# ARTIFICIAL NEURAL NETWORK ALGORITHMS FOR ACTIVE NOISE CONTROL APPLICATIONS

**PACS: 43.50 Ki**

FERNÁNDEZ FERNÁNDEZ, ALEJANDRO; COBO PARRA, PEDRO

Instituto de Acústica. CSIC.

C/ Serrano 144

28006. MADRID

SPAIN

Tfn: 34-91-561 88 06

Fax: 34-91-411 76 51. E-Mail: iacf319@ia.cetef.csic.es

## ABSTRACT

This paper shows the use of several methods commonly applied to training Artificial Neural Networks (ANN) in Active Noise Control (ANC) systems. Although ANN are usually focused on off-line training, real-time systems can take advantage of modern microprocessors in order to use these techniques. A theoretical study of which of these methods suit best in CAR systems is presented. The results of several simulations will show their effectiveness.

## INTRODUCTION

The objective of Active Noise Control or ANC systems is to generate an acoustical signal or "anti-noise" that is capable of cancelling the primary noise when both signals interfere. This task is accomplished by the use of electronic controllers which are able of filtering a reference signal (feedforward ANC) or the error signal (feedback ANC) so that their output (secondary signal) cancels the primary noise. This type of controllers usually needs to estimate transfer functions between the different outputs and inputs of the system (figure 1). This estimation is commonly done by the use of a FIR or IIR filter whose weights are adapted in order to minimize an error signal; a noise signal is emitted through the secondary source and measured by the error

microphone. It also feeds the adaptative filter as reference signal. The controller then tries to minimize the sum of the error microphone signal and the output of the filter, normally using an LMS (Least Mean Square) steepest-descent algorithm applied to the instantaneous error.

Although simple and stable, the LMS algorithm and its variations (normalized LMS, leaky LMS, etc.) are slow. The speed of their convergence depends on the step size (μ), as can be seen in the weight actualization equation:

$$W(k+1) = W(k) + m \cdot e(k) \cdot X(k)$$

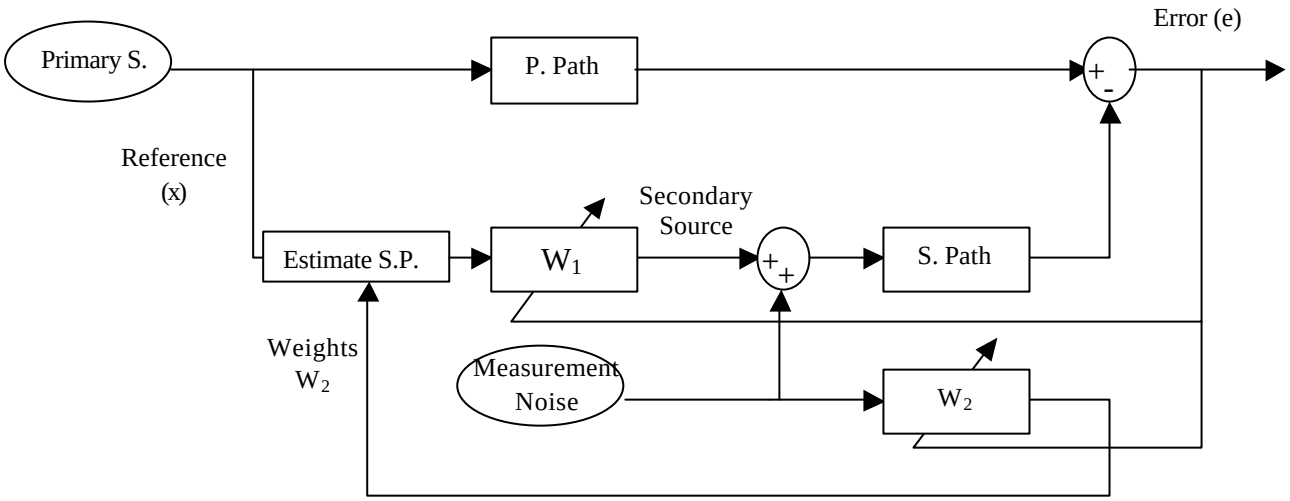


**Figure 1:** FXLMS controller scheme. This controller needs an estimation of the secondary path (transfer function between the secondary source and the error microphone).

