Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Sliding Mode Algorithm for On-line Learning in Fuzzy Rule-based Neural Networks

Andon V. Topalov*[1], Okyay Kaynak,**
Nikola G. Shakev*, Suk K. Hong***

**Technical University Sofia, Plovdiv branch, 25, Tsanko Dyustabanov str., 4000 Plovdiv, Bulgaria (Tel: +359-32-659528; e-mail: topalov@tu-plovdiv.bg).*
[1]currently with the School of Electrical and Computer Engineering, College of Information Technology, Ajou University, Suwon, Rep. of Korea
*** Department of Electrical and Electronic Engineering, Bogazici University, 34342 Bebek, Istanbul, Turkey (email: okyay.kaynak@boun.edu.tr)*
**** School of Electrical and Computer Engineering, Ajou University, 442-749 Suwon, Rep. of Korea, (e-mail: skhong@ajou.ac.kr)*

Abstract: A new, variable structure systems theory based, algorithm has been developed for on-line training of fuzzy-neural networks. Such computationally intelligent structures are widely used for modeling, identification and control of nonlinear dynamic systems. The algorithm is applicable to fuzzy rule-based neural nets of Takagi-Sugeno-Kang type with a scalar output. Its convergence is established and the conditions are given. Differently from other similar approaches which are limited to the adaptation of the parameters of the network defuzzification part only, the proposed algorithm tunes also the parameters of the implemented membership functions. The zero level set of the learning error variable is considered as a sliding surface in the space of network learning parameters. The effectiveness of the proposed algorithm is shown when applied to on-line learning of nonlinear functions approximation.

1. INTRODUCTION

During the last decade the concept of incorporating fuzzy logic into neural network has emerged and has become a popular research area (Farag *et al*., 1998), (Lin *et al*., 1998), (Lin *et al*., 2001), (Wang, 1997). Fuzzy neural networks (FNN) combine the advantages of both techniques. They are capable to deal with fuzzy reasoning when working with imprecise information (Wang, 1997) and possess the learning ability of neural networks as well (Narendra *et al*., 1990). Like the fuzzy systems and neural networks, FNNs have been proven to be universal approximators too (Lin *et al*., 1996). A fuzzy neural network can be considered as a fuzzy system that can be trained in similar way like the neural networks are. The typical approach of building FNNs is to implement standard neural networks which are designed to approximate a fuzzy system through their structure. Such structures are also known as fuzzy rule-based neural networks or neuro-fuzzy networks. The investigations carried out during the last years are showing that FNN can be very successfully applied for modelling and control of complex systems (Wang *et al*., 1997), (Wu *et al*., 2000).

Despite of the existing variety of learning algorithms used for training of FNN, they can be mainly classified into two groups: 1) gradient descent type algorithms that include computation of partial derivatives or sensitivity functions; they can be also viewed as different variants of the well known backpropagation learning algorithm (Rumelhart *et al*., 1986), and 2) algorithms based on evolutionary computations with genetic algorithms (GAs) being the most widely used among them (Aliev *et al*., 2001). When considering the application of FNN structures in different adaptive schemes where on-line learning is required the existing methods have some major drawbacks among which in particular are (i) the difficulty to obtain analytical pertaining to the convergence and stability of the learning schemes and (ii) the slow convergence speed. Recent investigations on neural and neuro-fuzzy networks control applications have begun to address stability issues more rigorously. Research in this area has been split over two main directions. It has been shown in several works that Lyapunov approach can be directly implemented to obtain robust training algorithms for continuous-time neural networks (Kosmatopoulos *et al*., 1995), (Suykens *et al*., 1999). Another proposed way to design a robust learning scheme is to utilize the Variable Structure Systems (VSS) theory in constructing the parameter adaptation mechanism of the NNs and FNNs (Parma *et al*., 1998), (Yu *et al*., 2004). The results obtained in this direction (Shakev *et al*., 2003) have shown that the convergence of the learning strategies can be significantly improved. Such intelligent systems exhibit the robustness and invariance properties inherited from Variable Structure Control (VSC) technique while still maintaining good approximation capability and flexibility.

