Reducing Mismatches in the Analog Signal by Using Levenberg-Marquardt Back Propagation Algorithm

Bhuvana B.P. Arulselvi
Department of ECE, Bharath University, Chennai 73, India

Abstract: Analog devices are compact and use less power and area when compared with the digital circuits. A novel approach to minimize error in Analog VLSI Signals. The accuracy of analog Signals is reduced by the limitation of analog hardware. The accuracy and efficiency of analog signals can be increased by training the signals till we achieve our desired efficiency. Hence, simple techniques are used to increase the accuracy and performance of analog signals by the neural network.

Key words: Analog VLSI - Neural network

INTRODUCTION

As of today we are in the advanced hi-tech world that requires system without any fault in it. The output of the system is expected to be very accurate without any error in it. Modern electronics uses artificial neural networks and fuzzy logics to optimize the performance of electronic devices. Performance of analog circuits can be optimized by the neural networks. These analog circuits consume very less power when compared with the digital circuits. Implementation of advanced algorithms related to very large scale integrated (VLSI) circuits solves the area and power constraints. These are specially used in advanced portable systems [1].

In my paper I used Levenberg–Marquardt Backpropagation algorithm to train the analog signals. Nonlinear Auto Regressive with external input(NARX) is chosen for training. Voice samples have been collected for training purpose. I have trained a particular analog signal and the error is found to be 2.4117xe-5. When a different analog signal is applied to the network the mean square error is high to a rate of 8.2310xe-5. Various analog signals is given to the network and all are found to have mean square error greater than 7.2311xe-5. By using Levenberg–Marquardt Backpropagation algorithm the network is trained. As the number of epoch increases the error starts to decrease. The number of hidden layers is chosen as 10 hidden layers and the delay as 60. As we are increasing the delay the time taken for training increase but the error keeps on reducing as the number of the iteration increases [2].

Neural network time series prediction method is used for training the network. Levenberg–Marquardt algorithm (LMA) is also known as the damped least-squares (DLS) method. It provides a numerical solution to the problem of minimizing a function, generally nonlinear, over a space of parameters of the function. These minimization problems arise especially in least squares curve fitting and nonlinear programming. The LMA interpolates between the Gauss–Newton algorithm (GNA) and the method of gradient descent. The LMA is more robust than the GNA, which means that in many cases it finds a solution even if it starts very far off the final minimum. For well-behaved functions and reasonable starting parameters, the LMA tends to be a bit slower than the GNA. LMA can also be viewed as Gauss–Newton using a trust region approach [3-5].

The LMA is a very popular curve-fitting algorithm used in many software applications for solving generic curve-fitting problems. However, the LMA finds only a local minimum, not a global minimum. It is often the fastest backpropagation algorithm in the toolbox and is highly recommended as a first-choice supervised algorithm, although it does require more memory than other algorithms [6-8].

The input data is classified into three types before training. They are Training data, validation data and testing data. Training data is repeatedly used to estimate
the weights of design. Validation data is used repeatedly used to estimate the non-training performance error. Also, it is used to stop training once the non-training validation error estimate stops decreasing. Test data is used once and only once on the best design to obtain an unbiased estimate for the predicted error of unseen non-training data [9].

The standard NARX network is a two-layer feed forward network, with a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. This network also uses tapped delay lines to store previous values of the input and output sequences. The output of the NARX network is fed back to the input of the network. This has two advantages. The first is that the input to the feed forward network is more accurate. The second is that the resulting network has a purely feed forward architecture and therefore a more efficient algorithm can be used for training [10].

Image and sound are analog signals that can be processed by the biological system. Artificial Neural Networks is used to depict the character of biological neurons. Cyril Prasanna Raj P proposed a neural network architecture with new technology for weight storage and weights are updated so as to reduce the error. This error is the difference between the output and the external signal processing applications [6]. Mauro Tucci and Marco Raugi presented a new approach for self-organising maps. Here a neuron act as finite impulse response (FIR) and during sequential learning process the co-efficient of filters are updated [7].

Analog device have device mismatch, charge leakage and nonlinear transfer function. Gonzala Carrajal presented an analysis of these effects on the residual error by LMS algorithm [8]. Miguel Figueroa described adaptive signal processing in mixed signal VLSI based on anti hebbian learning.

Neural network is used for face recognition based applications. This system usually uses a dimensionality-reduction network. The co-efficient can be learned or programmed to perform principal component analysis (PCA) and linear discriminant analysis (LDA) for dimensionality reduction [10]. Yuzo Hirai and Kuninori Nishizawa fabricated principal component analysis (PCA) learning network by asynchronous PDM in a FPGA circuits. Differential equations and the circuits are used to express the generalized hebbian algorithm [11].

