# Flexible Models with Evolving Structure

Plamen P. Angelov, Dimitar P. Filev

*Abstract*— A type of flexible models in the form of a neural network (NN) with evolving structure is treated in the paper. We refer to models with amorphous structure as flexible models. There is a close link between different types of flexible models: fuzzy models, fuzzy NN, and general regression model. All of them are proven universal approximators and some of them (Takagi-Sugeno fuzzy model with singleton outputs and radial-basis function NN) are interchangeable. The evolving NN (eNN) considered here makes use of the recently introduced on-line approach to identification of Takagi-Sugeno fuzzy models with evolving structure (eTS). Both *e*R and *e*NN differ from the other model schemes by their gradually evolving structure as opposed to the fixed structure models, in which only parameters are subject to optimization or adaptation. The learning algorithm is incremental, and combines unsupervised on-line recursive clustering and supervised recursive on-line output parameter estimation. eNN has potential in modeling, control (if combined with the indirect learning mechanism), fault detection and diagnostics etc. Its computational efficiency is based on the non-iterative and recursive procedure, which combines Kalman filter with proper initializations, and on-line unsupervised clustering. *e*NN has been tested with data from a real air-conditioning installation. Applications to real-time adaptive non-linear control, fault detection and diagnostics, performance analysis, time-series forecasting, knowledge extraction and accumulation, etc. are possible directions of their use in the future research.

*Index Terms*— Artificial Neural Networks, Takagi-Sugeno and Evolving Rule-based Models, Subtractive Clustering

## I. INTRODUCTION

BOTH fuzzy models and NN have been proven effective universal approximators [1], [2]. There is also a close link between fuzzy models and so-called fuzzy NN, especially between Takagi-Sugeno models with singleton outputs and radial-basis function NN (RBF-NN). It is possible to demonstrate that all of them are also closely linked to the general regression models [3]. We will refer to these models further as *flexible* models emphasizing their amorphous structure as opposed to the conventional first principle models, which have a deterministic structure [7].

Significant attention is paid recently [4]-[12] to the so-called 'data-driven' techniques for generation of *flexible* models. It forms a new direction in their design, which is closer to the practice, where row data exist in huge quantity and various forms and are extremely easy to transmit to any point of the globe almost instantaneously. This tendency in the design of *flexible* models is based on the recent achievements of the data mining and related knowledge extraction techniques [13]. It can also use the subjective expert knowledge, if such one exists, but this is not a prerequisite. The drawbacks of the later approach are well known and during the 80's and early '90s, the efforts have been mostly directed to tuning and static optimization of rules generated by experts.

These techniques, however, are still mostly applied to classification and *off-line* modeling and control [4], [6], [10]-[13], [18]. At the same time, currently there are clear demands for effective approaches to design of *autonomous*, *self-developing* and *self-enriching* systems, which in the same

The first author is with the Department of Civil and Building Engineering, Loughborough University, Sir Frank Gibb Building, Ashby Road, Loughborough, Leicestershire, LE11 3TU, UK (e-mail: P.P.Angelov@Lboro.ac.UK).

The second author is with Ford Motor Co., 24500 Glendale avenue, Detroit, MI 48239, USA (D.Filev@Ford.com).

time are *flexible* and *robust* [14]. Their *computational efficiency* and *compactness* are pre-requisites of a practical application. The problems of *on-line* applications are mostly related to the non-linear nature of the rule-base/network structure and computational expenses of the training technique (normally back-propagation or evolutionary algorithms), that hampers development of recursive, adaptive schemes [8], [14]. Some practical applications of fuzzy models and fuzzy NN are called *self-learning* or *adaptive*, but they are rather *self-adjusting* and *self-tuning.* They, normally, suppose structure of the model to be fixed. The lack of a true adaptivity hampers industrial applications, including portability (the ability to use a system designed for one specific application for another quite similar one with as little modifications as possible) [14].

Algorithms for *on-line* applications with *self-constructing* or ***evolving*** structure have been reported very recently and independently for the fuzzy [8]-[17] and NN models [5],[9],[21]-[22]. In [5], however, the learning scheme is iterative and tends to fall into local minimums, because it is a gradient-based error back-propagation. The structure learning in [8]-[10], [15] is determined by the descriptive potential of the data, while in [9], [21]-[22] it is based on the distance between the data and the nearest rule center. In [5] it is related to the error, which the previous structure had, which does not ensure the generality of such structure changes - the results will be good if the future character of the data is the same or similar. If new data pattern has appear, the error would be high for many new data samples, but not all of them need to be added as new neurons/rules to the model structure. The mechanism for pruning used in [5] is based on similarity and it tolerates the first new data samples rather than the most informative ones. In [5] there is no replacement of the neurons/rules, which has been accepted once in the network structure/rule-base. Additionally, the model in [5] as well as with the other NN models including GR-NN [3] supposes singletons as outputs, while more general linear functions, representing local linear models, are considered in [8]-[10], [15].

In this paper, we introduce eNN as a twin model structure to the eTS models [23]. We demonstrate the close links between these *flexible* models. The incremental learning mechanism, based on computationally very efficient technique, applicable on-line in real-time is also presented. The proposed eNN has wide range of potential applications. Their effectiveness is demonstrated on the example of modeling the air-conditioning plant serving a real building, located in Iowa, USA.

## II. EVOLVING NEURAL NETWORK

### *A. Structure*

The proposed eNN (Fig.1) have six layers. The first, input layer serves to pass the input signals to the respective neurons/rules from the next layers for further transformation. The second layer (one of the two most important ones, given in Fig.1 in bold) corresponds to a particular fuzzy set having its own label. The third layer, the another most important one (also given in bold in Fig.1), represents the set of rules. Each neuron in this layer corresponds to a fuzzy rule forming the rule base of this *flexible*

model. Inputs to each neuron are equal to the degree of membership to the respective fuzzy set every input signal has. The neurons/rules perform logical AND operation represented here with multiplication. The fourth layer of the network forms the local linear sub-models, which correspond to the consequent of the Takagi-Sugeno model. Links to and from this layer are given with different type of arrows in Fig.1 to distinguish between the antecedent and consequent parts of the rules. The fifth layer combines the antecedent and the consequent part of each rule. Finally, the last layer forms the total output of this *flexible* model. It performs a weighted summation of local sub-models.

