# ADC nonlinearity compensation based on Neural Networks

C. Schmidt J.E. Cousseau O. Agamennoni J.L. Figueroa

Instituto de Investigaciones en Ingeniería Eléctrica "Alfredo Desages"
UNS – CONICET, Avda. Alem 1253, 8000 Bahía Blanca, Argentina
*figueroa@uns.edu.ar*

***Abstract*— In this paper the problem of nonlinear system compensation is presented. In particular our interest is the compensation of high speed analog to digital converters (ADCs). These converters present strong nonlinearities which deteriorate the performance of a digital system in terms of the signal to noise and distortion ratio (SINAD). Two traditional compensation methods for this kind of system are reviewed. A compensation based on Neural Networks (NNs) is proposed and analized.**

***Keywords*— Analog to Digital Converters, Nonlinearities, Models.**

## 1. INTRODUCTION

The actual trend of migration towards digital signal processing systems, even for applications that were traditionally restricted to the analog domain because of their high operation frequencies and large bandwidth, has created a demand for ADCs of very high speed and low distortion.

In general, high speed ADCs present strong nonlinear effects in their transfer function which cause distortion in the discrete output signal [5]. These nonlinearities are originated in different subsystems of the converter, and affect the output determining a joint response [1]. The result of this is the presence of harmonic distortion that degrades the dynamic range over the frequency band of interest in the converter, determining a noise floor (Fig. 1). Hence, compensation of nonlinearities is needed.

Two compensation lines are found in the literature [2-3]. The first one is external calibration, where no detailed information is needed about the physics that rule the nonlinearities present at the output. Their joint behavior is measured and an external circuit is applied to compensate at the required operation point. As a disadvantage of this alternative, the solution obtained is strongly dependent on the operation point in which the converter is used, i.e. the dynamic and frequency range of the input signal. This means that calibration has to be redone when significant changes in the input signal occur, with the computational burden and efficiency loss that it implies. In other words, a periodic training of the calibration circuitry is required.

The second approach consists in modeling each subsystem of the ADC in circuital terms to identify the nonlinear behavior of each element. This allows the design of internal compensations based on the physical laws that rule these phenomena and the interaction between them. The result is a much more accurate compensation, independent of the input signals applied to the converter. Another advantage is that once the design is finished including internal compensations, it can all be integrated in the same chip.

The cost of this solution is a higher complexity in the design and the lack of flexibility. If one wishes to calibrate a different ADC, the whole process of study, modeling and design must be performed again.

The goal of this article is to present the results reached in the compensation of different ADC models [1] using a NN. Neural Networks are universal approximators, and they are known to have good generalization properties [4]. Previous work has been carried out on the subject [6] to compensate a mathematical model with its parameters adjusted by measurements, but no information is provided about the specific structure of the NN.

In Section 2, the most commonly used compensation structures are briefly reviewed. The proposed compensation technique based on NN is described in Section 3, and results based on simulations are presented in Section 4. This article ends in Section 5 with some conclusions.

## 2. BASIC CONCEPTS ON LUTs

Post-compensation by look-up tables (LUTs) is one of most frequently proposed methods for error correction in ADCs [8]. In this section, some basic concepts on LUTs are briefly reviewed, as they will be used in the following sections. The basic idea behind a LUT correction system is to use the output samples of the converter to generate a memory address (or index). This address gives access to a particular table entry where a correction value (that adds to, or replaces, the output sample) has previously been saved (Fig. 2).

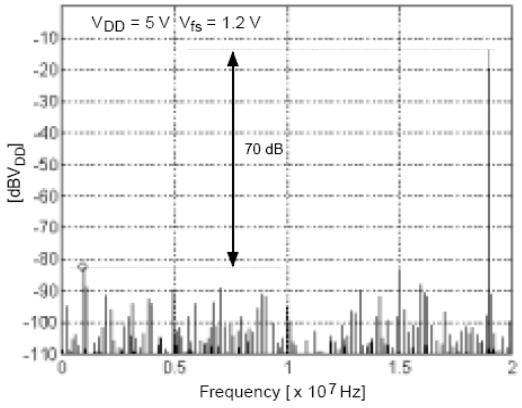


Fig. 1. Spectrum of the output signal of an ADC for a sinusoidal input signal.

