15th Triennial World Congress, Barcelona, Spain

POLYNOMIAL ARTIFICIAL NEURAL NETWORK AND GENETIC ALGORITHM FOR THE IDENTIFICATION AND CONTROL OF NONLINEAR SYSTEMS

E. Gómez-Ramírez, A. S. Poznyak* & R. Lozano**

LIDETEA, UNIVERSIDAD LA SALLE
Benjamín Franklin No. 47 Col. Condesa
CP 06140, México, D.F., México
E-mail: egomez@ci.ulsa.mx

**CINVESTAV-IPN, Sección de Control Automático,*
Av. IPN 2508 AP 14-740,
CP 07000, México D.F., México
E-mail: apoznyak@ctrl.cinvestav.mx

***Université de Compiegne*
Centre de Recherches de Royallieu BP 20529
60205 Compiegne Cedex France
E-mail: Rogelio.Lozano@hds.utc.fr

Abstract: In Adaptive Control Theory there are different procedures to identify a linear system. The fundamental problem is that in the real world many systems are nonlinear and it is not easy to obtain a mathematical model. In this work, an identification procedure for nonlinear systems is presented using the properties of Artificial Neural Networks and Genetic Algorithms to optimize the architecture of the network. A new technique of Adaptive Control to cancel the nonlinear dynamics of the system is proposed to set the poles of the system in a desire position. The behavior of the algorithm for the linear and nonlinear case is presented with the analysis of the theory and operational importance of these techniques.

Keywords: Adaptive Control, Neural Networks Models, Genetic Algorithm, Nonlinear Systems

1 INTRODUCTION

In adaptive control systems is very important to find an optimal model to ensure the convergence of the algorithms to the optimal controller parameters. In many cases these models require some kind of previous knowledge of the plant to be identified to define the type of model structure that is going to be used. This means that it is possible to find many different combinations of algorithms or functions, which represent the dynamics of the system. In nonlinear case such mapping cannot be realized exactly and unmodelled dynamics terms obligatory exist in any mathematical model description. For the adaptive controller, an ANN or Fuzzy Logic system is used to *learn* the behavior of the new dynamic. These actions require an adaptive and optimization process inside the *intelligent mechanism* to get the minimal estimation error.

In this paper we propose a methodology to identify Nonlinear Systems using a Polynomial Artificial Neural Network (PANN) and Genetic Algorithm. With the model and the information of the network an Adaptive controller is proposed to cancel the undesired dynamics. To describe this problem, the following structure is proposed: section 2 describes the theory of PANN and some useful definitions, in section 3 some concepts of Genetic Algorithm and variations of the method used to obtain the optimal Architecture of PANN (Gómez et al, 1999) are introduced. With this network is possible to identify nonlinear systems. Section 4 explains how we can use the structure of PANN to design an adaptive controller to cancel the nonlinear dynamic of the plant identified and a new placing of the transfer function poles is

proposed. In next section some simulations results conclude this study.

2 POLYNOMIAL ARTIFICIAL NEURAL NETWORK

The model of PANN can be described by (Gómez et al, 1999):

$$\hat{y}_k = [\phi(x_{1,k}, x_{2,k} \quad ...x_{n_i,k}, \quad x_{1,k-1}, \quad x_{2,k-1}, \quad .... \quad , \quad x_{n_i,k-n_1} \quad ..... \quad y_{k-1}, \quad y_{k-2}, \quad ....y_{k-n_2})]_{\phi_{\min}}^{\phi_{\max}} \tag{1}$$

where: $\hat{y}_k \in \Re$ is the estimated function, $\phi(x,y) \in \Re$ is a nonlinear function, $x_i \in X$ are the inputs, for $i=1,...,n_i$; and n_i is the number of inputs, $y_{k-j} \in Y$ are the previous values of the output, for $j=1,..,n_2$, and n_1 is the number of delays of the input, n_2 is the number of delays of the output, X,Y are compact subsets of $\Re$. In order to simplify the notation let us define:

$$z = \{x_{1,k}, x_{2,k},..,x_{n_1,k},..,y_{k-1}, y_{k-2},..,y_{k-n_2}\} = \{z_1, z_2, z_3,..z_{n_V}\} \tag{2}$$