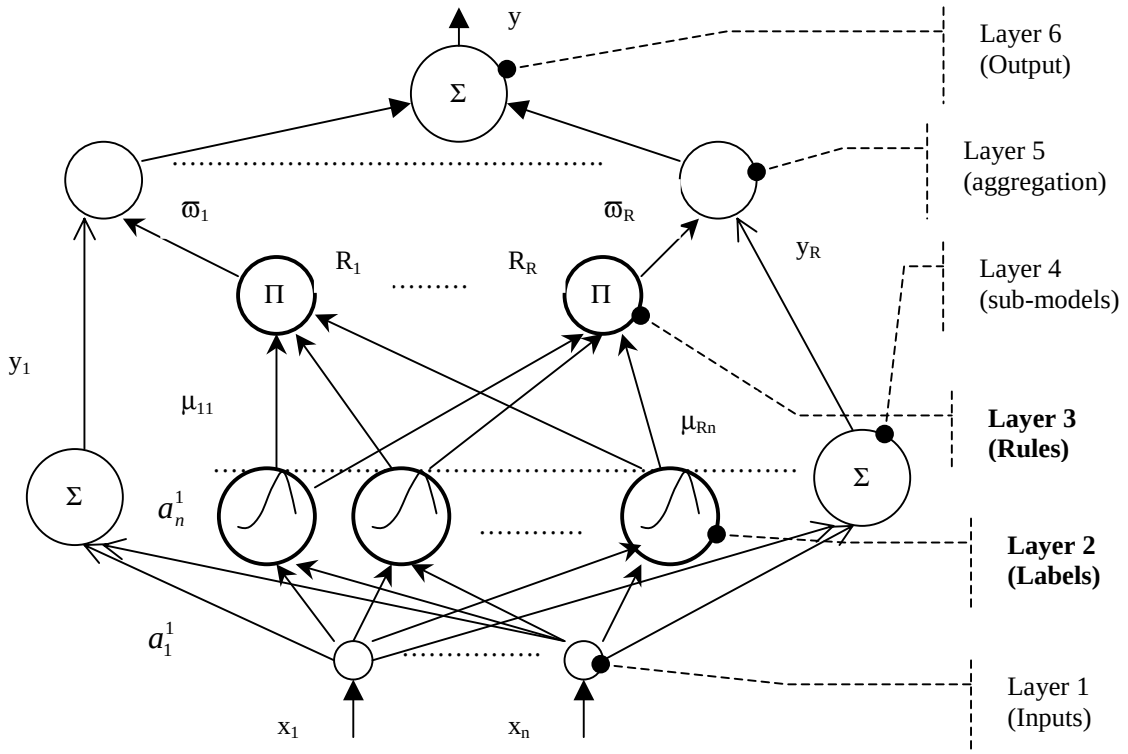


Fig.1 eNN diagram.

The eNN is a fuzzy NN based on the Takagi-Sugeno model [18] with evolving structure. Therefore, the description of each component of the network is similar to the respective Takagi-Sugeno model component:

The output ($y$) of the network is defined as a weighed sum, performed at layers 5 and 6:

$$y = \frac{\sum_{i=1}^{R} \varpi_i y_i}{\sum_{i=1}^{R} \varpi_i} \quad (1)$$

where $y$ is the output of the *flexible* model;

$y_i$ is the output of the local linear sub-model;

$R$ denotes the number of rules (neurons of the layer 3) in the model.

The degree of membership to the $i^{th}$ sub-model ($\varpi_i$) is determined at layer 3 by aggregation of the membership functions for each fuzzy set ($\mu_{ij}$, j=1,2…,n) for this sub-model. The commonly used

product operator is applied:

$$\varpi_i = \prod_{j=1}^{n} \mu_{ij} \tag{2}$$

The degree of membership to a fuzzy set ($\mu_{ij}$) from the other hand is determined at layer 2 by the Gaussian low, which ensures the greatest possible generalization:

$$\mu_{ij} = e^{-\alpha_{ij}(x_i - x_j^*)^2} \tag{3}$$

The fuzzy rules ($\Re_i$; $i=1,2,\ldots,R$), which labels ($L_{ij}$) are encoded in the neurons of the second layer, aggregated at layer 3, and which consequent part is formed by links from the input neurons ($x_i$; $j=1,2,\ldots,n$) through layer 4 (given in Fig. 1 with different type of arrows) describe together with (1) the overall action of this *flexible* model:

$$\Re_i: \textbf{IF } (x_1 \text{ is } L_{i1}) \textbf{ AND } \ldots \textbf{ AND } (x_n \text{ is } L_{in}) \ldots \textbf{ THEN } (y_i = a_{i1}x_1 + \ldots + a_{in}x_n + b_i) \qquad i=1,\ldots,R \tag{4}$$

where $a_{ij}$ are parameters of the linear sub-models (weights of the links to neurons of layer 4).

This expression illustrates the close link between eNN, eTS models as forms of *flexible* models. When the consequent part is represented by singletons ($a_{ij}=0$; $i=1,2,\ldots,R$; $j=1,2\ldots,n$) the closeness to the RBF-NN and the general regression models [3] are obvious.

*B. Learning*

The learning of the *e*NN is based on the procedure for *on-line* identification of the eTS models [15]. When the training data are collected continuously, some of them will reinforce and confirm the information contained in the previous data, but the other could bring new information [7]-[8]. This new information could concern a change in operating conditions, development of a fault or simply more significant change in the dynamics of the process. They may posses enough new information *to form a new neuron/rule or to modify* an existing one. The value of the information they bring is closely related to the information, which is possessed in the data collected so far. In cases when the new data are more informative, than the data used as prototypes for forming the antecedent part of the rules (neurons at layers 2 and 3) an adaptation or *evolution* of the model structure (rule-base) is necessary. It is based on the *on-line* clustering of the input-output data space with gradually evolving (shifting) regions of interest.

The learning procedure has two main parts performed *recursively* in *on-line*:

- ✓ To determine the structure of the layers 2 and 3 as in Fig.1 (antecedent part of the rule-base);
- ✓ To estimate the weights to and from the layer 4 (parameters of the linear sub-models).

The first part is performed as *incremental* unsupervised learning by a *recursive on-line* data space

clustering. This procedure exploits a recursive formula for informative potential of a data point calculation, which express the accumulated spatial proximity information and thus the potential of a data point to be used as a representative prototype.

When the structure of the layers 2 and 3 (antecedent part of the related fuzzy model) is determined and fixed, the second part is solvable using least squares estimation technique, known as Kalman filter [19]. Appropriate initialization needs to be made for the Kalman filter in order to take into account the variation of the antecedent part when the structure changes.

