Proceedings of the VI Brazilian Conference on Neural Networks - VI Congresso Brasileiro de Redes Neurais
pp. 211–216, June 2–5, 2003 – Centro Universitário da FEI, São Paulo - SP - Brazil

A Backpropagation with Automatically Generated Momentum Method

Atair Rios Neto[1], Osmar Vogler[2]
[1]INPE- Instituto Nacional de Pesquisas Espaciais
12227-010 São José dos Campos, SP, Brasil
[2]ITA- Instituto Tecnológico de Aeronáutica
12228-900 São José dos Campos, SP, Brasil
E-mails: atairrn@uol.com.br, vogler@ele.ita.br

Abstract

A method is developed for the iterative parallel solution of the feedforward neural networks supervised training problem. Stochastic optimal linear estimation is used go get a backpropagation with momentum method, where the momentum weighting is automatically done. The estimation of neural network weights is done avoiding the difficulties characteristic of Kalman filtering type of algorithms and related with the adjustment and calculations involving the a priori covariance matrix of estimation errors. Preliminary numerical testing indicates that the method is a competitive choice in terms of effectiveness, efficiency and facility of use when compared to the backpropagation method.

1. Introdution

The search for effective and efficient feedforward neural networks supervised training methods is certainly relevant and still of present interest. The theoretical results guaranteeing that these neural networks are universal approximators ([1], [2], [3]) are not constructive, that is, they do not provide the procedure for constructing the neural network to approximate a given mapping. The usual approach of treating the problem as one of optimization where a functional of fitting errors is minimized allows the use of existing numerical optimization methods. However, due to the characteristics of the problem there is still much to be done in the search for methods giving a better compromise in terms of effectiveness, efficiency and facility of use.

Recently an analogy with multilayer feedforward networks training and stochastic optimal linear estimation were used to develop a new method for the parallel solution of linear algebraic systems of equations ([4], [5]). This was done exploring analogies with the problem of supervised training of artificial neural networks when local training Kalman algorithms are used (e.g.: [6], [7], [8], [9], [10], and [11]). The method resulted to be a generalization of Jacobi's method, even when its most recent and elaborated form as presented by Björck ([12]) is considered.

In this paper, exploring analogies in the reverse way, a method is developed for the iterative parallel solution of the problem of feedforward neural networks supervised training. The resulting method turns out to be a stochastic optimal linear estimation version of a backpropagation with momentum method, where the momentum weighting is automatically done. The motivation is to have a parallel processing, stochastic optimal estimation based method, more efficient than the usual backpropagation, but which avoids the difficulties of dealing with the a priori information covariance matrix in the Kalman filtering approach.

2. Neural Network Training Approach

When a feedforward neural network is used to approximate a given mapping its training is done by supervised learning from input-output patterns data sets:

$$\{(x(t),y(t)) : y(t) = f(x(t)+\varepsilon(t), t=1,2,...,L\} \quad (1)$$

where, $\varepsilon(t)$ is a zero mean random variable representing the accuracy with which the approximation is to be attained. Adjusting (estimating) the neurons weight parameters to approximately fit the artificial neural net correspondent computational model to this data can be viewed and treated as a parameterized mapping:

$$\hat{y}(t) = \hat{f}(x(t),w) \quad (2)$$

where w is the vector of neuron weight parameters to be estimated.

A usual approach to solve the problem of supervised training is to minimize, with respect to the vector of weights *w*, the functional:

$$J(w) = 1/2[\sum_{t=1}^{L}(y(t)-\hat{f}(x(t),w))^T R^{-1}(t) \\ (y(t)-\hat{f}(x(t),w)] \quad (3)$$

where, the input-output data *{x(t),y(t): t=1,2,….,L}* is given, and the weight matrices $R^{-1}(t)$ can be taken as the inverse of the covariance matrices of $\varepsilon(t)$.