where: n_v is the total number of elements in z description:

$$n_v = n_i + n_1 n_i + n_2 \tag{3}$$

The nonlinear function $\phi(z) \in \Phi_p$ belongs to a family Φ_p of polynomials that can be represented as:

$$\Phi_p(z_1, z_2,.., z_{n_v}) = \left\{ \begin{matrix} \phi(z) : \phi(z) = a_0(z_1, z_2,.., z_{n_v}) + a_1(z_1, z_2,.., z_{n_v}) \\ + a_2(z_1, z_2,...., z_{n_v}) ... + a_p(z_1, z_2,...., z_{n_v}) \end{matrix} \right\} \tag{4}$$

The subindex p is the maximum power of the polynomials expression and $a_i(z_1, z_2,...., z_{n_v})$ are homogeneous polynomials of total degree i, for $i=0,...,p$. Every homogeneous polynomial can be written as:

$$\begin{aligned} &a_0(z_1, z_2,..., z_{n_v}) = w_0 \\ &a_1(z_1, z_2,..., z_{n_v}) = w_{1,1} z_1 + w_{1,2} z_2 + ... + w_{1,n_v} z_{n_v} \\ &a_2(z_1, z_2,..., z_{n_v}) = w_{2,1} z_1^2 + w_{2,2} z_1 z_2 + w_{2,3} z_1 z_3 + ...z_1 z_{n_v} + \\ &...z_2^2 + ...z_2 z_3 ... + w_{2,N_2} z_{n_v}^2 \\ &: \\ &a_p(z_1, z_2,..., z_{n_v}) = w_{p,1} z_1^p + w_{p,2} z_1^{p-1} z_2 + ... + w_{p,N_p} z_{n_v}^p \end{aligned} \tag{5}$$

where w_{ij} is the associated weight of the networ. The term w_0 corresponds to the *input bias* of the network. The homogeneous polynomial $a_1(z)$ is equivalent to *weight* the inputs. From $a_2(z)$ to $a_p(z)$ represent the *modulation* between the inputs and the *power* of every one.

Here N_i is the number of terms of every polynomial

$$N_0 = 1, N_1 = n_v, N_2 = \sum_{i=1}^{n_v} i, N_3 = \sum_{s_1=0}^{n_v-1} \sum_{i=1}^{n_v-s_1} i, \tag{6}$$

$$N_p = \underbrace{\sum_{s_{p-2}=0}^{n_v-1} ... \sum_{s_2=0}^{n_v-s_3} \sum_{s_1=0}^{n_v-s_2} \sum_{i=1}^{n_v-s_1} i}_{p-1}$$

The dimension N_Φ of each family Φ_p can be computed:

$$N_\Phi = \sum_{i=0}^{p} N_i \tag{7}$$

The activation function is given by:

$$[\phi(z)]_{\phi_{min}}^{\phi_{max}} = \begin{cases} \phi_{max} & \phi(z) \geq \phi_{max} \\ \phi(z) & \phi_{min} < \phi(z) < \phi_{max} \\ \phi_{min} & \phi(z) \leq \phi_{min} \end{cases} \tag{8}$$

where ϕ_{max} and ϕ_{min} are upper and lower bounds respectively.

2.1 Learning of PANN

To introduce the learning concepts in PANN first it is necessary to describe some terms that will be used:

The approximation error of PANN can be defined as:

$$err_n(y^n, \phi(z)) := \frac{1}{n} \sum_{k=1}^{n} (y_k - \phi(z_k))^2 \qquad y^n = (y_1, y_2,.., y_n) \tag{9}$$

where y_n is the target output, $\phi(z_k) \in \Phi_p$ and n is the number of points. The optimal error is defined by:

$$opterr_n(y^n, \phi(z)) := \min_{\phi \in \Phi_p} err_n(y^n, \phi(z)) = err_n(y^n, \phi^*(z)) \tag{10}$$

where $\phi^*(z) \in \Phi_p$ is the optimal estimation of y^n.

