Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp248-253)

Trend-removal in Corrosion Processes using Neural Networks.

MILTON MARTINEZ LUACES
Facultad de Ingeniería
Universidad ORT Uruguay
Cuareim 1451, Montevideo
URUGUAY
http://webs/montevideo.com.uy/miltonmartz

VICTOR MARTINEZ LUACES, MAURICIO OHANIAN
Facultad de Ingeniería
Universidad de la República
Julio Herrera y Reissig 565, Montevideo
URUGUAY
victorml@fing.edu.uy mohanian@fing.edu.uy

Abstract: Lifetime evaluation of metallic components is one of the main subjects for many industries due to its importance for development of metal protection and conservation methods. Looking forward this information, several electrochemical techniques have been applied by practitioners. One of the most important is the so called *Electrochemical Noise Study*, which makes possible to measure potential or real fluctuations produced by kinetic variations along the corrosion process.

This technique makes necessary to apply signal processing methods, including low frequency *trend removal*. Many statistical methods have been proposed to do so. In order to assess each method performance, it is a must to know exactly both trend and noise for a particular signal. Then, a statistical comparison can be carried out between noise extracted with different methods and *real* noise. With this purpose, sometimes signal is simulated by computer data generation.

In this paper, a new approach is proposed for *trend removal*, using artificial intelligence techniques instead of statistical methods. With this purpose, we combined an interval signal processing with *backpropagation* neural networks. Data pre-processing, used topologies, optimizations and training process are exposed in detail. Results are analysed and conclusions drawn.

Keywords: electrochemical noise, signal processing, trend removal, noise filtering, neural networks, backpropagation.

Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp248-253)

1. Introduction.

The study of corrosion process in metals plays a main role in many industries. This happens obviously because metal corrosion by the environment is a factor that must necessarily be quantified to do an accurate determination of an industrial product lifetime. This is the case of metal structures widely used in construction industry, and the same occurs with metal components of automotive, naval and aeronautic industries. In all these, many studies have been made with the goal of finding more accurate methods for measuring and predicting oxidation processes [1].

One of the most employed processes is the so-called *Electrochemical Noise s*tudy (EN). In this process, an electrochemical signal that follows a particular trend is perturbed with a certain noise. In real conditions noise appears as a result of oxidation reactions. Therefore, to understand corrosion behaviour it is crucial to detrend the signal function isolating this noise. This procedure is known by the name of *trend removal* [3]. Several methodologies have been applied to implement it, especially statistical procedures [20]. Despite they have been proved to be useful in some cases, a really accurate procedure for *trend removal* is still an open problem. As a matter of fact, it is relatively easy to remove noise from a noisy function, by polynomial approximation, *Artificial Neural Networks* (ANN) [6] or genetic programming [11] [10]. However, it is extremely difficult to do the inverse filtering. The fact is that *trend removal* cannot be done by polynomial approximation because if we do so, low-level relative errors in *trend* approximation, become unacceptable high-level ones in case of *noise* approximation because noise values have a higher magnitude order. In this paper, we propose to combine computer data simulation for both *trend* and *noise* with ANN building a feasible and also adaptable *trend-removal* tool for different purposes. We first survey existing statistical methods, even one we proposed in a previous paper [17]. Use of non-trivial functions is proposed and random noise with a distribution function according to real conditions. Data simulators were implemented for training data set generation. With this set, *backpropagation* ANN are trained splitting the signal in intervals and then training three different neural networks, depending on the mean derivative of each segment. Once done, a data testing set is generated for crossed validation. Finally, results are analysed and conclusions extracted.

2. Trend Removal in EN.

In majority of studies, electrochemical techniques are applied to characterize oxide products properties. One of this techniques recommended by many researchers [15] is based on EN, where real or potential fluctuations (produced by corrosion kinetics) are measured. This technique has several advantages:

1) No need of external signal. This allows testing the system in electrochemical equilibrium conditions.

2)Technical equipment is less expensive than other[3].

3) Information provided by EN technique allows the researcher to determine different parameters related with kinetics and localization of the studied process. A point to be noted is that both parameters play a key role in metallic structures design [16].

The main disadvantage of this technique is the high dispersion observed on experimental results [4]. In a previous work [17] three different factors (*electrolyte*, *frequency simulation* and *trend removal*) were analysed in order to determine causes for the observed high dispersion. The experiments done modifying these parameters, showed that *trend removal* method is the one which best explain this phenomenon.