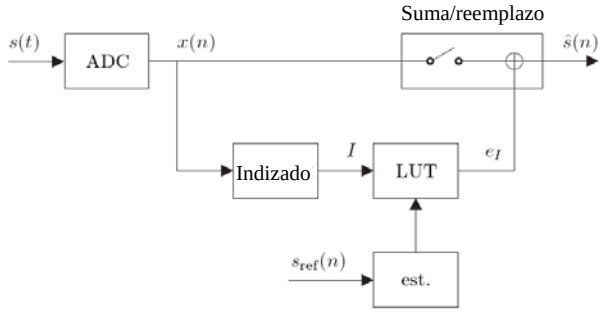


Fig. 2. General scheme of a look-up table correction system.

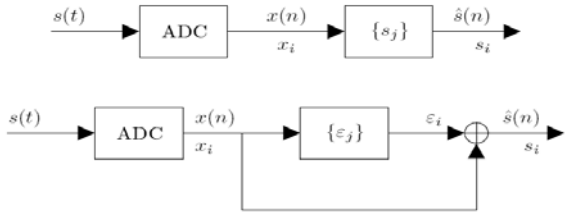


Fig. 3. a) Correction based on replacement. b) Correction by addition of the estimated error.

In this manner, the LUT models the error between the converter under test and an ideal ADC. The elements that compose the look-up table correction system can be divided in:

- **Indexing scheme:** determines the manner in which the table address index *I* is generated from the present output sample of the converter.
- **Correction or replacement:** the table can be used either to save the correction values to be added to the output sample, or to save values that directly replace the output sample from the converter (Fig. 3).
- **Reference signal:** There are different ways to obtain a reference signal. It is used to calibrate the table and therefore its selection is very relevant.
- **Estimation methods:** Strategies on how to obtain the correction values with which the table will be filled from the reference signal.

***Indexing schemes:*** they represent the key part of a LUT system, given the fact that they determine the size and structure of the table that will be implemented. The difference between two indexing schemes is basically how they do the mapping of the ADC samples into the corresponding memory address. Some examples are:

*Static index:* the index *I* is directly the *n* bit digital word corresponding to the present sample provided by the converter. It does not depend on the values from past samples. As an advantage, it requires the minimum amount of memory. However, this scheme does not take into account the dynamics of the sampled signal, and it has been proven that it can enhance the performance of the converter at some frequencies while deteriorating it at others.

*State space indexing:* one way to introduce information about the dynamics of the input signal consists of using both the present and previous sample to generate the address index *I*. Basically, if the converter provides an *n* bit digital word, the present and previous samples are combined and an index of length equal to *2n* bits is obtained.

This method can be generalized using *k* past samples along with the present sample to generate the index *I*, but the memory required to save the table grows exponentially as a function of *k*.

*Phase-plane indexing:* In this case, the address index for the table is generated from a combination of the present sample and an estimate of the slope of the input signal. The estimation of the slope can be obtained as the difference between the present sample and the previous one, using a FIR filter as a derivator, or using an analog derivator sampled by another ADC. This method can also be generalized to use the first *k* derivatives of the input signal.

## 3. COMPENSATION BY NEURAL NETWORKS

As previously mentioned, one of the most well known methods for compensation of errors due to nonlinearities in AD converters is post-correction using LUTs. The method consists on using the present output sample of the ADC as a memory address for a table where the estimated error from the converter has previously been saved. Then, the estimated error is subtracted from each output word from the converter to obtain an output word equal to that of an ideal ADC. However, depending on the dynamics of the system, the ADC may produce different errors for the same output word. Taking this into consideration, the memory address for the table is often obtained by using not only the present output word from the ADC but also some amount of previous samples. In this manner, the dynamics of the input signal is taken into account during the correction process. However, the size of the table grows exponentially as a function of the number of past samples included. In addition, the amount of memory required also grows exponentially as a function of the number of bits of resolution in the converter.

Another disadvantage of this method is that a very intensive experimental work is needed to detect the value of the errors with which the table will be filled.

