Indirect Neural Control:
A Robustness Analysis Against Perturbations

ANDRÉ LAURINDO MAITELLI
Federal University of Rio Grande do Norte
Natal/RN, BRAZIL, ZIP Code 59072-970

OSCAR GABRIEL FILHO
Potiguar University
Natal/RN, BRAZIL, ZIP Code 59056-000

Abstract: - In the area of robust intelligent control one of the focus is on controllers development that can maintain good performance even if there exists a poor model of the plant and if there are some plant parameter variations. The aim of this paper is an efficient address to this question, more specifically to the indirect approach using neural networks with on-line training, in present a criterion to evaluate the control system robustness against perturbations caused by modelling errors and plant parameter variations, since convergence and stability of neural networks are assured. The robustness was tested using computer simulations applied to control a nonlinear system obtained on technical literature.

Key-Words: - Robust Intelligent Control, Neural Control, Artificial Neural Networks, Robustness

1 Introduction

Control systems based on Artificial Neural Networks - ANN's have been developed by several researchers in all the world, each one emphasizing an implementation aspect that make it very attractive under some practical viewpoint. However only a few of them are worried about fundamental problems related with robustness analysis.

To overcome these problems it is presented an indirect control method using ANN's - the Indirect Neural Control system, in which the main purpose hereafter is achieving how much this methodology is robust with respect to neural networks learning errors and plant parameter variations.

The outline of this paper is: Section 2 is devoted to a brief presentation of the ANN's. Section 3 presents a structure for modelling both the plant and the controller. Section 4 details an indirect control scheme using ANN's, called Indirect Neural Control. Section 5 presents some concepts related with robustness, critical perturbations and stability margin, which is the main contribution of this paper. To validate these concepts, Section 6 presents the computer simulation results of a nonlinear plant control, and finally, Section 7 presents the conclusions.

In this paper, the control system and learning algorithm are based on discrete-time approach.

2 Artificial Neural Networks – ANN's

The ANN's have the purpose of simulate the human nervous system behaviour through software and/or hardware, to benefit of the powerful intelligent systems that human life is endowed [1]. The great development occurred on microprocessor technology of late years has made propitious an efficient software development able to implement the ANN's, i.e., to implement the mathematical elements [2], [3] which believes the human intelligence is based on.

The neural networks are composed of many processing elements connected each one to another of some manner, in such a fashion that making possible its parallel operation. These elements are based on nervous human system, and are called neurons.

The manner how the neurons are connected defines the network architecture. This paper uses a multilayer arrangement depicted in Fig.1 as follows

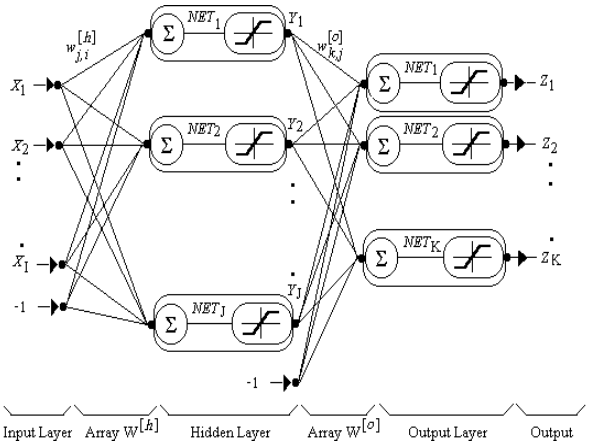

Fig.1: Artificial Neural Network (3-Layers)

In forward direction, the first layer is the input layer only, the intermediary layers are the hidden layers, and the last layer is the output layer.

According [4], the multilayer perceptrons are good to learning mathematical relationships from a set of input-output data, therefore being considered universal approximators.

2.1 Error Backpropagation Algorithm

Among the algorithms used to perform supervised learning, the backpropagation algorithm has emerged as the most widely and successfully algorithm for designing multilayer feedforward networks [4], where the learning process rests on synaptic weight adjustments so that the difference between the desired output and the current output is a minimum acceptable value performed on all training points. The adjustments of the synaptic weights are computed in backward direction, according the following equations:

$$w^{[c+1]} = w^{[c]} + (1-\alpha)\Delta w^{[c]} + \alpha \Delta w^{[c-1]} \quad (1)$$

$$\Delta w = -\eta \frac{\partial E}{\partial w} \quad (2)$$

$$E = \frac{1}{2}\sum_{p=1}^{P}\sum_{k=1}^{K}\left[e_{p,k}\right]^2 \quad (3)$$

$$e_{p,k} = o_{p,k} - z_{p,k} \quad (4)$$

where c is the training cycle (or epoch), η is the learning rate, α is the momentum constant, E is the cost function (or learning global error), $p = 1, 2, ..., P$ input-output learning data, $k = 1, 2, ..., K$ neural network outputs, e is the learning local error, o and z are the desired and current output for the output layer neuron, respectively. Note that z is computed in the forward direction.