Signal processing includes removal of continuous components and/or removal of low-frequency signals. For method performance evaluation, it is a must to know exactly both trend and noise for a particular signal. Then, a statistical comparison among different methods by the error-level obtained can be done. With this purpose, trend is computer simulated, and also noise, using random data generation, with inverse Gaussian distribution (with a null mean and 0.5 of standard deviation in this case).

One important condition is to avoid using trivial functions for trend sampling. In this paper, we name trivial (from a trend removal viewpoint), such functions as polynomial, sinusoid, exponential ones, and others that can be easily approximated by elementary mathematical methods such as polynomial regression. In a previous work [17], several non-trivial functions were tested, and as a result, Lorentz function proved to be the most interesting case. Indeed, methods such as polynomial regression or digital filtering obtained standard deviations of 1.8689 y 0.6489 respectively, while real standard deviation was 0.5043 (with a Lorentz trend). Obviously, these error levels (more than 300% and 29%) are absolutely unacceptable. Moreover, these poor results were later confirmed by the respective PSD (Potential Spectral Density). This showed that in both cases, these methods had a lack of performance for low frequency *trend removal* [17]. Therefore, taking our previous works into account, with a reuse-oriented approach this time we allowed the use of different trend functions, parametrizing the detrend tool for different domains. In this paper, particularly Lorentz trend function is analysed. This function is computed with the following expression:

Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp248-253)

$$f(x) = y_0 + \frac{2A}{\pi} \frac{w}{4(x - x_c)^2 + w^2}$$

Different values were applied for the parameters y_0, A, w_c and x_c, to obtain a signifying variance. Once parametrized, functional values are obtained by discretization of x on 0.1 intervals until 400 functional values are collected. The following chart shows the result:

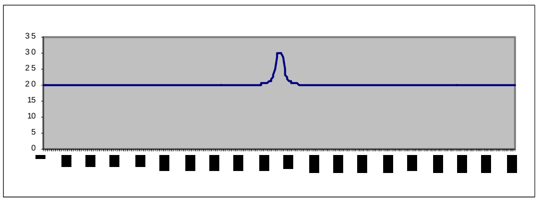


Chart 1. Lorentz Function.

3. Noise distribution

As in case of *trend*, the goal with *noise* is simulating real conditions as well as possible. In real conditions, noise is unpredictable, however its distribution is known. Taking this into account, a random function for noise data generation was implemented, but with an inverse Gaussian distribution, according to real conditions. For doing so a numeric algorithm was applied [19]. It consists in splitting into three sections the Gaussian function according to the derivative level (low, medium and high) and then applying different algorithms for each section for error minimization. Once done, *noise* generated function is added to *trend* function, obtaining a noisy signal adequate for *trend-removal* testing. It can be mentioned that in some cases, this kind of methodology is applied for statistical method validation using spreadsheets, so it seems to be adequate to establish comparisons using a similar technique in ANN. Anyway, in our case, these functions were implemented into the tool, in order to make possible the "self-training" of the ANN with the generated data. Indeed, the effort is well worth the invest, because this approach not only has the advantage of allowing the self-training, but also as a result of this a more exhaustive testing can be done. In the following chart, it can be seen the Lorentz function once perturbed with the noise data generated.

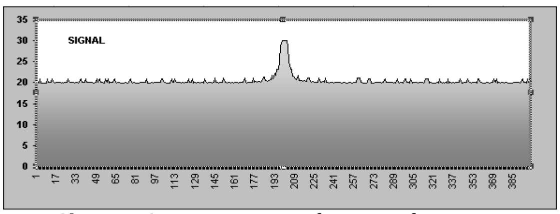


Chart 2. A noisy version of Lorentz function.

4. Statistical Methods

As we mentioned before, there are many methodologies for *trend removal* to isolate the EN. Mainly, four different statistical methods for *trend* removal have been applied:

1) *Polynomial fitting* is a well-known technique widely used [15]. Trend is fitted using squares regression, and then noise is obtained as the difference between experimental and predicted data.

2) *MAR-5*. In this method, noise is computed using a moving average [16].

