Adaptation of Weights in a Neuron Using an Integrated Filter

Karen Alicia Aguilar Cruz[1], María Teresa Zagaceta Álvarez[2], José de Jesús Medel Juárez[1]

[1] Instituto Politécnico Nacional, Centro de Investigación en Computación (CIC), México D. F, Mexico

[2] Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica (ESIME-A), México D. F, Mexico

karen_ali320@hotmail.com, mtza79@yahoo.com.mx, jmedeljj@yahoo.com

Abstract. This paper presents a description of application of stochastic weights in a neuron, problem solved through the adaptive estimation achieved with dynamical combination between the identification and estimation; having an adaptive structure that updates the estimated parameters into the integrated filter. The weights are dynamically adjusted in the neuron based on stochastic gradient, affecting the neuronal performance allowing that its response converges to the reference signal. In addition, the error is applied in identification as an innovative gain adjusting the neuron in its inputs and consequently its dendrites signals that are applied into gradient filter adjusting the neuron weights in accordance with the desired signal requirement. Such that the gradient estimation is built based on the Black-box scheme with unknown internal weights. All simulations were developed using Matlab® software.

Keywords: Estimation, stochastic systems, neural net, digital filter, identification.

1 Introduction

An artificial neural net is a computational model that imitates the biological actions: observing that the neurons adapt their gains using the learning process as it occurs in the brain or neural sensor subsystems. Different effects depend on the output stimuli (Nikola, 1996) (Medel, 2008). So, the artificial net considers the weights adaptation as a requirement, in accordance with the reference signal and the stimuli of the inputs.

The neuron maintains an electrical potential interval from 35×10^{-3} to 65×10^{-3}volts; but when a neuron is fired, an electrical impulse is increased; this is an electric energy generated by chemical effects, releasing an electrical potential from 90×10^{-3} to 110×10^{-3} volts. This impulse through the neuron is transmitted from 5×10^{-1} to 1×10^{2} metres per second and is distributed on average in a $1 \times$

10^{-3} *second*. In addition, the fast repetition rate corresponds on average to 10×10^{-3} seconds per firing. A computer, where signals travel on average at $2.0 \times 10^{8}\,\frac{m}{s}$ (electrical speed energy in a wire is *0.7* faster than in air), may repeat an impulse each 10×10^{-9}*seconds*. So, the computer device has in average two thousand times more speed in signal transmission and a thousand times in the fire signal repetition with respect to natural neuron action (Passion, 1998), because it uses the solid state instead of chemical reactions. But, for example, the main advantage of the brain with respect to other electronic devices is the possibility of "*self-programming*" with the changes of external stimuli, known as "*adaptability*". In other words, it can learn dynamically and in variable conditions. Naturally, the brain neurons change their response to new stimuli, having similar responses to similar events. The brain adaptability corresponds to survival actions, while a device that just accomplishes a sequence of commands.

1.1 Neural Network Structure

The computational neural net structures are based on biological neural configurations. The basic neural net is based in a neuron model, shown in Figure 1, consisting of Multiple Inputs and a Single Output (MISO form).

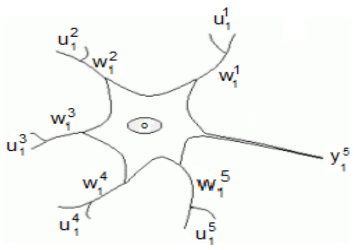

Fig. 1. Neuron model.

Each input is modified by a *weight*, which multiplies the input values. A neuron combines dendrite weight inputs and if the soma biological actions exceed a threshold, then the nucleus (in a biological sense) activates a function and determines its output answer. In a computational device, as shown in Figure 2, a behavioural additional condition has the answer close to the real neuron actions (Rajen, 2006).

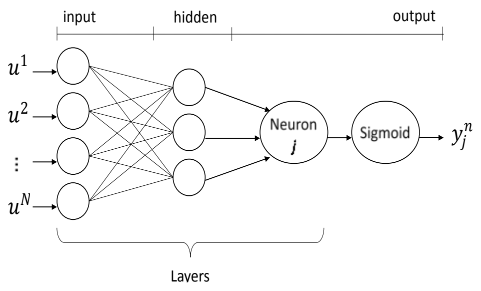

Fig. 2. Neuron device computational model.

Meanwhile, understanding how an individual neuron operates, many researches generate the way neurons organize themselves and the mechanisms used by neuron arrays to adapt their behaviour to external bounded stimuli. There are a huge number of experimental neural nets, and actually, laboratories and researchers continue building new neural net configurations in order to develop intelligent and autonomous systems.