We consider the following procedure for *on-line* clustering [15]. First, we establish the first data point as the first neuron at layer 3 (centre of the first cluster) and its potential is supposed to be equal to 1:

$$k:=1;\ R:=1;\ x^{1*}:=x_k;\ P^{1*}:=1 \tag{5}$$

where $k$ denotes the time step in the *on-line* mode;
$x^{1*}$ is the focal point for the first rule;
$P^{1*}$ is the potential of the first rule.

Starting from the next data point onwards the potential of each new data point is calculated *recursively* by [15]:

$$P_k(z_k)=\frac{k-1}{(k-1)(b_k+1)+g_k-2h_k} \tag{6}$$

where $P_k(z_k)$ denotes the potential of the data point ($z_k$) calculated at the moment $k$;

$d_{ik}^j = z_i^j - z_k^j$, denotes projection of the distance between two data points ( $z_i^j$ and $z_k^j$ ) on the axis $z^j$.

$$b_k=\sum_{j=1}^{n+1}(z_k^j)^2;\ g_k=\sum_{i=1}^{k-1}\sum_{j=1}^{n+1}(z_i^j)^2;\ h_k=\sum_{j=1}^{n+1}z_k^j\ p_k^j;\ p_k^j=\sum_{i=1}^{k-1}z_i^j\ .$$

Parameters $b_k$ and $h_k$ are calculated from the current data point $z_k$, while $p_k^j$ and $g_k$ are recursively updated by $g_k = g_{k-1}+\sum_{j=1}^{n+1}(z_{k-1}^j)^2;\ p_k^j = p_{k-1}^j+z_{k-1}^j$.

When a new data are collected in *on-line* mode, they influence the potentials of the established neurons (centres of the clusters), $z_i^*$, $i=1,2,\ldots,R$, because by definition the potential depends on the distance to all data points, including the new ones. The *recursive* formula for up-date of the potentials of the existing centres is given by [15]:

$$P_k(z_i^*)=\frac{(k-1)P_{k-1}(z_i^*)}{k-2+P_{k-1}(z_i^*)+P_{k-1}(z_i^*)\sum_{j=1}^{n+1}\left(d_{k(k-1)}^j\right)^2} \tag{7}$$

where $P_k(z_i^*)$ is the potential of the at the moment ***k*** of the neuron/cluster, which is a prototype of the $i^{th}$ rule.

Then potentials of the *new* data points are compared to the up-dated potential of the neurons at layer 3 (centres of the existing clusters). If the potential of the *new* data point is *higher* than the potential of the *existing* neurons (centres) then the *new* data point is added as a **new neuron/centre** and a *new rule* is formed ($R:=R+1; x_R^* = x_k$). The condition to have higher potential is a very strong one, which restricts form an excessively large rule base being formed. If in addition to the previous condition the new data point is close to an old neuron/centre then the new data point ($z_k$) replaces this neuron/centre ($z_j^* := z_k$).

This *on-line* clustering approach ensures an *evolving* rule-base by dynamically up-grading and modifying it while inheriting the bulk of the rules (*R-1* of the rules are preserved even when a modification or an up-grade take place).

Assumed that the NN is *gradually evolving* the number of neurons as well as the parameters will vary. Then weighted recursive least squares estimation [15] could be used to solve the second sub-task - to identify parameters of these loosely coupled linear sub-models [15]:

$$\hat{\pi}_{ik} = \hat{\pi}_{ik-1} + c_{ik} x_{ek-1} \lambda_i(x_{k-1}) \; (y_k - x_{ek-1}^T \hat{\pi}_{ik-1}); \; k=2,3,\ldots \tag{8}$$

$$c_{ik} = c_{ik-1} - \frac{\lambda_i(x_{k-1}) \; c_{ik-1} x_{ek-1} x_{ek-1}^T c_{ik-1}}{1 + \lambda_i(x_{k-1}) \; x_{ek-1}^T c_{ik-1} x_{ek-1}}; i = 1,2,..., R \tag{9}$$

with initial conditions $\hat{\pi}_1 = 0$ and $c_{i1} = \Omega I$ where $\Omega$ is a large positive number; C is a *R(n+1)× R(n+1)* co-variance matrix. Parameters of the newly added rule are determined as weighted average of the parameters of the rest *R* neurons by [15]:

$$\hat{\pi}_{R+1k} = \sum_{i=1}^{R} \lambda_i \hat{\pi}_{ik-1} \tag{10}$$

The co-variance matrix respective to the newly added neurons at layer 3 (new rules) is initialized by

$$c_{R+1k} = \Omega I \tag{11}$$

Parameters of the other *R* neurons are inherited ($\pi_{ik} := \pi_{i(k-1)}; i = [1, R]$). When a nauron at layer 3 (a rule) is replaced by another rule, then parameters of all rules are inherited ($\pi_{ik} := \pi_{i(k-1)}; i = [1, R]$). The co-variance matrices of the rest *R neurons at layer 3 (*rules) are also inherited ($c_{ik} := c_{i(k-1)}; i = [1, R]$).

The procedure for the *on-line* rule-base adaptation could be summarised as follows:

1. Initialisation of the NN structure (layers 2 and 3 determining the antecedent part of the fuzzy rules).

2. Reading the *next* data sample at the next time step.

3. *Recursive* calculation of the potential of each new data sample by (6).

4. *Recursive* update of the potentials of the established neurons at layer 3 (rule centres) by (7) because of the influence of the *new* data.

5. Possible *up-grade* or *modification* or *of* the layers 2 and 3 (rule-base structure) based on the comparison between the potential of the *new* data sample and the potential of the *existing* neurons at layer 3 (centres of the rules).

6. *Recursive* calculation of the consequent parameters (weights between layers 1 and 4 and between layers 4 and 5) by (8)-(9).

The algorithm continues for the next time step from step 2.

In this way, at each moment in *on-line* layers 2 and 3 of the network (effectively, the antecedent part of the fuzzy rule base) has been specified *recursively* and *non-iteratively*. Consequent part of the model, formed by neurons at layers 4 and 5, is determined non-iteratively. The learning procedure is in one pass for each new data sample. This effect is due to the exploitation of the *quasi-linear* nature of the Takagi-Sugeno models [18] and separation of the identification problem for the antecedent part from this of the consequent.