3) *Butterworth*, which uses analogical filters [17]. It is part of the *Matlab Signal Processing Toolbox©*.

4) *MICS* is a *trend removal* technique that we proposed and extensively tested in a previous work [17]. This technique consists of computing medians over intervals and then fitting by cubic splines.

In most of these statistical methods, the relative error level is acceptable for *trend*, but usually unacceptable and useless in case of noise, due to low relative errors in case of *trend* that become high ones in case of noise because of their different magnitudes.

5. Using neural networks

From their very beginning ANN were mostly employed on function approximation [7], [22]. From this viewpoint, *trend-removal* can be seen as a particular case of function approximation. ANN are also widely used as a statistical analysis tool. In our case, statistical analysis has already been done in previous works, so the main idea of this paper is to apply an ANN methodology to improve our *trend removal* method. Moreover, our domain (environmental corrosion) is essentially dynamic, due to high variance of functional trends, so it seems to be especially adequate the use of ANN, because of its inherent dynamic nature. Because a big quantity of both patterns and output data, we preferred *feed-forward* ANN, that have proved to be especially useful in function approximation. According to Universal Approximation Theorem [5], feed-forward ANN of three or more layers with *backpropagation* algorithm can approximate every continuous function (such as Lorentz ones) so it is specially recommended its use in our case.

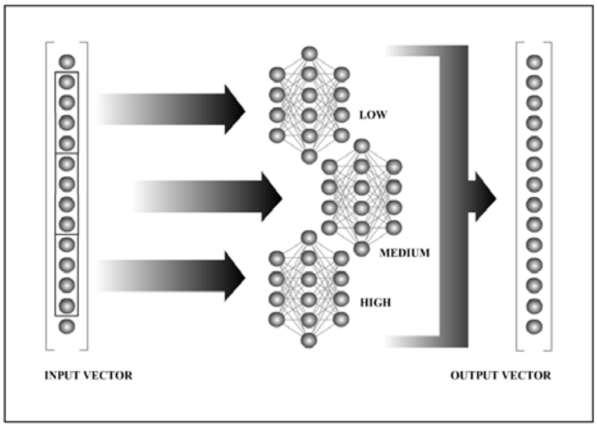
6. The proposed methodology

Simulated noise has a Gaussian distribution with null media and standard deviation of 0.5. As we mentioned before, function must not be a trivial one for a proper *trend-removal* test. First, in order to simulate real conditions, data generation was implemented with Lorentz function, being itself a non-trivial function.

In noise-removal methods, a classical application of ANN, the main fact is to obtain an accurate measure of the *trend*, for training-set. Once trained, it is possible to use the ANN for noise-removal. This kind of technique has been early applied in ANN, by well-known researchers, particularly Bernard Widrow [24]. Because in our case the goal is to find a method for *trend*-removal, we must focus in finding an accurate measure of the noise, as Widrow himself suggested us [24]. Therefore, we put special attention in simulating noise as realistic as possible, implementing for this purpose a random data generator, but with an inverse Gaussian distribution, according to real conditions observations. As our goal is to approximate noise function minimizing error, we implemented a *Backpropagation* ANN. This is a supervised-learning ANN, meaning that real results are provided with the training data set. Then, the ANN applies an algorithm based in the descendent gradient, in order to minimize the *mean square error* (MSE), the classical error measure in these cases. In the input layer, we began with 50 data (thus, 50 neurons, one for each input data), but in this case we obtained poor convergence, so we later reviewed this parameter, deciding to increase resolution several times, until we fixed it in 400 input neurons which proved to be the best.

About the topology used, according to Funahashi's Universal Approximation Theorem, already mentioned, one hidden layer (three layers ANN) is enough to approximate the noise function [12][13]. Anyway, for performance reasons, in tuning phase we added a second hidden layer, a practice recommended in especially complex problems [14]. Learning rate was fixed in 0.1 and a threshold was also implemented in the *Backpropagation* ANN. A *momentum* factor was also added, and fixed in 0.3 [18], in order to accelerate convergence and avoid oscillations. The ANN developed is available in the Web site for testing.