The equation used to adjust the $w_{k,j}$ synaptic weights that connect the *j*-neurons of the hidden layer to the *k*-neurons of the output layer, such that E is less than or equal to $\varepsilon \geq 0$ a beforehand given tolerance, i.e., $E \leq \varepsilon$, is

$$\Delta w_{k,j}^{[c]} = \left[\eta\left(o_k - z_k\right) f_o'\left(\sum_j WY\right) y_j\right]^{[c]} \quad (5)$$

with $j = 1, 2,..., J+1$ and $k = 1, 2,..., K$, where $f_o'(.)$ is the derivative of the output layer neurons activation function. The equation used to adjust the $w_{j,i}$ synaptic weights from *i*-neurons of the input layer to the *j*-neurons of the hidden layer, is

$$\Delta w_{j,i}^{[c]} = \left[\eta f_h'\left(\sum_i WX\right) x_i \sum_k \delta_k w_{k,j}\right]^{[c]} \quad (6)$$

with $i = 1, 2,..., I+1$ and $j = 1, 2,..., J$, where $f_h'(.)$ is the derivative of the hidden layer neurons activation function and $\delta_k = \left(o_k - z_k\right) f_o'(\sum_j WY)$ is the error signal back propagated from *k*-neuron. The backpropagation algorithm can suffer convergence problems [5]. To overcome these problems, are used the η-adaptive and the normalized momentum approaches [2], [3].

3 Modelling Using Neural Networks

In external plant model, the inputs and the plant outputs are represented by its derivatives, where the derivative orders will establish the plant dynamical behaviour. The symbols used to recognize the input and plant output orders are defined as follows: n_y = output order, n_u = input order, n = plant order (= output order, that is, $n = n_y$) and d = plant transport delay ($d = n_y - n_u$).

Taking account the discrete-time nonlinear model (here k is a time domain) using ANN's given by

$$y(k+d) = g[\varphi(k,d,n), W] + e(k+d) \quad (7)$$

where g is a mathematics function chosen by the designer to model the plant dynamic behaviour, φ is the regression vector, e is the estimation error, i.e., $e(k+d) = y(k+d) - \hat{y}(k+d)$, and W is the matrix of neural network adjustable parameters (synaptic weights), from what we immediately deduce that the neural estimate of the plant output is given by the following equation

$$\hat{y}(k+d) = g[\varphi(k,d,n), W] \quad (8)$$

The physical systems identification using ANN's can be done employing one of the several model structures presented in [6]. In this paper, it will be adopted the model known as Neural Network Autoregressive with Exogeneous Inputs - NNARX explained as follows.

3.1 Neural Network AutoRegressive with eXogeneous inputs - NNARX Model

The extended nonlinear NNARX model is given by the equation (8), in which the regression vector φ is defined as

$$\varphi(k,d,n) = [y(k+d-1), \ldots, y(k+d-n) \; u(k), u(k-1), \ldots, u(k+d-n), -1] \quad (9)$$

where the -1 augmented component is the input to the additional weight that is made equal to the threshold. The NNARX architecture is depicted in Fig.2 below.

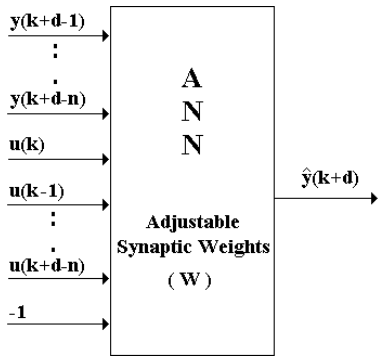

Fig.2: NNARX architecture

This model is BIBO (Bounded Input Bounded Output) stable because it does not have feedback of the estimate output. The absence of stability related problems makes the NNARX model the preferred choice when the system is deterministic or the noise level is not too significant [6].

3.2 Estimate Jacobian Computing

The computing of the plant estimate Jacobian is done after each neural identifier training, according to the chosen neural network architecture. In this work, we use only the followings:

• Neural identifier without hidden layer (only input and output layers). The plant estimate Jacobian is given by

$$\hat{J}(k+d-1) = f_o^{'}(NET_{o1}).w_{1,2n-d+1} \tag{10}$$

where $NET_{o1} = \sum_{i=1}^{2n-d+2} WX$ and i is the input layer index, being $w_{1,\ 2n\text{-}d+1}$ the synaptic weight that connects the input $u(k\text{-}1)$ to the plant estimate output $\hat{y}(k+d-1)$. It is important to point out that we are using the most recent known input-estimate output pair, $[u(k-1), \hat{y}(k+d-1)]$, to compute the plant estimate Jacobian.