Minimizing the cost function of Eq. (3) can heuristically be seen as a solution to the estimation problem:

$$y(t) = \hat{f}(x(t), w) + \varepsilon(t), t = 1,2,...,L \qquad (3a)$$

If the mapping of Eq.(3) is expanded in a Taylor series, in a typical ith iteration, it results:

$$y(t) \cong \bar{y}(t,i) + \hat{f}_w(x(t), \bar{w}(i))(w - \bar{w}(i)) + o(2) + \varepsilon(t), t = 1,2,...,L \qquad (4)$$

where, $o(2)$ indicates the high order terms; $i = 1,2,...,I; \bar{w}(i)$ is the a priori estimate of w coming from the previous iteration, starting with $\bar{w}(1) = \bar{w}; \bar{y}(t,i) = \hat{f}(x(t), \bar{w}(i));$ $\hat{f}_w(x(t), \bar{w}(i))$ is the matrix of first partial derivatives with respect to w.

Retaining only the first order terms of the series expansion, and in order to better attend the linear perturbation condition, it is reasonable that in a typical ith – iteration:

$$a(i)[y(t) - \bar{y}(t,i)] \cong \hat{f}_w(x(t), \bar{w}(i))[w(i) - \bar{w}(i)] + \varepsilon(t) \qquad (4a)$$

where, $0 \langle a(i) \leq 1$ is a rate-of-learning parameter to be adjusted in order to guarantee the hypothesis of linear perturbation. The resulting approximation of $J(w)$ in Eq.(3), in a ith iteration, is then:

$$J(w(i)) = 1/2[\sum_{t=1}^{L} (z(t,i) - H(t,i)w(i))^T R^{-1}(t) (z(t,i) - H(t,i)w(i))] \qquad (5)$$

where the following compact notation was adopted:

$$z(t,i) \doteq a(i)[y(t) - \bar{y}(t,i)] + \hat{f}_w(x(t), \bar{w}(i))\bar{w}(i) \qquad (6)$$

$$H(t,i) \doteq \hat{f}_w(x(t), \bar{w}(i)) \qquad (7)$$

The minimization of the functional of Eq.(5) with zero weighting of the a priori knowledge of $\bar{w}(i)$ is formally equivalent (see, e.g.,[13]) to the optimal solution of the following stochastic linear estimation problem:

$$z(t,i) = H(t,i)w(i) + \varepsilon(t) \qquad (8)$$

$$E[\varepsilon(t)] = 0, E[\varepsilon(t)\varepsilon^T(t)] = R(t) \qquad (9)$$

with, $t=1,2,\ldots,L$, and $\varepsilon(t)$ taken as a Gaussian white noise sequence.

In what follows an approximate parallel processing solution will be proposed for the estimation problem of Eqs. (8).

3. Proposed Parallel Solution Method

Following closely Rios Neto and Rios Neto ([4], [5]), consider the system of redundant linear algebraic equations when Eqs. (8) are combined for all values of *t*, resulting:

$$z(i) = H(i)w(i) + \varepsilon \qquad (10)$$

where *H(i)* is a given mxn rank n real matrix; *w(i)* is the complete nx1 vector of unknown neuron weight parameters in the ith iteration; *z(i)* is a given mx1 real vector; and ε with the distribution as in Eq. (9) is to represent the numerical accuracy expected to be attained.

In an inside (relative to each outside iteration in i for each i=1,2,…, I) iterative solution scheme, the problem of Eq. (10) can be treated, in a typical i_i inside iteration, as:

$$\beta_t(z(i) - H(i)\bar{w}(i_i)) = H(i)(w(i) - \bar{w}(i_i)) + \varepsilon \qquad (11)$$

where, $i_i = 1,2,...,I_i$; $\bar{w}(i_i)$ is the value from the previous inside iteration, with an initial guess reasonably taken as $\bar{w}(1_i) = \bar{w}(i)$; and $0 < \beta_t \leq 1$ is to be chosen to adjust the step size in a given iteration of the linear system iterative solution.