The new on-line learning method for neuro-fuzzy networks, suggested in the current investigation, controls the error dynamics. The latter is defined as a difference between the current and the desired output signal of the fuzzy rule-based neural network and it is described using a differential equation. Differently from the gradient-based learning

methods which aim to minimize an error function, here the learning parameters are tuned by the proposed algorithm in a way to enforce the error to satisfy this stable equation.

The present work consists of four sections. Section 2 presents the developed new method for parametric adaptation of fuzzy rule-based neural networks with a scalar output using the Sliding Mode Control (SMC) theory. The results from simulations are shown in section 3. Finally, section 4 summarizes the obtained results.

2. THE SLIDING MODE LEARNING

2.1 The Neuro-Fuzzy Network

A neuro-fuzzy network with three inputs and one output will be considered for simplicity (see Fig. 1). The obtained results can be generalized for any number of network inputs. The incoming signals are fuzzyfied by using Gaussian membership functions. The input signals $x_1(t)$, $x_2(t)$ and $x_3(t)$ are associated accordingly with I, J and K numbers of fuzzy labels which are determined by their corresponding membership functions μ. Then the total number of the fuzzy *if-then* rules will be $I \times J \times K$.

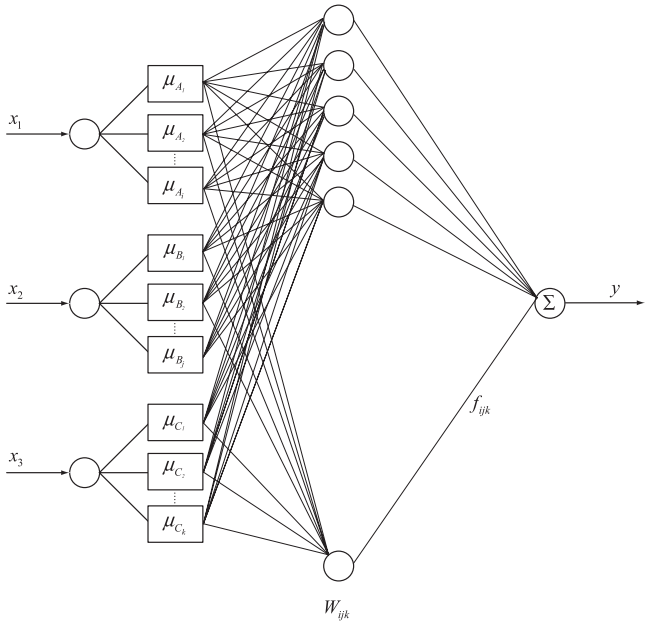

Fig. 1. The fuzzy-neural network

A fuzzy *if-then* rule base of Takagi-Sugeno type is used where the fuzzy sets are included in the premise part only. In this case the corresponding rule R_{ijk} can be expressed as:

$$R_{ijk}: \textbf{\textit{if}}\ x_1 \text{ is } A_i \text{ and } x_2 \text{ is } B_j \text{ and } x_3 \text{ is } C_k$$

$$\textbf{\textit{then}}\ f_{ijk} = a_i x_1 + b_j x_2 + c_k x_3 + d_{ijk} \qquad (1)$$

where $i = 1,...,I$; $j = 1,...,J$; $k = 1,...,K$.

It is further assumed that the output of each fuzzy *if-then* rule consists of a constant d_{ijk} only (i.e. a_i, b_j and c_k are all equal to zero and $f_{ijk} = d_{ijk}$) which is a widely used simplification.

The Gaussian membership function is described by two parameters – the center c and the distribution σ which are among the tunable parameters of the fuzzy-neural structure.