The common computational neural net used is named as a *back-propagation network* and is characterized with a mathematical structure model that knows its behavioural stability conditions (bounded inputs and bounded output, BIBO conditions).

Intuitively it is built taking a number of neurons and arrays them forming a *layer*. A layer is formed having all inputs and nodes interconnected with others nodes, but not both within the same node. A layer finishes with a node connected with a succeeding layer or outputs giving the answer. The multiple layers are arrayed as an input layer, multiple intermediate layers and an output layer is shown in Figure 3, where the intermediate layers do not have inputs or outputs to the external world and are called *hidden layers* (Marcek, 2004).

Back-propagation neural networks are usually *fully connected* to improve the learning process. This means that each neuron is connected to every output from the preceding layer.

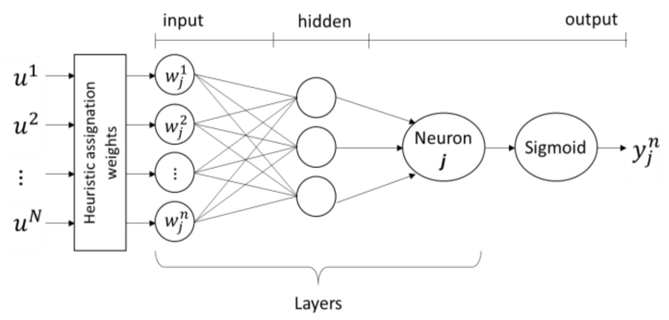

Fig. 3. MISO Back-propagation Network with three layers.

The layers are described as: input, distributing signals from the external world; hidden, categorizing the signals; and the output, collecting all features detected and producing a response. However, the problem of the layers has many descriptions considering the set of optimal weights.

1.2 Neural Network Operation

The output of each neuron is a function of its inputs and weights, with a layer as described recursively in (1) (Huang, 2006).

$$W_j^N = w_j^n u_n + W_j^{N-1}, \tag{1}$$

where the basic function has the form $W_j^{N-1} = \sum_{n=1}^{N-1} w_j^n u_n$.

The output neural net answer is a convolution operation, shown in (2).

$$Y_j^N = (F \circ W)_j^N. \tag{2}$$

The W_j^N value is convoluted with a threshold value giving an approximate biological neural net answer; but in a computational sense, it is active considering a t_j^N known as an activation function. The activation function usually is the sigmoid function.

The output vector answer Y_j^N is the neural net response, observing that the threshold function corresponds to biological electrical potential of 90×10^{-3} to $110 \times 10^{-3}\ volts$ needed in synopsis operations.

The biological or computational fire answers correspond to threshold conditions that accomplish the excitation functions generating an answer giving many inputs. Generally, the weights are selected intuitively in the first step; but with adaptive considerations, they can be adjusted to seek the desired answer (García, 2008).

2 Net Adapting its Weights Using Stochastic Filtering

Adaptation in a neural net means adjusting its weights with a law action, seeking the convergence to the output desired. The difference between the desired and actual response is known as convergence error, defined as (3) and shown in figure 4.

$$e_j^N = \hat{Y}_j^N - Y_j^N. \tag{3}$$

The filtering action could be a sliding mode, proportional gain in its weight and other non-linear models that allow the neural net convers to the desired answer with respect to the input set, but instead of it, in this paper we propose the identification technique shown in figure 4, that adjusts the inputs, predicting how many gain is required to minimize the inputs with respect to the desired reference signal.

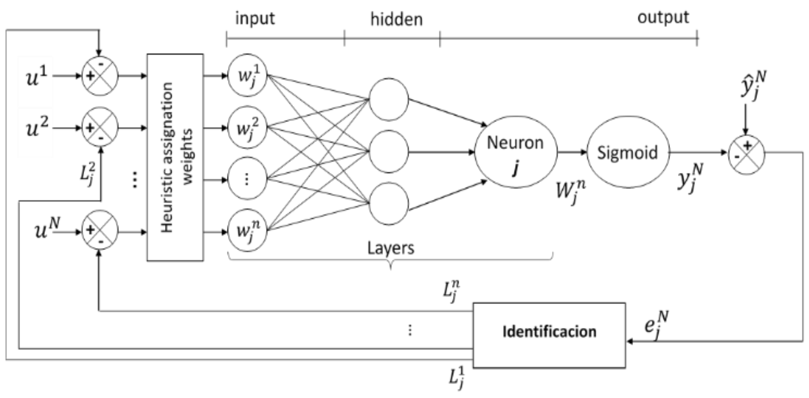

Fig. 4. Neural weights adjustment using an identification action.