• Neural identifier with just one hidden layer. The plant estimate Jacobian is given by

$$\hat{J}(k+d-1) = \sum_{j=1}^{J} f_o^{'}(NET_{o1}).f_h^{'}(NET_{hj}).w_{h,j,2n-d+1}.w_{o,1,j} \tag{11}$$

where J (without hat) is the number of neurons in the hidden layer, $NET_{o1} = \sum_{j=1}^{J+1} WY$, $NET_{hj} = \sum_{i=1}^{2n-d+2} WX$ and the indexes i and j refer to the input and hidden neurons, respectively. It is important to point out that the synaptic weights indicated by $w_{h,j,2n-d+1}$ correspond to the connection of the input $u(k\text{-}1)$, that occupied the position $2n\text{-}d+1$ enumerated from top to bottom, with the j-neurons of the hidden layer. The underwritten index 1 that appears in (10) and (11) is due to the neural identifier possessing only one neuron in the output layer.

4 Indirect Neural Control Scheme

Up to this section, the purpose was to present some basic knowledge about multilayer neural networks and the used structure for nonlinear modelling applied to unknown plant dynamics identification, to assemble a theoretical underground for hereafter formulation of the indirect control scheme based on neural networks.

The term indirect arises because the controller parameters adjustment is based on estimate plant model instead on nominal plant model, probably due to the following reasons: I) the controller impossibility to direct attains the real plant parameters values, and/or II) the difficult to achieve acceptable plant parameters values, resulting from nonlinear unknown dynamics occurrence.

It is used an Indirect Neural Control Scheme that is basically formed with a neural controller (nc) and a neural identifier (ni) [2], [3], [7], [8], [9], both arranged as depicted in the following figure:

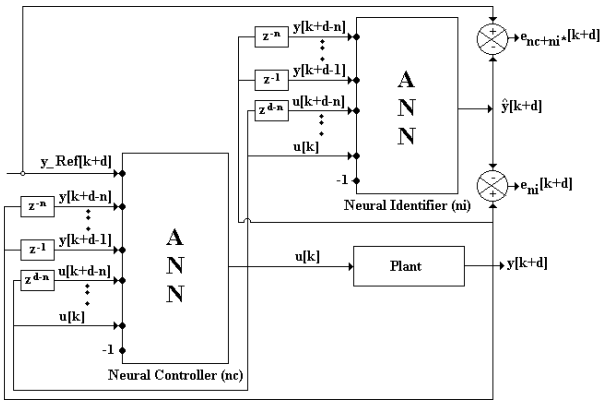

Fig.3: Indirect Neural Control Scheme

Firstly the neural identifier is training, in such manner to minimize the error e_{ni} as shown in Fig.3. In sequence, it is performed the controller training coupled with the neural identifier, being used now the error e_{nc+ni*} instead, and getting the identifier

synaptic weights with fixed values, what were attained after the last neural identifier training stage. The asterisk that appears in ni* means that the identifier remains with constant synaptic weights during all controller training steps.

5 Robustness Analysis

The robustness analysis developed hereafter is with respect to the modelling error that could happen in neural networks learning process, both the identifier and the controller in cascade with a fixed identifier, and a plant parameter variations. The robustness analysis is an important contribution of this paper, because it addresses a challenger question that becomes very attractive of late years.

Theorem 5.1 (Control Error Robustness): "Consider the Indirect Neural Control of a plant, BIBO stable, given by its input-output data [$u(k)$,$y(k+d)$]. If Ω_c is the control errors set, i.e., $\Omega_c(k+d)=\{\omega_c(k+d)\}$, where $\omega_c(k+d)=\left|y\_\mathrm{Ref}(k+d)-\hat{y}(k+d)\right|$, then we can affirm that

$$\sup_{\substack{M\to\infty \\ \Delta u(k)=0,\forall k>k_c}} \Omega_c(k+d+M)=\left|\sqrt{2\varepsilon_{ni}}\right|+\left|\sqrt{2\varepsilon_{nc+ni^*}}\right|$$

where $\{.\}$ is a set, k_c is such that $\Delta u(k)=0,\forall k>k_c$, M is the settling horizon beginning from the instant k_c+d and $\varepsilon_{ni},\varepsilon_{nc+ni^*}\geq 0$ are the beforehand tolerances chosen for neural identifier and neural controller training, respectively."