The strength of the rule R_{ijk} is obtained as a *T*-norm of the membership functions in the premise part (by using a multiplication operator):

$$W_{ijk} = \mu_{A_i}(x_1)\,\mu_{B_j}(x_2)\,\mu_{C_k}(x_3) \qquad (2)$$

The output signal of the fuzzy-neural network $y(t)$ is calculated as a weighted average of the output of each rule:

$$y(t) = \frac{\sum_{i=1}^{i\max}\sum_{j=1}^{j\max}\sum_{k=1}^{k\max} W_{ijk} f_{ijk}}{\sum_{i=1}^{i\max}\sum_{j=1}^{j\max}\sum_{k=1}^{k\max} W_{ijk}} \qquad (3)$$

In the above expression $\mu_{A_i}(x_1)$, $\mu_{B_j}(x_2)$ and $\mu_{C_k}(x_3)$ are the Gaussian membership functions of the inputs x_1, x_2 and x_3 respectively and have the following appearance:

$$\mu_{A_i}(x_1) = \exp\left[-\frac{(x_1 - c_{A_i})^2}{\sigma_{A_i}^2}\right],\ \mu_{B_j} = \exp\left[-\frac{(x_2 - c_{B_j})^2}{\sigma_{B_j}^2}\right]$$

$$\text{and}\quad \mu_{C_k} = \exp\left[-\frac{(x_3 - c_{C_k})^2}{\sigma_{C_k}^2}\right] \qquad (4)$$

Therefore (2) can be rewritten also as follows

$$W_{ijk} = \exp\left[-\frac{(x_1 - c_{A_i})^2}{\sigma_{A_i}^2} - \frac{(x_2 - c_{B_j})^2}{\sigma_{B_j}^2} - \frac{(x_3 - c_{C_k})^2}{\sigma_{C_k}^2}\right] \qquad (5)$$

After the normalization of (3) the output signal of the neuro-fuzzy network will acquire the following form:

$$y(t) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} f_{ijk}\bar{W}_{ijk} \qquad (6)$$

where $\bar{W}_{ijk}$ is the normalized value of the output signal of the neuron ijk from the second hidden layer of the network:

$$\bar{W}_{ijk} = \frac{W_{ijk}}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} W_{ijk}} \tag{7}$$

2.2 Initial Assumptions

The following vectors have been specified:

$X(t) = \left[x_1(t)\, x_2(t)\, x_3(t) \right]^T$ – vector of the time varying input signals;

$W(t) = \left[W_{111}(t)\, W_{112}(t) \ldots W_{211}(t) \ldots W_{ijk}(t) \ldots W_{IJK}(t) \right]^T$ – vector of the output signals of the neurons from the second hidden layer;

$\sigma_A = \left[\sigma_{A_1}\ \sigma_{A_2} \ldots \sigma_{A_i} \ldots \sigma_{A_I} \right]^T$ – vector of the parameters defining the distribution of the Gaussian membership functions relevant to the first input of the network;

$\sigma_B = \left[\sigma_{B_1}\ \sigma_{B_2} \ldots \sigma_{B_j} \ldots \sigma_{B_J} \right]^T$ – vector of the parameters defining the distribution of the Gaussian membership functions relevant to the second input of the neuro-fuzzy network;

$\sigma_C = \left[\sigma_{C_1}\ \sigma_{C_2} \ldots\ \sigma_{C_k} \ldots \sigma_{C_K} \right]^T$ – vector of the parameters defining the distribution of the Gaussian membership functions relevant to the third input of the neuro-fuzzy network;

$c_A = \left[c_{A_1}\ c_{A_2} \ldots c_{A_i} \ldots c_{A_I} \right]^T$ – vector of the parameters defining the centers of the Gaussian membership functions relevant to the first network input;

$c_B = \left[c_{B_1}\ c_{B_2} \ldots c_{B_j} \ldots c_{B_J} \right]^T$ – vector of the parameters defining the centers of the Gaussian membership functions relevant to the second network input;

$c_C = \left[c_{C_1}\ c_{C_2} \ldots c_{C_k} \ldots c_{C_K} \right]^T$ – vector of the parameters defining the centers of the Gaussian membership functions relevant to the third network input;

$f(t) = \left[f_{111}(t)\, f_{112}(t) \ldots f_{211}(t)\, f_{212}(t) \ldots f_{ijk}(t) \ldots f_{IJK}(t) \right]$ - vector of the time variable weight coefficients of the connections between the neurons from the second hidden layer and the output neuron (node) of the fuzzy rule-based neural network.

