one. Traditionally, these phases are done secuencially, the parameter updating is employed after the structure is decided. It is suitable only for off-line operation. Most of structure identification methods are based on data clustering, such as fuzzy C-means clustering [14], [16], mountain clustering [10], substractive clustering [3]. This approach requires that all input output data are ready before we start to identify the plant. So the structure identification approaches are off-line.

There are a few on-line methods in the literature. In [6] the input space is partitioned according to a aligned clustering-based algorithm. After the number of rules is decided, the parameters are tuned by recursive least square algorithm, it is called SONFIN. In [7] it is the recurrent case of the above case, it is called RSONFIN. In [15] the input space is automatically partitioned into fuzzy subsets by adaptive resonance theory mechanism. Fuzzy rules that tend to give high output error are split in two, by a specific fuzzy rule splitting procedure. In [8] he proposes that the radius to make clustering updates. In [19] they consider each group as a one rule, and each rule is trained by its group data, they give a time varying learning rate for backpropagation algorithm in order to prove the parameter learning error is stable.

In this paper, we propose a new on-line clustering fuzzy neural network. Learning structure and parameter learning are updated at the same time in our algorithm, we do not make difference in structure learning and parameter learning. It generate groups with a given radius. The center is updated in order to get that the center is near to the incoming data in each iteration, in this way, It does not need to generate a new rule in each iteration, i.e., it does not generate many rules and It does not need to prune the rules. We give a time varying learning rate for backpropagation training in the case of the centers and the widths. We use extended Kalman filter to train the center of sets in the THEN part.

2 Fuzzy systems

Nowadays, fuzzy inference systems (FIS) are one of the most famous applications of fuzzy logic and fuzzy sets theory [21]. They can be helpful to achieve classification tasks, offline process simulation and diagnosis, online decision support tools and process control. The strength of FIS relies on their twofold identity. On the one hand, they are able to handle linguistic concepts. On the other hand, they are universal approximators able to perform nonlinear mappings between inputs and outputs. These two characteristics have been used to design two kinds of FIS.

The first kind of FIS to appear focused on the ability of fuzzy logic to model natural language [22]. These FIS contain fuzzy rules built from expert knowledge and they are called fuzzy expert systems or fuzzy controllers, depending on their final use. Fuzzy logic allows gradual rules to be introduced into expert knowledge based simulators. It also points out the limitations of human knowledge, particularly the difficulties in formalizing interactions in complex processes. This kind of FIS offers a high semantic level and a good generalization capability. Unfortunately, the complexity of large systems may lead to an insufficient accuracy in the simulation results. Expert knowledge only based FIS may show poor performances. Even so, the major benefits of fuzzy techniques are the convenient method to model technical systems and the good interpretability of the system description by using linguistic rules. However, a fuzzy system implementation could be very time consuming because there are no systematic methods to determine its parameters (fuzzy sets and fuzzy rules). In [23] some methods are described to construct fuzzy inference systems from input-output data, which do not propose a systematic procedure to determine the number of rules of the fuzzy system. The gradient descent method for example, fixes the number of rules before training, while the table look-up method and the recursive least squares method fix the IF-part fuzzy sets before training, which in turn sets a bound for the number of rules. One of the most common problems at the time of designing fuzzy rule-based systems is how to get the desired fuzzy rule base. A set of essential design issues such as the number of if-then fuzzy rules, partition of universes, and membership functions, must be addressed. In many application tasks, fuzzy rules are manually gotten from human expert knowledge, and the resulted system is then tuned by monitoring its performance through trial and error. However, this approach is not suitable or even feasible when there is no linguistic knowledge available or expert knowledge must be tuned to data as well as becomes impractical for large dimension problems. To solve this problem, several methods have been recently proposed to automate the process of generating fuzzy rules based on numerical training data [24], [25].

3 Neural networks

The second class of simulation tools is based on automatic learning from data. This study is restricted to supervised learning and observed outputs are part of the training data. Thus, a numerical performance index can be defined which is usually based on the mean square error. Neural networks have become very popular. Their main advantage is the numerical accuracy while a major drawback is their black box behavior. Indeed, they provide a numerical model, whose coefficients have no meaning for experts. Sugeno [26] was one of the first to propose self-learning FIS and to open the way to a second kind of FIS; those designed from data. Even if the fuzzy rules, which are automatically generated from data, are expressed in the same form as expert rules, there is generally a loss of semantic. Since Sugeno's early work, a lot of researchers have been involved in designing fuzzy systems from databases. It proposes several approaches to generate fuzzy rules of numeric data [27] - [32]. [33] Introduced the main methods for designing fuzzy inference systems from data. All these methods can be considered as rule generation techniques. Rule generation can be decomposed into two main steps: 1) rule induction and 2) rule-base optimization. Originally, automatic induction methods were applied to simple systems with a few variables. In these conditions, there is no need for

optimizing the rule base. The situation is different for large systems. The number of induced rules becomes enormous and the rule description is complex because of the number of variables. Obviously, the rules will be easier to interpret if they are defined by the most influential variables and the system behavior will be easier to understand as the number of rules is getting smaller. Variable selection and rule reduction are, thus, two important steps of the rule generation process. They are usually referred as structure optimization. Apart from structure optimization, a FIS has many parameters that can also be optimized, i.e., membership functions parameters and rule conclusion adjustment. This is called parameter optimization. A thorough study has been done by various authors [31], their respective advantages and drawbacks are well known.