Proof: Turning back to (3), that gives the learning cost function, and conform it for neural controller (nc) coupled with a single output neural identifier (ni*), the latter being maintained with fixed synaptic weights during all controller training steps, we have

$$E_{nc+ni^*}(k+d)=\frac{1}{2}\sum_{p=1}^{P}\left[e_{nc+ni^*}(k+d-p+1)\right]^2$$

whose development gives

$$\begin{aligned}E_{nc+ni^*}(k+d)=\frac{1}{2}\{&\left[e_{nc+ni^*}(k+d\right]^2\\&+\left[e_{nc+ni^*}(k+d-1\right]^2+\ldots\\&+\left[e_{nc+ni^*}(k+d-P\right]^2\}\end{aligned}$$

Using a conservative procedure in taking account only the most recent neural identifier training input-output pair (p=1), we obtain

$$e_{nc+ni^*}(k+d)\leq\sqrt{2E_{nc+ni^*}(k+d)}$$

According to the scheme shown in Fig.3, we obtain $e_{nc+ni^*}(k+d)=y\_\mathrm{Ref}(k+d)-\hat{y}(k+d)$, what gives

$$y\_\mathrm{Ref}(k+d)-\hat{y}(k+d)\leq\sqrt{2E_{nc+ni^*}(k+d)} \qquad (12)$$

Since $\hat{y}(k+d)=y(k+d)-e_{ni}(k+d)$, from (12) we have

$$\begin{aligned}&y\_\mathrm{Ref}(k+d)-y(k+d)+e_{ni}(k+d)\\&\leq\sqrt{2E_{nc+ni^*}(k+d)}\\&y\_\mathrm{Ref}(k+d)-y(k+d)\\&\leq\sqrt{2E_{nc+ni^*}(k+d)}-e_{ni}(k+d)\end{aligned} \qquad (13)$$

Assuming that the estimate error $e_{ni}(k+d)$ is not available, because it just depends on unknown output $y(k+d)$, it is necessary to dispose the estimate error in recursive form, and so goes back if $d\geq 1$. Without generalization loss, we can take $e_{ni}(k+d)=e_{ni}(k+d-1)+\Delta e_{ni}(k+d)$. In this case, we have

$$\begin{aligned}&y\_\mathrm{Ref}(k+d)-y(k+d)=\\&\sqrt{2E_{nc+ni^*}(k+d)}-\left[e_{ni}(k+d-1)+\Delta e_{ni}(k+d)\right]\end{aligned} \qquad (14)$$

Now applying the absolute value operator in (14), and knowing that the control error is given by $\omega_c(k+d)=\left|y\_\mathrm{Ref}(k+d)-y(k+d)\right|$, it becomes

$$\begin{aligned}&\left|y\_\mathrm{Ref}(k+d)-y(k+d)\right|=\\&\left|\sqrt{2E_{nc+ni^*}(k+d)}-\left[e_{ni}(k+d-1)+\Delta e_{ni}(k+d)\right]\right|\\&\omega_c(k+d)=\\&\left|\sqrt{2E_{nc+ni^*}(k+d)}-\left[e_{ni}(k+d-1)+\Delta e_{ni}(k+d)\right]\right|\end{aligned}$$

Next, taking the triangular inequality and reordering, we get

$$\begin{aligned}\omega_c(k+d)\leq&\left|\left[e_{ni}(k+d-1)+\Delta e_{ni}(k+d)\right]\right|\\&+\left|\sqrt{2E_{nc+ni^*}(k+d)}\right|\end{aligned} \qquad (15)$$

In (15), the term correspondent to estimate error change, $\Delta e_{ni}(k+d)$, tends to be null as soon as the

time rises from $k_c + d$ instant, where k_c is the time when the settled control signal becomes constant forever, i.e., $\Delta u(k) = 0, \forall k > k_c$, there exists a positive integer M, such that

$$\omega_c(k+d+M) \leq \left\| \begin{bmatrix} e_{ni}(k+d+M-1) \\ +\Delta e_{ni}(k+d+M) \end{bmatrix} \right\| + \left| \sqrt{2E_{nc+ni^*}(k+d+M)} \right|$$

whose limit, as soon as M tends to infinity, is given by

$$\lim_{\substack{M\to\infty \\ \Delta u(k)=0, \forall k>k_c}} \omega_c(k+d+M) \leq \left| e_{ni}(k+d+M-1) \right| + \left| \sqrt{2E_{nc+ni^*}(k+d+M)} \right| \quad (16)$$