The following assumptions have been used in this investigation:

Both, the input signals $x_1(t)$, $x_2(t)$ and $x_3(t)$, and their time derivatives $\dot{x}_1(t)$, $\dot{x}_2(t)$ and $\dot{x}_3(t)$ will be considered bounded:

$$x_1(t) \le B_x,\ x_2(t) \le B_x,\ x_3(t) \le B_x \quad \forall t \tag{8}$$

$$\dot{x}_1(t) \le B_{\dot{x}},\ \dot{x}_2(t) \le B_{\dot{x}},\ \dot{x}_3(t) \le B_{\dot{x}} \quad \forall t \tag{9}$$

where B_x and $B_{\dot{x}}$ are known positive constants.

The parameters of the Gaussian membership functions are also bounded as follows:

$$|\sigma_A| \le B_\sigma;\ |\sigma_B| \le B_\sigma;\ |\sigma_C| \le B_\sigma;$$

$$|c_A| \le B_c;\ |c_B| \le B_c;\ |c_C| \le B_c \tag{10}$$

where B_σ and B_c are known positive constants.

It follows then from (5) and (7) that the normalized value of the output signals of the neurons from the second hidden layer will be also bounded:

$$\left|\bar{W}_{ijk}\right| \le B_{\bar{W}} \tag{11}$$

where the positive constant $B_{\bar{W}}$ is bounded by the following inequality:

$$B_{\bar{W}} \le \exp\left[-3\frac{(B_x - B_c)^2}{B_\sigma^2} \right] \tag{12}$$

It is also assumed that the elements of the vector $f(t)$ are bounded at each moment of time t by a given known positive constant B_f.

$$\left|f_{ijk}(t)\right| \le B_f \tag{13}$$

The scalar signal $y_d(t)$ represents the time-varying desired output of the neural network. It will be assumed that $y_d(t)$ and $\dot{y}_d(t)$ are bounded signals, i.e.,

$$\left|y_d(t)\right| \le B_{y_d},\ \left|\dot{y}_d(t)\right| \le B_{\dot{y}_d} \quad \forall t \tag{14}$$

where B_{y_d} and $B_{\dot{y}_d}$ are positive constants.

2.3 The Sliding Mode Learning Algorithm

Let us define the learning error of the fuzzy-neural network as the difference between the network's current output $y(t)$ and its desired value $y_d(t)$:

$$e(t) = y(t) - y_d(t) \tag{15}$$

Using the theory of Sliding Mode Control of Variable Structure Systems (Utkin, 1992) the zero value of the learning error coordinate $e(t)$ can be defined as time-varying sliding surface, i.e.,

$$s(e(t)) = e(t) = y(t) - y_d(t) = 0 \tag{16}$$

Condition (16) guarantees that the output $y(t)$ of the neuro-fuzzy network coincides with the desired output signal $y_d(t)$ for all time $t > t_h$, where t_h is the hitting time of $e = 0$.

Definition: A sliding motion will have place on a sliding manifold $s(e(t)) = e(t) = 0$ after a time t_h, if the condition $s(t)\dot{s}(t) = e(t)\dot{e}(t) < 0$ is satisfied for all t in some nontrivial semi-open subinterval of time of the form $[t, t_h) \subset (-\infty, t_h)$.

It is desired to devise a dynamical feedback adaptation mechanism, or online learning algorithm for the neuro-fuzzy network parameters such that the sliding mode condition of the above definition is enforced.

Theorem: If the learning algorithm for the parameters of the membership functions with a Gaussian distribution is chosen respectively as:

$$\dot{c}_{A_i} = -\frac{\sigma_{A_i}}{\sigma_A^T \sigma_A} \alpha sign(e) \tag{17}$$

$$\dot{c}_{B_j} = -\frac{\sigma_{B_j}}{\sigma_B^T \sigma_B} \alpha sign(e) \tag{18}$$

$$\dot{c}_{C_k} = -\frac{\sigma_{C_k}}{\sigma_C^T \sigma_C} \alpha sign(e) \tag{19}$$

$$\dot{\sigma}_{A_i} = -\frac{s_{A_i}}{s_A^T s_A} \alpha sign(e);\ s_{A_i} = x_1 - c_{A_i};\ s_A = \left[s_{A_1}\, s_{A_2} \ldots s_{A_I} \right]^T \tag{20}$$