The PANN learns uniformly the target output with accuracy ε if:

$$P\{err_n(y^n, \phi(z)) - err_n(y^n, \phi_n^*(z)) > \varepsilon\} = 0 \quad \varepsilon > 0$$

Now the learning problem is how to find a specific $\phi \in \Phi_p(z)$ that fulfilled the previous inequality. Let define a vector of components $M(z)$ in the same way that in (4):

$$M(z) = \{z_1, z_2,.., z_{nv}, z_1^2, z_1 z_2,.., z_{nv}^2, z_1^3, z_1^2 z_2 ....., z_{nv}^p\} \tag{11}$$

Then the nonlinear function $\phi \in \Phi_p(z)$ described in (4) can be represented as:

$$\phi = \langle W', M(z) \rangle, W' = W .* W_b^T, \forall w_i^b \in \{0,1\} \tag{12}$$

are the weights of PANN, W_b is a boolean vector.

The product .* is defined by:

$$W .* W_b^T = \begin{cases} w_{ij} \ \ if \ w_j^b = 1 \\ 0 \ \ if \ w_j^b = 0 \end{cases} \tag{13}$$

For example, for W and W_b as:

$$W = [w_1 \quad w_2 \quad w_3 \quad w_4] \quad W_b^T = [1 \quad 0 \quad 1 \quad 0]^T$$

$$W .* W_b^T = [w_1 \quad 0 \quad w_3 \quad 0]$$

Equation (12) means that ϕ only has specific terms of Φ_p that can be selected by W_b in such way, that the optimal structure of PANN ϕ^* can be calculated as:

$$\phi^*(z) = \left\langle \left(W^{'}\right)^*, M(z) \right\rangle = \left\langle W.*\left(W_b^T\right)^*, M(z) \right\rangle$$
$$= \left\langle W, M(z).*\left(W_b^T\right)^* \right\rangle \tag{14}$$

$$\Rightarrow err_n\left(y^n, \phi(z)\right) = err_n\left(y^n, \left\langle \left(W^{'}\right), M(z) \right\rangle\right) \tag{15}$$

$$\Rightarrow \underset{W^{'}}{opt}\, err_n\left(y^n, \phi(z)\right) = \underset{W^{'}}{opt}\, err_n\left(y^n, \left\langle \left(W^{'}\right)^*, M(z) \right\rangle\right) \tag{16}$$

Using (9)-(16), the learning problem for this specific structure can be formulated to the following two steps optimization problem:

$$\min_{W_b} \min_{W \in \Re} err_n\left(y^n, \phi(z)\right)\Big|_{W_b} \tag{17}$$

where $err_n\left(y^n, \phi(z)\right)\Big|_{W_b}$ is the error value given by (9) under a fixed value of W_b. The values of the parameter W can be obtained using least square method:

$$W\big|_{W_b} = \arg\min_{W \in \Re^N} err_n\left(y^n, \phi(z)\right)\Big|_{W_b}$$
$$W\big|_{W_b} = \Gamma^N \sum_{k=1}^{n} y_k \left(M_b(z_k)\right), \quad M_b(z_k) = M.*W_b^T \tag{18}$$
$$\Gamma^N = \left(\sum_{k=1}^{n} M_b(z_k) M_b(z_k)^T\right)^{-1}$$

In our case $N=N_\Phi$ and the searching space has dimension 2^N. It is clear, that in many cases the searching space can be very big and for many possible combinations of $\phi(z)$, and only one or one set of this group fulfills the equation (17). The next section describes how can be obtained the value of the array W_b^* using GA.

3 GENETIC ALGORITHM

To find the optimal structure of the network the crossover or sexual recombination, the mutation and other special process called add parents and add Random Father (Gómez, 2002) are used. Next Sections describe these processes in detail.

3.1 Crossover

To explain the multipoint crossover for each fixed number $g=1,..,n_g$, where n_g is the number of total generations, let introduce the matrix F_g which is the set of parents of a given population. This matrix is boolean of dimension $F_g : n_p x n_b$, where n_p is the number of parents of the population at the generation g and n_b is the size of every array (chromosomes). Let $C\left(F_g, n_t\right)$ be the crossover operator which can be defined as the combination between the information of the parents set considering the number of intervals n_t of each individual and the number of sons n_s such that:

$$n_s = n_p^{\,n_t} \tag{19}$$

then $C\left(F_g, n_t\right) : n_p x n_b \rightarrow n_s x n_b$. To show how the crossover operator can be applied the following example is introduced. Let F_g has $n_p=2$ and $n_t=2$. This means that the array (the information of one father) is divided in 3 sections and every section is determined with a_i and b_i respectively for $i=1,..,n_t$. It's important to appoint that with this operator the parents F_g of the population g are included in the result of the crossover:

$$F_g = \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} \Rightarrow C(F_g, 2) \begin{bmatrix} a_1 & a_2 \leftarrow \\ a_1 & b_2 \\ b_1 & a_2 \\ b_1 & b_2 \leftarrow \end{bmatrix}$$