The maximum value of the step size is a function of the power of the reference signal X and the number of weights (Widrow and Stearns, 1985):

$$0 < m < \frac{1}{(N+1) \cdot Power(X(n))}$$

where N is the number of weights. This means that the greater the number of weights, the lower the step size and the convergence speed of the filter. This can be a drawback in applications where the distance between the secondary source and the error microphone is large. In this situation the delay of the acoustical signal is significant, which forces to use a huge number of weights (although many of them tend to zero) and thus, a small step size. Another consequence is the fact that no matter how powerful and fast the hardware becomes, the speed of convergence of this type of controllers will remain unchanged.

**NEURAL NETWORKS**

A neural network is a mathematical model of biological neural systems. Its main feature is the ability to adapt or *learn* when the network is trained. The individual blocks which form the neural networks are called *neurons* (figure 2). The neuron consists of a linear combiner followed by a nonlinear function (Haykin, 1996). Each input to the neuron is multiplied by a *synaptic weight* (which is the adjustable parameter) and added afterwards with the other weighted inputs. The output of the linear combiner plus an external bias is then fed to the nonlinear function.

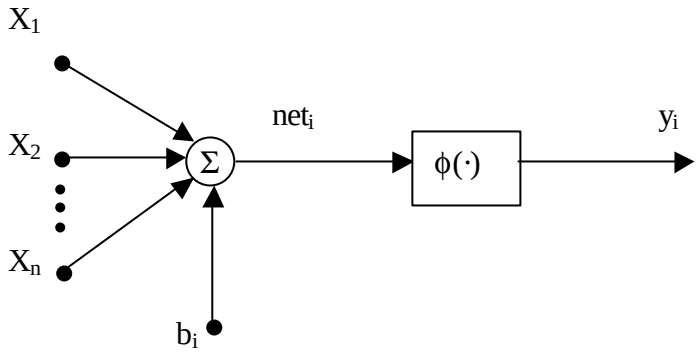


**Figure 2:** Model of a neuron (Haykin, 1996)

The most popular network is the *multilayer perceptron* (MLP). It consists of an *input layer* followed by one or more *hidden layers* whose nodes are neurons like the one described before and an *output layer* that also contains neurons (figure 3).

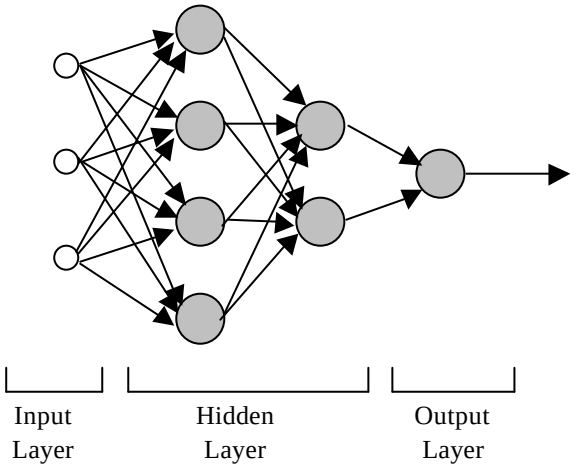


**Figure 3:** Multilayered perceptron with three inputs, two hidden layers and one output.

Using this terminology, an adaptative FIR filter can be described as a neural network whose inputs are delayed samples of the signal, one linear neuron as output and no hidden layer. To implement an IIR filter the delayed samples of the output of the network should also be included in the inputs.

The training of neural networks is carried out in a similar way to what FXLMS algorithm do. The main difference is that instead of adapting the weights each time a new sample arrives, a significant number of samples are used to calculate the mean square error (FXLMS uses the instantaneous square error). The main advantage of using the mean square error is that faster minimum search methods such as quasi-Newton methods can be utilized (figure 4). These

methods can not be implemented with the instantaneous square error because they usually do not converge.

