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

Nonlinear System Identification and Control Using an Input-Output Recurrent Neurofuzzy Network ⋆

Marcos Ángel González-Olvera * **Yu Tang** **

* *National Autonomous University of Mexico (UNAM). Engineering Faculty (e-mail:mangel@verona.fi-p.unam.mx). Tel (52)-55-56223013*
** *(e-mail:tang@servidor.unam.mx)*

Abstract: In this work we develop an input-output recurrent neurofuzzy network in discrete-time for identification and control of nonlinear systems. The structure is linear in the consequent parameters and nonlinear in the antecedent ones. The training of the antecedent parameters is achieved by linearizing them around a suboptimal value, assuming that the only known data are input-output signals obtained directly from measurements of the system, as well as some information about its structure (local stability and time delays). The training algorithm is based on a Kalman filter, stable under certain assumptions. It is also presented a theorem to check the stability of the resulting network in the Lyapunov sense, and a predictive control design. The performance of the network is shown by the identification and control of a nonlinear benchmark system.

Keywords: Identification, Neural-Network Models, Predictive Control, Fuzzy Systems, Neural Control

1. INTRODUCTION

Neural networks and fuzzy systems have shown, as noted by Cybenko [1989], to be very useful tools in identification due to their approximation capabilities when dealing with piece-wise continuous functions. For this reason, they have been used modeling and designing control systems, such as in Narendra and Parthasarathy [1990], where their ability to learn and adapt have been exploited.

Nevertheless, neurofuzzy networks (regarded as an interpretation of fuzzy systems as multilayer neural networks) have been used not only as static input-output functions, but also as dynamic models by adding feedback connections in some or all layers within the network. This architecture is known as *recurrent* or *dynamic* network, and are assumed to be useful when modeling and identifying dynamic systems. One of the very first approaches was given by Hopfield [1982].

The structural definition of neural networks depends on the problem to be solved. Narendra and Parthasarathy [1990] proposed networks with feedback within the internal layers, Weibel et al. [1989] used time-delay networks and Williams and Zipser [1989] used fully recurrent networks, where all layers are considered to be inputs. Networks such as globally static-locally recurrent structures have been proposed by Weibel et al. [1989], Wan [1990], Back and Tsoi [1990], using neurons with linear filters in the synapsis.

However, Mastorocostas and Theocharis [2002] have noted that these structures do not guarantee the stability of the training algorithms. Also, their low convergence speed and the eventual parameter explosion (known as the *curse of dimensionality*) are challenges to be faced when training. In order to overcome the previous challenges, some authors have proposed simpler structures by focusing on the specific task to be analyzed. For example, on control and identification tasks, Mouzouris and Mendel [1997] and Jang [1992] proposed externally recurrent networks; and Gorrini and Bersini [1994] studied structures with with internal feedback via fuzzy systems.

Other authors have proposed nonlinear structures for identification with a state-space representation, taking into account that state variables are often used in the control area. In this sense, Juang [2002] proposed to consider internal states to model the dynamics.

However, not many structures take into account the stability of the resulting architecture as well as the algorithm convergence. Regarding training, general algorithms such as Backpropagation-Trough-Time (first seen in Rumelhart et al. [1986]) and Real-time recurrent learning (proposed by Williams and Zipser [1989] as a simplification of the previous algorithm) have been used. As these algorithms are slow and lack stability, linear approximations such as recursive-least-squares (cited by Johansson [1993]) and Kalman filters (some described by Haykin [2001]) have been proposed and successfully applied. Authors as Poznyak et al. [2006], Yu and Li [2004], Mastorocostas and Theocharis [2002] guarantee stable training algorithms, relying the specific structure of the network.

⋆ This work is partly supported by UNAM-PAPIIT IN106206. The work of Marcos A. González Olvera is supported by DGEP-UNAM.

In this paper these challenges are considered: the presented structure is a recurrent neurofuzzy network with an input-output representation and state-space interpretation, as a great part of the control theory is based in this representation. Having this in mind, Gonzalez Olvera and Tang [2007] focused on a Controllable-Form Recurrent NeuroFuzzy Network (CReNN). This scheme is stated to be simple in its analysis, with easiness in the interpretation of the resulting parameters. In this paper, a modification of this scheme to an Input-Output CReNN (IOCReNN) is considered, in order to make the training and its application for predictive control more transparent, in which only the signals to be measured and controlled are considered in the structure.