$$\dot{\sigma}_{B_j} = -\frac{s_{B_j}}{s_B^T s_B} \alpha\, sign(e);\ s_{B_j} = x_2 - c_{B_j};\ s_B = \left[s_{B_1}\, s_{B_2} \ldots s_{B_J} \right]^T \tag{21}$$

$$\dot{\sigma}_{C_k} = -\frac{s_{C_k}}{s_C^T s_C} \alpha\, sign(e);\ s_{C_k} = x_3 - c_{C_k};\ s_C = \left[s_{C_1}\, s_{C_2} \ldots s_{C_K} \right]^T \tag{22}$$

and the adaptation of the connection weights between the second hidden layer and the output layer of the neuro-fuzzy network is chosen as follows:

$$\dot{f}_{ijk} = -\frac{\bar{W}_{ijk}}{\bar{W}^T \bar{W}} \alpha sign(e) \tag{23}$$

with $$\bar{W}^T = \left[\bar{W}_{111}\, \bar{W}_{112} \ldots \bar{W}_{IJK} \right] \tag{24}$$

where α is a sufficiently large positive number satisfying the inequality

$$\alpha > \frac{2nB_{\bar{W}} B_r B_f \left(1 + nB_{\bar{W}}\right) + B_{\dot{y}d}}{1 - 4nB_{\bar{W}} B_q B_f \left(1 + nB_{\bar{W}}\right)} \tag{25}$$

where $n = I \times J \times K$,

then, given an arbitrary initial condition $e(0)$, the learning error $e(t)$ will converge to zero during a finite time.

Proof: Consider the following Lyapunov function candidate:

$$V(e(t)) = \frac{1}{2} e^2(t) \tag{26}$$

The time derivative of $V(e(t))$ is given by

$$\dot{V}(e(t)) = e\dot{e} = e(\dot{y} - \dot{y}_d) = e\left[\frac{d}{dt}\left(\sum_i \sum_j \sum_k f_{ijk} \bar{W}_{ijk} \right) - \dot{y}_d \right] =$$

$$= e\left[\sum_i \sum_j \sum_k \left(\dot{f}_{ijk} \bar{W}_{ijk} + f_{ijk} \dot{\bar{W}}_{ijk} \right) - \dot{y}_d \right] \tag{27}$$

It can be easily shown that

$$\dot{\bar{W}}_{ijk} = -2\bar{W}_{ijk} K_{ijk} + 2\bar{W}_{ijk} \sum_i \sum_j \sum_k \left(\bar{W}_{ijk} K_{ijk} \right) \tag{28}$$

where

$$K_{ijk} = A\dot{A} + B\dot{B} + C\dot{C} \tag{29}$$

and

$$A = \frac{x_1 - c_{A_i}}{\sigma_{A_i}},\ B = \frac{x_2 - c_{B_j}}{\sigma_{B_j}},\ C = \frac{x_3 - c_{C_k}}{\sigma_{C_k}} \tag{30}$$

Then $\dot{V}(e(t))$ can be further expressed as follows:

$$\dot{V} = e\left\{ \sum_i \sum_j \sum_k \left[\dot{f}_{ijk} \bar{W}_{ijk} + f_{ijk} \left(-2\bar{W}_{ijk} K_{ijk} + 2\bar{W}_{ijk} \sum_i^3 \sum_j^3 \sum_k^3 \bar{W}_{ijk} K_{ijk} \right) \right] - \dot{y}_d \right\} =$$

$$= e\left\{ \sum_i \sum_j \sum_k \dot{f}_{ijk} \bar{W}_{ijk} - 2\sum_i \sum_j \sum_k \bar{W}_{ijk} \left(A\dot{A} + B\dot{B} + C\dot{C} \right) f_{ijk} + \right.$$

$$\left. + 2\sum_i \sum_j \sum_k \left[\bar{W}_{ijk} f_{ijk} \sum_i \sum_j \sum_k \bar{W}_{ijk} \left(A\dot{A} + B\dot{B} + C\dot{C} \right) \right] - \dot{y}_d \right\} =$$

