ON THE INFLUENCE OF THE NUMBER OF LAYERS ON THE PERFORMANCE AND CONVERGENCE BEHAVIOR OF THE BACK PROPAGATION ALGORITHM

Mohamed IBNKAHLA

National Polytechnics Institute of Toulouse
ENSEEIHT, 2, Rue Camichel, 31071 Toulouse Cedex, France
E-mail: ibnkahla@len7.enseeiht.fr

Abstract

In many neural network applications to signal processing, the back propagation (BP) algorithm is used for the training process [4, 6]. Recently, several authors (e.g. [1, 2, 5]) have analyzed the behavior of the BP algorithm and studied its properties.

The influence of the number of layers on the performance and convergence behavior of the BP algorithm remains, however, not well known. The paper tries to investigate this problem by studying a simplified multilayer neural network used for adaptive filtering. The analysis is based upon the derivation of recursions for the mean weight update which can be used to predict the weights and mean squared error over time. The paper shows also the effects of the algorithm step size and the initial weight values upon the algorithm behavior. Computer simulations display good agreement between the actual behavior and the predictions of the theoretical model. The properties of the BP algorithm are illustrated through several simulation examples and compared to the classical LMS algorithm [3].

1. INTRODUCTION

The neural structure studied in this paper is presented in figure 1. It is composed of L layers, each layer has a single scalar neuron.

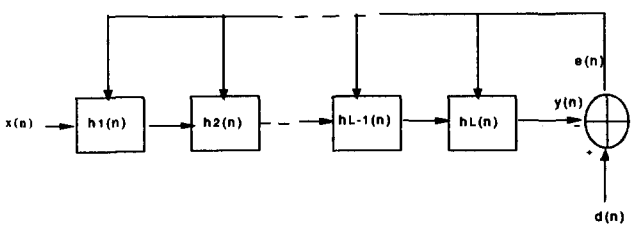

Figure 1: An order 1 multi-layer adaptive filter.

The network input is a scalar signal $x(n)$, its output is then:

$$y(n) = \prod_{k=1}^{L} h_k(n)\, x(n),$$

which is equivalent to an order 1 filtering (multiplication by a scalar $h(n)$ with $h(n) = \prod_{k=1}^{L} h_k(n)$).

The weights are updated using the BP algorithm, which minimizes the error $e(n)$ between the reference $d(n)$ and the output $y(n)$, as:

$$h_k(n+1) = h_k(n) - \mu \frac{\partial e^2(n)}{\partial h_k(n)}, \quad k = 1, .., L.$$

Which can be written as:

$$h_k(n+1) = h_k(n) + 2\mu e(n) \left(\prod_{j=1, j\neq k}^{L} h_j(n) \right) x(n)$$

In order to simplify the statistical analysis of the algorithm, we suppose that all the neurons are initialized with the same positive value at time $n = 0$: $h_1(0) = ... = h_L(0) > 0$. We suppose also that h_{opt} is positive. It can be easily seen that: $h_i(n) = h_j(n)$ for all i and j. The updating equations become then:

$$h_k(n+1) = h_k(n) \left(1 + 2\mu e(n)\, x(n)\, h_k^{L-2}(n)\right).$$

The global coefficient h is then updated as:

$$h(n+1) = h(n) \left(1 + 2\mu e(n)\, x(n)\, h^{\frac{L-2}{L}}(n)\right)^{L}.$$

2. MEAN WEIGHT BEHAVIOR

2.1. THEORY

We define a logarithmic weight error (between the current filter coefficient $h(n)$ and the optimal one h_{opt} [3])

as: $\mathbf{v}(n) = \ln(h(n)/h_{opt})$. $v(n)$ has then the following recursion:

$$v(n+1) = v(n) + L\ln\left(1 + 2\mu e(n)\, x(n)\, h^{\frac{L-2}{L}}(n)\right).$$

Under some assumptions (which will be detailed in the final paper), we have:

$$v(n+1) = v(n) + 2\mu L(e_{opt}(n) + h_{opt}(1 - \exp(v(n)))\, x(n))x(n)\, h_{opt}^{\frac{L-2}{L}} e^{\frac{L-2}{L}v(n)}$$

Let $m(n) = E(v(n))$, it can be demonstrated that :

$$m(n+1) = m(n) + \beta\exp(\alpha m(n))\{1 - \exp(m(n))\} \quad (1)$$

where $\alpha = 1 - \frac{2}{L}$ et $\beta = 2\mu L h_{opt}^{1+\alpha} r_X$.