## III. EXPERIMENTAL RESULTS

### *A. Air-conditioning*

Two engineering examples are considered in order to illustrate the proposed technique. The first one is a typical engineering example of modeling the temperature difference across a cooling coil in an air-conditioning plant serving a real building has been considered as a demonstration of the application of *e*NN.

A part of the air-conditioning plant processing the incoming air flow prior its submission to an occupied zone is represented graphically in Fig.2. The outlet from the coil temperature is normally a specified by the design constant value, while the inlet air temperature is varying. Hence, the temperature difference is varying mostly because of the inlet temperature variations and the load in the zone. The inlet temperature is normally related or equal to the ambient temperature, which have significant seasonal variations.

When the model is build from the data of all possible seasons, it gives some averaged, lower precision. If the data are from a specified season then using the resultant model is limited to this season only. Due to its gradually evolving structure, however, *e*NN is able to adapt to the changing data pattern.

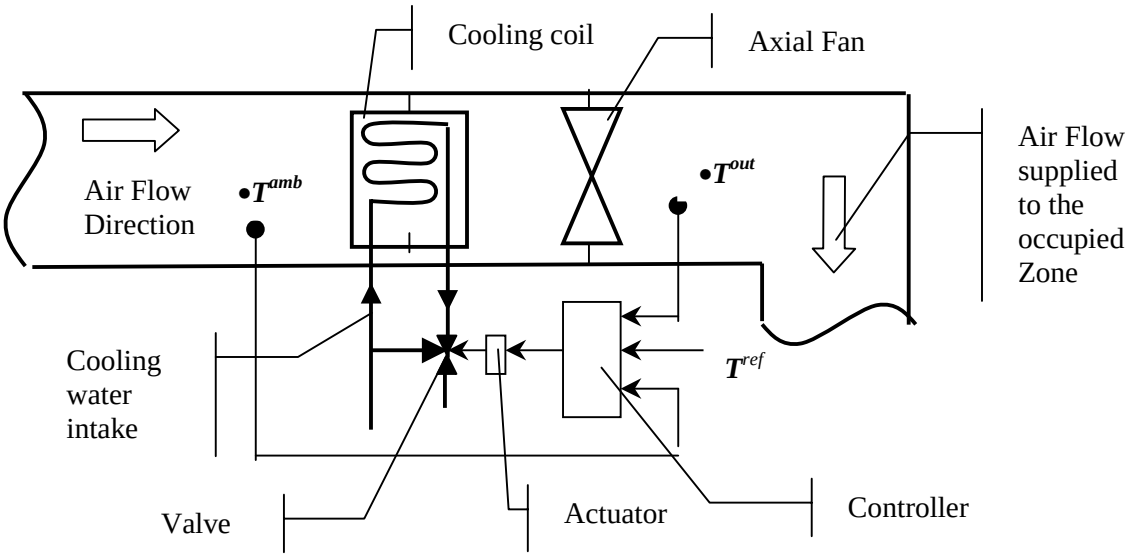


Fig.2. Air-conditioning unit.

The inputs considered in this realization are:

- mass flow rate of the inlet air ($m$);
- temperature of the chilled water ($T_w$);
- moisture content of the inlet air ($g$);
- valve position ($Ucc$).

The output is the temperature difference across the coil ($\delta T = T^{out} - T^{in}$).

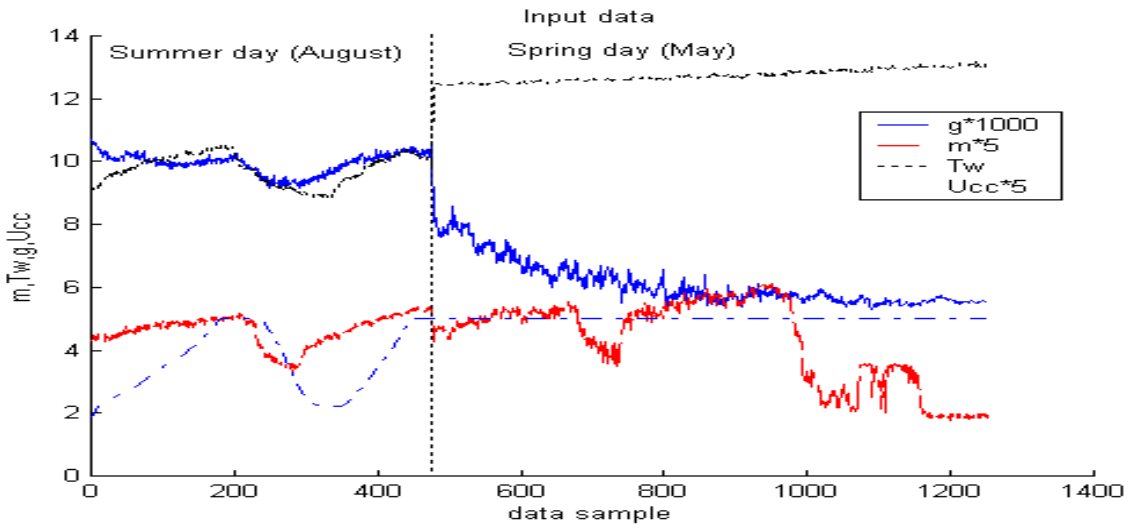


Fig. 3 Input data (airflow rate, moisture content, cooling water temperature and the valve position).

Real data from operating the facility in a typical summer day of August 22, 1998 (left part of the plots) and in a typical spring day of May 23, 1998 (right part of the plots) have been used. The structure of the *e*NN has evolved *on-line* (bottom plot in Fig. 4). Starting from one neuron in the layer 3 (one linear model in the output of the Takagi-Sugeno model) six more neurons (linear sub-models) has been added (right part of the plots) *on-line*.

On the top plot of Fig. 4, the real data for the temperature (in $^oC$) is depicted with dashed line and

the prediction by *e*NN with a solid line. On the next plot, the absolute error (in $^{o}C$) has been depicted. The overall RMS error is equal to 0.33262 $^{o}C$ (non-dimensional error index, NDEI is equal to 0.088), which is a good level for real data, modeled in *on-line*. On the bottom plot, the structure evolution (number of neurons in layer 3 representing fuzzy rules) has been illustrated.