In order to solve this issue, the use of a smaller table to train a NN is proposed in [6]. The basic idea is to use the trained NN as a substitute for the bigger sized table that would be necessary otherwise. The goal is to train the NN with a series of data consisting in some sets of present and previous output words and their corresponding error, such that given any other combination of present and previous samples, the NN gives a good estimate of the error committed by the ADC. Hence, it is a generalization problem.

The NN that will be used is a single layer perceptron with as many inputs as output words from the ADC are used in the estimation, and a single output representing the error value to be subtracted from the present output word (Fig. 5). Two models developed by the authors in a previous article [1] will be used to test the compensation scheme. The first model is a circuital model of a 4

bit flash ADC and the second is the circuital model of a 7 bit pipelined ADC based on the first model. These models offer a good representation of the static and dynamic nonlinearities of a real ADC [5, 7].

The compensation process can be divided into three stages. The first stage is the calculation of the error committed by the converter compared to an ideal ADC (Fig. 6) for some sets of data and its storage in a small sized LUT. The indexing scheme for this LUT will be of state-space type with a memory of $k$ past samples, according to the memory effect observed in the circuital ADC model used [1]. However, the LUT will not be completely filled but only some random values will be used, and the rest of the table values will be predicted by the NN. Thus, a small sized LUT can be used to train a NN that will be equivalent to a much bigger table. This procedure is illustrated in Figure 4.

Once the table has been filled, the error data set is divided in two parts, one for training the NN and the other to perform a validation of the results. The NN used is a single layer perceptron with $N$ neurons, $k$+1 inputs, and a single output. The inputs are the present and $k$ past output words from the ADC under test, and the desired output is the measured error that occurs for that combination of inputs (Fig. 5).

Second stage is then to train the NN that will replace the table in the compensation scheme. Given the data set of $k$+1 inputs and the corresponding desired output, the training of the NN is performed using a back-propagation algorithm [4], where the weighs and bias are computed to minimize the error gradient between the response of the net and the desired output (Fig. 6).

Third stage is the validation phase of the NN as compensator. In order to perform the validation, a data set different to the one used for training is considered, and the performance of the compensation is evaluated. This is achieved by comparing the output of the compensated system with that of an ideal ADC (Fig. 7). Ideally, there should be no difference between them.

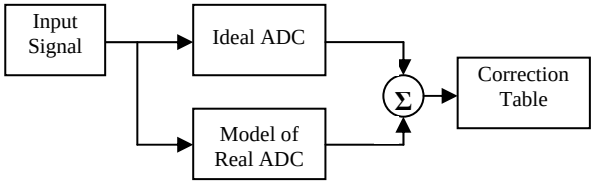


Fig. 4. Modeling process of the error signal between an ideal ADC and the converter to be calibrated.

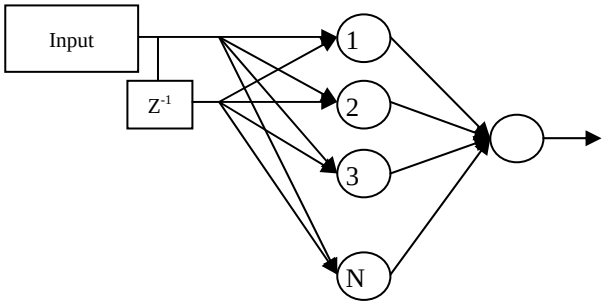


Fig. 5. Topology of the NN used.

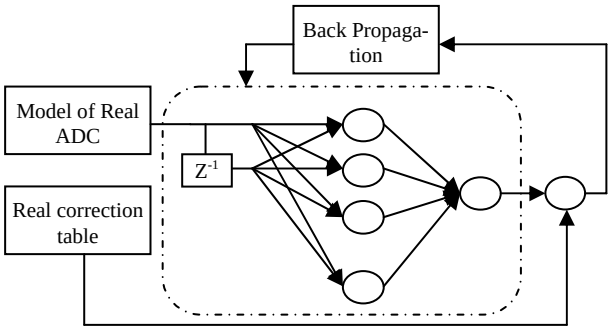


Fig. 6. Training of the Neural Network.