4 Clustering

Therefore, the number of rules in the fuzzy system can be seen as a design parameter and can be determined it based on the input-output pairs, proposal that is the base of the design of fuzzy systems using clustering [34], [35].In order to obtain a set of N fuzzy conditional rules capable of representing the system to be studied, clustering algorithms are particularly suited, since they permit a scatter partitioning of the input–output data space, which results in finding only the relevant rules. The clustering algorithms have the advantage of avoiding the explosion of the rule base, a problem known as the "curse of dimensionality." From a general and conceptual point of view, grouping means to partitioning a disjoint data collection in groups or subgroups; with a specific datum in just one group, having several specific properties to distinguish it from the others data of the data of other groups. The aim of grouping is extracting natural set groups, in order to produce a concise representation of the system behavior. The clustering algorithms have been used extensively to organize, classify and compress data and to construct models. The clustering algorithms can be divided in two classes: batch and on-line. The batch algorithms process data off line, therefore, the temporary their structure is ignored. On the other hand the on-line algorithms, assume that the data are produced by a stationary process, that is why they do not move nor vary. In this situation the data can be sampled and grouped with a batch algorithm. Jang [31] describes four of the more representative off line clustering techchniques: K-means clustering, fuzzy clustering C-means (FCM), the mountain clustering method, and subtractive clustering.

5 Combination of fuzzy systems and neural networks

Combinations of neural networks, and fuzzy systems (or neurofuzzy systems for short) have been recognized as a powerful alternative approach to develop fuzzy systems. Some neurofuzzy networks are capable to learn and to provide if–then fuzzy rules in linguistic or explicit form [36], [37], [38]–[41]. However, most of the current neurofuzzy approaches address parametric identification or learning only. In general, the designer chooses membership functions shape and the respective parameters are adjusted. In many neurofuzzy design techniques, the fuzzy sets involved are defined in multidimensional spaces. Very often, this turns rule interpretation very difficult. In addition, some of them are not consistent with fuzzy set and fuzzy reasoning theory.

6 Fuzzy neural networks for nonlinear identification

Consider following unknown discrete-time nonlinear system:

$$y(k-1) = f[X(k-1)] \qquad (1)$$

Where $X(k-1) = [x_1(k-1),..., x_N(k-1)] =$ [y(k - 2), …., y(k - n -1), u (k - 2)…., u(k - m -1] $\in \Re^N$, (N=n+m) is the input vector, $|u(k)|^2 \le \bar{u}$, $y(k)$ is the output, $|u(k-1)|^2 \le \bar{u}$, y(k-1) is the output of the plant, f is a general nonlinear smooth function $f \in C^\infty$. A generic fuzzy model is presented as a collection of fuzzy rules in the following form (Mandani fuzzy model [16])

$$R_j : \text{IF } x_1 \text{ is } A_{1,j} \text{ and } x_2 \text{ is } A_{2,j} \text{ and. . . } x_N \text{ is } A_{N,j} \quad (2)$$
$$\text{THEN } v \text{ is } B_j$$

We use M (j = 1, 2…M) fuzzy IF-THEN rules and N fuzzy sets for each rule to perform a mapping from an input linguistic vector $X(k-1) = [x_1(k-1),..., x_N(k-1)] \in \Re^N$, (N=n+m) to an output linguistic scalar $v \in \Re$. $A_{1,j}…A_{N,j}$ and B_j are standard fuzzy sets. Each input variable x_i has N fuzzy sets. In the case of [14], [16]

we know, by using product inference, center-average defuzzifier and center fuzzifier, called Sugeno fuzzy inference system with weighted average (FIS), the output of the fuzzy logic system can be expressed as

$\hat{y}$ (k - 1) = a(k - 1)/b(k - 1)

$$a(k-1) = \sum_{j=1}^{M} v_j(k-1) z_j(k-1),$$

$$b(k-1) = \sum_{j=1}^{M} z_j(k-1) \quad (3)$$

$$z_j(k-1) = \prod_{i=1}^{N} \exp\left[-\left(\frac{x_i(k-1) - c_{ij}(k-1)}{\sigma_{ij}(k-1)} \right)^2 \right]$$

where $x(k-1)$ are inputs of system (1), (i = 1…N), c_{ij}(k-1) and $\sigma_{ij}(k-1)$ are the centers and the widths of the membership function of **IF**, respectively, (j = 1…M), $v_j(k-1)$ is the center of the membership function of THEN. If we define [17]

$$\phi_j(k-1) = z_j(k-1) / b(k-1) \quad (4)$$

Then (3) can be written as follows

$$\hat{y}(k-1) = \sum_{j=1}^{M} \phi_j(k-1) u_j(k-1) = \Phi^T(k-1) V(k-1) \quad (5)$$

Where

$V(k-1) = [v_j(k-1), ..., v_M(k-1)]^T \in \Re^M$

And $\Phi(k-1) = [\phi_j(k-1), ..., \phi_M(k-1)]^T \in \Re^M$.