Chris Diorio, Paul Hasler developed a complimentary pair of four terminal silicon synapses for analog learning applications. A non-volatile memory is used. Electron tunneling and hot electron injection allow bidirectional memory updates [12]. Chun-Hsien Chen developed neural network architecture for syntax analysis. Artificial Neural Networks can be used for symbol processing applications. A new approach is proposed for lexical analysis, stack and parse tree construction [13].

Digital perturbative learning in analog VLSI neural network was proposed by Vincent F Koosh. He used two feed forward neural networks. First uses analog synapses and second uses a computer for controlling global operations [14]. Momchil Milew proposed an analog implementation of artificial neural network with quadratic non-linearity of synapses [15].

We have selected a specific application to apply neural networks in analog circuits. A pattern was assumed by Widrow and Hoff. It is an adaptive algorithm. Filter co-efficient are adjusted by LMS for minimizing the cost function. Recursive least square algorithm uses matrix operations, hence operations. But LMS doesn’t use such operations; hence it uses less time and resources. LMS algorithm is simple and less complicated. It also uses gradient based steepest decent method. It does not require correlation function calculation and matrix inversions. For understanding the concepts of LMS algorithm. Let us consider a linear Perceptron. Here the architecture with new technology for weight storage and weights are updated so as to reduce the error. This error is the difference between the output and the external reference [16].

**MATERIALS AND METHODS**

**Levenberg–Marquardt Backpropagation Algorithm:**
Backpropagation is also called as the generalized delta rule. This is a gradient descent approach to minimize the total squared error of the output which is computed by the network. Process of training in backpropagation involves three steps. They are

- Feed forward of the input vector
- Calculation and backpropagation of the error
- Adjustment of the weight

Many changes have been made to the algorithm to increase the speed. One hidden layer is enough but in case if problem is very critical, then more than one hidden layer is needed.

As discussed earlier, backpropagation training involves three major steps. They are Feed forward of the input vector, Calculation and backpropagation of the error, adjustment of the weight. During the feed forward operation, the input vectors receive the input signal and broadcasts it to the hidden layers B1...Bp. This is followed by the computation of hidden layer activation function and it broadcasts it to the output layer [10].
Each output layer then computes the activation function which will be the output of the network for the given input pattern. Now the output vector compares \( c_k \) with the target value \( t_k \) for finding the error for the corresponding input pattern. The factor \( \delta_k \) should be calculated so that the error can be distributed at the output vector \( C_k \) back to all vectors in the hidden layer [17-19]. This is used to update the weight between the output and the hidden layer.

Similarly, the factor \( \delta_i \) should be calculated so that the error can be distributed at the hidden vector \( B_i \) back to all vectors in the input layer. This is used to update the weight between the hidden and the input layer. Once all \( \delta \) values are determined then the weight for all values are adjusted [20]. The weight \( w_{jk} \) is adjusted and the adjustment is based on the factor \( \delta_k \) and activation function \( a_k \) of the hidden unit \( B_k \). Similarly, The weight \( v_{ij} \) is adjusted and the adjustment is based on the factor \( \delta_i \) and activation function \( a_i \) of the input unit \( A_i \).

**Activation Function:** Some of the features of the activation function which is used in the back propagation are as follows. It should be

- Continuous
- Differentiable
- Monotonically non-decreasing

For easy computation, the derivative used should be easy to compute. The value of these derivatives can be expressed in terms of the value of the function. One of the typical activation function which suits all requirements is the binary sigmoid function which has a range of \((0,1)\) and it can be defined as:

\[
f_j(a) = \frac{1}{1 + \exp(-a)}
\]

With,

\[
f(xx)
\]

Another activation function is bipolar sigmoid function, which has a range of \((-1,1)\) and is defined as:

\[
f_2(a) = \frac{2}{1 + \exp(-a)} - 1
\]

with \( f_2(a) = \frac{1}{2}[1 + f_2(a)][1 - f_2(a)] \)

Bipolar function is closely related to the function

\[
tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}
\]

This function is illustrated in figure

\[
\text{f(x)}
\]

**Random Initialization:** The choice of initial weights shows whether the neural net reaches a minimum error and if it reaches minimum error then how fast it converges. Weight update between any two units depend on both the derivative of the upper unit activation function and the lower unit activation function.

Hence, it should be noted that the initial weights which makes activations or the derivative of activation function zero should be avoided. Initial weight values should not be too large also. A common method to initialize the weight is to assign random values between -0.5 and 0.5. These values can be either positive or negative [21].