3.2 Mutation

The mutation operator just changes some bits that were selected in a random way from a fixed probability factor P_m; in other words, we just vary the components of some *genes*. This operator is extremely important, because assures the maintenance of the diversity inside the population, which is basic for the evolution (Altenberg, 1995)(Banzhaf et al, 1997). This operator $M : n_s x n_b \rightarrow n_s x n_b$ changes with probability P_m a specific population in the following way:

$$M\left(F_{ij}, P_m\right) = \begin{cases} \overline{F}_{ij} & r(\omega) \le P_m \\ F_{ij} & r(\omega) > P_m \end{cases} \tag{20}$$

where $r(\omega) \in U(0,1)$ is a random variable with uniform distribution $i=1,..,n_s$; $j=1,..,n_b$ defined on a probability space $(\Omega, \Im, P)$, $\omega \in \Omega$. The mutation operator ensures that the probability of finding any point in the search space is never zero. If the probability is very high the information that every generation founds can be lost and the method has the same behavior that a traditionally random search.

3.3 Add Parents Mutated

In this part the parents mutated are added to the result of crossover process, then the population A_g at the generation g can be obtained like:

$$A_g = \begin{bmatrix} C\left(F_g, n_t\right) \\ M\left(F_g, P_m\right) \end{bmatrix} \tag{21}$$

Note that A_g has the best individuals of A_{g-1} because the parents, the best individuals of the previous generation, are included in this population. This step and the previous one ensure that the algorithm does not diverge.

3.4 Add Random Parents

To avoid local minima a new scheme is introduced and it is called add random parents. If the best individual of one generation is the same than the previous one, a new random individual is included like parent of the next generation. This step increases the population because when crossover is applied with the new random parent, the number of sons increases by the relation (19). This step is tantamount to have a

very big mutation probability and to search in new one points of the solution space.

3.5 Selection Process

The Selection Process S_g computes the objective function O_g that represents the performance condition to maximize or minimize and selects the best n_p individuals of A_g as:

$$S_g\left(A_g, n_p\right) = \min^{n_p} O_g\left(A_g\right) \tag{22}$$

Then, the parents of the next generation can be calculated by:

$$F_{g+1} = S_g\left(A_g, n_p\right) \tag{23}$$

Notice that the following operator can obtain the best individual of the generation g:

$$S_g\left(A_g, 1\right) \tag{24}$$

In resume the Genetic Algorithm proposed can be describe with the following steps:

1. For the initial condition $g=0$ select the A_0 in random way with dimension $A_0 : n_s \, x \, n_b$
2. Compute the objective function and select the best individuals for the initial population $F_1 = S_0(A_0, n_p)$
3. Obtain the new population in the generation g with the crossover and mutation process A_g
4. Calculate the objective function and select the best individual of the generation with S_g
5. Return to step 3 until the maximal number of generations is reached or one of the individuals of S_g obtains the minimal desire value of O_g.

For our case the application to the theory of PANN is automatic if we consider like W_b^* the array searched. Then the problem of learning can be translated obtaining the optimal structure of PANN using GA. Notice that the optimal error in generation g is obtained by:

$$\left(\underset{W' \in \Re^N}{opt\ err_n} \right)_g = \min_{W_b \in A_g} err_n\left(y^n, \left\langle \left(W'\right)_{W_b}, M(z) \right\rangle \right) \tag{25}$$