Consider now the analogy where the linear algebraic system is viewed as a multilayer perceptron neural network, with a first layer of n fan out neurons and an output layer of m perceptrons with identity activation functions (e.g.,[14]). The condition of Eq. (11) is that obtained in a typical iteration if a stochastic optimal linear parameter estimation training approach is applied to this particular neural network (see e.g., [11]), to determine the input *w* associated with the output vector *z(i)*, in a situation where the output layer weights $h_{jk}(i)$ are assumed known. For this analogous neural network, one can now take an approximate in parallel iterative solution ([11]), to get in each inside iteration convergence contributing increments in the components $w_k(i)$:

$$\beta(z(i) - H(i)\bar{w}(i_i)) = H_k(i)(w_k(i) - \bar{w}_k(i_i)) + \varepsilon_k \qquad (12)$$

and for *j=1,2,...,m* and *k=1,2,...,n*:

$$\beta(z_j(i) - \sum_{l=1}^{n} h_{jl}(i)\overline{w}_l(i_i)) = h_{jk}(w_k(i) - \overline{w}_k(i_i)) + \varepsilon_{kj} \tag{13}$$

An equivalent form of Eq.(11), from which Eq.(12) would heuristically follow, is:

$$\sum_{k=1}^{n} \beta_k(z(i) - H(i)\overline{w}(i_i)) = \sum_{k=1}^{n} [H_k(i)(w_k(i) - \overline{w}_k(i_i)) + \varepsilon_k] \tag{14}$$

where, $H_k(i)$ is the kth column of *H(i);* and $\beta = \beta_k$ and ε_k sum up to β_t and ε respectively.

In the linear parameter estimation problem of Eqs.(13) the errors ε_{kj} are thus zero mean, normally distributed not correlated random errors (see Eq. (9)), with variances:

$$E[\varepsilon_{kj}^2] = \overline{r}_j = n^{-2}E[\varepsilon_j^2] = n^{-2}r_j \tag{15}$$

With this modeling approach, the problem of solving for the generic component $w_k(i)$ in Eq.(12) can be viewed as one of stochastic linear parameter estimation, in each i_i inside iteration. For each $k = 1,2,...,n$, the observation like Eq.(12) can then in each i_i iteration be processed in parallel, using a without a priori information Gauss-Markov estimator (see, for example, [15]or[13]), to get an estimate of the components $w_k(i)$:

$$\hat{w}_k(i_i) = \hat{w}_k(i,i_i) = \overline{w}_k(i_i) + \beta[H_k^T(i)\overline{R}^{-1}H_k(i)]^{-1}H_k^T(i)\overline{R}^{-1}(z(i) - H(i)\overline{w}(i_i)) \tag{16}$$

where, as defined before, $H_k(i)$ is the kth column of *H(i)*; and

$$\overline{R} = diag.[\overline{r}_j = n^{-2}r_j : j = 1,2,...,m] = n^{-2}R \tag{17}$$

Noticing that the without a priori information form of the estimator in Eq. (16) allows to cancel out the factor n^{-2} in $\overline{R}$, and that *R* is the same for *k=1,2,…,n,* it is then possible to combine the estimates $\hat{w}_k(i,i_i)$, of the parallel processing estimation, to get the equivalent following estimator for the whole vector *w(i)* in the inside iteration i_i :

$$\hat{w}(i_i) = \hat{w}(i,i_i) = \overline{w}(i_i) - \beta S \nabla f_L{}^T(\overline{w}(i_i)) = \overline{w}(i_i) - \beta S g(\overline{w}(i_i)) \tag{18}$$

$$S = diag.[(H_k^T(i)R^{-1}H_k(i))^{-1} : k = 1,2,...,n]$$

$$f_L(w(i)) = 1/2[H(i)w(i) - z(i)]^T R^{-1} [H(i)w(i) - z(i)] \tag{19}$$

$$\nabla f_L{}^T(\overline{w}(i_i)) = H^T(i)R^{-1}(H(i)\overline{w}(i_i) - z(i)) \hat{=} g(\overline{w}(i_i))$$

and where $0 < \beta$ can be chosen such as to minimize $f_L(w(i))$ in a given inside iteration, if it is taken as ([16]):

$$\beta = (\overline{g}^T S\overline{g})/(\overline{g}^T SH(i)^T R^{-1}H(i)S\overline{g}), \quad \overline{g} = g(\overline{w}(i_i)) \tag{20}$$