Because the method needs to approximate different functions, we avoided including the whole function range in the input vector of the ANN. This is crucial, because if we do so, the ANN would memorize the *trend-removal* function, then our method would become useless for other trend cases. Opposite to that, we applied an interval processing (a sort of moving-window, commonly employed for temporary series). We used a 40-data interval, with the result that the original vector is split in 10 vectors of 40 data each. These 40 data vectors are then used as inputs for the ANN. In the following schema we can a graphical representation of the procedure decrypted:



Schema 1. Split and join of signal segments.

By this approach, the ANN improved. Many optimization techniques had been suggested for *feed-forward*, one of them is the use of an ACON structure that is, using many sub-nets depending on data entries. In our case, we used three sub-nets, of equal topology, but with different weight matrix. In the input layer, we loaded 40-data vectors in one of the three sub-nets depending on the media signal derivative on the interval (we divided the function range in three sectors, namely low, medium and high derivative, using finite differences). Finally, obtained vectors in the output layer are joined into one vector of 400 data-elements, composing the final solution, as we could see in the previous schema. In the training-phase, different values of learning rate and *momentum* were tested. In the case of values higher than 0.3 in learning rates, we observed oscillations as it could be expected, and then it was finally fixed in 0.15, with the result that the ANN advances uniformly to the solution, with little oscillations. Moreover, the steps are more and littler near a local minimum, with the momentum factor added, but it also accelerates de convergence when it is still far from a local minimum.

Then, the ANN was integrated with the data generators to complete the tool.. The data inputs and outputs were implemented with text files or spreadsheets.

7. Training process

For training process, we first used a single expression for the noise, training the ANN continuously with it. In this first phase we could prove the ANN memorizing ability as a first step. Of course, with single noise data, the ANN could approximate to any desired error level. On a second phase, we trained the net with a Lorentz function with fixed parameters, but with random noise (inverse Gaussian distributed). We loaded the ANN with different noise values at each step, according to real conditions. In this phase, data files were saved after 10, 100, 1000, 5000, 10000 y 30000 iterations. These results showed a certain oscillation level, but with a clear media and standard deviation descendent direction of the MSE. Based in this second phase, it can be seen that with 30.000 training epochs, a 0.2% *trend* relative error and a 7% noise relative error which is pretty good, considering that the function is random, and thus unpredictable itself. In a third and final phase, we randomised both noise and Lorentz function parameters into a determined interval according to real conditions. In this case, the most exhaustive testing, relative errors were increased, as it could be expected, but still acceptable in comparison with statistical methods we used before. The following chart shows a comparison at different training levels.

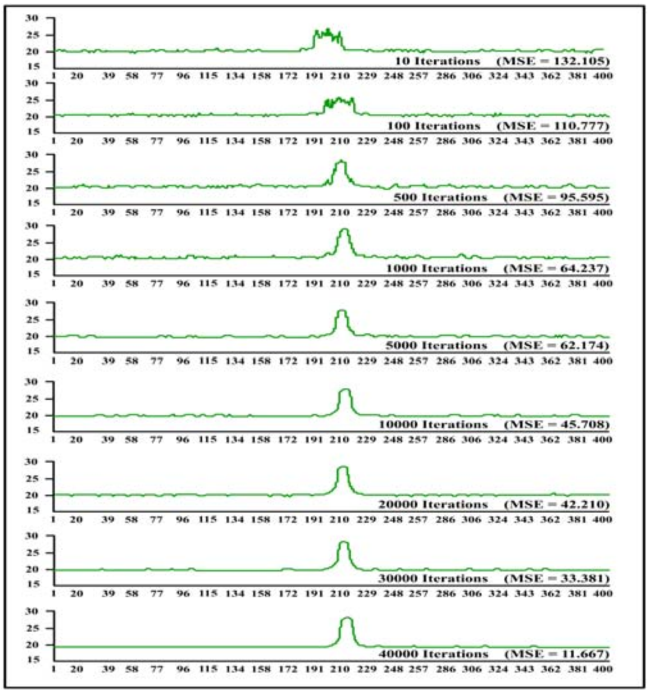


Chart 3. Predicted trend at different iterations

7. Validation

Several statistical measures are useful for ANN validation: media, standard deviation are primary metrics. More information can be obtained with *kurtosis* and PSD. Crossed validation [5][8] was also used, but not with the classical procedure of splitting data in *training* and *testing sets*, but generating a *testing-set* once concluded the training phase, as the data generated, we can always obtain fresh data for testing. On-line cross validation was also applied, as we have already done in previous ANN works [21], but in this case using new testing data each step. With the optimizations applied the convergence was good, although it can be improved with second order[9] or even genetic algorithms [11][23]).