Remark 1 *The structure is similar to the given in [17],[18], but they train it with the gradient in all parameters. In this paper since* c_{ij}(k-1) and $\sigma_{ij}(k-1)$ *are nonlinear in parameters we use the gradient. Since* v_j*(k-1) is linear in parameters we use the extended Kalman filter which is proven in [12] and [20] that it is better than the gradient.*

7 Structure identification

Let x_i(k-1) are newly incoming pattern, then we get

$$p(k-1) = \max_{1 \le j \le M} z_j(k-1) \quad (6)$$

If $p(k-1) < r$, then a new rule is generated (each rule correspond to each center) and M= M + 1 where r is a selected radius, $r \in (0,1)$. Once a new rule is generated, the next step is to assign initial centers and widths of the corresponding membership functions, a new density with value 1 is generated for this rule.

$$c_{i,M+1}(k) = x_i(k) \;\; \sigma_{i,M+1}(k) = rand \in (0,1) \quad (7)$$
$$v_{M+1}(k) = y(k), \;\; d_{i,M+1}(k) = 1$$

If $p(k-1) \ge r$, then a rule is not generated and in the case that $z_j(k-1) = p(k-1)$ we have the winner rule j^*, the centers and density of this rule are updated as

$$c_{ij^*}(K) = c_{ij^*}(k-1) + \frac{1}{1 + x_i^2(k-1) + c_{ij^*}^2(k-1)} \left(x_i(k-1) - c_{ij^*}(k-1) \right) \quad (8)$$
$$d_{i,j^*}(k) = d_{i,j^*}(k-1) + 1$$

Remark 2 *The structure identification is a little similar to the given in [6], [7], but they do not take the max of* z_j*(k-1) as (6), this idea is taken from the competitive learning of ART recurrent neural network [4], [14] in order to get the winner rule (in the case of ART is the winner neuron). If the algorithm of [6], [7] do not generate a new rule, it does nothing, in this paper the center is updated as in (8) in order to get that the center is near to the incoming data in each iteration, in this way, It does not need to generate a new rule in each iteration, i.e., it does not generate many rules and It does not need to prune the rules. This idea is similar to the updating of weights in the Kohonen recurrent neural network [4], (in this case they speak of weights and we speak about the center of the membership functions).*

8 Parameter identification

We need the stability of identification parameters because this algorithm works on line. We .rst analyze the stability of centers and the widths of the membership function of the IF part, later we analyze the stability of the centers of the membership function of the THEN part. We assume from [16] and [19] that fuzzy is a general approximator of nonlinear functions, then (1) can be written as

$$y(k-1) = a*(k-1)/b*(k-1) - \mu(k-1)$$
$$a*(k-1) = \sum_{j=1}^{M} u_j^*(k-1) z_j^*(k-1),$$
$$b*(k-1) = \sum_{j=1}^{M} z_j^*(k-1) \quad (9)$$
$$z_j^*(k-1) = \prod_{i=1}^{N} \exp\left[-\left(\frac{x_i(k-1) - c_{ij}^*(k-1)}{\sigma_{ij}^*(k-1)} \right)^2 \right]$$

where u_j^*, $c_{ij}^*(k-1)$ and $\sigma_{ij}^*(k-1)$ are unknown parameters which may minimize the modeling error $\mu(k-1)$. In the case of two independent variables, smooth function has Taylor formula as

$$f(x_1,x_2) = f(x_{10},x_{10}) + \frac{\partial f(x_1,x_2)}{\partial x_1}(x_1 - x_{10}) + \frac{\partial f(x_1,x_2)}{\partial x_2}(x_2 - x_{20}) + \zeta(k-1) \quad (10)$$

Where $\zeta(k-1)$ is the remainder of the Taylor formula. If we let x_1 and x_2 correspond $c_{ij}(k-1)$ and $\sigma_{ij}(k-1)$, x_{10}, x_{20} correspond $c_{ij}^*(k-1)$ and $\sigma_{ij}^*(k-1)$, and define $\tilde{c}_{ij}(k-1) = c_{ij}(k-1) - c^*{}_{ij}(k-1)$, $\tilde{\sigma}_{ij}(k-1) = \sigma_{ij}(k-1) - \sigma^*{}_{ij}(k-1)$, then apply Taylor formula to (3) and (9) and gives

$$\hat{y}(k-1) = y(k-1) + \frac{\partial \hat{y}(k-1)}{\partial c_{ij}(k-1)} \tilde{c}_{ij}(k-1) + \frac{\partial \hat{y}(k-1)}{\partial \sigma_{ij}(k-1)} \tilde{\sigma}_{ij}(k-1) + \zeta(k-1) \quad (11)$$