Since S and $H(i)^T R^{-1}H(i)$ are positive definite matrices, the estimator of Eq.(18) is equivalent to a modified Newton method applied to the functional $f_L(w(i))$.

Convergence of the parallel processing method of Eqs.(16) can now be verified considering its Newton method equivalent form of Eq. (18), using the Kantorovich inequality and concluding that in each iteration (see for example [16], pp. 261-262):

$$Q[\hat{w}(i,i_i)] \le \gamma^2 Q[\overline{w}(i_i)] \tag{21}$$

$$Q[w(i)] = 1/2(w(i) - w^*(i))^T H(i)^T R^{-1} H(i)(w(i) - w^*(i)) \tag{22}$$

$$\gamma = (\lambda_M - \lambda_m)/(\lambda_M + \lambda_m) \tag{23}$$

$w^*(i)$ being the value of $w(i)$ that leads to the minimum of $f_L(w(i))$ in Eq.(19); and λ_M, λ_m the largest and smallest eigenvalues of the positive definite matrix $SH(i)^T R^{-1}H(i)$.

Notice that for each neuron weight $w_k(i)$ the estimate of Eq.(16) is the result of processing in batch the L observations like Eqs.(13). Since the $\varepsilon(t)$ are taken as Gaussian white noise sequences, the matrices *R(t)* are diagonal and the result of the estimate of Eq.(16) can equivalently be obtained by recursively processing Eqs.(13) (see e.g., [15], pp. 161-163). Thus, the proposed parallel solution method turns out to be a backpropagation with automatically generated momemtum type of method.

The numerical implementation of the method can be done in the following three steps algorithm:

(i) In an outside $i \le I$ iteration, and for all the *t=1,2,…,T,* with $x(t), y(t), R(t)$ given; and with the a priori estimate $\overline{w}(i)$

coming from the previous (*i-1*) iteration or starting with $\overline{w}(1)=\overline{w}$, calculate the $\overline{y}(t,i)=\hat{f}(x(t),\overline{w}(i))$. If the $(y(t)-\overline{y}(t,i))$ are inside the 3 sigma limits given by the distribution of the $\varepsilon(t)$, make $\hat{w}(i)=\overline{w}(i)$ and stop, convergence was reached. If not: calculate the $\hat{f}_w(x(t),\overline{w}(i))$; adjust $0\langle a(i)\leq 1$ to guarantee the hypothesis of linear perturbation; calculate the $z(t,i),H(t,i)$ and form $z(i),H(i)$; calculate S (Eq. (19)) and $SH(i)^T R^{-1}H(i)S$. Go to (ii).

(ii) For the given *i*, in a given inside $i_i \leq I_i$ iteration, with $\overline{w}(i_i)=\hat{w}(i_i-1)$ the a priori estimate coming from the previous iteration in i_i and starting with $\overline{w}(1_i)=\overline{w}(i)$ calculate $(z(i)-H(i)\overline{w}(i_i))$, and for those components inside the 3 sigma limits, take

$$(z_j(i)-H_j(i)\overline{w}(i_i))=$$
$$(sign(z_j(i)-H_j(i)\overline{w}(i_i)))3r_j^{1/2}$$

If all components are inside the 3 sigma limits given by the distribution of ε, increment *i*, and for the incremented value of *i* redefine $\overline{w}(i)=\overline{w}(i_i)$, reset $i_i=1$ and go back to (i). If not, calculate $g(\overline{w}(i_i))$ (Eq. (19)) and β (Eq. (20)), go to (iii).