Considering that the identifier is a single output neural network, from (3) it becomes

$$\begin{aligned} E_{ni}(k+d+M-1) &= \frac{1}{2}\sum_{p=1}^{P}\left[e_{ni}(k+d+M-p)\right]^2 \\ &= \frac{1}{2}\{\left[e_{ni}(k+d+M-1\right]^2 \\ &\quad + \left[e_{ni}(k+d+M-2\right]^2 + \ldots \\ &\quad + \left[e_{ni}(k+d+M-P\right]^2\} \end{aligned}$$

where E_{ni} is the neural identifier cost function, and P is the number of training input-output data. Now, using a conservative reasoning in taking account only the most recent neural identifier training input-output pair (p=1), and knowing that the convergence condition is $E_{ni}(k+d+M-1) \leq \varepsilon_{ni}$, such that $\varepsilon_{ni} \geq 0$ is the beforehand tolerance chosen for neural identifier training, we obtain

$$\begin{aligned} \left| e_{ni}(k+d+M-1) \right| &\leq \left| \sqrt{2E_{ni}(k+d+M-1} \right| \\ &\leq \left| \sqrt{2\varepsilon_{ni}} \right| \end{aligned} \quad (17)$$

Now taking account (17), and also considering that $E_{nc+ni^*}(k+d+M) \leq \varepsilon_{nc+ni^*}$, with $\varepsilon_{nc+ni^*} \geq 0$, it comes from (16) that

$$\lim_{\substack{M\to\infty \\ \Delta u(k)=0, \forall k>k_c}} \omega_c(k+d+M) \leq \left| \sqrt{2\varepsilon_{ni}} \right| + \left| \sqrt{2\varepsilon_{nc+ni^*}} \right| \quad (18)$$

Finally, using the *supremum* operator into control errors set given by (18), we obtain

$$\sup_{\substack{M\to\infty \\ \Delta u(k)=0, \forall k>k_c}} \Omega_c(k+d+M) = \left| \sqrt{2\varepsilon_{ni}} \right| + \left| \sqrt{2\varepsilon_{nc+ni^*}} \right| \quad (19)$$

where M is the settling horizon of the plant beginning from instant k_c+d and $\varepsilon_{ni}, \varepsilon_{nc+ni^*} \geq 0$ are the beforehand tolerances chosen for neural identifier and neural controller training, respectively. □

Definition 5.2: "The Critical Perturbation – CP of the Indirect Neural Control system is the maximum deviation that can occur on plant output without loss of stability."

Theorem 5.3 (Critical Perturbation): "The Critical Perturbation – CP of the Indirect Neural Control system is given by

$$CP = \left| \Delta u(k).\hat{J}(k+d-1) \right| + 2\left| \sqrt{2\varepsilon_{ni}} \right| + \left| \sqrt{2\varepsilon_{nc+ni^*}} \right| \quad (20)$$

where $\Delta u(k)$ is the possible variation of the control signal according actuator range, $\hat{J}(k+d-1)$ is the plant estimate Jacobian and $\varepsilon_{ni}, \varepsilon_{nc+ni^*} \geq 0$ are the beforehand tolerances chosen to neural identifier and neural controller training, respectively."

Proof: The training of the neural controller is performed coupled with the neural identifier, the latter taken with fixed synaptic weights during all the controller training process. In this case, the plant estimate Jacobian can be given by [10], [11]

$$\hat{J}(k+d-1) = \frac{y\_\mathrm{Ref}(k+d) - \hat{y}(k+d-1)}{u(k) - u(k-1)} \quad (21)$$

Manipulating the equation above, and taking account the Fig.3, in which it is immediately seen that $\hat{y}(k+d-1) = y(k+d-1) - e_{ni}(k+d-1)$, then we give

$$\begin{aligned} &\left[u(k) - u(k-1)\right].\hat{J}(k+d-1) = \\ &y\_\mathrm{Ref}(k+d) - \left[y(k+d-1) - e_{ni}(k+d-1)\right] \\ &\Delta u(k).\hat{J}(k+d-1) = \\ &y\_\mathrm{Ref}(k+d) - y(k+d-1) + e_{ni}(k+d-1) \end{aligned}$$

Now, we suppose that the plant output is settled on steady-state, y_{ss}, from instant k+d-1, and at a future

instant the plant suffer a perturbation that cause an abrupt variation, Δy_{ss}, that is

$$\begin{aligned}&\Delta u(k).\hat{J}(k+d-1)= \\ &y\_\mathrm{Ref}(k+d)-(y_{ss}+\Delta y_{ss})+e_{ni}(k+d-1)\\ \Delta y_{ss} &= -\Delta u(k).\hat{J}(k+d-1)\\ &\quad +\left[y\_\mathrm{Ref}(k+d)-y_{ss}\right]\\ &\quad +e_{ni}(k+d-1)\end{aligned} \tag{22}$$