Using chain rule, we get

$$\frac{\partial \hat{y}(k-1)}{\partial c_{ij}(k-1)} = \frac{\partial \hat{y}(k-1)}{\partial a(k-1)} \frac{\partial a(k-1)}{\partial z_j(k-1)} \frac{\partial z_j(k-1)}{\partial c_{ij}(k-1)} = \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]}{b(k-1)\sigma_{ij}^2(k-1)}$$
$$\frac{\partial \hat{y}(k-1)}{\partial \sigma_{ij}(k-1)} = \frac{\partial \hat{y}(k-1)}{\partial a(k-1)} \frac{\partial a(k-1)}{\partial z_j(k-1)} \frac{\partial z_j(k-1)}{\partial \sigma_{ij}(k-1)} = \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]^2}{b(k-1)\sigma_{ij}^3(k-1)}$$

We define the identification error as

$$e(k-1) = \hat{y}(k-1) - y(k-1) \quad (12)$$

So

$$\hat{y}(k-1) = y(k-1) + \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]}{b(k-1)\sigma_{ij}^2(k-1)} \tilde{c}_{ij}(k-1) + \quad (13)$$
$$+ \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]^2}{b(k-1)\sigma_{ij}^3(k-1)} \tilde{\sigma}_{ij}(k-1) + \zeta(k-1)$$

If we define:

$$D_1(k-1) = \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]}{b(k-1)\sigma_{ij}^2(k-1)} \quad (14)$$
$$D_2(k-1) = \frac{2u_j(k-1) z_j(k-1) \left[x_i(k-1) - c_{ij}(k-1) \right]^2}{b(k-1)\sigma_{ij}^3(k-1)}$$

Then (13) is

$$\hat{y}(k-1) = y(k-1) + D_1(k-1)\tilde{c}_{ij}(k-1) + D_2(k-1)\tilde{\sigma}_{ij}(k-1) + \zeta(k-1) \quad (15)$$

In order to assure the stability of identification we use the following learning law to updated the weights of neural identifier

$$c_{ij}(k) = c_{ij}(k-1) - \eta(k-1) D_1(k-1) e(k-1) \quad (16)$$
$$\sigma_{ij}(k) = \sigma_{ij}(k-1) - \eta(k-1) D_2(k-1) e(k-1)$$

where j=1…M, i=1…N and D_1(k-1) and D_2(k-1) are given in (14). Note that D_i(k-1) are auxiliary terms to minimize the notation, but the right is to substitute D_i(k-1) from (14) into (16). The dead-zone is applied to $\eta(k-1)$ as

$$\eta(k-1) = \begin{cases} \dfrac{\eta}{1+q(k-1)} & if \quad e^2(k-1) \geq \dfrac{\zeta^{-2}}{1-\eta} \\ 0 \quad if \quad e^2(k-1) < \dfrac{\zeta^{-2}}{1-\eta} \end{cases} \quad (17)$$

Where $q(k-1) = D_1^2(k-1) + D_2^2(k-1), 0 < \eta \leq 1$

Remark 3 *The normal learning (14),(16) has similar form as backpropagation [16], the only difference is that we use normalizing learning rate $\eta(k-1)$; [16] use fixed learning rate. The time-varying learning rate can assure the identification stable. This learning rate is easy to get, no any prior information is required, for example we may select $\eta = 0.9$.*

Now, we prove the stability of the centers of the THEN part. From (4) and (5), (9) can be written as

$$\hat{y}(k-1) = \Phi^T(k-1) V*(k-1) + \mu(k-1) \quad (18)$$

Where $V*(k-1)$ is the optimal weight which can minimize the modeling error $\mu(k-1)$.

From (5), (12) and (18) we have

$$e(k-1)=\Phi^T(k-1)\tilde{V}(k-1)+\mu(k-1) \quad (19)$$

Where $\tilde{V}(k-1)=V^*(k-1)-V(k-1)$. We modify the extended Kalman filter [12], [20] as dead-zone Kalman filter

$$\begin{aligned} V(k)&=V(k-1)-\alpha_{k-1}P_{k-1}\Phi(k-1)e(k-1)\\ P_k&=R_1+P_{k-1}-\alpha_{k-1}P_{k-1}\Phi(k-1)\Phi^T(k-1)P_{k-1}\\ \hat{y}(k)&=\Phi^T(k-1)V(k-1) \end{aligned} \quad (20)$$

$$\alpha_{k-1}=\begin{cases}\dfrac{1}{R_{k-1}} & e^2(k-1)\ \geq\ 3\bar{\mu}^2\\ 0 & e^2(k-1)\ \langle\ 3\bar{\mu}^2\end{cases}$$