This paper is organized in the following sections: in Section 2 the proposed modified IOCReNN structure is presented and its stability in the sense of Lyapunov properties and learning algorithm are discussed. Also, a theorem for a predictive control is stated. Examples of the performance of the neural identifier are shown in Section 3 by identifying and controlling a nonlinear system. Finally, conclusions and future work are drawn in Section 4.

2. INPUT-OUTPUT RECURRENT NEUROFUZZY NETWORK

Let the following system to be identified with input u_k and output y_k, modeled by the difference equation be

$$y_{k+1} = f_d(y_k, \ldots, y_{k-n+1}, u_k, \ldots, u_{k-m+1}). \quad (1)$$

The following assumptions are made for this system

Assumption 1. For (1):

- The system is locally observable.
- y_k and u_k, for $k = 0 \ \ldots \ N$ are available.
- There is no direct measurement of the actual physical states.
- The function f_d is continuous.
- The order n and m of time-delays of the output and input, respectively, are estimated or known *a priori*.

The objective is to find a mapping $\hat{f}(\cdot)$ such that $\sup_k ||\hat{y}_k(\cdot) - y_k||$ be as small as possible. So, it is stated that the following neurofuzzy network with n_R rules

R_i: IF $\hat{y}_k$ is A_i THEN

$$\hat{y}_{k+1} = [\, c_{i1} \ldots c_{in} \,] \begin{bmatrix} \hat{y}_k \\ \vdots \\ \hat{y}_{k-n+1} \end{bmatrix} + [\, h_{i1} \ldots h_{im} \,] \begin{bmatrix} u_k \\ \vdots \\ u_{k-m+1} \end{bmatrix} \quad (2)$$

$$= \mathbf{c}_i^T \begin{bmatrix} \hat{y}_k \\ \vdots \\ \hat{y}_{k-n+1} \end{bmatrix} + \mathbf{h}_i^T \begin{bmatrix} u_k \\ \vdots \\ u_{k-m+1} \end{bmatrix} \quad (3)$$

can identify the system, where $\hat{y}$ is the estimate of the output signal, $A_i(x) = \exp(-\sigma_i^2(x-\varsigma_i)^2)$ are the membership functions of the antecedent part of the fuzzy rules, and the defuzzification is made by a weighted average. By rewriting the previous definition for the fuzzy rules, the system can be described by the following equations

$$\begin{aligned} \hat{y}_{k+1} &= \left(\varphi_k^1 \mathbf{c}_1^T + \ldots + \varphi_k^n \mathbf{C}_{n_R}^T\right) \mathbf{z}_k + \\ &\quad \left(\varphi_k^1 \mathbf{h}_1^T + \ldots + \varphi_k^n \mathbf{h}_{n_R}^T\right) \xi_k \\ &= (\varphi_k^T \otimes \mathbf{z}_k)\text{vect}(\mathbf{C}) + \\ &\quad (\varphi_k^T \otimes \xi_k)\text{vect}(\mathbf{H}) \end{aligned} \quad (4)$$

where $\otimes$ is the Kronecker matrix product, vect$(\cdot)$ is the row vectorization into a single column of the argument matrix, and the signals are given by

$$\mathbf{z}_k = [\hat{y}_k \ \ \hat{y}_{k-1} \ldots \hat{y}_{k-n+1}]^T \quad (5)$$

$$\xi_k = A_0 \xi_{k-1} + B_0 u_k \quad (6)$$

$$R_k^i = e^{-\sigma_i^2(\hat{y}_k - \varsigma_i)^2} \quad (7)$$

$$\varphi_k^i = \frac{R_k^i}{\sum_j R_k^j} \quad (8)$$

$$\mathbf{C} = [\, \mathbf{c}_1 \ldots \mathbf{c}_n \,]^T \quad (9)$$

$$\mathbf{H} = [\, \mathbf{h}_1 \ldots \mathbf{h}_m \,]^T \quad (10)$$

$$A_0 = \begin{bmatrix} 0_{1\times m} \\ I_{(m-1)\times(n-1)} \ 0_{(m-1)\times 1} \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 1 \\ 0_{(m-1)\times(m-1)} \end{bmatrix}$$