In [5] we have proposed a second order approximation of equation 1 (on the variable $m(n)$). The mean weight evolution have been modelled by the following 2nd order recursion: $m(n+1) = am(n) + bm^2(n)$, where $a = 1 - 2\mu L h_{opt}^{1+\alpha} r_X$, and $b = 2\mu(4 - 3L)\, h_{opt}^{1+\alpha} r_X$. This model is simple and gives a direct and explicit interpretation of the algorithm behavior. Experimentally, it does not present a significant difference with equation 1. The non linear term $bm^2(n)$ controls the weights evolution at the beginning of the learning process (if the initialization is far from the optimum, i.e. $|m(0)| \ll m^2(0)$). Because of this quadratic term, we can not deal with a time constant for $L > 1$. This is the main difference with LMS. The linear term, $am(n)$, governs the evolution of the weights at the end of the learning process (or during the whole process if it is initialized near the optimum).

In this paper we use the analytical model given by equation 1. The evolution of $m(n)$ is then governed by a non linear equation: $m(n+1) = f(m(n))$, where $f(x) = x + \beta\exp(\alpha x)\{1 - \exp(x)\}$.

2.2. SIMULATIONS

In the simulations below we apply the network to a system identification problem. The input signal $x(n)$ is a white gaussian noise ($\sigma_x^2 = 1$). The reference is: $d(n) = h_d x(n) + b(n)$, where b is a gaussian noise ($\sigma_b^2 = 0.04$). The adaptive filter is used in order to identify the unknown scalar filter h_d (note that $h_{opt} = h_d$).

In the following, we compare the theoretical model for the evolution of $m(n)$ (equation 1) and Monte Carlo simulations for different values of L, μ, h_{opt}, and $h(0)$.

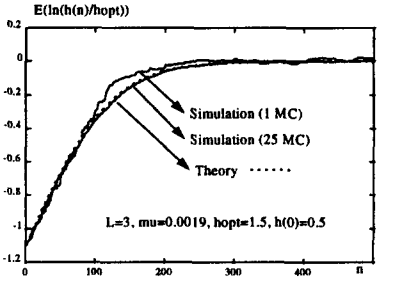

Figure 2: Evolution of $E(v(n))$, $L = 3$.

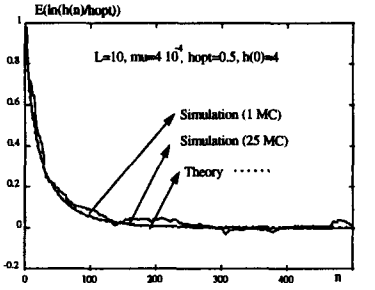

Figure 3: Evolution of $E(v(n))$, $L = 10$.

3. STABILITY CONDITIONS

The stability conditions can be determined by studying lipschitz conditions of the function [7]:

$$f(x) = x + \beta\exp(\alpha x)(1 - \exp(x)).$$

It was demonstrated [7] that a necessary convergence condition (regards μ) is:

$$0 \le \mu \le \mu_{max} = \frac{1}{Lh_{opt}^{2-\frac{2}{L}} r_X} \quad (2)$$

The study of f determines explicitly the influence of the initial weight values on the algorithm behavior. Figure 4 presents the functions $f_L(x)$, the parameters μ_L were chosen such that the steady state MSE are the same for all layers.

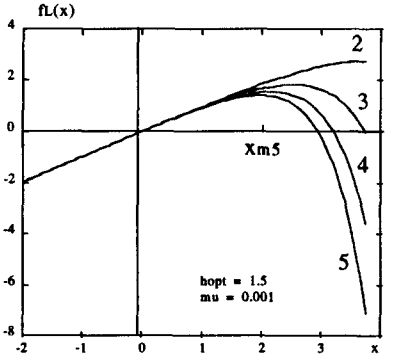

Figure 4: $f_L(x)$.

In this paper we will give some experimental results that illustrate the algorithm behavior. The following figures use the same identification model as above, with $\sigma^2 = 0.04$, $h_{opt} = 1.5$. The case $L = 3$ is studied (we did also comparisons with LMS initialized with 0, the learning rates μ_{LMS} and μ_3 have been taken such that we have the same steady state MSE for both algorithms). The figures show that the algorithm ($L = 3$) is slow for initializations smaller than the optimum. The comparison with LMS (initialized at 0) show that the latter is faster even if the neural net is initialized near the optimum h_{opt} (e.g. $h(0) = 0.1, 0.25$, and 0.5). On the other hand, the neural net is faster for the initializing values which are greater than h_{opt} (e.g. $h(0) = 4$).