(iii) In parallel, for *k=1,2,…,n*, calculate the estimates of $w_k(i)$ (Eq. (16)). Redefine $\overline{w}_k(i_i+1)=\hat{w}_k(i,i_i)=\hat{w}_k(i_i)$, increment i_i and go back to (ii).

4 Preliminary Numerical Test

The very simple xor problem is considered, for a preliminary evaluation of the method developed as compared to Matlab versions of the Backpropagation and the Levenberg Marquadt methods. A multilayer perceptron neural network with two inputs, two hidden hyperbolic tangent sigmoid neurons and one linear output neuron was trained, initializing the weights with random outcomes of a zero mean and variance one normal distribution. In what follows, the results, correspondent to the value of the rate of learning parameter (alfa) for which each method exhibited the best performance, are presented.

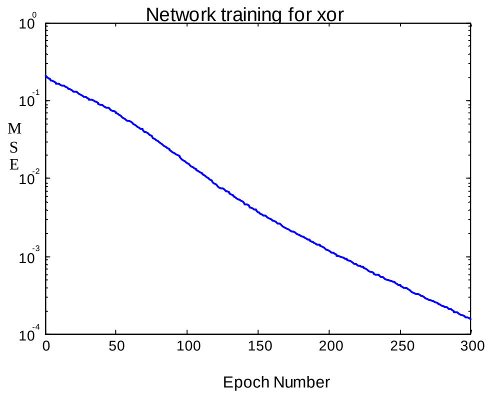

Figure 1: Backpropagation best result, $\alpha=0.1$

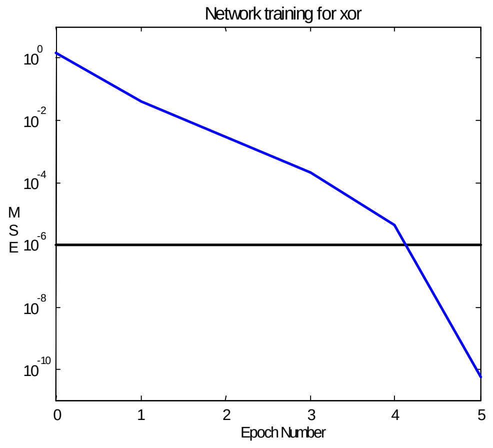

Figure 2: Levenberg Marquadt best result, $\alpha=0.6$

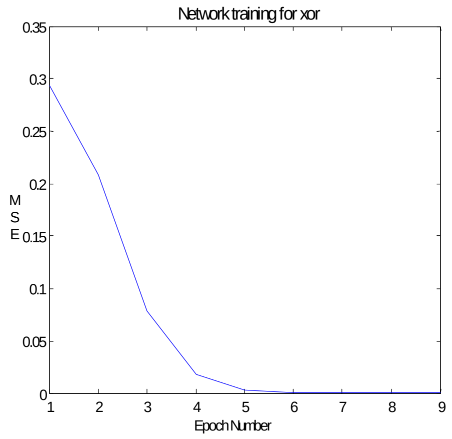

Figure 3: New Method best result, $\alpha=0.8$

These results indicate that the new proposed method is competitive as compared to the usual Backpropagation, as expected, since it is a backpropagation with momentum type of method. The results also indicate that the new method can be competitive as compared to the Levenberg Marquadt method, depending upon optimization of the numerical programming of its numerical algorithm.

Though numerical tests in more complex situations have not yet been done, it is expected that for neural networks with more complex architectures, where a large number of neurons is involved, the proposed method will suffer limitations similar to those of the Levenberg Marquadt method, and related to memory requirements and time spent in each iteration calculations. It is recognized that further efforts have to be made to simplify the calculations to get β in Eq. (20).