The parameters to be identified are

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_1^T \\ \vdots \\ \mathbf{c}_n^T \end{bmatrix}, \ \mathbf{H} = \begin{bmatrix} \mathbf{h}_1^T \\ \vdots \\ \mathbf{h}_n^T \end{bmatrix}, \ \sigma = \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_n \end{bmatrix}, \ \varsigma = \begin{bmatrix} \varsigma_1 \\ \vdots \\ \varsigma_n \end{bmatrix},$$

with the rule functions defined as

$$\begin{aligned} R_k^i &= e^{-\sigma_i^2(y_k - \varsigma_i)^2} && \text{rule activation} \\ \varphi_k^i &= \frac{\mu_{R_k^i}}{\sum_j \mu_{R_k^j}} && \text{normalized activation} \end{aligned}$$

Given that the approximation to the real system with output signal y_{k+1} is given by

$$y_{k+1} = f_1(\mathbf{C}, \sigma, \varsigma, \mathbf{z}_k) + f_2(\mathbf{H}, \sigma, \varsigma, \xi_k) + \bar{\zeta}_k \quad (11)$$

$$= f(\mathbf{C}, \mathbf{H}, \sigma, \varsigma, \mathbf{z}_k, \xi_k) + \bar{\zeta}_k \quad (12)$$

The approximation error $\bar{\zeta}_k$ will be minimum ($\zeta_k = \min_\theta \bar{\zeta}_k$) if the optimal values for $\mathbf{C}^T$, $\mathbf{H}^T$, σ y ς are found, i.e.

$$y_{k+1} = f_1(\mathbf{C}^*, \sigma^*, \varsigma^*, \mathbf{z}_k) + f_2(\mathbf{H}^*, \sigma^*, \varsigma^*, \xi_k) + \zeta_k \quad (13)$$

$$= f(\mathbf{C}^*, \mathbf{H}^*, \sigma^*, \varsigma^*, \mathbf{z}_k, \xi_k) + \zeta_k \quad (14)$$

2.1 Stability analysis

The network stability analysis in the Lyapunov sense can be further made if the network is rewritten as a state-space representation. It can be proven that its form is (taking $u_k \equiv 0$).

$$\mathbf{z}_{k+1} = \sum_i \varphi_k^i \begin{bmatrix} \mathbf{c}_i^T \\ I_{(n-1)\times(n-1)} \;\; 0_{(n-1)\times 1} \end{bmatrix} \mathbf{z}_k \quad (15)$$

In this way, the stability can be analyzed by using the Theorem for Standard Aditive Models (SAM) proposed by Kosko [1992], that enounces:

Theorem 1. Let a fuzzy dynamic system be defined with a linear system in state-space representation in each fuzzy rule, with D_i as the system matrix for the i-th subsystem. The system is stable in the sense of Lyapunov if $\exists\ P = P^T > 0$ such that the matrix inequalities

$$D_i P D_i - P < 0 \quad (16)$$

hold $\forall\ i = 1, \ldots, n_R$, where n_R is the number of rules in the fuzzy system.

2.2 Linearization of the antecedent parameters

The accessibility to the antecedent parameters is not an easy task as there are nonlinear relations to the output and the error signal $e_k = y_k - \hat{y}_k$. If the network is near a local optimum for the antecedent parameters, then the algorithm of linearization proposed by Yu and Li [2004] can be used. For this purpose, an initialization algorithm will be discussed in Section 2.3.

Taking the general continuous function $h(x)$, the Taylor expansion theorem states that it can be expressed as $h(x) = h(x_0) - \frac{\partial h}{\partial x}\Big|_{x=x_0} + R(x, x_0)$, where $R(x, x_0)$ are the higher-order terms in the expansion. In this sense, we propose to expand the optimal IOCReNN (that is, with the parameters $\theta = \theta^*$), around a suboptimal value θ. So, the function $f = f(\theta^*)$ defining the optimal network will be expressed, around its parameters θ, as

$$f = f(\theta^*) = f(\theta) + (\theta^* - \theta)^T \left.\frac{\partial f}{\partial \theta^*}\right|_{\theta^*=\theta} + R(\theta, \theta^*) \quad (17)$$

where $\theta = \left[\, \mathrm{vect}(\mathbf{C})^T \;\; \mathrm{vect}(\mathbf{H})^T \;\; \sigma^T \;\; \varsigma^T \,\right]$ is the vector that contains all parameters to be identified, and $R(\theta, \theta^*)$ represents the higher-order terms in the expansion. Given that direct measurements of $y_k, \ldots, y_{k-n+1}$ are available, it can be considered for training that
$\mathbf{z}_k = [y_k \;\ldots\; y_{k-n+1}]^T$.