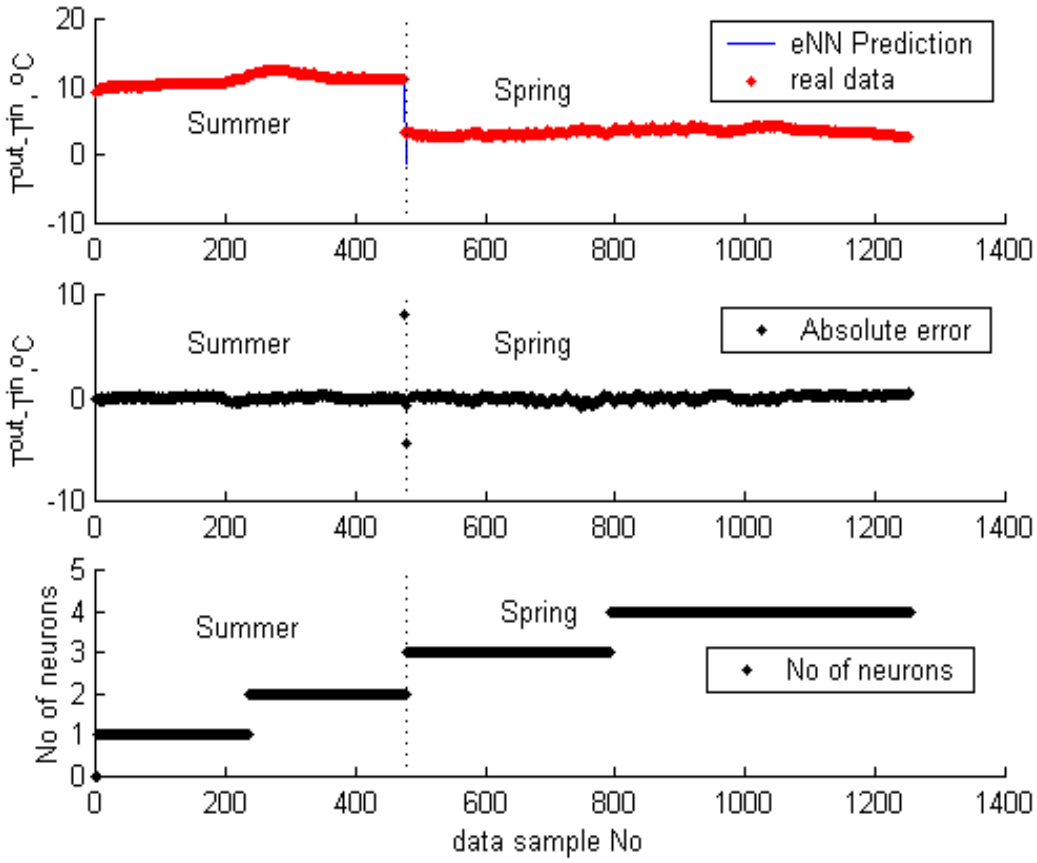


Fig. 4 RMS and absolute error in prediction (two upper plots) and the structure of *e*NN (number of neurons in the layer 3) evolution in *on-line* prediction (bottom plot).

The error is more significant when the seasons change the data pattern dramatically and this is for one step only. As it could be seen from this figure, the flexible *e*NN model is able to adjust its structure (bottom plot) so that quickly to minimize the error and to make good *on-line* predictions.

The final model includes 4 transparent and interpretable fuzzy rules:

$R_1$:**IF** ($m$ is $\boldsymbol{m_1}$) **AND** ($T_w$ is $\boldsymbol{T_1}$) **AND** ($g$ is $\boldsymbol{g_1}$) **AND** ($U_{cc}$ is $\boldsymbol{U_1}$)

**THEN** $\delta T = a_{11}m+a_{12}T_w+a_{13}g+a_{14}Ucc+b_1$

$R_2$: **IF** ($m$ is $\boldsymbol{m_2}$) **AND** ($T_w$ is $\boldsymbol{T_2}$) **AND** ($g$ is $\boldsymbol{g_2}$) **AND** ($Ucc$ is $\boldsymbol{U_2}$)

**THEN** $\delta T=a_{21}m+a_{22}T_w+a_{23}g+a_{24}Ucc+b_2$

$R_3$: **IF** ($m$ is $\boldsymbol{m_3}$) **AND** ($T_w$ is $\boldsymbol{T_3}$) **AND** (g is $\boldsymbol{g_3}$) **AND** ($Ucc$ is $\boldsymbol{U_3}$)

**THEN** $\delta T=a_{31}m+a_{32}T_w+a_{33}g+a_{34}Ucc+b_3$

$R_4$: **IF** ($m$ is $\boldsymbol{m_4}$) **AND** ($T_w$ is $\boldsymbol{T_4}$) **AND** ($g$ is $\boldsymbol{g_4}$) **AND** ($Ucc$ is $\boldsymbol{U_4}$)

**THEN** $\delta T=a_{41}m+a_{42}T_w+a_{43}g+a_{44}Ucc+b_4$

The neurons at layer 2 are represented by a Gaussian function (3) with centers (focal points or prototypes of the clusters, $x_j^*, j = 1,2,3.4$ ) given in Table I and spread $\alpha = 0.5(x_{max} - x_{min})$.

TABLE I
RULE CENTERES

| | *m, kg/s* | $T_w$,$^oC$ | *g, *10$^{-3}$* | $U_{cc}$, - |
|---|---|---|---|---|
| $R_1$ | 0.98695 | 9.7755 | 9.7535 | 1.000 |
| $R_2$ | 1.06769 | 9.8966 | 10.349 | 1.000 |
| $R_3$ | 1.13808 | 12.6427 | 6.3848 | 1.000 |
| $R_4$ | 0.38794 | 13.0061 | 5.5480 | 1.000 |

The weights of the links between neurons from layer 1 and 4 (parameters of the consequent part) has evolved in *on-line* mode according to RLS procedure (8)-(11) as illustrated in Fig.5