**Training of Neural Net:** Aim of the backpropagation net is to achieve a balance between the correct response to training pattern and the good response to the new input pattern and also it is not mandatory to continue the process of training till the mean square error reaches the minimum value.

Hecht-Nielsen suggested two sets of data during the training process. They are as follows, a set of training patterns and a set of training –testing patterns. These are disjoint sets. Based on the training patterns weights are adjusted. But the error is calculated by the training–testing data. Training continues as long as the error in training-testing decreases [21]. When the error is increased, net memorizes the training patterns and training is terminated at this point.

**Number of Training Pairs:** Consider \( P \) as the number of training patterns available, \( W \) as the number of weights to be trained and \( e \) is the expected accuracy of classification. Some of the questions which are asked regarding the number of training pairs include “how can a net classify the percentage of testing and training patterns correctly?”
if there is enough testing pattern available, then the net will generalise properly. Training patterns are determined by:

\[ \frac{W}{P} = e \]

**Data Representation:** In some cases, input and output vectors same range values in some components. This is because of the fact that a single factor in the weight correction expression is the activation of the lower unit whose activation function is zero. From this it is clear that learning may be improved if the inputs can be represented in the bipolar form and bipolar sigmoid function is used [10].

In the applications of the neural network, data may be represented in the form of continuous valued variable or a set or ranges. Consider an example the temperature of a food can be represented by the actual temperature or by the ranges of temperature like: hot, room temperature, chilled and frozen. This can be represented in neural network with the former case stating a single neuron could be used in general and in later case four neurons each with bipolar values could be appropriate [22].

**Number of Hidden Layers:** When a neural network has more than one hidden layer, only small modifications has to be done in the value of \( \delta \). Calculation of the value of \( \delta \) is repeated for each extra hidden layers and summing over the \( \delta \)'s for the units in the previous layer that feed into the current layer for which \( \delta \) can be calculated. In the algorithm described earlier, step 4 can be repeated for every hidden layer in the feed forward stage and step 7 can be repeated for every hidden layer in the backpropagation stage.

**Neural Network Implementation:** Artificial Neural Network can be built using analog, digital or hybrid electronic hardware. Neurocomputing algorithms can be implemented in dedicated or general purpose. One of the commercial neurocomputer involves simulation of error back-propagation. In conventional programmable host computer, neurons are trained. The digital circuitry performs subsequent transfer and storage of weights which result from the training.

However, through analog computation information stored in the network is recalled within the electronic circuitry. Fast recall is performed by the network based on some features. Those features are the analog computation mode and parallel form of information processing. Numerical simulation is done in off chip in the neurocomputer. Digital transfer and data storage circuitry and a dedicated analog neural network perform a recall.

The most commonly encountered operations are scalar and outer product vector multiplication and matrix vector multiplication. These operations are performed are as a series of multiply and add operations. Generation of the non-linear activation function involves additions and subtractions of matrices and vectors and ordinary multiplications and they, are less frequent, but are also indispensible for most learning and recall task. . Efficiency of training can be improved by the following reasons [11].

- Programmable computers have dedicated hardware architectures.
- Parallel architectures of densely connected neural nodes contains multiply and multiply-add processors.
- By using generalization property of networks, training data volume data is reduced.
- Design of Neurocomputing IC is done by analog, digital or digital/analog arrays.

Sequence of steps are illustrated in Fig. 4 showing the problem-algorithm-model flowchart. Alternatively, the users problem can be transferred directly to the artificial neural system model for execution but sometimes artificial neural system model is done through modelling of the algorithm after neural processing of algorithm. Solution is obtained by processing the artificial neural system model [12].

![Fig 1: Flow chart of processing Steps](image-url)
RESULT

Matlab 7.12.0 have been used to realize the characteristic waveform of analog signal. The neural network structure with 60 delay elements and 10 hidden layers is shown below.

Fig. 2:

The neural network structure with 59 delay elements and 10 hidden layers is shown below, this is one step ahead view of neural network

Fig. 3:

Performance Validation Plot:

Fig. 4

Error Histogram Plot:
Fig. 5

Correlation Between The Input And Error Waveform:

Fig. 6

Autocorrelation Plot:

Fig. 7
The work done in this paper evaluates the impact of errors in analog signals and it shows the method of error rectification by neural networks. In particular, we reduce the error by using Levenberg–Marquardt Backpropagation algorithm.

Error reduction is necessary in all electronic devices as error causes the efficiency of the circuits to diminish. Here, for an analog signal the mean square error is reduced by training the signals by using Levenberg–Marquardt Backpropagation algorithm and
increasing delays in neural network time series tool and the corresponding performance plot, error histogram plot, auto correlation and output performance are shown in the results.

REFERENCES