$$= e\left\{ -\alpha\, sign(e) - 2\sum_i \sum_j \sum_k \bar{W}_{ijk}\, f_{ijk} \left(\frac{A}{\sigma_{A_i}^2} \left(\dot{x}_1 \sigma_{A_i} + 2\alpha\, sign(e) \right) + \right. \right.$$

$$+\frac{B}{\sigma_{B_j}^2}\left(\dot{x}_2\sigma_{B_j}+2\alpha\, sign(e)\right)+\frac{C}{\sigma_{C_k}^2}\left(\dot{x}_3\sigma_{C_k}+2\alpha\, sign(e)\right)\Big)+$$

$$+2\sum_i\sum_j\sum_k\Bigg[\bar{W}_{ijk}f_{ijk}\sum_i\sum_j\sum_k\bar{W}_{ijk}\Bigg(\frac{A}{\sigma_{A_i}^2}\left(\dot{x}_1\sigma_{A_i}+2\alpha\, sign(e)\right)+$$

$$+\frac{B}{\sigma_{B_j}^2}\left(\dot{x}_2\sigma_{B_j}+2\alpha\, sign(e)\right)+\frac{C}{\sigma_{C_k}^2}\left(\dot{x}_3\sigma_{C_k}+2\alpha\, sign(e)\right)\Bigg)\Bigg]-\dot{y}_d\Bigg\}=$$

$$=e\Bigg[-\alpha\, sign(e)-2\sum_i\sum_j\sum_k\bar{W}_{ijk}f_{ijk}r_{ijk}-4\alpha\, sign(e)\sum_i\sum_j\sum_k\bar{W}_{ijk}f_{ijk}q_{ijk}+$$

$$+2y\sum_i\sum_j\sum_k\bar{W}_{ijk}r_{ijk}+4y\alpha\, sign(e)\sum_i\sum_j\sum_k\bar{W}_{ijk}q_{ijk}-\dot{y}_d\Bigg]=$$

$$=e\Bigg[-\alpha sign(e)-2\sum_i\sum_j\sum_k\bar{W}_{ijk}r_{ijk}\left(f_{ijk}-y\right)-$$

$$-4\alpha\, sign(e)\sum_i\sum_j\sum_k\bar{W}_{ijk}q_{ijk}\left(f_{ijk}-y\right)-\dot{y}_d\Bigg]=$$

$$=\Bigg[-\alpha-4\alpha\sum_i\sum_j\sum_k\bar{W}_{ijk}q_{ijk}\left(f_{ijk}-y\right)\Bigg]|e|-$$

$$-\Bigg[2\sum_i\sum_j\sum_k\bar{W}_{ijk}r_{ijk}\left(f_{ijk}-y\right)+\dot{y}_d\Bigg]e\le$$

$$\le-\alpha|e|+4\alpha nB_{\bar{W}}B_q\left(B_f+nB_{\bar{W}}B_f\right)|e|+$$

$$+2|e|nB_{\bar{W}}B_r\left(B_f+nB_{\bar{W}}B_f\right)+B_{\dot{y}_d}|e|=$$

$$\le|e|\Big\{-\alpha\left[1-4nB_{\bar{W}}B_qB_f\left(1+nB_{\bar{W}}\right)\right]+$$

$$+2nB_{\bar{W}}B_rB_f\left(1+nB_{\bar{W}}\right)+B_{\dot{y}_d}\Big\}<0 \qquad (31)$$

where r_{ijk} and q_{ijk} are defined as follows:

$$r_{ijk}=\frac{A}{\sigma_{A_i}}\dot{x}_1+\frac{B}{\sigma_{B_j}}\dot{x}_2+\frac{C}{\sigma_{C_k}}\dot{x}_3 \qquad (32)$$

$$q_{ijk}=\frac{A}{\sigma_{A_i}^2}+\frac{B}{\sigma_{B_j}^2}+\frac{C}{\sigma_{C_k}^2} \qquad (33)$$

$$\left|r_{ijk}\right|\le B_r,\ \ \left|q_{ijk}\right|\le B_q \qquad (34)$$

and the positive constants B_r and B_q are bounded by the following inequalities:

$$B_r\le 3B_{\dot{x}}\frac{B_x+B_c}{B_\sigma^2},\quad B_q\le 3\frac{B_x+B_c}{B_\sigma^3} \qquad (35)$$

The inequality (31) means that the controlled trajectories of the learning error $e(t)$ converge to zero in a stable manner.