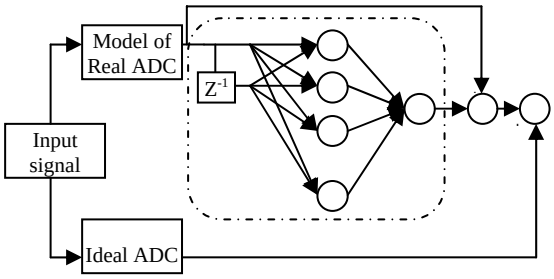


Fig. 7. Validation of the Neural Network.

## 4. SIMULATIONS AND RESULTS

The ADC models for simulation described in [1] were used in order to analyze and test the proposed ADC compensation scheme. The first model is the circuital model of a 4 bit flash ADC. The input-output data from this converter are obtained by circuital simulations in Spice. Based on this model, it is also obtained the circuital model of a 7 bit two stage pipelined ADC [1, 9].

An ideal ADC was simulated in MATLAB to allow for comparison of both models with an ideal converter and estimate the resulting error for each of them.

First, the amplitude quantization of a sinusoidal signal was simulated using the circuital, mathematical and ideal ADC models. Then, the time quantization of the signals was performed using a clock determining a sampling frequency 5 times higher than the frequency of the signal. In this manner, a data set of 1650 input-output values was for each of the three mentioned ADC models.

Next, given the same input signal, an error vector was calculated as the difference in the output data between the ideal and the circuital models. This error signal was then saved in a table for later usage. The circuital model was chosen to perform the compensation because of the more severe distortion that it presents.

This error vector represents the desired output needed for training the NN, and once it has been obtained, the first phase of the compensation process finishes as described in the previous section. This process was repeated for the circuital model of a 7 bit pipelined ADC, with an effective sampling frequency 20 times higher than the frequency of the input signal.

From these data series, two third parts were used to train the neural network and the rest for validation. For the

training phase, a data series of present and past output samples from the circuital models were used as the input to the NN and the error vector was used as the desired output.

### *3.1.1. Circuital model 1 – 4 bit flash ADC*

The NN used is a single layer perceptron, with sigmoidal activation functions for the neurons of the hidden layer and a linear activation function for the output neuron.

Several NN of this type were simulated and the Mean Squared Error (MSE) in training and validation was computed as a function of the number of neurons in the hidden layer, for the case of one single past sample used to model the memory of the system. The plot is shown in Fig. 8.

It can be seen from the figure that the MSE both in training and validation decreases as a function of the number of neurons in the first layer until 8 neurons are used, and then it remains constant. However, the MSE is not as small as expected.

In order to further reduce the error, a longer memory effect was included. Thus, the process was repeated taking into account a memory of 15 past samples as input to the NN. The results are shown in Fig. 9. In this case, an important reduction in the MSE is observed, which means the memory of the system is longer than one. In addition, the decreasing tendency of the error as a function of the number of neurons continues beyond 20 neurons.

Then, several NNs with 25 neurons in the hidden layer were simulated varying the number of past samples used as inputs to the net, in order to determine the memory of the system. The MSE in training and validation was computed again under these conditions (Fig. 10). There is an evident reduction of the error as a function of the number of past samples included as inputs. It can also be seen that this reduction is significant until the memory is augmented to 25 samples, where the minimum error in validation is reached. The MSE in training continues to drop as an over-adjustment occurs

From the previous analysis we conclude that the best perceptron type NN to compensate the circuital model of the 4 bit flash ADC must have 25 neurons in the hidden layer and use a 25 tap delay line at the input to model the memory of the system.

Finally, a simulation of the compensated system was carried out and the error obtained was analyzed compared to the error of the system without compensation.

For this purpose, a data set not included in training was used. Fig. 11 shows the error for both cases normalized to a Least Significant Bit (LSB). It should be noticed that only an error larger than ± 0.5 LSB will produce an error on the output digital word from the converter. Hence, the compensated system will produce less error.

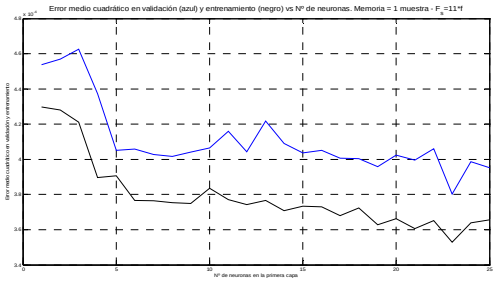


Fig. 8. MSE in validation (blue) and MSE in training (black) vs. Number of neurons for a memory of one sample.