The adaptive *back-propagation* procedure is described in (4):

$$u_j^{n\prime} = u^n - L_j^n, \tag{4}$$

where L_j^n corresponds to identification action considered by neural net designer.

Now, applying the concept considered above with respect to neural net, it adjusts its weights using stochastic estimation giving a great advantage over traditional inference weights assignation heuristically.

The neural net has adaptive weights based on an identification with its estimation, associating the output filter information with the neuron answer (Huang, 2006), building the control volume described as $T^N = \{(y_j^n, \hat{y}_j^n)\}_{n-\overline{1,N}} \subseteq R^2$ where a variant scheme has the form $G_N: (Y_j^N \times \hat{Y}_j^N) \times T \to \{((y_j^n, \hat{y}_j^n), \tau)\}\Big|_{n-1}^{N} \subseteq R^3$ (Margaliot, 2000), with dynamical adjusted moments (Gustafsson, 2001) in accordance with the reference previously defined in a distribution sense.

The neuro-stochastic filter is based on the back-propagation algorithm, because its weights have a dynamic actualization (Ali, 2003) (Amble, 1987) (Haykin, 1996) with different levels for each interval iteration (Huang, 2006), using the error described partially as $e_j^n \in R$ defined as $e_j^n := \hat{y}_j^n - y_j^n$, considering that its distribution function (Marcek, 2004) (García, 2008) is bounded and the statistical results have stationary conditions. Filter is shown in Figure 5, using the estimation weights (Passino, 1998) (Medel, 2008).

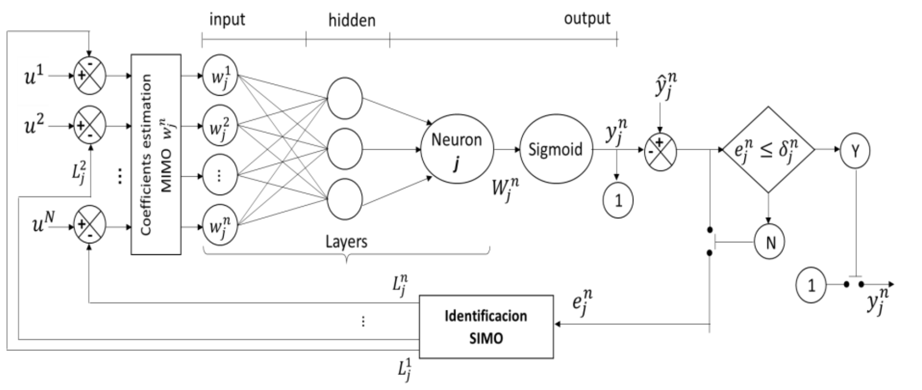

Fig. 5. Neuro-stochastic Digital Filter Process.

The error $(|e_j^i|)$ has an interval limit $[0, \varepsilon]$ and ε is described as a positive value with $\inf\{ |e_j^i| : i, j \in Z_+\} \xrightarrow[n\to\infty]{} \delta_j^n$ (Morales, 2002).

Stochastic filter applied into neuron considers the concepts described in (Abraham, 1991) and (García, 2011), having the elements needed in its basic description: back propagation neural net scheme, adaptive weights considering the estimation and identification, convergence answer, the error as an innovation process e_j^i with its bounded probability moments, in a metric sense, [19]. Activation function is the stage where the answer filter is transformed into a natural answer approximating to minimal convergence error region, and neuro-stochastic filter has a natural actualization obtaining its weights dynamically based on second probability moment into the basic estimation action (5) considering de gradient description.

$$J^n = \frac{1}{n^2}\left[e_j^{n^2} + (n-1)J^{n-1}\right], \in R_{[0,1)} \;\; n \in Z_+. \tag{5}$$

The functional error J^n has an exponential convergence and stationary conditions if the weights set into filter established a stationary reference $\lim_{n\to\infty} |J^n| \to m$, considering that $0 < \{|e_j^i|\} < 1$ and (6).