3. APPLICATION TO ONLINE LEARNING OF NONLINEAR FUNCTION

In order to demonstrate the functionality of the proposed learning algorithm a simulation experiment has been carried out in the Matlab/Simulink programming environment. A fuzzy rule-based neural network has been trained with the proposed SMC based algorithm to approximate the function:

$$F(x)=e^{-\frac{x}{10}}\sin(3x) \qquad (36)$$

The latter is commonly used as a benchmark since it is non-monotonic.

The neuro-fuzzy network topology used in the experiment has been with one input and the argument x of the above function has been used to feed it. The network input has been associated with three fuzzy labels defined by Gaussian membership functions with initial values of the parameters $\sigma_A=\sigma_B=\sigma_C=1$, $c_A=-0.5$, $c_B=0.2$ and $c_C=1$. Small random initial values have been generated for the weight coefficients f_{ijk}.

The obtained results are presented on Fig. 2 and Fig. 3. As it can be seen from Fig. 2, the learning error converges rapidly towards close to zero values.

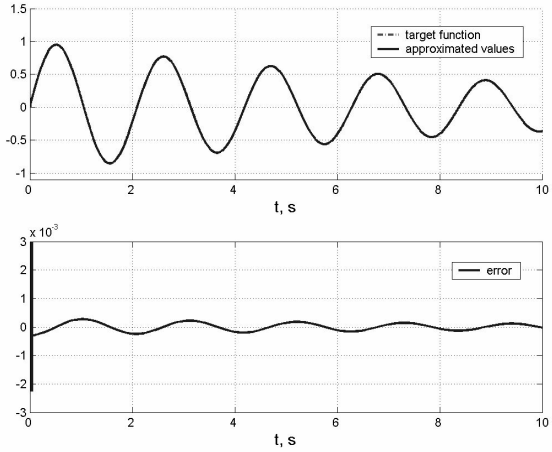

Fig. 2. Online learning of a decaying sinusoidal function. The approximated function is plotted with dashed line, the output of the FNN with SMC learning algorithm is plotted with solid line on the same plot. The error is separately plotted with solid line.

Fig. 3 illustrates the changes that occurred in the membership functions which initial shapes and locations are shown with dashed lines and those at the end of the learning period – with solid lines.

As it can be seen, the fuzzy rule-based neural network has an adaptive behaviour by adjusting itself to track the function

presented. The implementation of the sliding mode concept has introduced speed-up in network learning and after a small period of time the network error is very small.

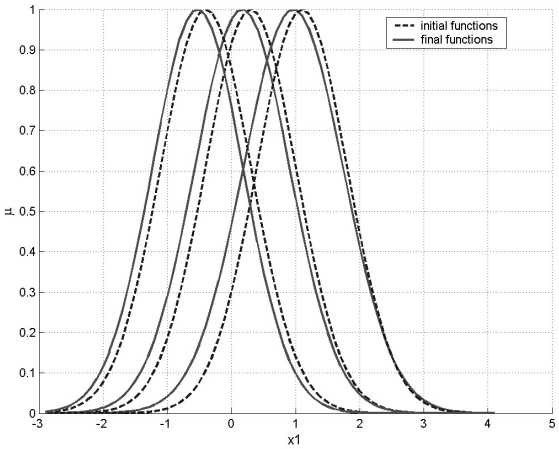

Fig. 3. Changes in the shapes and locations of the Gaussian membership functions associated with the input of the neuro-fuzzy network during the approximation of decaying sinusoidal function. Initial values are presented with dashed lines.

4. CONCLUSIONS

In this paper a new learning algorithm has been proposed for fuzzy rule-based neural networks with scalar output which robustly drives the learning error to zero in finite time. The weights adaptation scheme is based on sliding mode control concept and it represents a simple, yet robust, mechanism for guaranteeing finite time reachability of zero learning error condition. The convergence of the algorithm has been analyzed and simulation results have been presented to show its effectiveness. In contrast with off-line learning algorithms, the algorithm proposed can be used to train the network as it interacts with the external environment.