By developing (14) and given that f is linear in the consequent parameters $\mathbf{C}$, $\mathbf{H}$, the expansion can be written as

$$\begin{aligned} f(\mathbf{C}^*, \mathbf{H}^*, \sigma^*, \varsigma^*, \mathbf{z}_k, \xi_k) = {} & f(\mathbf{C}, \mathbf{H}, \sigma, \varsigma, \mathbf{z}_k, \xi_k) \\ & + f(\mathbf{C}^* - \mathbf{C}, \mathbf{H}^* - \mathbf{H}, \sigma, \varsigma, \mathbf{z}_k, \xi_k) \\ & + (\sigma^* - \sigma)^T \left.\frac{\partial f}{\partial \sigma^*}\right|_{\sigma,\varsigma,\mathbf{c}} \\ & + (\varsigma^* - \varsigma)^T \left.\frac{\partial f}{\partial \varsigma^*}\right|_{\sigma,\varsigma,\mathbf{c}} + R \end{aligned} \quad (18)$$

Given that $y_{k+1} = f(\mathbf{C}^*, \mathbf{H}^*, \sigma^*, \varsigma^*, \mathbf{z}_k, \xi_k) + \zeta_k$ and $\hat{y}_{k+1} = f(\mathbf{C}, \mathbf{H}, \sigma, \varsigma, \mathbf{z}_k, \xi_k)$, subtracting both equations,

$$\begin{aligned} \tilde{y}_{k+1} \triangleq y_{k+1} - \hat{y}_{k+1} = {} & \tilde{\sigma}^T \mathbf{g}_k + \tilde{\varsigma} \mathbf{h}_k \\ & + \varphi^T (\mathbf{C}^* - \mathbf{C})\, \mathbf{z}_k + \varphi^T (\mathbf{H}^* - \mathbf{H})\, \xi_k + \vartheta_k \end{aligned} \quad (19)$$

where $\vartheta_k = R - \zeta_k$, $\mathbf{g}_k = \left.\frac{\partial f}{\partial \sigma^*}\right|_{\sigma,\varsigma}$, $\mathbf{h}_k = \left.\frac{\partial f}{\partial \varsigma^*}\right|_{\sigma,\varsigma}$.

Rewriting the equations, the following linearized model respect to the parameters is obtained

$$\tilde{y}_{k+1} = \tilde{\theta}^T \mathbf{G}_k + \vartheta_k = \tilde{\theta}_k^T \mathbf{G}_k + \vartheta_k \quad (20)$$

where

$$\tilde{\theta} \triangleq \theta^* - \theta \quad (21)$$

$$\theta = \begin{bmatrix} \mathrm{vect}(\mathbf{C}) \\ \mathrm{vect}(\mathbf{H}) \\ \sigma \\ \varsigma \end{bmatrix} \quad (22)$$

$$\mathbf{G}_k = \begin{bmatrix} \varphi \otimes \mathbf{z}_k \\ \varphi \otimes \xi_k \\ \mathbf{g}_k \\ \mathbf{h}_k \end{bmatrix} \quad (23)$$

The equation (2.2) express the calculus of the partial derivatives of the function with respect to the parameters σ. In the same way, the partial derivatives $\left.\frac{\partial f_1}{\partial \sigma^*}\right|_{\sigma^*=\sigma} = \frac{\partial \varphi^T}{\partial \sigma} \mathbf{C}\mathbf{z}(k)$ of the function with respect to ς can be obtained, but due to space limitations are not shown.

2.3 Parameter Initialization Algorithm

The objective in the parameter initialization algorithm is to put the parameters near a *good* local minimum, in order to make $||\theta^* - \theta||$, and consequently R, as small as possible.

The proposed algorithm is based the work made by Gonzalez Olvera and Tang [2007], which can be summarized in the following steps

(1) Train a static fuzzy system by using ANFIS (developed by Jang [1993]), whose output is given by the known outputs y_k, and its inputs are n and m time-delays of the output and input u_k respectively.
(2) As the trained static fuzzy system has $n + m$ input fuzzy sets and n_R fuzzy rules, initialize the IOCReNN with n_R rules and n *states*.
(3) Initialize the A_i fuzzy sets with those corresponding to y_{k-1}in the static fuzzy system.
(4) As the obtained fuzzy system involves linear functions in each rule, obtain its consequent parameters and initialize $\mathbf{C}$ y $\mathbf{H}$ correspondingly.

Remark 1. The ANFIS algorithm is suggested to be used used as it involves both a fuzzy space partition and training of consequent parameters by an hybrid algorithml, but other algorithms that make similar tasks can be used.