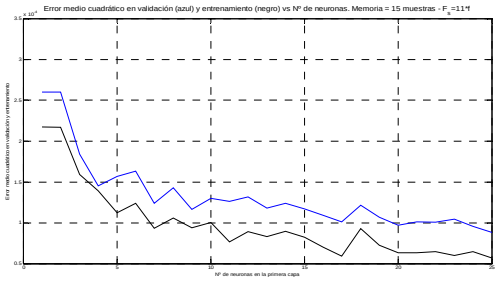


Fig. 9. MSE in validation (blue) and MSE in training (black) vs. Number of neurons for a memory of 15 samples.

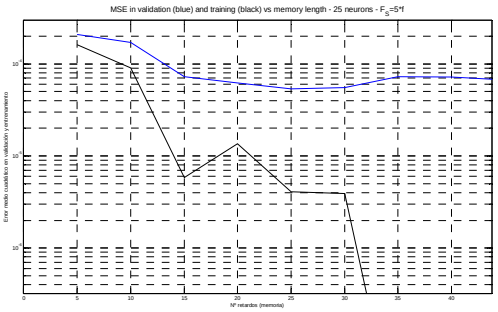


Fig. 10. MSE in validation (blue) and MSE in training (black) vs. memory length for a NN with 25 neurons.

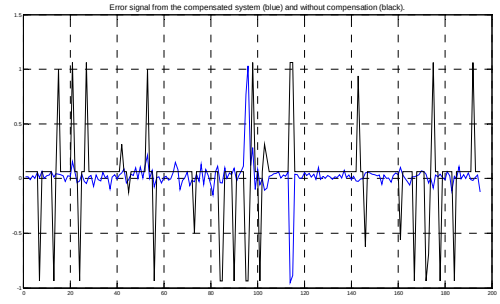


Fig. 11. Error signal form the compensated system (blue) and without compensation (black).

On the other hand, the memory needed in this compensation scheme is less than 2KB. Considering that $k = 25$ past samples were used to model the dynamic behavior of the system, and that the memory required in a traditional LUT compensation scheme grows exponentially with $k$, an equivalent LUT system would require a memory of $2^{25.4} = 2^{100} = 1.26 \times 10^{30}$ bytes. This means that the implementation of an equivalent solution using a plain LUT for correction is not feasible.

### *3.1.2. Circuital model 2 – 7 bit pipelined ADC*

In the same manner as in the previous case, several single layer perceptron type NNs were simulated. Then, the MSE in training and validation was computed as a function of the number of neurons used for a memory length of 10 samples. The plot is shown in Fig. 12.

As can be seen from the figure, in this case the number of neurons needed is 15, as no reduction in the MSE is obtained for a larger amount of neurons. Again, the process of simulation was repeated for several NN of fixed number of neurons, but varying the amount of past samples used to model the memory of the system. It can be seen that the MMSE in validation is obtained for a memory of 10 samples, as shown in Fig. 13.

Figure 14 shows the real error committed by the pipelined ADC and the prediction given by the neural network, both obtained for a data set different from the one used for training of the NN. The parameters of the neural network used were 15 neurons in the hidden layer and 10 tap delay line to represent the memory of the system.

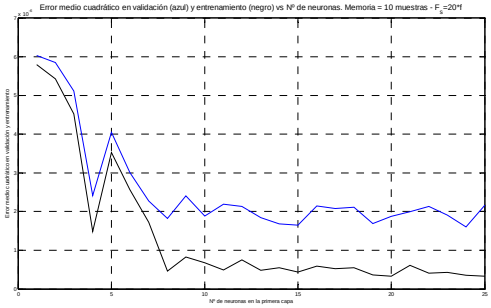


Fig. 12. MSE in validation (blue) and MSE in training (black) vs. Number of neurons for a memory of 10 samples.