Remark 1

If $O_g^i = err_n^i\left(y^n, \left\langle \left(W'\right)_{W_b}^i, M(z) \right\rangle \right), i = 1,...n_s + n_p$ represents the error between the target y^n and the estimated by PANN. Then $\min O_g^i = \min\ err_n^i\left(y^n, \left\langle \left(W'\right)_{W_b}^i, M(z) \right\rangle \right)$ represents the error of the optimal individual in the generation g that can be selected by $W_b^g = S_g\left(A_g, 1\right)$ due to $S_g\left(A_g, n_p\right) = \min^{n_p} O_g\left(A_g\right)$

Remember that the population A_g of generation g and the parents F_{g+1} can be described by:

$$A_g = \begin{bmatrix} C\left(F_g, n_t\right) \\ M\left(F_g, P_m\right) \end{bmatrix} \; F_{g+1} = S_g\left(A_g, n_p\right)$$

$$A_{g+1} = \begin{bmatrix} C\left(F_{g+1}, n_t\right) \\ M\left(F_{g+1}\right) \end{bmatrix} \; F_{g+2} = S_{g+1}\left(A_{g+1}, n_p\right)$$

Notice that the best individuals of the generation g are in the next generation due to the crossover operator, then:

$$\begin{aligned} &\min_i err_n^i\left(y^n, \left\langle \left(W'\right)_{W_b^{g+1}}^i, M(z) \right\rangle \right) \\ &\le \min_i err_n^i\left(y^n, \left\langle \left(W'\right)_{W_b^{g+1}}^i, M(z) \right\rangle \right) \end{aligned} \tag{26}$$

$$\Rightarrow \ \min_i O_{g+1}^i \le \min_i O_g^i \tag{27}$$

with $\phi_g \in \Phi_p$ can be described like:

$$\phi_g = \left\langle W, M(z).* S_g\left(A_g, 1\right)^T \right\rangle \tag{28}$$

$$err\left(y, \phi_{g+1}(z)\right) \le err\left(y, \phi_g(z)\right) \tag{29}$$

This means that the minimal error in the generation g is less or equal than the error of the previous generation for the best individual of the population. With this part we only prove convergence, but not convergence to the optimal error value.

It is possible to show that if mutation Probability P_m is strictly more than zero then for any small enough $\varepsilon > 0$ we can guarantee not only (29), such that:

$$err\left(y, \phi_{g+1}(z)\right) + \varepsilon \le err\left(y, \phi_g(z)\right) \tag{30}$$

Theorem 1

Let define $W_b^g \overset{\Delta}{=} S_g\left(A_g, 1\right)$ (31)

where W_b^g is the best individual in the generation g, i. e., the optimal structure of PANN at this moment. Then the approximation error *(9)* can be rewritten as:

$$err\left(y^n, \phi_g(z)\right) = \frac{1}{n} \sum_{k=1}^{n} \left(y_k - \left\langle W, M(z).* \left(W_b^g\right)^T \right\rangle \right)^2 \tag{32}$$

where y^n is the target output, $\phi_g(z)$ is the optimal representation in the generation g, and:

$$W\big|_{W_b} = \arg\min_{W \in \Re^N} err_n\left(y^n, \phi(z)\right)\Big|_{W_b = W_b^g}$$

If Mutation Probability $P_m > 0$ then PANN learns uniformly the target output, such that:

$$\lim_{g \to \infty} P\left\{ err\left(y, \phi_g(z) - opterr\left(y, \phi(z)\right) \ge \varepsilon \right\} = 0, \ \ \varepsilon > 0 \tag{33}$$

Proof of Theorem 1:

Let define:

$$\Delta_g := err_n\left(y^n, \phi_{g+1}(z)\right) - opterr_n\left(y^n, \phi_g(z)\right) \tag{34}$$

To simplify the notation let use:

$$\Delta_g := err^g - opterr^g \tag{35}$$

$$\begin{aligned} \Delta_{g+1} &:= err^{g+1} - err^g + err^g - opterr^g \\ &= err^{g+1} - err^g + \Delta_g \end{aligned}$$

$$\Rightarrow \ \Delta_{g+1} = \Delta_g + err^{g+1} - err^g$$

$$\Delta_g = \Delta_{g-1} + err^g - err^{g-1}$$

$$\Delta_{g-1} = \Delta_{g-2} + err^{g-1} - err^{g-2}$$
$$\Delta_{g-2} = \Delta_{g-3} + err^{g-2} - err^{g-3}$$
$$\vdots$$
$$+\Delta_1 = \Delta_0 + err^1 - err^0$$

$$\Delta_{g+1} = \Delta_0 + \sum_{k=0}^{g}\left(err^{k+1} - err^k\right) = \Delta_0 - \sum_{k=0}^{g}\left(err^k - err^{k+1}\right) \tag{36}$$