2.4 Kalman-filter based training

Once the IOCReNN has been initialized, we make the following assumption

Assumption 2. If the nonlinear neurofuzzy network is initialized near a local minimum, the difference $\theta^* - \theta$ and R are also small.

$$\frac{\partial \varphi^T}{\partial \sigma} = 2 \begin{bmatrix} \frac{\sigma_1 (y_k - \varsigma_1)^2}{\left(\sum_j \mu_l\right)} \mu_1 \\ \vdots \\ \frac{\sigma_n (y_k - \varsigma_n)^2}{\left(\sum_j \mu_l\right)} \mu_n \end{bmatrix} \otimes \begin{bmatrix} \mu_1 & \dots & \mu_n \end{bmatrix} - 2diag \left(\begin{bmatrix} \frac{\mu_1}{\left(\sum_l \mu_l\right)} \left(\sigma_1 (y_k - \varsigma_1)^2\right) \dots \frac{\mu_n}{\left(\sum_l \mu_l\right)} \left(\sigma_n (y_k - \varsigma_n)^2\right) \end{bmatrix} \right) \tag{24}$$

With this in mind, the optimized linearized model (20) can be shown as

$$\tilde{\theta}_{k+1} = \tilde{\theta}_k \tag{25}$$

$$\tilde{y}_k = \mathbf{G}_k \tilde{\theta}_k^* + \zeta_k \tag{26}$$

There is no direct knowledge about $\tilde{\theta}$, and there exist some *perturbations* such as ζ_k and R_k whose effect must be taken into account. For these reasons, a Kalman filter-based algorithm was used to train the network and then find the estimates $\hat{\theta}$. The filter is defined by the equations

$$\hat{\theta}_k = \hat{\theta}_{k-1} + K_k(y_k - \mathbf{G}_k \hat{\theta}_{k-1}) \tag{27}$$

$$K_k = \frac{P_{k-1}\mathbf{G}_k^T}{R_2 + \mathbf{G}_k P_{k-1} \mathbf{G}_k^T} \tag{28}$$

$$P_k = P_{k-1} - \frac{P_{k-1}\mathbf{G}_k^T \mathbf{G}_k P_{k-1}}{R_2 + \mathbf{G}_k P_{k-1} \mathbf{G}_k^T} + R_1 \tag{29}$$

where K_k is the Kalman gain matrix, P_k is the conditional covariance matrix, $R_2 \leq 1$ is the forgetting factor, and $R_1 = R_1^T$ is a matrix that handles the covariance of the process noise (regarded as ζ and R). It is well known that if $R_1 = 0$, the traditional least-squares algorithm is recovered.

The applications of the Kalman filter to neural networks have proven to be very helpful, as they intend to provide with an approximation to the minimum variance estimate of the parameters in the linear case. In the nonlinear case, even though the algorithm may be caught into a local minima and then result in a non-optimal estimation, its convergence and performance (as studied and compared to other algorithms by Hagner et al. [2000]) have proven to be very helpful. Nevertheless, it must be noticed that even though the stability of the Kalman filter has been studied under certain conditions (Guo [1990]), it is still a difficult task to be accomplished, as noted by de Jesús Rubio and Yu [2007].

2.5 Predictive control

The trained IOCReNN can be used to control a previously identified system by using a predictive control architecture. Let the IOCReNN be rewritten as

$$\hat{y}_{k+1} = (\varphi_k^T \otimes \mathbf{z}_k)\text{vect}(\mathbf{C}) + \varphi_k^T(\mathbf{h}_{c1} u_k + \bar{\mathbf{H}} \bar{\xi}_k), \tag{30}$$

where $\mathbf{h}_{c1}$ is the first column of the matrix $\mathbf{H}$, $\bar{\mathbf{H}} = [\mathbf{h}_{c2} \dots \mathbf{h}_{cn_R}] = \bar{\mathbf{H}}$ is the matrix as $\mathbf{H}$ with the first column removed, and $\bar{\xi}_k = [u_{k-1} \cdots u_{k-m+1}]^T$. If after the training the network complies with $\varphi^T \mathbf{h}_{c1} \neq 0$ (it is supposed that the system has not singular points identified by the neural network), a model reference predictive control can be applied, so the next theorem is stated

Theorem 2. Predictive control design. Having that the IOCReNN defined by (30) has no singular points for its control (i.e. $\varphi^T \mathbf{h}_{c1} \neq 0 \ \forall \ k$), the following predictive control

$$u_k = \frac{1}{\varphi \mathbf{h}_{c1}} \left(- \left(\varphi_k^T \otimes \mathbf{z}_k\right) \text{vect}(\mathbf{C}) - \varphi^T \bar{\mathbf{H}} \bar{\xi}_k + y_{d(k+1)} \right). \tag{31}$$

makes the network achieve any desired output in finite time. If the control is applied to the real system, the tracking error $e_{k+1} = y_{d(k+1)} - y_{k+1}$ is bounded when the perturbations R and ζ are bounded.