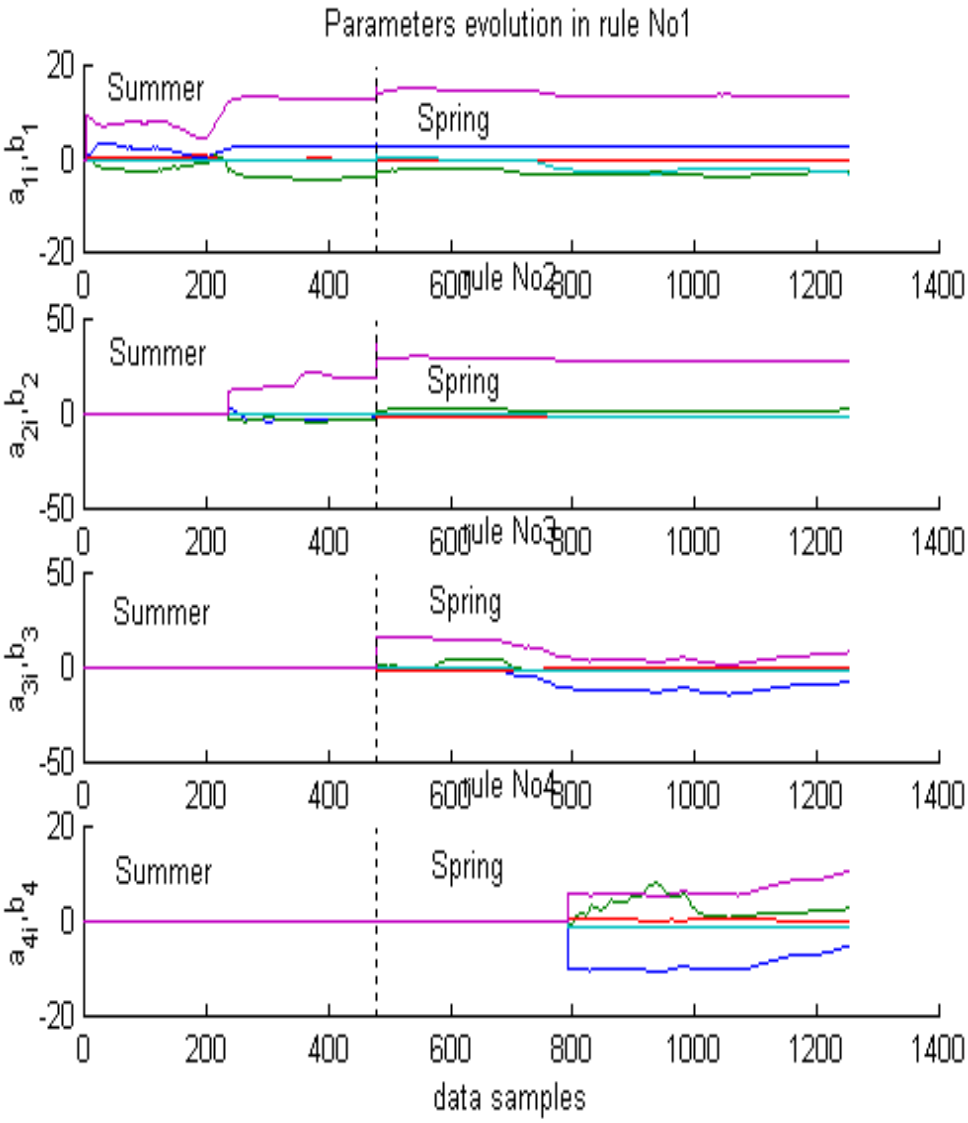


Fig. 5 Parameters ($a_{ij}$,$b_i$; *i=1,2,…,4; j=1,2,3,4*) evolution. The values represent on-line evolution of parameters of the linear sub-models that form the Takagi-Sugeno structure of the eNN.

*B. Fermentation process*

Another example is the on-line modelling the fermentation of *Kluyveromyces marxianus var. lactis MC 5* in lactose oxidation [7]. The model of this particular process includes the dependence between concentrations of the basic energetic substrates: lactose (S) and dissolved oxygen concentration (DO). This process is not well studied. There does not exist a general mathematical model of the microbial synthesis because of the extreme complexity and great variety of living activity of the micro-organisms, although various models of different parts of the whey fermentation exist. Six runs of fermentation where carried out in aerobic batch cultivation of *Kluyveromyces lactis*. A laboratory bio-reactor ABR 02M with capacity 2 *l* has been used [7].

The cell mass concentration ($X_k$) is modelled *on-line* using data for $S_k$ and $DO_k$:

$$y_{k+1} = X_{k+1};\ x_{k+1} = \{S_{k+1}, DO_{k+1}\} \tag{12}$$

The eNN has evolved to 8 neurons at the layer 3 (centres of the antecedent part are given in Table 1) in *on-line*:

*$R_1$: **IF** (S is Extremely High) **AND** (DO is Very High)* ***THEN*** $X = a_0^1 + a_1^1 S + a_2^1 DO$

*$R_2$: **IF** (S is Very High) **AND** (DO is Rather High)* ***THEN*** $X = a_0^2 + a_1^2 S + a_2^2 DO$

*$R_3$: **IF** (S is High) **AND** (DO is Very High)* ***THEN*** $X = a_0^3 + a_1^3 S + a_2^3 DO$

*$R_4$: **IF** (S is Medium) **AND** (DO is High)* ***THEN*** $X = a_0^4 + a_1^4 S + a_2^4 DO$

*$R_5$: **IF** (S is Rather High) **AND** (DO is Very Low)* ***THEN*** $X = a_0^5 + a_1^5 S + a_2^5 DO$

*$R_6$: **IF** (S is Rather Low) **AND** (DO is Rather Low)* ***THEN*** $X = a_0^6 + a_1^6 S + a_2^6 DO$

*$R_7$: **IF** (S is Very Low) **AND** (DO is Low)* ***THEN*** $X = a_0^7 + a_1^7 S + a_2^7 DO$

*$R_8$: **IF** (S is Low) **AND** (DO is Extremely)* ***THEN*** $X = a_0^8 + a_1^8 S + a_2^8 DO$

TABLE II
RULE CENTERS AND LINGUISTIC LABELS

| | ***$S_k$*** | ***g/l*** | ***$DO_k$*** | **-** |
|---|---|---|---|---|
| **Centre** | **Linguistic label** | **Value** | **Linguistic label** | **Value** |
| $x_1^*$ | *Extremely High* | 41.3 | *Extremely High* | 5.59 |
| $x_2^*$ | *Very High* | 35 | *Rather High* | 2.54 |
| $x_3^*$ | *High* | 32.4 | *Very High* | 4.04 |
| $x_4^*$ | *Medium* | 21.4 | *High* | 2.80 |
| $x_5^*$ | *Rather High* | 24.0 | *Very Low* | 0.13 |
| $x_6^*$ | *Rather Low* | 18.8 | *Rather Low* | 0.14 |
| $x_7^*$ | *Very Low* | 10.5 | *Low* | 0.18 |
| $x_8^*$ | *Low* | 15.1 | *Extremely Low* | 0.07 |

The RMS error in prediction of the cell mass concentration is 2.125 g/l (Fig. 6), which is a satisfactory value taking into account the high complexity of the process.