5. Conclusions

The adoption of an iterative stochastic parameter estimation approach allows the problem of neural network supervised training to be reduced, in each iteration, to one of solving a stochastic linear estimation problem. The possibility of getting an approximate parallel iterative solution can be explored if in each iteration this problem is looked at as one of solving a stochastic system of redundant linear algebraic equations and if an analogy is made to view the linear system as modeled by a multilayer perceptron neural network.

In this paper these possibilities were explored to develop a stochastic backpropagation with automatic momentum weighting method.

The motivation was to have a method giving a good compromise in terms of effectiveness, efficiency and facility of use. That is, a method with effectiveness of convergence close to Kalman filtering type methods, but with the efficiency of backpropagation with momentum methods, however without the difficulties of heuristically having to adjust the weighting of the momentum term.

The preliminary testing results presented are only indicative of method's expected performance. However, the fact that a numerical formal equivalence with a modified Newton method exists is a guarantee of competitive performance as compared to gradient based, first order methods.

References

[1] G. Cybenko. Approximation by superposition of a sigmoidal function, *Mathematics of Control, Signals and Systems*, Vol. 2, No. 3, pp. 303-314, 1989.

[2 K. Hornik, M. Stinchcombe, H. White. "Multilayer feedforward networks are universal approximators", *Neural Networks*, Vol. 2, No. 5, pp. 359-366, 1989.

[3] K. Funahashi. On the approximate realization of continuous mappings by neural networks, *Neural Networks*, Vol. 2, No. 3, pp. 183-192, 1989.

[4] A. Rios Neto and W. Rios Neto. An Optimal Linear Estimation Approach to the Parallel Solution of Linear Algebraic Systems. SBA *Controle & Automação*, 11(1), ISSN 0103-1759, pp. 61-67, 2000.

[5] A. Rios Neto and W. Rios Neto. A Neural Network Analogy and a Stochastic Parameter Estimation Approach in the Parallel Solution of Linear Algebraic Systems of Equations, in *Advances in Space Dynamics*, Antonio Fernando Bertachini Almeida Prado, editor, ISBN:85-901487-1-8, pp.262-271, 2000.

[6] Y. Iiguni and H. Sakai. A Real Time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter, *IEEE Transactions on Signal Processing*, 40(4), pp. 959-966, 1992.

[7] R.S. Scalero and N. Tepedelenlioglu. A Fast New Algorithm for Training Feedforward Neural Networks, *IEEE Transactions of Signal Processing*, 40(1), pp. 202-210, 1992.

[8] G. Chen and H. Ögmen. Modified Extended Kalman Filtering for Supervised Learning, *Int. J. Systems Sci.*,24(6), pp. 1207-1214, 1993.

[9] F. Lange. Fast and Accurate Training of Multilayer Perceptrons Using an Extended Kalman Filter (EKFNet)", *internal paper*, DLR(German Institute Aerospace Research Establishment), Institute for Robotics and Systems Dynamics, 1995.

[10] A. Rios Neto. Kalman Filtering Stochastic Optimal Estimation Algorithms and Usual Backpropagation in Neural Nets Training. *Proc. Second Brazilian Congress in Neural Networks*, Curitiba, PR, Brazil, pp. 139-144, 1995.

[11] A. Rios Neto. Stochastic Optimal Linear Parameter Estimation and Neural Nets Training in Systems Modelling. *RBCM- J. of the Braz. Soc. Mechanical Sciences*, XIX(2), pp. 138-146, 1997.

[12] A. Björck. *Numerical Methods for Least Squares Problems*. SIAM- Society for Industrial and Applied Mathematics, Philadelphia, 1996.

[13] A.H. Jazwinski. *Stochastic Process and Filtering Theory*. Academic Press, NY, 1970.

[14] J.M. Zurada. *Introduction to Artificial Neural Systems*, West Pub. Co, 1992.

[15] P.B. Liebelt. *An Introduction to Optimal Estimation*. Addison-Wesley, 1967.

[16] D.G.Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.