Proof 1. When identifying the nonlinear system $y_{k+1} = f(\cdot)$, the network's approximation is given by Eq. (4). When applying the proposed control, the terms are cancelled and the resulting control makes the output in time $k+1$ equal to

$$y_{k+1} = y_{d(k+1)} + \vartheta_k \tag{32}$$

Under the assumption that both R_k and ζ_k are bounded, the control error will be bounded by $|e_{k+1}| = |\vartheta_k| \leq |R_k| + |\zeta_k|$.

As the identified parameters can not be assumed to be the ideal ones (the algorithm only searches near a local optimum), an extra PID compensation term is added to this control in order to deal with the uncertainties of the resulting network. In this sense, the proposed compensation control is

$$\bar{u}_k = u_k + \frac{1}{\varphi \mathbf{h}_{c1}} \left(-K_p e_k - K_i \sum_{j=1}^{k} e_j \right). \tag{33}$$

3. EXAMPLES

To analyze the identification capacity of the network, a linear and a nonlinear systems were identified. In both cases, the identification algorithm shown in Section 2.3 and a pseudo random input signal $|u| \leq 1.02$ were used, with 4,000 data for the first system and 8,000 data for the second one. The control scheme was used in the second system.

3.1 Linear system identification

The system to be identified is given by the transfer function in discrete time

$$H(z) = \frac{(z - 0.5)(z + 0.5)}{(z + 0.9)(z - 0.8 \pm 0.2i)} \tag{34}$$

In this case, it was decided to train a net by using the same information as the known system, with $n_R = 4$. The obtained results are shown in Figure 1, as well as the value of the roots of the characteristic polynomial for each linear system per rule.

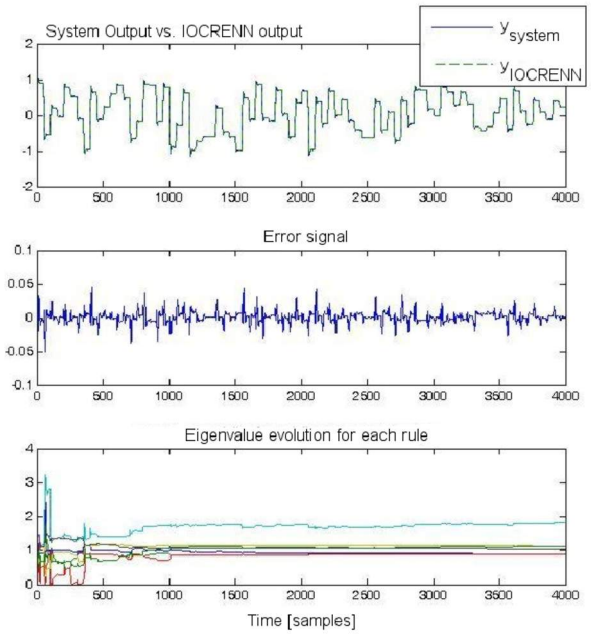

Fig. 1. Linear system identification

3.2 Nonlinear system identification

To test the identification performance of the IOCReNN, a nonlinear system proposed by Narendra and Parthasarathy [1990] was identified, which is given by the nonlinear difference equation

$$y_k = f_N(y_{k-1}, y_{k-2}, y_{k-3}, u_k, u_{k-1}) \tag{35}$$

where

$$f(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_3^2 + x_2^2} \tag{36}$$

In the identification structure and algorithm, $n = 3$, $m = 2$ and 4 rules were considered, resulting in 28 parameters to be trained. The results are shown in Fig. 2, as well as the evolution of the nonlinear parameters σ, ς and roots of each characteristic polynomial per rule. A RMS error of 0.03692 was achieved, comparable with previous results reported by Juang and Lin [1999], with RMS error of 0.0248 (using 36 parameters), and RMS of 0.0084 using a Genetic-Algorithm-based training and 33 parameters.