Where
$Q_{k-1}=R_2+\Phi^T(k-1)P_{k-1}\Phi(k-1)$,
$R_{k-1}=Q_{k-1}+\Phi^T(k-1)P_{k-1}\Phi(k-1)=$
$R_2+2\Phi^T(k-1)P_{k-1}\Phi(k-1), R_2\rangle 0\in\Re$. $\bar{\mu}$ is the upper bound of the uncertainty $\mu(k-1)$, $|\mu(k-1)|<\bar{\mu}$. $P_{k-1}\in\Re^{MxM}$ is the covariance matrix, $R_1=\alpha I\in\Re^{MxM}$ is a diagonal matrix where $\alpha>0$ is small.

9 Algorithm Proposed

The final algorithm is proposed in the next steps:

0) Select the following parameters: $R_2=\beta\rangle 0\in\Re$, $0<r<1\in\Re$, $0<\eta<1\in\Re$, $\bar{\varsigma}^2\in\Re$, $\bar{u}^2\in\Re$.

1) For the first data k=1, M=1, $v_1(1)$ = y(1), $d_1(1)$=1, (i = 1…N), $c_{i1}(1)$ = $x_i(1)$, $\sigma_{ij}(1)=rand\in(0,1)$, $P_1=100\in\Re^{1x1}$, $R_1=\alpha\in\Re^{1x1}$.
2) For the other data $k\geq 2$, evaluate z_j(k-1) and b(k-1) with (3), evaluate by $\hat{y}(k-1)$ with (4) and (5), evaluate p(k-1) with (6), update c_{ij}(k-1) and $\sigma_{ij}(k-1)$ with (14) (16) and (17). Note that D_i(k-1) are auxiliary terms to minimize the notation, but the right is to substitute D_i(k-1) from (14) into (16), update V(k) with (20).
3) If p(k-1) < r, then a new rule is generated (M=M+1) where $r\in(0,1)$, assign initial values to $c_{i,M+1}$(k), $\sigma_{i,M+1}(k)$, v_{M+1}(k) and d_{M+1}(k) with (7), $P_{M+1,M+1,k}=100\in P_k\in\Re^{(M+1)x(M+1)}$ and $R_{M+1,M+1,1}=\alpha\in R_1\in\Re^{(M+1)x(M+1)}$, go to 2.
4) If $P(k-1)\geq r$, then a rule is not generated and in the case that z_j(k-1) = p(k-1) we have the winner rule $j*$, the value of $d_{j*}(k)$ of this rule is updated with (8), go to 2.

10 Simulation

In this section, the suggested on-line self organized algorithm proposed is applied to nonlinear system identification.

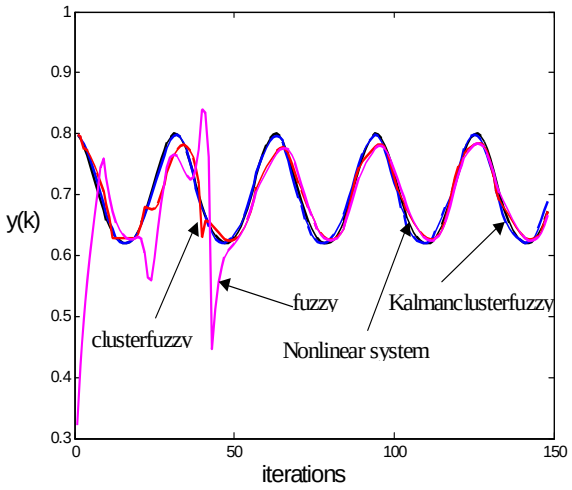

Figure 1: Comparison in identification for (21).

Example 1 Consider the nonlinear system iven in [16] and [19]:

$$y(k)=0.52+0.1x_1+0.28x_2-0.06x_1x_2 \quad (21)$$

with $x_1(k)=\sin^2(10/k)$ and $x_2(k)=\cos^2(10/k)$. The parameters of our algorithm are $\alpha=\beta=0.01$, $r=0.1$. We compare our algorithm called Kalmanclusterfuzzy with the version given in [19] called clusterfuzzy (in this case use the gradient in all cases to train parameters and they do not use (8)) and with the version given in [16] called fuzzy (in this case they do not cluster and they use gradient with constant learning rate), The identification is given in Fig.1 and the least mean square for fuzzy is 0.0026, for clusterfuzzy is $1.5832x10^{-4}$ and for Kalmanclusterfuzzy is $1.1921x10^{-5}$. In Fig. 2 is given the membership functions of the Kalmanclusterfuzzy algorithm. The clusterfuzzy algorithm generate 5 rules and the Kalmanclusterfuzzy algorithm generate 2 rules, it is because Kalmanclusterfuzzy uses (8).