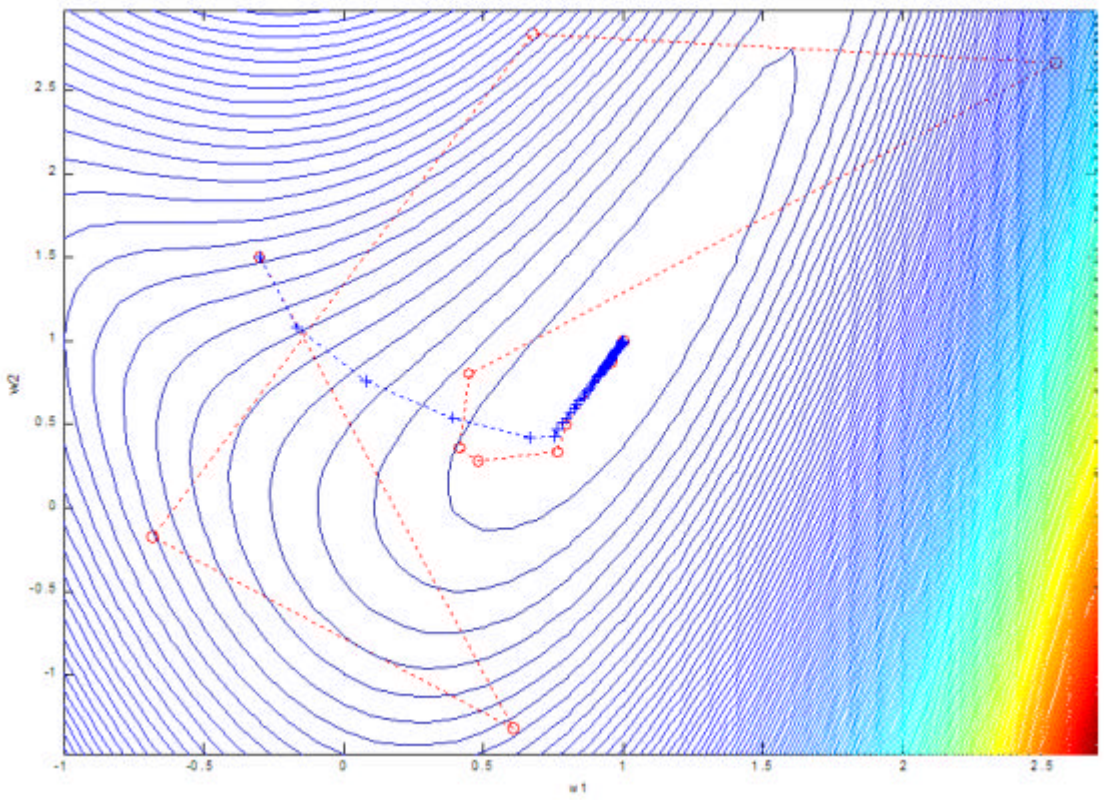


**Figure 4:** Minimization process of Rosenbrock's function using an LMS algorithm (in blue) and a quasi-Newton iBFGS (in red). Although chaotic at first, the iBFGS algorithm reaches the minimum in 14 iterations, while it takes 123 with the LMS algorithm.

## PROPOSED METHOD

To implement this type of training in real-time system for identifying a transfer function, a buffer is used to store several number of samples of the measurement noise of figure 1. Then this data can be used as training data for the iterative optimization process (the training process is done off-line). A compromise must be met between speed and precision; the more data collected, the better the precision but with the cost of having to spend more time acquiring data. The main advantage of this method versus the LMS is the fact that with the same amount of data, this method takes advantage of the fast convergence of quasi-Newton methods and the iterative process. A limitation is that while acquiring the training data, the system is idle, and it has to spend some time afterwards training the network; however, if the microprocessor controlling the process is fast enough this is done in much less time than what it takes the LMS to converge. In this system, the increase in speed of computer hardware produces lower adapting times.

Figure 5 shows a comparison between this method and normalized LMS. The algorithms have to identify a transfer function which is simply a delay of 75 ms, which corresponds to a distance between the secondary source and the error microphone of 25 m. The sample time is 1 ms, so at least 76 weights are needed. As said before, the maximum step size of the LMS algorithm depends on the number of weights; this makes LMS based algorithms slower when there are significant delays because as the iterations are led by the sampling clock (in this case one every

millisecond). On the other hand, the off-line training does not have this problem; if we have to use a small step size each iteration can be done in less than one millisecond if the hardware is powerful enough. As can be seen, the normalized LMS algorithm needs almost 5 times more time than off-line training.

A sensitivity test can also be implemented afterwards the off-line training in order to eliminate non significant weights. This will free some resources of the controller and can eliminate part of the residual signal produced by not making these weights zero when they actually are zero. If the system presents nonlinearities, a multilayer perceptron with nonlinear neurons can be used to implement them.