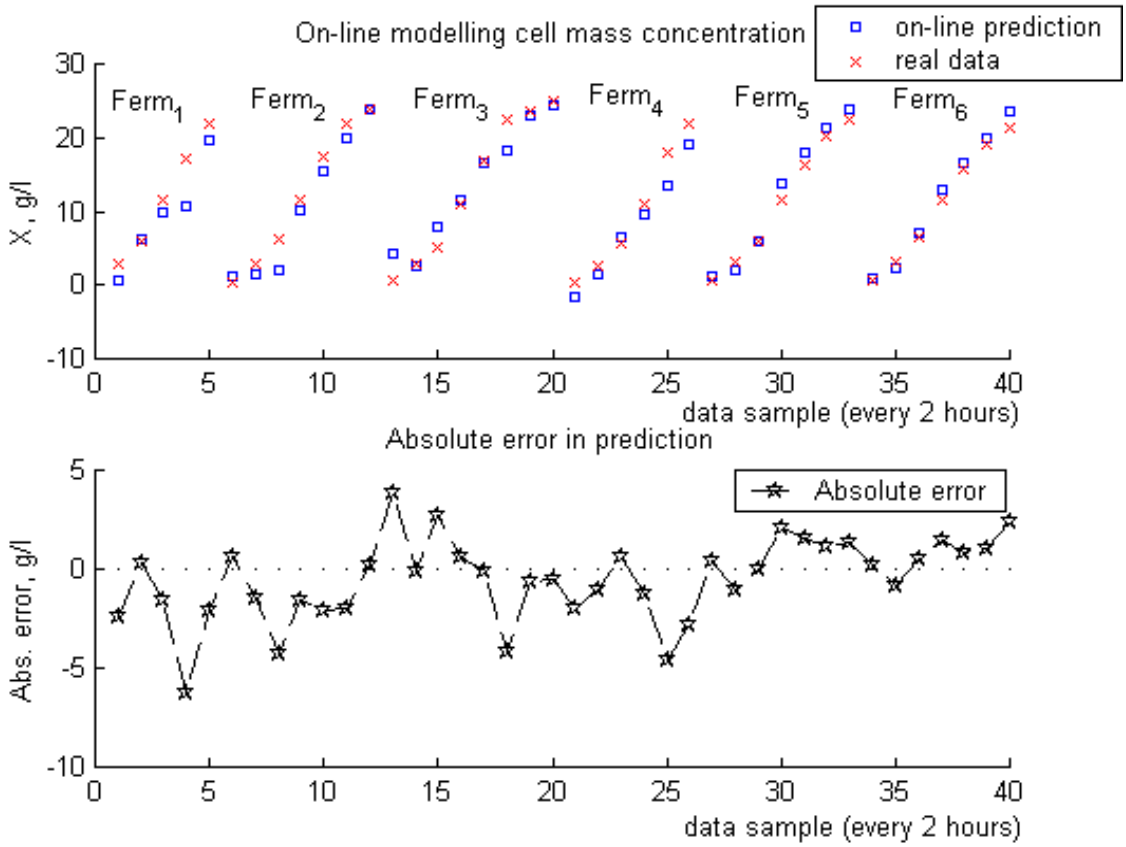


Fig. 6 Absolute error in prediction the cell mass concentration

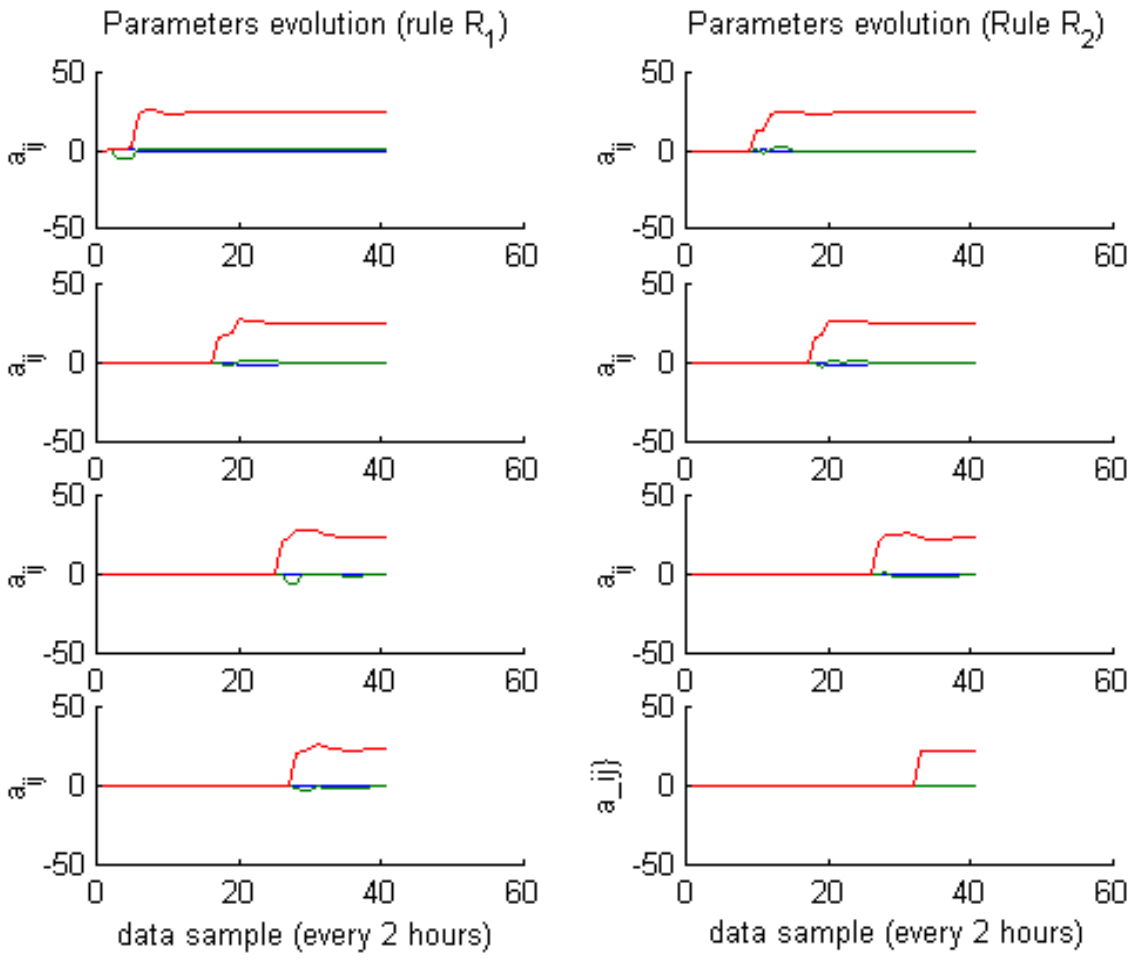


Fig. 7 Weights (consequent parameters) evolution

## VI. Conclusion

A type of flexible model in the form of a neural network (NN) with evolving structure is treated in the paper. We refer to models with amorphous structure as flexible models. There is a close link between different types of flexible models: fuzzy models, fuzzy NN, and general regression model. All of them are proven universal approximators and some of them (Takagi-Sugeno fuzzy model with singleton outputs and radial-basis function NN) are interchangeable. The evolving NN (eNN) considered here makes use of the recently introduced on-line approach to identification of Takagi-Sugeno fuzzy models with evolving structure (eTS). Both *e*R and *e*NN differ from the other model schemes by their gradually evolving structure as opposed to the fixed structure models, in which only parameters are subject to optimization or adaptation. The learning algorithm is incremental, and combines unsupervised on-line recursive clustering and supervised recursive on-line output parameter estimation. eNN has potential in modeling, control (if combined with the indirect learning mechanism), fault detection and diagnostics etc. Its computational efficiency is based on the non-iterative and recursive procedure, which combines Kalman filter with proper initializations, and on-line unsupervised clustering. *e*NN has been tested with data from a real air-conditioning installation and on-line modelling of a biotechnological process of protein synthesis. Applications to real-time adaptive non-linear control, fault detection and diagnostics, performance analysis, time-series forecasting, knowledge extraction and accumulation, etc. are possible directions of their use in the future research.

## VII. Acknowledgment

The authors express their gratitude to Dr. R. Buswell who provided the experimental data, generated from the research project RP1020 courtesy of ASHRAE.

## References

[1] K. Hornik, Approximation capabilities of multilayer feedforward network, *Neural Network*, vol. 4, pp.251-257, 1991.

[2] L.-X. Wang, “Fuzzy systems are universal approximators”, *Proc. of the International Conference on Fuzzy Systems*, San Diego, CA, USA, pp.1163-1170, 1992.

[3] D. F. Specht, “A general regression neural network”, *IEEE Transactions on NN*, vol. 2 no. 6, pp. 568-576, 1991.

[4] W. S. Lin, C.-H. Tsai, "Self-organizing fuzzy control of multi-variable systems using learning vector quantization network", *Fuzzy Sets and Systems*, vol.124, pp.197-212, 2001.

[5] F.-J. Lin, C.-H. Lin, P.-H. Shen, "Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive", *IEEE Trans. on Fuzzy Systems*, vol.9, no.5, pp.751-759, 2001.