Taking the absolute value of equation (22), and applying the triangular inequality, we give

$$\begin{aligned}|\Delta y_{ss}| &\le \left|\Delta u(k).\hat{J}(k+d-1)\right|\\ &\quad +\left|\left[y\_\mathrm{Ref}(k+d)-y_{ss}\right]\right|\\ &\quad +\left|e_{ni}(k+d-1)\right|\end{aligned}$$

Applying Theorem 5.1, and also knowing that $\left|e_{ni}(k+d-1)\right| \le \left|\sqrt{2\varepsilon_{ni}}\right|$, where $\varepsilon_{ni} \ge 0$ is a beforehand chosen tolerance for neural identifier convergence, we have

$$\begin{aligned}|\Delta y_{ss}| &\le \left|\Delta u(k).\hat{J}(k+d-1)\right|\\ &\quad +\left|\sqrt{2\varepsilon_{ni}}\right|+\left|\sqrt{2\varepsilon_{nc+ni^*}}\right|+\left|\sqrt{2\varepsilon_{ni}}\right|\\ |\Delta y_{ss}| &\le \left|\Delta u(k).\hat{J}(k+d-1)\right|\\ &\quad +2\left|\sqrt{2\varepsilon_{ni}}\right|+\left|\sqrt{2\varepsilon_{nc+ni^*}}\right|\end{aligned} \tag{23}$$

It means that the plant output variation can not exceed the value computed by equation (23), under the penalty to lose the stability. Using Definition 5.2, we finally conclude that

$$CP = \left|\Delta u(k).\hat{J}(k+d-1)\right| + 2\left|\sqrt{2\varepsilon_{ni}}\right| + \left|\sqrt{2\varepsilon_{nc+ni^*}}\right|.$$

□

Definition 5.4: "The Stability Margin – SM of the Indirect Neural Control system is the factor that can be multiplied by the steady-state plant output without loss of stability."

Theorem 5.5 (Stability Margin): "The Stability Margin - SM of the Indirect Neural Control system is given by

$$SM = 1+\frac{CP}{y_{ss}}." \tag{24}$$

Proof: Using Definition 5.4, it comes that

$$SM.y_{ss} = y_{ss} + CP \tag{25}$$

Manipulating (25), finally we obtain

$$\begin{aligned}SM &= \frac{y_{ss}+CP}{y_{ss}}\\ &= 1+\frac{CP}{y_{ss}}\end{aligned}$$

□

6 Computer Simulation Results

To evaluate the robustness criteria presented in this work, it was performed the indirect neural control of a nonlinear plant through computer simulations. For this purpose, we used a neural identifier (4/1 – linear activation function) and a neural controller (4/8/1 – tangent hyperbolic and linear activation functions, in this order from input to output layer), whose learning rates were adjusted using the η-adaptive method, subject to the stability assurance [12].

The system to be controlled was obtained from [11], and it is a second order nonlinear plant with unitary delay, whose equation is the following:

$$\begin{aligned}y(k+1) &= 0.2\cos[0.8(y(k)+y(k+1))]+\\ &+0.4\sin[0.8(y(k)+y(k-1))+2u(k)+u(k-1)]\\ &+0.1[9+y(k)+y(k-1)]+\left[\frac{2(u(k)+u(k-1))}{1+\cos(y(k))}\right]\end{aligned}$$

The set-point is $y\_\mathrm{Ref} = 0.8$, for $0 \le k \le 250$. In the instant k=251, they were applied some perturbations in the plant, to evaluate the robustness of the Indirect Neural Control system. The simulations presented the following results:

1st case: Perturbation of $\Delta y_{ss} = +0.5 y_{ss}$

Before perturbation be applied, the plant was stabilized at $y_{ss} = 0.8$, when in the instant k=251 it suffer a perturbation equivalent to 50% of the steady-state output value. The plant input signal can be settled from $u_{min} = -0.5$ to $u_{max} = 0.2$. One time instant before the occurrence of the perturbation, the value of the input signal was $u(250) = -0.23$ and the estimate value of the plant Jacobian was $\hat{J}(250) = 1.5456$. Using (20) and (24), we obtain

$$\begin{aligned}CP &= \left|\left\{\left[(-0.5)-(0.2)\right]/2\right\}1.5456\right| + 2(0.0447) + 0.0447\\ &= 0.6751 = 67.51\%\end{aligned}$$

$$SM = 1 + (0.6751)/(0.8)$$
$$= 1.8439$$

Based on CP and SM values computed as above, it is seen to come that the system control has the property to return to the stable condition, once the applied perturbation (50%) was less than the critical perturbation (67.51%). The results are shown in the figure below:

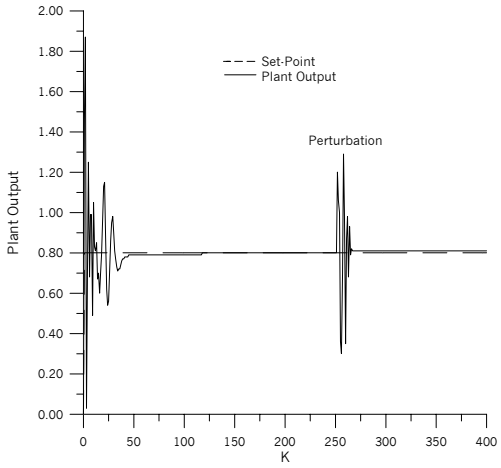

Fig.4: Plant output (perturbation=0.5y_{ss})

The results shown in figure 4 confirmed the Indirect Neural Control robust behaviour, whose values are recorded in Table 1.

Table 1: Error Limit *versus* Control Error

k (instant)	ε_{ni} , ε_{nc+ni*} (tolerance)	*sup* Ω_c (error limit)	$\omega_c(k)$ (control error)
250	1.0E-03	89.44E-03	1.76E-03
400	1.0E-03	89.44E-03	7.71E-03

2nd case: Perturbation of $\Delta y_{ss} = +0.8 y_{ss}$

The basic difference with respect to the first case is that now it is applied a perturbation equivalent to 80% of the steady-state output value. One time instant before occurrence of the perturbation, the value of the input signal was $u(250)$ = -0.23 and the estimate value of the plant Jacobian was $\hat{J}(250) = 1.9402$. Using (20) and (24), we obtain

$$CP = \left|\left\{\left[(-0.5) - (0.2)\right]/2\right\}1.9402\right| + 2(0.0447) + 0.0447$$
$$= 0.8132 = 81.32\%$$
$$SM = 1 + (0.8132)/(0.8)$$
$$= 2.0165$$

Based on CP and SM values computed as above, it is seen to come that the system control has the property to return to the stable condition, once the applied perturbation (80%) was less than the critical perturbation (81.32%). The results are shown in the figure below:

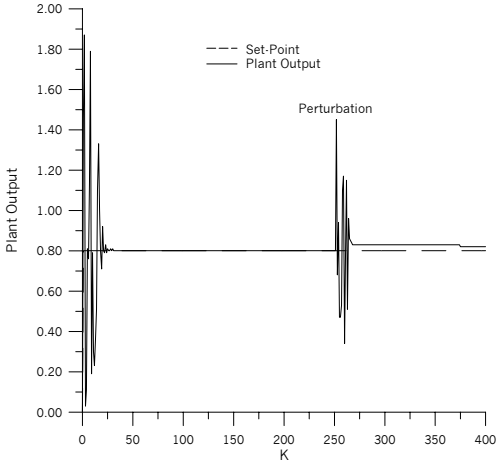

Fig.5: Plant output (perturbation=0.8y_{ss})

The results shown in figure 5 confirmed the Indirect Neural Control robust behaviour, whose values are recorded in Table 2.

Table 2: Error Limit *versus* Control Error

k (instant)	ε_{ni} , ε_{nc+ni*} (tolerance)	*sup* Ω_c (error limit)	$\omega_c(k)$ (control error)
250	1.0E-03	89.44E-03	3.07E-03
400	1.0E-03	89.44E-03	22.63E-03

3rd case: Perturbation of $\Delta y_{ss} = +1.0 y_{ss}$

In this last case, the difference with respect to the formers is that now it is applied a perturbation equivalent to 100% of the steady-state output value. One time instant before occurrence of the perturbation, the value of the input signal was $u(250)$ = -0.23 and the estimate value of the plant Jacobian was $\hat{J}(250) = 1.7485$. Using (20) and (24), we obtain

$$CP = \left|\left\{\left[(-0.5) - (0.2)\right]/2\right\}1.7485\right| + 2(0.0447) + 0.0447$$
$$= 0.7461 = 74.61\%$$
$$SM = 1 + (0.6751)/(0.8)$$
$$= 1.8439$$

Based on CP and SM values computed as above, it is seen to come that the system control does not have the property to return to the stable condition, once the applied perturbation (100%) was greater than the critical perturbation (74.61%).