8. Results

Once implemented the ANN and the generators, we applied the self-training of the tool with the generated data. Because Lorentz function is randomly parametrized, and later perturbed with random noise (inverse Gaussian distributed), a certain oscillation-level takes place, although it tends to stabilize after the 10000 epochs. In the following chart it can be seen the MSE value during training process.

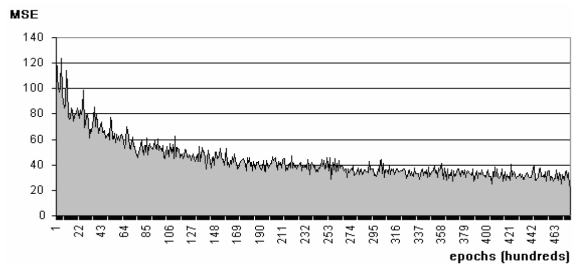


Chart3. Evolution of the MSE expressed in hundred of epochs

And in the following chart, media and standard deviation of MSE values (in groups of 10 values), are shown [2].

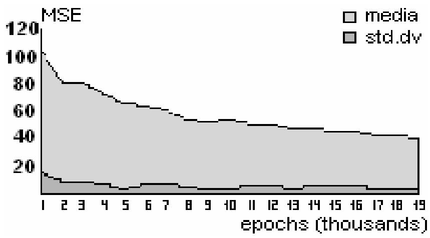


Chart 4. Evolution of media and standard deviation of the MSE.

According to the prediction power of the ANN, it can be seen that the ANN successes in isolating noise with an acceptable relative error level, as this charts show:

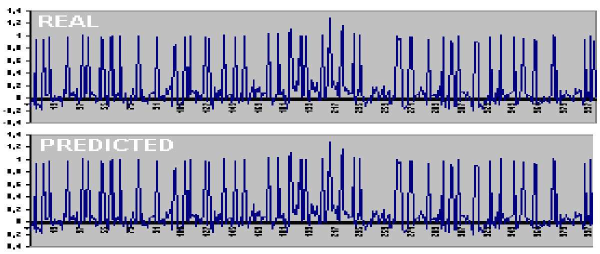


Chart 5. Comparison between real and ANN predicted noise.

As it can be seen, noise function generally has high frequency, so the ANN without ignoring the complete function still has the ability to identify frequency variances and so distinguish noise from trend function. Nevertheless, results must be checked with PSD, a suitable method for assessing *trend removal* methods.
In the following chart, we can see PSD at the end of training process.

Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp248-253)

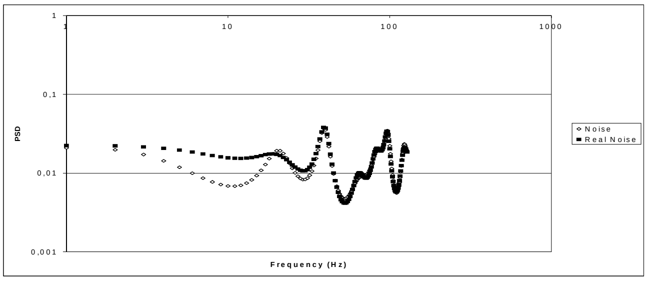


Chart 6 PSD at 30000 iterations.

To prioritize inference instead of memorization, a suitable alternative is to perform earlier stop training. Indeed, at 1000 epochs MSE remains low, with the advantage of improving ANN generalization ability. At this level, the PSD is as it shows next chart:

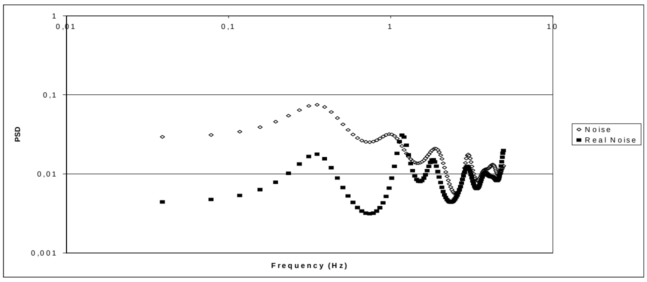


Chart 7 PSD at 1000 iterations.