3.3 Nonlinear system control

By using the trained IOCReNN from the preceding section, the same system was controlled. After checking the nonexistence of singular points, a linear reference model was used. The results are shown in Fig. 3. As it can be seen, there exists a steady-state error; and the RMS error between the desired and obtained signal is 0.03945. By using the compensation PID control shown in (33), using the empirical values $K_p = 0.2$ and $K_i = 0.12$, the RMS error was reduced to 0.01744. The results for the compensation control are shown in Fig. 4.

4. CONCLUSIONS

This work has presented a recurrent neural network-based structure for identification and control of nonlinear systems, with a state-space representation and analysis but

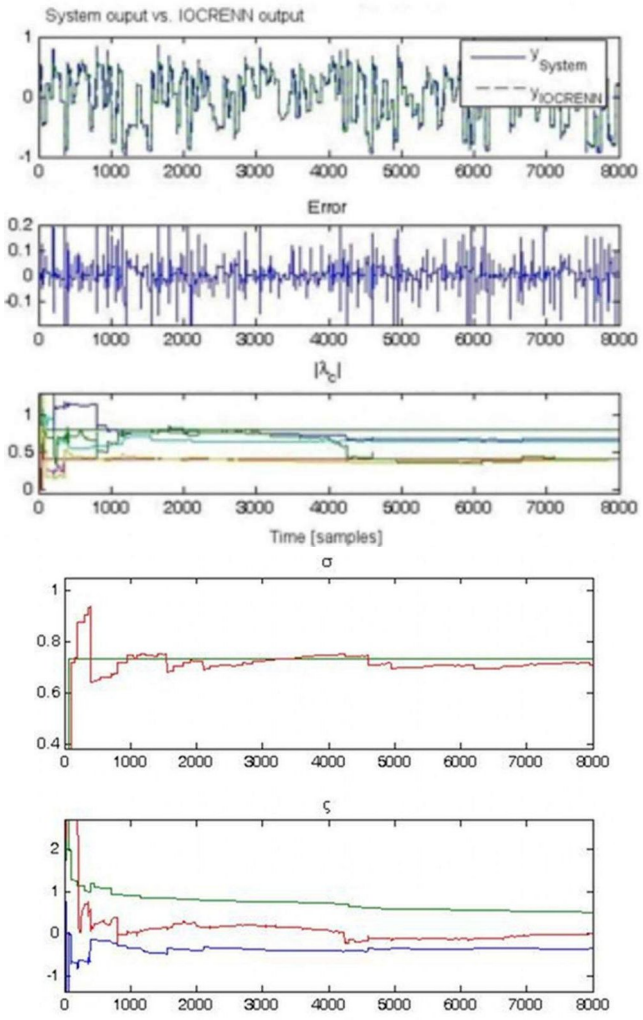

Fig. 2. Nonlinear system identification

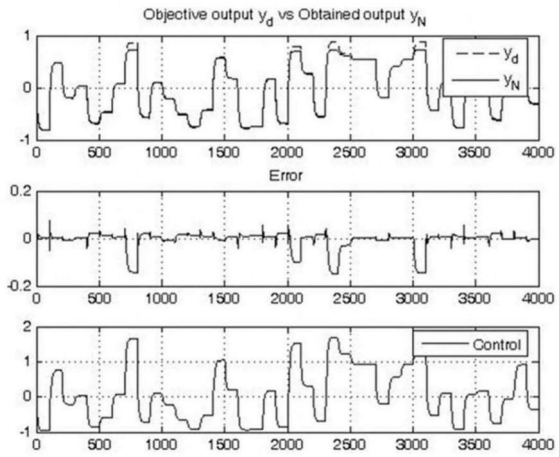

Fig. 3. Nonlinear system control

with input-output interpretation. The same results regarding Lyapunov stability for state-space dynamic fuzzy systems can be used in this structure.

On the one hand, it was presented a training algorithm that is able to get an approximation to a real plant (under certain assumptions) using only input-output data and with online training capabilities. When a proper