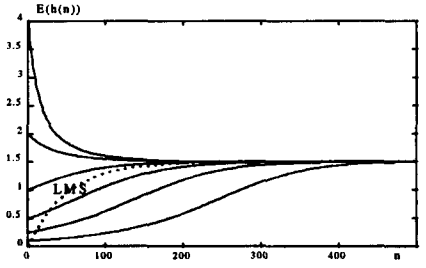

Figure 5: Evolution of $E(h(n))$, for different $h(0)$, $L = 3$, and comparison with LMS.

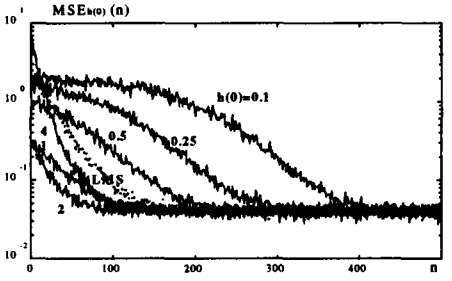

Figure 6: Corresponding MSE's.

We will now see what happens when $h(0)$ is very high. In the simulation below we took $h(0) = 20$, $L = 3$ and $h_{opt} = 1.5$. The algorithm will rapidly go (in average) to values smaller than h_{opt}, than it converges slowly to h_{opt}. In this cas, the neural net is slower than LMS (initialized with 0). The behavior of function f confirms these experimental results.

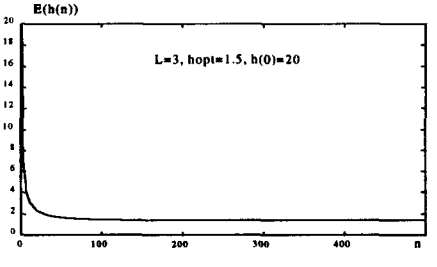

Figure 7: Case where $h(0) \gg 1$.

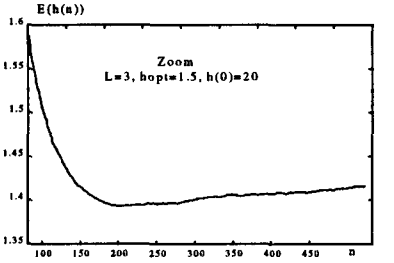

Figure 8: Case where $h(0) \gg 1$, zoom.

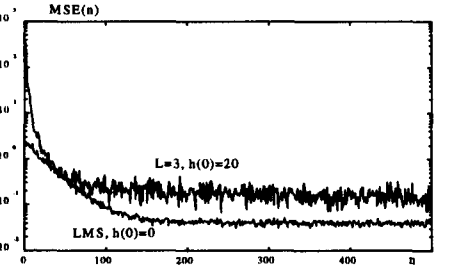

Figure 9: MSE, case where $h(0) \gg h_{opt}$, comparison with LMS, $h_{LMS}(0) = 0$.

4. STEADY STATE MSE

The SSMSE is expressed as:

$$E_R = \frac{E_{min}}{1 - \mu L h_{opt}^{2\left(1-\frac{1}{L}\right)} r_X}. \tag{3}$$

Note that for LMS ($L = 1$), the SSMSE is:

$$E_R = \frac{E_{min}}{1 - \mu r_X}.$$

The figure below compares the theoretical expression 3 and the estimated error for $L = 2$.

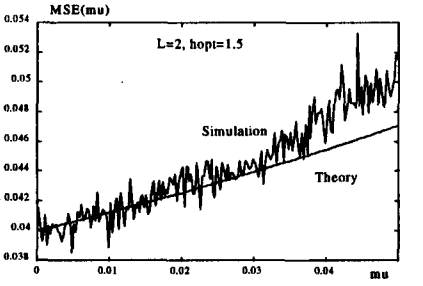

Figure 10: $MSE(\mu)$, $L = 2$.