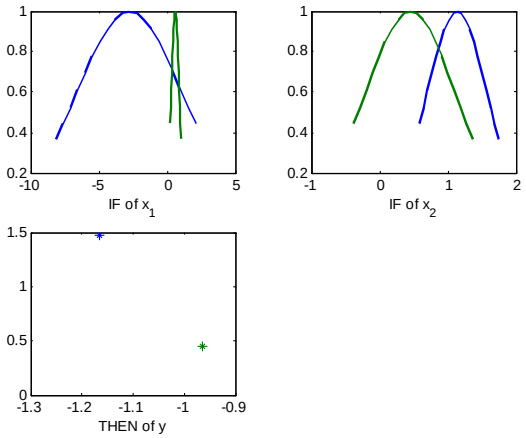

Figure 2: Membership functions for Kalmanclusterfuzzy algorithm.

Example 2 Consider the nonlinear system:

$$y(k)=\frac{y^2(k-3)+y^2(k-2)+y^2(k-1)+\tanh(u(k))+1}{y^2(k-3)+y^2(k-2)+y^2(k-1)+1} \quad (22)$$

Where
$u(k)=0.6\sin(3\pi kTs)+0.2\sin(4\pi kTs)+1.2\sin(\pi kTs)$,
$Ts=0.01, x_1(k)=y(k-1), x_2(k)=y(k-2), x_3(k)=y(k-3), x_4(k)=u(k)$
. The parameters of the our algorithm are $\alpha=\beta=0.01$, $r=1x10^{-6}$. We compare our algorithm called Kalmanclusterfuzzy with the version given in [19] called clusterfuzzy (in this case use the gradient in all cases to train parameters and they do not use (8)) and with the version given in [16] called fuzzy (in this case they do not cluster and they use gradient with constant learning rate), The identi.cation is given in Fig.3 and the least mean square for fuzzy is 0.0013, for clusterfuzzy is 0.0019 and for Kalmanclusterfuzzy is $2.2x10^{-5}$. In Fig. 4 is given the membership functions of the Kalmanclusterfuzzy algorithm. The clusterfuzzy algorithm generate 6 rules and the Kalmanclusterfuzzy algorithm generate 2 rules, it is because Kalmanclusterfuzzy uses (8).

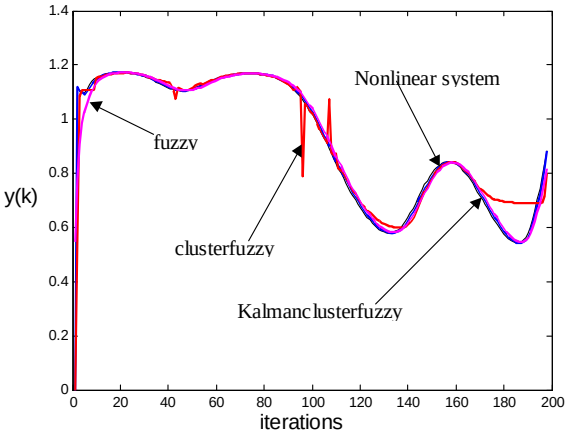

Figure 3: Comparison in identification for (22).

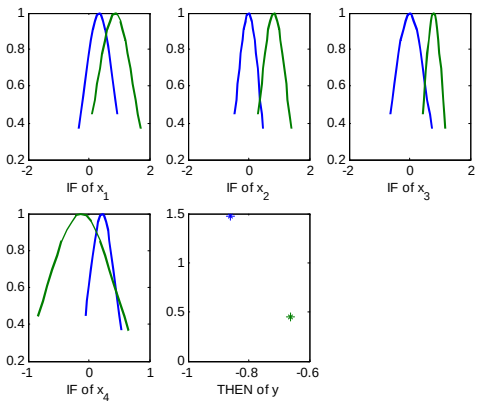

Figure 4: Membership function for Kalmanclusterfuzzy algorithm.

11 Conclusions

In this paper we presented a quick and efficient approach for system modelling using clustering and neuro fuzzy networks. The structure identification and parameter learning are done on line. From a dynamic system point of view, such training can be useful for all neural network applications requiring real-time updating of the weights. In the future, we will design a pruning algorithm

References:

[1] M.F. Azem, M.Hanmandlu and N. Ahmad, Structure identification of generalized adaptive neuro-fuzzy inference systems, IEEE Trans. on Fuzzy Syst., Vol.11, No.6, 666-681,2003.

[2] M. Brown, C.J. Harris, Adaptive Modeling and Control, Macmillan Pub.Co., Prentice Hall, New York, 1994.

[3] S.L. Chiu, Fuzzy Model Identification based on cluster estimation, Journal of Intelligent and Fuzzy Systems, Vol.2, No.3, 1994.