Let us define: $\partial_k := err^k - err^{k+1} \geq 0$ (37)

Notice that to prove theorem it is sufficient to state that:

$$\sum_{k=0}^{\infty}\partial_k \overset{a.s.}{=} \infty \tag{38}$$

$$S_n(\omega) = \sum_{k=0}^{n}\partial_k(\omega) = S_{n-1}(\omega) + \partial_n(\omega) \tag{39}$$

To prove (38) we should prove that:

$$\sum_{k=0}^{\infty}E\{\partial_k|_{\Im_{n-1}}\} \overset{a.s.}{=} \infty \tag{40}$$

Then $E\left\{S_n(\omega)|_{\Im_{n-1}}\right\} \overset{a.s.}{=} S_{n-1}(\omega) + E\left\{\partial_n(\omega)|_{\Im_{n-1}}\right\}$ (41)

where $\Im_{n-1}=\sigma(\delta_0,\delta_1,\delta_2,\ldots\delta_{n-1})$ is the sigma-algebra generated in the corresponding process and ω is a random event such that:

$$\int\partial_n(\omega)dP(\omega|\Im_{n-1}) = \int_{\omega:\partial_n\geq\varepsilon}\partial_n(\omega)dP(\omega|\Im_{n-1}) + \int_{\omega:\partial_n<\varepsilon}\partial_n(\omega)dP(\omega|\Im_{n-1}) \tag{42}$$

$$\int\partial_n(\omega)dP(\omega|\Im_{n-1}) \geq \int_{\omega:\partial_n\geq\varepsilon}\partial_n(\omega)dP(\omega|\Im_{n-1}) \geq \varepsilon P\{\partial_n \geq \varepsilon|\Im_{n-1}\} = \varepsilon P_n \tag{43}$$

where $P_n = P\{\partial_n \geq \varepsilon|f_{n-1}\}$

Remember that for a random variable $r(\omega)$ the following conditions is fulfilled:

$$M\left(F_{ij},P_m\right) = \begin{cases}\overline{F}_{ij} & r(\omega)\leq P_m\\ F_{ij} & r(\omega)>P_m\end{cases}$$

and using:

$$\Rightarrow P\{A_{g+1} = A_g\} = 0 \ \ \forall P_m > 0$$
$$\Rightarrow P\left\{\left(err^g - err^{g+1}\right)\geq\varepsilon|_{\Im_{n-1}}\right\} > 0$$
$$\Rightarrow P\{\partial_n \geq \varepsilon|\Im_{n-1}\} \geq p > 0 \ \ \forall P_m > 0$$
$$\Rightarrow \text{If } P_n \geq p > 0 \qquad \therefore \sum_{n=1}^{\infty}P_n \overset{a.s.}{=} \infty$$

and if $\sum_{k=0}^{\infty}\partial_k \overset{a.s.}{=} \infty \Rightarrow$

$\exists g = n_0(\omega) < \infty : \Delta_{n_0(\omega)+1} = 0 \ \ \forall\Delta_0$. Theorem is proved

$$\Rightarrow \lim_{g\to n_0(\omega)} err_n\left(y^n,\phi_g(z)\right) = opterr_n\left(y^n,\phi(z)\right) \ \square$$

Remark 2

For practical case theorem 1 can be written using definition 1 as:

If Mutation Probability $P_m>0$ then PANN learns uniformly the target output with accuracy ε in a finite number of generations n_g if:

$$\lim_{g\to n_g} P\{err(y,\phi_g(z) - opterr(y,\phi(z)) > \varepsilon)\} = 0 \quad \varepsilon > 0 \tag{44}$$

4 ADAPTIVE CONTROL OF NONLINEAR SYSTEMS

The ARMAX model is a very common algorithm to identify linear dynamic systems. PANN with the parameter $p=1$ can be related with this representation and in similar way NARMAX model for $p\geq 2$. One advantage of PANN, which was described in the previous section, is that we obtain the optimal one and it is possible to identify the linear and nonlinear part of a nonlinear dynamic system. Using the previous theory, in this section a methodology to control nonlinear systems using PANN is described.