5. ACKNOWLEDGEMENTS

The work of A. V. Topalov and N. G. Shakev was supported by the Ministry of Education and Science of Bulgaria Research Fund Project BY-TH-108/2005. The work of O. Kaynak was supported in part by the Bogazici University Research Fund Project 03A202 and in part by the TUBITAK Project 100E042.

REFERENCES

Aliev, R.A., B. Fazlollahi and R.M. Vahidov (2001). Genetic algorithm-based learning of fuzzy neural networks. Part I: Feed-forward fuzzy neural networks. *Fuzzy Sets Syst*., **vol**. **118**, **no**. **3**, pp. 351-358.

Farag W.A., V.H. Quintana and G. Lambert-Torres (1998). A genetic-based neuro-fuzzy approach for modeling and control of dynamic systems. *IEEE Trans. Neural Networks*, **vol**.**9**, pp. 756-767.

Kosmatopoulos E.B., M.M. Polycarpou, M.A. Christodoulou and P.A. Ioannou (1995). High-order neural network structures for identification of dynamical systems. *IEEE Trans. on Neural Networks*, **vol**. **6**, **no**. **2**, pp.431-442.

Lin C.T. and C.S. George Lee (1996). *Neural Fuzzy Systems*. Englewood Cliffs, NJ.

Lin F.-J., R.F. Fung and R.-J. Wai (1998). Comparison of sliding mode and fuzzy neural network control for motor-toggle servomechanism. *IEEE/ASME Trans. Mechatron*., **vol**. **3**, pp. 302-318.

Lin F.-J, K.-K Shyu and R.-J. Wai (2001). Recurrent-fuzzy-neural-network sliding-mode controlled motor-toggle servomechanism. *IEEE/ASME Trans. Mechatron*., **vol**. **6**, **no**. **4**, pp. 453-466.

Narendra K.S. and K. Parthasarathy (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. Neural Networks*, **no**. **1**, pp. 4-27.

Parma G.G., B.R. Menezes and A.P. Braga (1998). Sliding mode algorithm for training multilayer artificial neural networks. *Electronics Letters*, **vol**. **34**, **no**. **1**, pp. 97-98.

Rumelhart D., D. Hinton and G. Williams (1986). Learning internal representations by error propagation, In: *Parallel Distributed Processing. 1*, D. Rumelhart and F. McClelland eds., pp. 318-362. Cambridge, MA, MIT Press.

Shakev N.G., A.V. Topalov and O. Kaynak (2003). Sliding mode algorithm for on-line learning in analog multilayer feedforward neural networks. In: *Artificial Neural Networks and Neural Information Processing. Lecture Notes in Computer Science vol. 2714*, pp. 1064-1072. Springer-Verlag.

Suykens J.A.K, J. Vandewalle and B. De Moor (1999). Lur'e systems with multilayer perceptron and recurrent neural networks: absolute stability and dissipativity. *IEEE Trans. on Automatc Control*, **vol**. **44**, pp. 770-774.

Wang L.X. (1997). *A course in Fuzzy Systems and Control*. Prentice-Hall, Englewood Cliffs, NJ.

Wang W.Y., T.T. Lee, C.L. Liu and C.H. Wang (1997). Function approximation using fuzzy neural networks with robust learning algorithm. *IEEE Trans. Systems, Man, and Cybernetics – Part B*, **vol**. **27**, pp. 740-747.

Wu S.Q. and M.J. Er (2000). Dynamic fuzzy neural networks – a novel approach to function approximation. *IEEE Trans. on Systems, Man and Cybernetics*, *Part B*, **vol**. **30**, pp. 358-364.

Yu S., X. Yu and Z. Man (2004). A fuzzy neural network approximator with fast terminal sliding mode and its applications. *Fuzzy Sets and Systems*, **vol**. **148**, pp. 469-486.

Utkin V. I. (1992). *Sliding Modes in Control Optimization*, Springer-Verlag, New York.