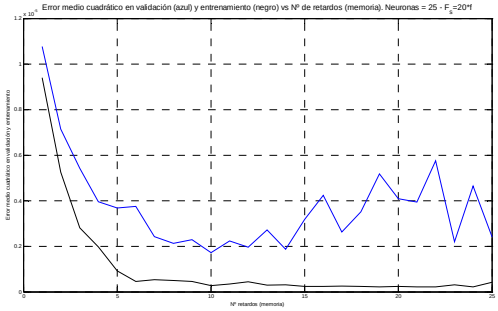


Fig. 13. MSE in validation (blue) and MSE in training (black) vs. memory length for a NN with 15 neurons.

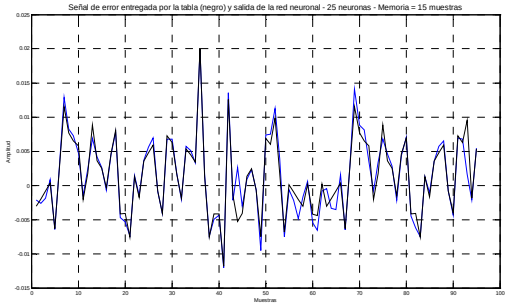


Fig. 14. Error signal form the compensated system (black) and prediction from the NN (blue)

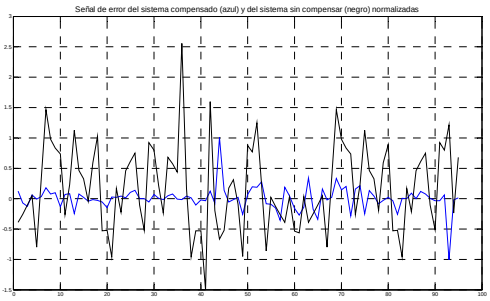


Fig. 15. Error signal form the compensated system (blue) and without compensation (black).

Finally, in Fig. 15 the error signal from the 7 bit two stages pipelined ADC is shown, before and after compensation is applied.

It is important to remark that the number of neurons in the hidden layer and the number of past samples used to model the memory of the system are determined by trial and error. These parameters should be analyzed for each different ADC under consideration.

## 5. CONCLUSIONS.

In this work, ADC compensation techniques are described and the results obtained for a compensation based on artificial neural networks are detailed, both for the case of a 4 bit flash ADC and a 7 bit two stage pipelined ADC. These models are extracted from previous work of the authors. It is shown that this compensation scheme leads to good results in the prediction and cancellation of errors with a computational cost that is not excessively high. In addition, the optimal parameters for the NN used can be uniquely determined for each model, leading to the smallest error possible.

## REFERENCES


[1] C. Schmidt, J.E. Cousseau and J.L. Figueroa, “Modelos de Conversores Analógicos Digitales,” enviado al XIII RPIC, Rosario, 2009.

[2] Y. Oh and B. Nurmann, "System Embedded ADC Calibration for OFDM Receivers," *IEEE Transactions on Circuits and Systems*, Vol. 53 (8), pp. 1693-1703, 2006.

[3] H.S. Lee, D.A. Hodges and P.R. Gray, "A self Calibrating 12b 12us CMOS ADC," *Proc. ISSCC Digital Tech. Papers*, pp. 64-65, Feb. 1984.

[4] “Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models”, O. Nelles, Springer – Verlag, 2001.

[5] R. Van de Plassche, *CMOS Integrated Analog to Digital and Digital to Analog Converters*, Kluwer Academic Publishers, 2003.

[6] A. Baccigalupi, A. Bernieri and C. Ligouri, "Error-Compensation of AD Converters Using Neural Networks,” *IEEE Transactions on instrumentation and Measurements*, Vol. 45 (2), pp. 640-644, 1996.

[7] N. Björsell, “Modeling Analog to Digital Converters at Radio Frequency,” Doctoral Thesis in Telecommunications, Stockholm, Sweden 2007

[8] “Characterization and Correction of Analog-to-Digital Converters”, H. F. Lundin, Doctoral Thesis in Signal Processing, Stockholm, Sweden 2005.

[9] C. Hammerschmied, “Pipelined A/D Converters for Telecommunication Applications,” *Workshop on A/D Converters for Telecommunication*, Integrated Systems Laboratory, ETH Zürich, October 2001.