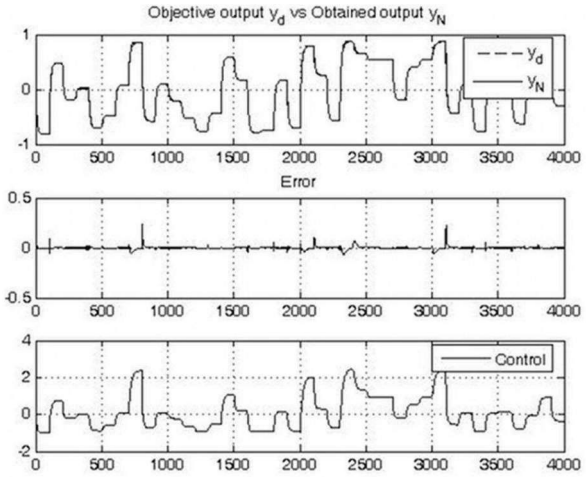

Fig. 4. Nonlinear system control with PID compensation

initialization algorithm is used, a convergent learning can be achieved. In the linear identification case, the achieved identification for each subsystem was slightly different from the *real* one. In the nonlinear case, there is an existent identification error that is supposed to have origin in the lack of nonlinearities included in the structure. We hypothesize that adding membership functions in each rule will help to reduce it, but having care in not falling into the *curse of dimensionality*. On the other hand, a predictive control was proposed that exploits the resulting trained network, with bounded tracking error, that was tested in a nonlinear system. A simple linear compensation via PID was successfully used in reducing the tracking error. It must be noted that still an stability analysis is needed for this implementation, which is part of the future work of this project.

REFERENCES

A. D. Back and A. C. Tsoi. A time series modeling methodology using fir and iir synapses. *Proc. Workshop on Neural Networks for Statistical and Economic Data*, pages 187–194, 1990.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Con. Signal Syst.*, 2:303–314, 1989.

J. de Jesús Rubio and W. Yu. Nonlinear system identification with recurrent neural networks and dead-zone Kalman filter algorithm. *Neurocomputing*, 70(13-15): 2460–2466, 2007.

M.A. Gonzalez Olvera and Y. Tang. A new recurrent neurofuzzy network for identification of dynamic systems. *Fuzzy Sets and Systems*, 158(10):1023–1035, 2007.

V. Gorrini and H. Bersini. Recurrent fuzzy systems. *Proc. IEEE Int. Conference on Fuzzy Systems*, pages 193–198, 1994.

L. Guo. Estimating time-varying parameters by the Kalman filter based algorithm: stability and convergence. *Automatic Control, IEEE Transactions on*, 35 (2):141–147, 1990.

David G. Hagner, Mohamad H. Hassoun, and Paul B. Watta. *Comparison of Recurrent Networks for Trajectory Generation*, chapter 10, pages 243–276. CRC Press, 2000.

S.S. Haykin. *Kalman Filtering and Neural Networks*. Wiley Chichester, 2001.

JJ Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

J.-S. R. Jang. Anfis: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23:665–685, May-June 1993.

J. S. R. Jang. Self-learning fuzzy controllers based on temporal back propagation. *IEEE Transactions on Neural Networks*, 3(5):714–723, 1992.

Rolf Johansson. *System Modeling and Identification*. Prentice Hall Information and System Sciences. Prentice Hall, 1st edition, 1993.

Chia-Feng Juang. A tsk-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms. *IEEE Transactions of Fuzzy Systems*, 10(2):155–170, April 2002.

Chia-Feng Juang and Chin-Teng Lin. A recurrent self-organizing neural fuzzy inference network. *IEEE Trans. Neural Networks*, 10(4):828–845, 1999.

Bart Kosko. *Fuzzy Engineering*. Prentice Hall, 1992.

Paris A. Mastorocostas and John B. Theocharis. A recurrent fuzzy-neural model for dynamic system identification. *IEEE Trans. onf Systems, Man and Cybernetics - Part B: Cybernetics*, 32(2):176–190, April 2002.

George C. Mouzouris and Jerry M. Mendel. Dynamic nonsingleton fuzzy logic systems for nonlinear modeling. *IEEE Trans. Fuzzy Systems*, 5(2):199–207, May 1997.

K.S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Tras. Neural Networks*, 1:4–27, March 1990.

A. Poznyak, T. Poznyak, and I. Chairez. Dynamic neural observers and their application for identification and purification of water by ozone. *Automation and Remote Control*, 67(6):887–899, 2006.

D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Learning Internal Representations by Error Propagation*, chapter 8. MIT Press, Cambridge, 1986.

E. A. Wan. Temporal back-propagation for fir neural networks. *Proc. of the International Joint Conference on Neural Networks*, pages 575–580, 1990.

A. Weibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. Phenomene recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37:328–339, March 1989.

R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 2(1):270–280, 1989.

Wen Yu and Xiaoou Li. Fuzzy identification using fuzzy neural network. *IEEE Trans. on Fuzzy Systems*, 12(3), June 2004.