[4] J.R. Hilera, V.J.Martines, Redes Neuronales Artificiales, Fundamentos, Modelos y Aplicaciones, Adison Wesley Iberoamericana, U.S.A , 1995.

[5] C.F. Juang, AFNFIS: Adaptive-Network-Based Fuzzy inference system, IEEE Trans. on Systems, Man and Cybernetics, Vol.23, 665-685, 1993.

[6] C.F. Juang and C.T. Lin, An On-line Self Constructing Neural Fuzzy Inference Network and Its Applications, IEEE Trans. on Fuzzy Syst., Vol.6, No.1, 12-32, 1998.

[7] C.F. Juang and C.T. Lin, A Recurrent Self-Organizing Fuzzy Inference Network, IEEE Trans. on Neural Networks, Vol.10, No.4, 828-845, 1999.

[8] N. Kasabov, Evolving Fuzzy Neural Networks for Supervised/Unsupervised Online Knowledge-Based Learning, IEEE Trans. on Systems, Man and Cybernetics, Vol.31,No.6, 902-918, 2001.

[9] C.T.Lin, Neural Fuzzy Control Systems with Structure and parameter learning, New York: World Scienti.c, 1994.

[10] S. Mitra, Y. Hayashi, Neuro-Fuzzy rule Generation: Survey in Soft Computing Framework, IEEE Trans. on Neural Networks, Vol.11, No.3, 748-769, 2000.

[11] I. Rivals and L. Personnaz, Neural Network Construction and Selection in non linear modeling, IEEE Trans. on Neural Networks, Vol.14, No.4, 804-820, 2003.

[12] J.J. Rubio and W. Yu, Dead-zone Kalman filter algorithm for recurrent neural networks, 44th IEEE Conference on Decision and Control, Spain, 2562-2567, 2005

[13] J.J. Rubio and W. Yu, A new discrete-time sliding mode control with time-varing gain and neural identification, Journal on Control, Vol. ,No. 2562-2567, 2006.

[14] J. Shing and C.T. Sun, Neuro-Fuzzy and Soft Computing, Prentice Hall, NJ 07458, 1997.

[15] S.G.Tzafestas and K.C. Zikidis, On-Line Neuro-Fuzzy ART-Based Structure and parameter learning TSK Model, IEEE Trans. on Systems, Man and Cybernetics, Vol.31, No.5, 797-803, 2001.

[16] L. X. Wang, A Course in Fuzzy Systems and Control, Prentice Hall, Englewood NJ 07458, 1997.

[17] W. Yu and X. Li, Fuzzy Identification Using Fuzzy Neural Networks with Stable Learning Algorithms, IEEE Trans. on Fuzzy Syst., Vol.12, No.3, 411-420, 2004.

[18] W. Yu and A. Ferreyra, System Identification with State- Space Recurrent Fuzzy Neural Networks, 43rd IEEE Conference on Decision and Control, Bahamas, 5106-5111, 2004.

[19] W. Yu and A. Ferreyra, On-line clustering for nonlinear system identification using fuzzy neural networks, IEEE International Conference on Fuzzy Systems, 678-683, 2005.

[20] W. Yu, J. J. Rubio and X. Li, Recurrent Neural Networks Training with Stable Risk Sensitive Kalman Filter Algorithm, International Joint Conference on Neural Networks, IJCNN.05, Montreal, Canada, 700-704, July-August 2005.

[21] L. A. Zadeh, “Fuzzy sets,” Inform. Control, vol. 8, pp. 338–353, 1965.

[22] E. H. Mamdani and S. Assilian, “An experiment in linguistic synthesis with a fuzzy logic controller,” Int. J. Man-Mach. Stud., vol. 7, pp. 1–13, 1975.

[23]Li Xin Wang, A course in Fuzzy Systems and control, (Prentice-Hall, PTR), 448 Pags, (1996).

[24] L. X. Wang and J. M. Mendel, Generating fuzzy rules by learning from example. IEEE Transactions on Systems, Man and Cybernetics, 22(6):1414-1427,1992

[25] S. Abe and M. Lan, Fuzzy rules extraction directly from numerical data for function approximation. IEEE Transactions on Systems, Man and Cybernetics, 25(1):119-129, January 1995.

[26] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” IEEE Tran. Syst., Man, Cybern., vol. SMC 15, pp. 116–132, 1985.

[27] M.F. Azeem, M.Hanmandlu, and N. Ahmad, Structure identification of generalized adaptive neuro-fuzzy inference systems, IEEE Trans. Fuzzy Syst., **vol.11**, No.6, 666-681, (2003).

[28] 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**, 260-270, (1995).

[29] C.F.Juang and C.Teng Lin, An on-Line self-constructing neural fuzzy inference network and its applications, IEEE Trans. Fuzzy Syst., **vol.6**, No.1, 12-32, (1983).

[30] J. S. Jang, ANFIS: Adaptive-network-based fuzzy inference system, IEEE Trans. Syst., Man, Cybern., **vol. 23**, 665-685, (1993).