$$J_{min} = \inf_m \{\min J(y_{i-0}^j, \hat{y}_j^i)\}_n. \tag{6}$$

Considering the gradient estimation in accordance with the desired signal and filter action, firstly, the filter process adjust the inputs, and these are applied into the gradient estimation adjusting the weights and generating in the same time the adaptive process guarantying the convergence rate (Rajen, 2006). Then, the weights $\{w_j^i\}_{i=\overline{1,n}, j=\overline{1,m},}, n, m \in Z_+$ affect the neuron elements and consequently will give the correct answer $\hat{y}(k)$ (Ash, 1970), with MISO (Multi Inputs Single Output) properties. It means that (5) without concurrence has the form $J^n = \mathbf{E}\{e_j^n\}^2 =, \in R_{[0,1)} \;\; n \in Z_+$, with $e_j^n = y_j^n - \hat{y}_j^n \; and \; y_j^n = Wy_j^{n-1} + Bu_j^{n\prime}$. Such that, the functional error with symmetric conditions has the form with explicit output results as $J^n = \mathbf{E}\left\{W^2 y_j^{(n-1)^2} + B^2 u_j^{n\prime 2} + \hat{y}_j^{n^2} - 2(Wy_j^{n-1} + Bu_j^{n\prime})\hat{y}_j^n\right\}$. The gradient of J^n, allows to have the neuron weights $\widehat{\boldsymbol{W}}_j^n = \left(\boldsymbol{E}\{y_j^{n-1}\hat{y}_j^n\}\right)\left(\boldsymbol{E}\left\{y_j^{(n-1)^2}\right\}\right)^{-1}$.

2.1 Weight Properties

The filter weights estimation uses the adaptive criterion, in order to adjust them dynamically, considering the stochastic properties and bounding each of them using a transition function maintaining the stability. The weights set $\{w_j^i\}_{i=\overline{1,n}, j=\overline{1,m},}, n, m \in Z_+$, in each layer accomplishes the condition $\sum_{i=1}^n w_j^i \le 1$, without losing the Transition Function (TF) (García, 2008):

i. Each weight has a Dynamic Transition Function (DTF): 1) $\ln(\Phi_j^i) < \infty$, 2)$\ln(\Phi_j^i) > 0$, 3)$\ln(\Phi_j^i)\, \tau^{-1} < 1$.

ii. The weight is described using the Transition Function (TF) in $w_j^{1-i_0} = \ln(\Phi_j^i)\left(\ln(\Phi_j^i)(i - i_0)\right)^{-1}$.

iii. The velocity changes are limited inside the transition function $\ln(\Phi_j^i) \le \ln(\Phi_j^{i_0})(i - i_0)^T$, $\ln(\Phi_j^i) \le \ln(\Phi_j^{i-1})\,(i-1)^T$.

The transition functions sum is bounded in each layer $0 \le \left|\sum_{i-1}^n \Phi_j^i\right| \le 1$. In accordance with the value of $\Phi_j^{i_0}$, the weights are bounded accomplishing with $w_j^{1-i_0} \le \ln \Phi_j^{i_0}$.

The identifier described as $\hat{x}_i = w_j^{\mathrm{i}}(i - i_0)\ \hat{x}_{i-1} + K_i\hat{\omega}^i$ considering from (i) to (iii), where K_i is the function gain and is a functional identification error, defined by the second probability moment (5), $\hat{\omega}^i$ is the innovation process with $\{\hat{\omega}^i\} \subseteq N\ (\mu_{\hat{\omega}^i}, \sigma^2_{\hat{\omega}^i} < \infty)$.

3 Results

The MISO stochastic filter considers the digital filter structure (Haykin, 1996) with the transition matrix bounded in accordance with the functional error criterion (Ash, 1970). The soft system (statistic in variance sense) considers the evolution times bounded and the processor performance at τ intervals with an average evolution time of 4×10^{-3} $sec \pm 2 \times 10^{-5} sec$. This section uses the first order difference discrete ARMA (1, 1) model (7) representing a reference system.

$$x^{i+1} = W_i x^i + \omega^i. \tag{7}$$

And the output described as (8):

$$y^i = Cx^i. \tag{8}$$

where $y^i \in R, W_i \in R^{[n\times n]}_{[0,1)}, \boldsymbol{x}^i, \boldsymbol{\omega}^i \epsilon R^{[n\times 1]}, C = I$. x^{i} is the internal states vector, W_i is the parameters matrix, $\{\omega^i\} \subseteq N\big(\mu_{\omega^i}, \sigma^2_{\omega^i} < \infty\big)$ is the vector noise into the system, y^{i} is the reference vector and $\hat{y}^i$ is the desired system signal. The filter process established the stochastic weights adjusted in agreement to the functional error convergence. Figure 6 describes the reference signal and its identification without knowing the internal matrix weights considering the estimation results $\widehat{\mathbf{W}}_{\mathrm{j}}^{\mathrm{n}}$. Figure 7 shows both overlapping densities considering the same time interval. Figure 8 shows the evolution functional error described in (5).