First, it is necessary to define a new kind of structure. The PANN model (1) can be rewritten as:

$$y_k = \phi_k(x) + \phi_k(y) + \phi_k(x,y) \tag{45}$$

where $\phi(x)$ and $\phi(y)$ represent the linear terms of $\phi(z)$. These terms are covered by the case of $p=1$, e.g. the case for the polynomials $a_0(z_1,z_2,\ldots z_{n_y})$ and $a_1(z_1,z_2,\ldots z_{n_y})$ and $\phi_k(x,y)$ represents the nonlinear relations. Then, if we substitute

$$y_k - \phi_k(y) = A\left(q^{-1}\right)y_k \tag{46}$$
$$\phi(x_k) = q^{-d}B\left(q^{-1}\right)x_k \tag{47}$$

where q^{-1} represents the delay operator. Then equation (45) can be written as:

$$A\left(q^{-1}\right)y_k = q^{-d}B\left(q^{-1}\right)x_k + \phi_k(x,y) \tag{48}$$

with

$$A\left(q^{-1}\right) = 1 + a_1q^{-1} + \ldots. + a_{n_2}q^{-n_2}$$
$$B\left(q^{-1}\right) = b_0 + b_1q^{-1} + \ldots. + b_{n_1}q^{-n_1} \tag{49}$$

where the parameters a_i and b_i are unknown and will be obtained using the theory described in the previous section. Notice that these parameters are equivalent to compute the weights of PANN using (17)-(18).

Let the following assumptions be fulfilled:

A.1 n_1,n_2 are fixed and the delay d are known

A.2 $q^{-d}B\left(q^{-1}\right)$ has all the zeros inside the unit circle and $b_0\neq 0$

Using a similar scheme of an indirect Adaptive Controller. Let the Euclidean division (Landau et al.,1998)

$$C\left(q^{-1}\right)= A\left(q^{-1}\right)S\left(q^{-1}\right)+ q^{-d} R\left(q^{-1}\right) \tag{50}$$

where:

$$C\left(q^{-1}\right)= 1 + c_1 q^{-1} + ... + c_{n_2+d-1} q^{-(n_2+d-1)}$$
$$S\left(q^{-1}\right)= s_0 + s_1 q^{-1} + ... + s_{d-1} q^{-(d-1)} \tag{51}$$
$$R\left(q^{-1}\right)= r_0 + r_1 q^{-1} + ... + r_{n_2-1} q^{-(n_2-1)}$$

With this polynomial identity we can compute the values for $S(q^{-1})$ and $R(q^{-1})$ for a fixed value of $C(q^{-1})$.

We calculate $C\left(q^{-1}\right)y_{t+d}$

$$C\left(q^{-1}\right)y = S\left(q^{-1}\right)A\left(q^{-1}\right)y + q^{-d} R\left(q^{-1}\right)$$
$$C\left(q^{-1}\right)y = S\left(q^{-1}\right)\left[q^{-d}\left\{Bx + \phi(x,y)\right\}\right] + q^{-d} R\left(q^{-1}\right)$$
$$C\left(q^{-1}\right)y = q^{-d}\left[S\left(q^{-1}\right)\left\{B\left(q^{-1}\right)x + \phi(x,y)\right\} + R\left(q^{-1}\right)\right]y = q^{-d} y^* \tag{52}$$

$$y^* = S\left(q^{-1}\right)B\left(q^{-1}\right)x + S\left(q^{-1}\right)\phi(x,y) + R \tag{53}$$

If we define the control action as: $x = X_1 + X_2$ (54)

with X_1 and X_2 such that:

$$\Rightarrow B(q^{-1})X_1 = -\phi(x,y) \tag{55}$$
$$\Rightarrow S(q^{-1})B(q^{-1})X_2 + R = y^* \tag{56}$$

In (56) the nonlinear part of the system is eliminated and controlled like a linear system.