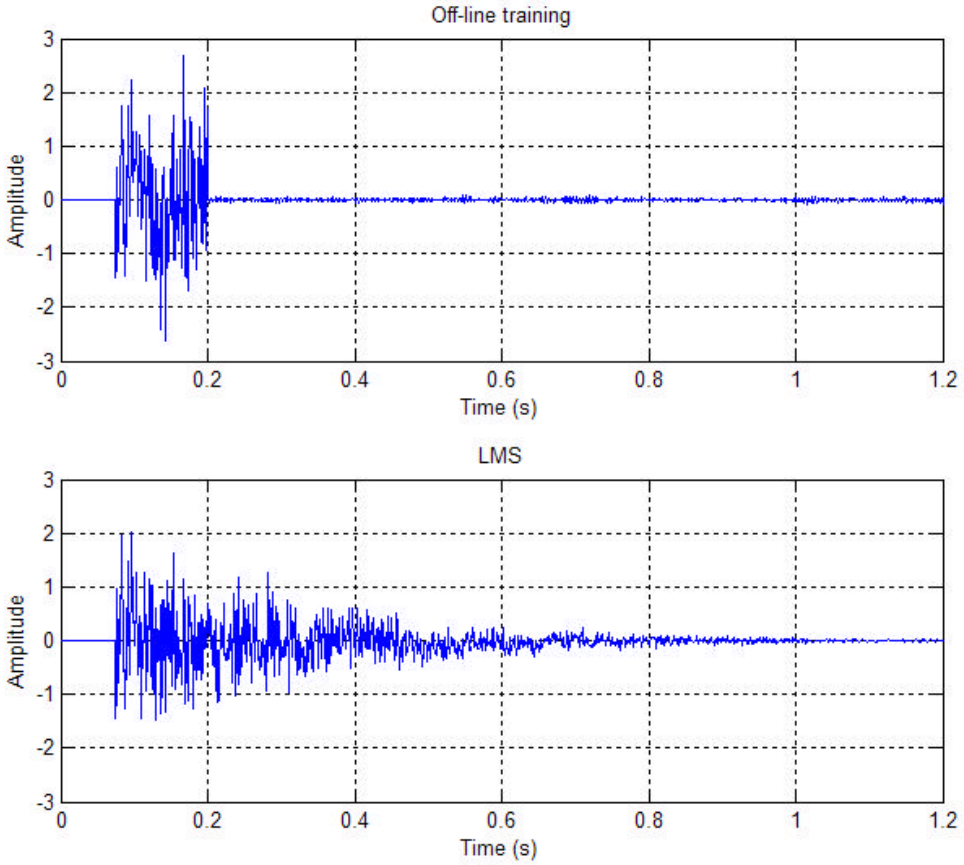


**Figure 5:** Residual signal after applying the proposed method (above) and normalized LMS algorithm (below) in the identification of a transfer function using white noise as input.

## CONCLUSIONS

ANC systems commonly use LMS based algorithms for the filter adaptation process. Although multichannel systems can take advantage of this low-computational-cost algorithm, systems with few channels cannot make use of modern powerful microprocessors because an increase in computational speed does not produce smaller adapting times. On the other hand, nonlinear systems cannot be correctly modelled by FIR or IIR filters with variable weights.

Neural network type training can be used to accelerate convergence times by the use of off-line training combined with fast minimum search algorithms and fast microprocessors. Quasi-Newton algorithms such as the iBFGS, cannot by applied directly to the instantaneous quadratic error as the LMS because in its time varying surface quasi-Newton methods fail to converge. The proposed off-line training method uses the mean square error, which is better suited than the instantaneous square error for the use of quasi-Newton methods. The iterations are no longer lead by the sampling clock which means that a faster microprocessor will calculate them faster, in comparison to FXLMS where a new iteration cannot began until a new sample is acquired. Sensitivity methods can free resources for the controller and reduce the residual signal and multilayer perceptrons can correctly model nonlinear systems. In other words, off-line training can take full advantage of computational power of today's microprocessors for low channel ANC systems, while providing smaller adaptive times and better models of nonlinear systems.

## ACKNOWLEDGMENTS

This work has been supported by the Comisión Interministerial de Ciencia y Tecnología (CICYT) under the project DPI2001-1613-C02-01.

## REFERENCES

**Haykin, S.** 1996. *"Adaptative Filter Theory"*. Prentice-Hall International, INC.,Upper Saddle River, NJ

**Widrow, B. and Stearns, S. D.** 1985. *"Adaptative signal processing"*. Prentice-Hall,Englewood Cliff, NJ