[6] R. Babuska, H. B. Verbruggen, H. Hellendoorn, "Promising fuzzy modeling and control methodologies for industrial applications", *Proc. of the ESIT'99*, AB-02, Crete, Greece, June 1999 (http://lcewww.et.tudelft.nl/~babuska/bib/index.html).

[7] P. P. Angelov, “*Evolving rule-based models: A tool for design of flexible adaptive systems”*, Heidelberg: Springer, Physica-Verlag, 2002.

[8] P. Angelov, R. Buswell, "Identification of evolving fuzzy rule-based models", *IEEE Transactions on Fuzzy Systems*, vol. 10, No.5, pp.667-677, 2002.

[9] N. K. Kasabov, Q. Song, DENFIS: Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction, *IEEE Transactions on Fuzzy Systems*, vol. 10, No.2, pp.144-154, 2002.

[10] J.S.R. Jang, "ANFIS: Adaptive network-based fuzzy inference systems", *IEEE Trans. on SMC*, vol. 23, no. 3, pp. 665-685, 1993.

[11] S. L. Chiu, "Fuzzy model identification based on cluster estimation", *Journal of Intelligent and Fuzzy Systems*, vol. 2, pp. 267-278, 1994.

[12] R R. Yager, D. P. Filev, "Learning of fuzzy rules by mountain clustering", *Proc. of SPIE Conference on Application of Fuzzy Logic Tech.*, Boston, MA, USA, pp. 246-254, 1993.

[13] K. J. Cios, W. Pedricz, R. W. Swinarski, "*Data mining methods for knowledge discovery"*, Boston, MA: Kluwer Academic Press, 1998.

[14] EUNITE: "*EUropean Network on Intelligent Technologies for smart adaptive systems"*, Contract No IST-2000-29207, Project Summary, p.4, 2000.

[15] P. Angelov and D. Filev, An Approach to On-line Identification of Takagi-Sugeno Fuzzy Models, *IEEE Transactions on Systems, Man and Cybernetics*, part B, submitted 2002

[16] D. P. Filev, "Rule-base guided adaptation for mode detection in process control", in joint 9th IFSA World Congress/20th NAFIPS Ann. Conf., July 2001, Vancouver, BC, Canada, pp.1068-1073.

[17] D. P. Filev, T. Larsson, and L. Ma, "Intelligent control for automotive manufacturing-rule based guided adaptation", *in IEEE Conf. IECON-2000*, Nagoya, Japan, October 2000, pp.283-288.

[18] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its application to modeling and control", *IEEE Trans. on SMC*, vol. 15, pp.116-132, 1985.

[19] K. Astrom, and B. Wittenmark, "*Computer controlled systems: theory and design"*, Englewood Cliffs, NJ: Prentice Hall, 1984.

[20] R. R. Yager, D. Filev, "*Essentials of fuzzy modeling and control"*, NY: John Willey and Sons, 1994.

[21] D. Deng, N. Kasabov, "Evolving Self-Organizing Maps for On-line learning, Data Analysis and Modeling", in *Proc. IJCNN'2000 Neural Networks, Neural Computing: New Challenges Perspectives New Millennium*, vol. VI, S.-I. Amari, C. L. Giles, M .Gori, and V. Piuri Eds., New York, 2000, pp.3-8.

[22] N. Kasabov, "Evolving Fuzzy Neural Networks - Algorithms, Applications and Biological Motivation", in *Methodologies for the Conception, Design and Application of Soft Computing*, T. Yamakawa and G. Matsumoto Eds., Singapore: world Scientific, 1998, pp.271-274.