[31] J.S.Jang, C.T.Sun and E.Mizutani, *Neuro-Fuzzy and Soft Compting: A computational approach to learning and machine intelligence*, (Prentice-Hall, Upper Saddle River, NJ), 614, (1997).

[32] C. T. Lin, *Neural Fuzzy Control Systems with Structure and Parameter Learning*, (World Scientific, New York), (1994).
[33] C. T. Lin, *Designing fuzzy inference systems from data: An Interpretability-Oriented Review*,IEEE Trans. On Fuzzy Syst., vol.9, No. 3, 426-443,(2001).
[34] Li Xin Wang, *Adaptive Fuzzy Systems and Control*, (Ed. PTR Prentice Hall, USA), 232 Pags, (1994).
[35] Hung T. Nguyen., Nadipuram R. Prasad, *Fuzzy Modeling and Control*, (Ed. CRC Press,USA) 426 Pags, (1999).
[36] W. Pedrycz, *Fuzzy Sets Engineering*. Boca Raton, FL: CRC, 1995.
[37] J. Nie, "Constructing fuzzy model by self-organizing counterpropagation network," *IEEE Trans. Syst., Man, Cybern.*,, vol. 25, pp. 963–970, 1995.
[38] J. Jang and C. Sun, "Neuro-fuzzy modeling and control," *Proc. IEEE*, 1995, vol. 83, pp. 378–406.
[39] H. Ishibuchi, R. Fujioka, and H. Tanaka, "Neural networks that learn from fuzzy if–then rules," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 85–97,1993.
[40] J. Keller and H. Tahani, "Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks," *Int. J. Approximate Reasoning*, vol. 6, pp. 221–240, 1992.
[41] J. Keller, R. Yager, and H. Tahani, "Neural network implementation of fuzzy logic," Fuzzy Sets Syst., vol. 45, pp. 1–12, 1992.

José de Jesús Rubio Avila was born in México City in 1979. He graduated in electrical engineering from the ESIME IPN in México in 2001. He received the master degree in automatic control in the CINVESTAV IPN in México in 2004 and the doctor degree in automatic control in the CINVESTAV IPN in México in 2007. Since 2006, he is a full time professor in the Autonomous Metropolitan University - Mexico City where he is a member of the researching group of intelligent systems and signal processing. He has published 7 chapters in books, 8 international papers and he has presented 13 papers in International Conferences. He is a member of the adaptive fuzzy systems task force. His research interests are primarily focused on evolving intelligent systems, nonlinear and adaptive control systems, neural-fuzzy systems, mechatronic, robotic and delayed systems.

Andrés Ferreyra Ramírez was born in México City in 1968. He received the professional degree of Electrical Engineering with speciality in Electronics and Communications in 1994 from the Escuela Nacional de Estudios Profesionales campus Aragón UNAM, in México. He received the M. S. degree in biomedical engineering in 2001 from the Universidad Autónoma Metropolita, Iztapalapa, in México. He received the Ph.D. in Automatic Control in 2005 from CINVESTAV IPN, in México. He has been a Professor in Electronics Engineering Department of the Universidad Autónoma Metropolitana, Azcapotzalco, Mexico City, since 1996, where he is currently the Coordinator of the researching group of intelligent systems and signal processing. He has published 8 international papers and he has been presented 24 papers in Internationals Conferences and nationals. His research has focused on connectionist evolving systems, neuro-fuzzy system, fuzzy systems, fuzzy control and fuzzy classifiers.

Carlos Avilés Cruz was born in Mexico City (march 13,1966). He is graduated in Electronics Engineering (1989) from the Autonomous Metropolitan University, He received the master degree in signal and image processing from the National Institute Polytechnic of Grenoble FRANCE (1993). He received the Doctor degree in signal and image processing from the National Institute Polytechnic of Grenoble- FRANCE (1997). Actually he is full time professor at the Autonomous Metropolitan University - Mexico City. His research interests are: computer vision, digital signal and image processing and High Order Statistics. He is author and co-author of about 30 nationals and internationals articles and also co-author of two books. He is also Member of the National Researcher System.

Ivan Vasquez Alvarez was born in Naucalpan, Mexico in 1974. He received the B.Sc. degree in electronics engineering from the Metropolitan Autonomous University – Azcapotzalco, Mexico City, in 1999; the M.Sc. degree in electrical engineering from CINVESTAV-IPN (Advanced Studies and Research Center of the National Polytechnic Institute), Guadalajara, in 2004, and the Ph.D. degree (Candidate of Sciences) in applied mechanics and control from the Moscow State University – Lomonosov, in 2007. Actually he is working as a visitor professor at the electronics department of the Metropolitan Autonomous University – Azcapotzalco in Mexico City where he is a member of the researching group of intelligent systems and signal processing. His research interests include mathematical modeling and simulation of mechanical systems, automotive systems and nonlinear control.