If the system is minimum phase then $|y_k| < \infty \Rightarrow |x_k| < \infty$. The parameters $A(q^{-1}),B(q^{-1})$, and $\phi_k(x,y)$ can be estimated by PANN using GA and after this, the dynamic of the system is adjusted placing the poles at the zeros of $C(q^{-1})$. The next example shows the results of the algorithm for the nonlinear case. All the programs used in the simulations were made in Matlab.

Example:

The procedure to apply an adaptive controller is the following:

1. Identification of the plant using a random signal $u(k)$ with uniform distribution U(-1,1). The parameters of PANN used are $p=2$, $n_1=n_2=2$, $P_m=0.2$, $n_p=3$,$n_t=3$.
2. Test the results using another input like $r=sin(t)$.
3. Separate the linear and nonlinear part of the model to obtain $A(q^{-1}),B(q^{-1})$ and $\phi_k(x,y)$ to cancel and adapt the dynamical system. In this particular case $\phi_k(x,y)=0$, this means that there is not a nonlinear part and therefore is like a typical adaptive control application. Choosing the following values for C

$$C(q^{-1}) = 1 - 1.9025z^{-1} + 0.9048z^{-2} \tag{57}$$

the setting time is $t_s=1$ sec. In this case we are going to use a very simple nonlinear discrete model first proposed by May in his paper (May, 1976). This model can be described by:

$$y_k = r(1 - y_{k-1}^2) + u_k \tag{58}$$

With $r=1.2$ and $y_0=0.5$ we can obtain a chaotic dynamics without introducing any input. Separating the linear and nonlinear part we can observe in figure 1 the results of the proposed methodology.

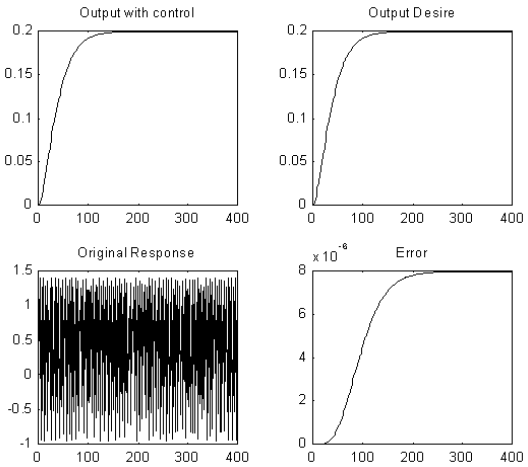

Figure 1 Adaptive Control Response

5 CONCLUSION

A methodology to identify nonlinear dynamics using PANN was proposed with the optimal structure obtained by GA. The modification to the algorithms of GA ensures at least with ε precision the convergence of the response.

The structure can identify linear and nonlinear systems choosing the suitable parameters. For the linear case $p=1$ the minimal identification model for the plant is obtained. In the case of time series prediction the procedure is the same and the results corresponds to the optimal delays required to minimize the estimation error. For the values of n_1 and n_2 only it is necessary to set the maximal values and the algorithm will itself select the optimal representation in this range. The Mutation probability is recommended to be between 0.10 and 0.15 to optimal convergence.

6 REFERENCES

Altenberg L., *Genome growth and the evolution of the genotype-phenotype map*, Springer-Verlag, Berlin, Germany, 1995.

Andre D. and Koza J., *Advances in Genetic Programming 2*, MIT Press, 1996.

Banzhaf W., Nordin P, Keller R. & Francone F., *Genetic Programming -- An Introduction On the Automatic Evolution of Computer Programs and its Applications*, Morgan Kaufmann, dpunkt.verlag, 1997.

E. Gómez-Ramírez, A. Poznyak, A. González Yunes & M. Avila-Alvarez. Adaptive Architecture of Polynomial Artificial Neural Network to Forecast Nonlinear Time Series. *CEC99 Special Session on Time Series Prediction*. Mayflower Hotel, Washington D.C., USA, July 6-9, 1999.

E. Gómez-Ramírez & X. Vilasis-Cardona, Adaptive Multiresolution Filtering to Forecast Nonlinear time series. *CEC2002 Congress on Evolutionary Computation*, May 12-17, Honolulu Hawaii, 2002

Landau, I. D., Lozano R. & Saad M. *Adaptive Control*. Springer Verlag. 1998

May, R.M., *Nature (London)*, vol. 261, pag. 459, June 1976.