5. INFLUENCE OF THE NUMBER OF LAYERS

5.1. ON THE STEADY STATE MSE

It is interesting to study the function $E_R(L)$ in order to see how the MSE behaves. We have:

$$\frac{dE_R(L)}{dL} = 0 \Longleftrightarrow L = L_0 = -2\ln(h_{opt}).$$

This means that $E_R(L)$ has a unique minimum (at $L_0 \geq 1$) when $h_{opt} < e^{\frac{-1}{2}}$. Whereas if $h_{opt} > e^{\frac{-1}{2}}$, the minimum L_0 is smaller than 1. Figures 11 and 12 illustrate the effect of L on the MSE.

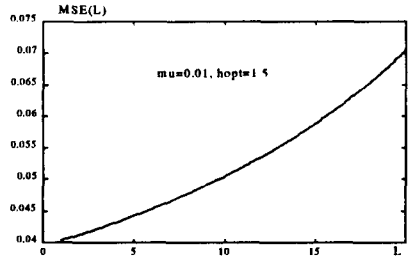

Figure 11: $E_R(L)$, case $h_{opt} > e^{\frac{-1}{2}}$.

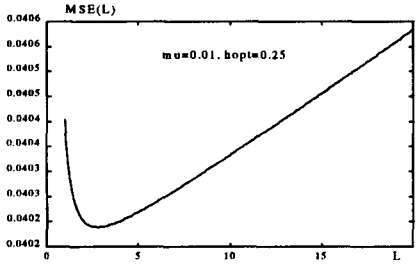

Figure 12: $E_R(L)$,case $h_{opt} < e^{\frac{-1}{2}}$.

5.2. ON THE TRANSIENT BEHAVIOR

The figures below compare the BP algorithm transient behavior to that of LMS. The step sizes $\mu(L)$ were taken such that the steady state MSE is the same for all networks (i.e. $E_R(L) = E_0$ fixed). It is shown that all the networks (initialized at $h(0) = 4$) are faster than LMS (for $h_{LMS}(0) = 4$ and $h_{LMS}(0) = 0$).

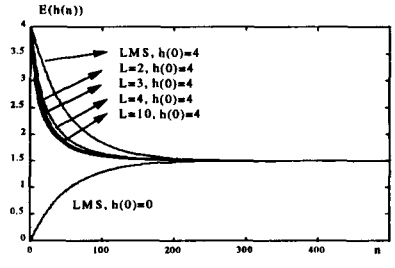

Figure 13: Influence of the number of layers on the transient behavior.

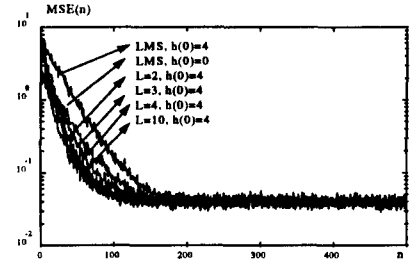

Figure 14: Corrensponding MSE's.

6. CONCLUSION

The paper studied the behavior of the backpropagation algorithm in the framework of linear adaptive filtering. We proposed a mathematical model to predict the mean weight evolution during the learning process. This model allowed to explicitly show the influence of the number of layers on the backpropagation behavior. Several simulation results were presented, they confirm our theoretical analysis and interpretation.

7. REFERENCES

[1] N. J. Bershad, J. J. Shynk, and P. Feintush, "Statistical analysis of the single-layer backpropagation algorithm: Part 1 and 2 ", IEEE Trans. Signal Processing, Vol. 41, No. 2, February, 1993.

[2] N. J. Bershad, M. Ibnkahla, and F. Castanié, "Neural Network Modelling and Identification of Nonlinear Channels with Memory: Part II - Analytic Models", submitted to IEEE Trans. Signal Processing, 1996.

[3] S. Haykin, *Adaptive Filter Theory*, 2nd ed. Englewood Cliffs, N.J., Printice-Hall.

[4] S. Haykin, *Neural Networks: a Comprehensive Foundation*, IEEE Press, 1994.

[5] M. Ibnkahla, Z. Faraj, F. Castanie and J.C. Hoffmann, "Multi-layer adaptive filters trained with back propagation: a statistical approach", Signal Processing, Vol. 40, pp. 65-85, 1994.

[6] M. Ibnkahla and F. Castanié, "Neural networks for digital communications and signal processing: overview and new results", In E. Biglieri and M. Luise Eds., *Signal Processing for Digital Communications*, Springer Verlag, London, 1996.

[7] M. Ibnkahla, *Neural Networks: New Structures and Applications to Digital Satellite Communications*, Ph. D. dissertation, National Polytechnics Institute of Toulouse, France, 1996.