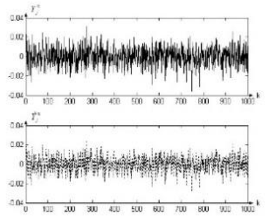

Fig. 6. Neuro signal Y_j^n and its identification $\hat{Y}_j^n$.

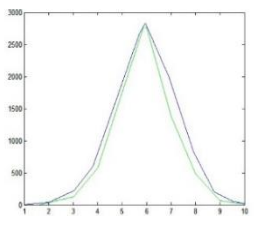

Fig. 7. Overlapping y_j^n and $\hat{y}_j^n$ densities.

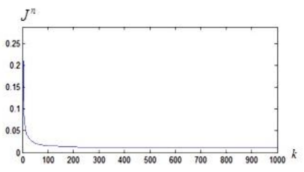

Fig. 8. Functional error (5).

The digital filter time evolution response was less than the reference process time state change, proposed with a value of 5×10^{-2}sec, and is delimited by the processor, considered in $(\hat{y}_j^n)$. The convergence time is 862×10^{-4} sec, described in (Medel, 2008).

4 Conclusion

Neural net in identification sense, considered the adaptation process adjusting the weights dynamically using the estimation condition. Nevertheless, in many cases, these applications generate convergence problems because the gains increase the neural net weights positive or negatively without converge to desired value. In the black-box computational scheme the internal weights are known; but in real conditions it is impossible and only has a desired or objective answer, adjusting in some sense to their dynamically needing estimation process with smooth movements with respect to functional and identification error (5). Therefore, an option considered to estimate these in the new environmental circumstances, is based on gradient structure without losing the stability with respect to a reference system and on the Hausdorff condition, where the filter converge to the desired output system in distribution sense.

References

1. Abraham K.: Stochastic Expert Systems. Florida, CRC Press (1991)
2. Ali H.S.: Fundamentals of Adaptive Filters. John Wiley & Sons, New Jersey, USA (2003)
3. Amble T.: Logic Programming and Knowledge Engineering. Addison Wesley, USA (1987)
4. Ash R.: Real Analysis and Probability. Dover Publications, USA (1970)
5. García J.C., Medel J.J., Sánchez J.C.: Neural stochastic digital filtering: MIMO case. Ingeniería e Investigación, 31: 184–192 (2011)
6. García J.C., Medel J.J., Guevara P.: Filtrado Digital Difuso en Tiempo Real. Computación y Sistemas, 11: 390–401 (2008)
7. Gustafsson F.: Adaptive Filtering and Change Detection. John Wiley and Sons, England (2001)
8. Haykin S.: Adaptive Filtering. Prentice Hall, USA (1996)
9. Huang G., Zhu K., Siew C.: Real-Time Learning Capability of Neural Networks. IEEE Transactions on Neural Networks, 17: 863–878 (2006)
10. Mamdani E.: Applications of Stochastic Algorithms for Control of Simple Dynamic Plant. Proc. IEEE, 121: 1585–1588 (1974)
11. Marcek D.: Statistical, Classical and Stochastic Neural Networks, Modeling Decisions for Artificial Intelligence. Springer Verlag, pp. 41–48 (2004)
12. Langholz M.M.: New Approaches to Stochastic Modeling and Control Design and Analysis. Singapore, World Scientific (2000)
13. Medel J.J., García J.C., Sánchez J.C.: Real-time Neuro-Stochastic Digital Filtering: Basic Concepts. WSEAS Transactions on Systems and Control, 3: 654–663 (2008)
14. Medel J.J., García J.C., Guevara P.: Real-time Stochastic Digital Filters (RTFDF) Properties for SISO Systems. Automatic Control and Computer Sciences, AVT, 42: 26–34 (2008)
15. Morales G.: Introducción a la Lógica Difusa. Cinvestav, Mexico (2002)
16. Nikola K.: Foundations of Neural Networks, Stochastic Systems, and Knowledge Engineering. Hong Kong, The MIT Press (1996)
17. Passino K.M.: Stochastic Control. USA, Addison Wesley (1998)
18. Rajen B., Gopal M.: Neuro-Stochastic Decision Trees. International Journal of Neural Filters, 16: 63–68 (2006)
19. Schneider M., Kandel A.: Stochastic expert systems tools. John Wiley & Sons, England (1996)

20. Takagi T., Sugeno M.: Stochastic Identification of Systems and its Applications to Modelling and control. IEEE Transactions and Systems, Man, and Cybernetics, 15: 116–132 (1986)
21. Yamakawa F.: Stochastic Neurons and Stochastic Neural Networks. (1989)