Analysing these two charts we can see that both are similar in medium potential order. The difference between predicted and real noise appears only in low frequencies while in high frequencies there is no influence of the superimposed frequency. In the beginning of training process, detrending is partial. It also should be seen that the more the ANN is trained the more concordance can be obtained between real and predicted noise spectral. At 30000 epochs differences are about 10 Hz, which is remarkable.

9. Conclusions

As we could see, ANN can be a useful tool for *trend-removal* in electrochemical noise studies. Several ANN topologies can be used, anyway, a 4-layer Perceptron ANN proved to be adequate. As training algorithm *backpropagation*, can be used, as in our case, or even second order algorithms to improve convergence [11]. Splitting data input vectors in three groups depending on the mean derivative also is an important contribution to error minimization. Variable learning rates and momentum accelerates convergence and at the same time helps in avoiding oscillation [5].

Several uses can be proposed for these techniques, such as corrosion levels prediction in steel alloys, or even different atmospheric conditions, thus providing a versatile evaluation tool for industrial viability studies.

References

[1] ALMEIDA, E. et al. *Electrochemical Noise Measurements for Corrosion Applications*, 1996, Ed. ASTM, West Conshohocken, PA, USA, pp 421-424.

[2] CANAVOS, G. *Probabilidad y Estadística, Aplicaciones y Métodos.* México: McGraw-Hill, 1998.

[3] COTTIS, R. A. *Interpretation of Electrochemical Noise Data*, Corrosion, NACE, 2001.

[4] DAWSON, J. L. *Electrochemical Noise Measurements for Corrosion Applications*, ASTM, West Conshohocken, PA, USA, pp 3-35, 1996.

[5] HAYKIN, S. *Neural Networks, a comprehensive foundation*, IEEE Press, U.S.A, 1994.

[6]HETCH-NIELSEN, R. *Neurocomputing*, Massachusetts: Addison-Wesley, 1991.

[7] HILERA, J. *Redes Neuronales.* Madrid, 1998.

[8] FREEMAN J., SKAPURA, M. *Neural Networks* , MIT Press, Massachusetts, 1991.

[9] ISASI, P.; GALVAN, I. *Redes de Neuronas Artificiale.* Madrid: Prentice-Hall, 2004.

[10] KOZA, J et al, *Genetic Generation of both the weights and Architecture for a Neural Network*, 2004.

[11] MANRIQUE, D. *Redes de Neuronas Artificiales y Computación Evolutiva*, Course ORT-UPM, 2005.

[12] MARTINEZ, M. *Metodologías en Redes Neuronales*, COMPUMAT, Cuba, 2003.

[13] MARTINEZ, M.; MARTINEZ V. *Teachers Assessment: A comparative study with Neural Networks*, DELTA, New Zealand, 2003.

[14] MOHAMMAD, H. *Fundamentals of Neural Networks.* Massachusetts: MIT Press, 1998.

[15] OHANIAN, M. et al. *Caracterización de Productos de Corrosión formados en ambiente marino subtropical y marino antártico*, 2004.

[16] OHANIAN, M., PhD. Thesis, *Corrosión Atmos-férica: Caracterización estructural y electroquímica de productos de corrosión formados sobre acero de baja aleación en intemperie marina*, UDELAR, 2004.

[17] OHANIAN, M., MARTINEZ, V. et al, *Highly dispersed electrochemical noise data*, The Journal of Corrosion Science and Engineering, Vol. 7, 2004.

[18]PICTON,P. *Neural Networks.* London,Rowe,2000.

[19] RALSTON, A., RABINOWITZ, P., *First Course in Numerical Analysis.*New York: Dover Publ., 1965.

[20] SACHS, L. *Estadística Aplicada* Springer, 1967.

[21] SALVETTO, P.; MARTINEZ, M. et al. *A Very Early Estimation of Software Development using Neural Networks.* Buenos Aires: CACIC, 2004.

[22] STERGIOU, C, SIGANOS, D. *Neural Networks.* London: Imperial College, 2001.

[23] VENTURA, D. *Using Evolutionary Computation to Generate Training Data Sets for NN.* Utah, 1995.

[24]WIDROW,B.*Sistemasconexionistas*, IEEE Conference, Buenos Aires, 2004.