The results are shown in the figure below:

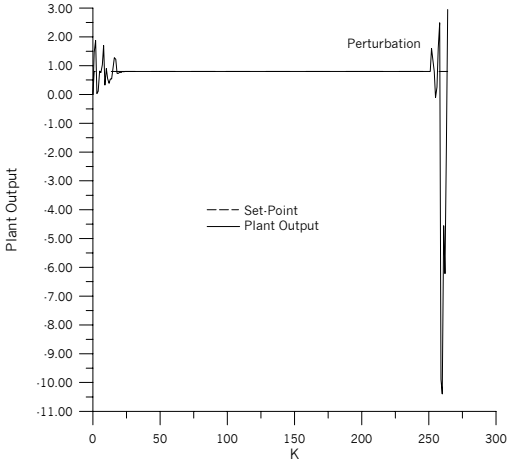

Fig.6: Plant output (perturbation=1.0y_{ss})

The results shown in figure 6 confirmed the Indirect Neural Control lack of robustness, whose values are recorded in Table 3.

Table 3: Error Limit *versus* Control Error

k (instant)	ε_{ni} , ε_{nc+ni*} (tolerance)	$sup\ \Omega_c$ (error limit)	$\omega_c(k)$ (control error)
250	1.0E-03	89.44E-03	0.74E-03
400	1.0E-03	89.44E-03	$\rightarrow \infty$

7 Conclusions

Through the analysis of the simulation results putting them on a parallel with the theoretical computed values, we can conclude that the values obtained for Critical Perturbation and the Stability Margin are conservatives, i.e., their values are less than the true values due to approximations made on deduce them. Consequently, the conclusion goes to meet a safety procedure.

In spite of the conservative value computed for the Stability Margin, it was shown that the Indirect Neural system is very robust, because it can absorber relative high perturbations comparing to the steady-state output values that existing immediately before their occurrence. This means that it is capable to support perturbations caused by modelling errors and plant parameter variations due, e.g., to the unknown dynamics and the aging of the physical components.

Finally, we conclude that the new methods for evaluation of the neural robustness presented in this work produced good results, and could be used as a tool for Indirect Neural Control design.

References:

[1] Rumelhart, D.E., McClelland, J.L. e The PDP Group. Parallel Distributed Processing: Explorations in the Microstruture of Cognition, Vols. 1 e 2.*The MIT Press*. Cambridge, Massachusetts. USA. 1986.

[2] Gabriel, O. Fo. *A Neural Adaptive Control Scheme with On-Line Training*. Master Dissertation. Federal University of Rio Grande do Norte. Natal/RN. Brazil. 1996.

[3] Maitelli, A.L. e Gabriel Fo, O. A Neural Adaptive Control Scheme with On-Line Training. In: *Proc. of the 7º Latin-American Congress of Automatic Control - LACC - IFAC*, Volumen 2, pp. 887-892. Buenos Aires. Argentina. 1996.

[4] Haykin, S. *Neural Networks*. Artmed Editora Ltda. Porto Alegre/RS. Brazil. 1999.

[5] Pansalkar, V.V. e Sastry, P.S. Analysis of the Back-Propagation Algorithm with Momentum. *IEEE Transactions on Neural Networks*, Vol. 5, No. 3, pp. 505-506. USA. 1994.

[6] Nϕrgaard, M., Rvn, O., Poulse, N.K. e Hansen, L.K. *Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag London Limited. London. England. 2001.

[7] Tanomaru, J. e Omatu, S. Process Control by On-Line Trained Neural Controllers. *IEEE Transactions on Industrial Electronics*, Vol. 39, No. 6. USA. 1992.

[8] Maitelli, A.L. e Gabriel, O. Fo. A Neural Adaptive Controller Applied to a Nonlinear Plant. In: *Proc. of the III Brazilian Congress of Neural Networks,* pp. 466-471. Florianópolis/SC. Brazil. 1997.

[9] Maitelli, A.L. e Gabriel, O. Fo. Stability and Robustness Analysis Applied to a Indirect Neural Control System. *Controle&Automação Journal* (still being reviewed). Brazil. 2003.

[10] Maitelli, A.L. e Gabriel, O. Fo. Indirect Hybrid Controller Based on Neural Networks – Part I: Development and Implementation. In: *Proc. of the 6º Intelligent Automation Brazilian Symposium,* pp. 183-188. Bauru, Brazil. 2003.

[11] Adetona, O., Sathananthan, S. e Keel, L. H. Robust Nonlinear Adaptive Control Using Neural Networks. In: *Proc of the American Control Conference*, pp. 3884-3889. Arlington, VA. USA. 2001.

[12] Ng, G.W. *Application of Neural Networks to Adaptive Control of Nonlinear Systems*. Research Studies Press Ltd. London. England. 1997.