STOCHASTIC OPTIMAL LINEAR PARAMETER ESTIMATION AND NEURAL NETS TRAINING IN SYSTEMS MODELING

Atair Rios Neto

Instituto Nacional de Pesquisas Espaciais - INPE

12201-970 São Jose dos Campos, SP, Brasil

atairrn@uol.com.br

Abstract

Supervised training of feedforward neural networks for nonlinear mapping and dynamical systems modeling is addressed. Viewing neural nets training as a stochastic parameter estimation problem, results in Kalman filtering are adapted to develop training algorithms. Many levels of approximation are considered to develop a range of full non parallel to simplified parallel processing versions of algorithms, together with an adaptive approach intended to give to these algorithms the features of good numerical behavior and of distributing the extraction of learning information to all training data.

Keywords: Neural Nets Supervised Training, Neural Nets and Systems Modeling, Stochastic Optimal Estimation Training Algorithms, Kalman Filtering Training Algorithms.

INTRODUCTION

In recent years, the experience with optimal linear parameter estimation procedures has been explored to develop neural networks supervised training algorithms having the structure of recursive least squares (Chen and Billings, 1992) and of Kalman filtering (Singhal and Wu, 1989; Watanabe, Fukuda and Tzafestas, 1991; Scalero and Tepedelenlioglu, 1992; Chen and Ögmen, 1993; Chandran, 1994; Lange, 1995).

In this paper the author further explores the possibility given by Kalman filtering. Previously full non local processing (Rios Neto, 1994) and local parallel processing (Rios Neto, 1995) feedfoward neural nets training algorithms are presented together with the development of an adaptive procedure. Extending the stochastic optimal parameter estimation solution of the neural net supervised training problem one models the weight parameters as random walk stochastic processes. Noise dispersion adaptation (Rios Neto and Kuga, 1985) is then used as an automatic way of conditioning the covariance matrix of parameters estimation errors, thus avoiding loosing the capacity to extract information of new data as the processing goes on. The use of this adaptive procedure is thus intended to be effective along the processing in distributing to all data the extraction of learning information.

FUNDAMENTALS: FEEDFORWARD NEURAL NETWORKS AND DYNAMIC SYSTEMS MODELLING

Among the types of feedforward artificial networks used for modeling and identification of systems (Chen and Billings,1992) the most basic and frequently used one is the Multilayer Perceptron made up of layers of basic artificial neurons connected forward, as illustrated in Fig.1, for the ith neuron of a kth hidden layer, with n_k neurons:

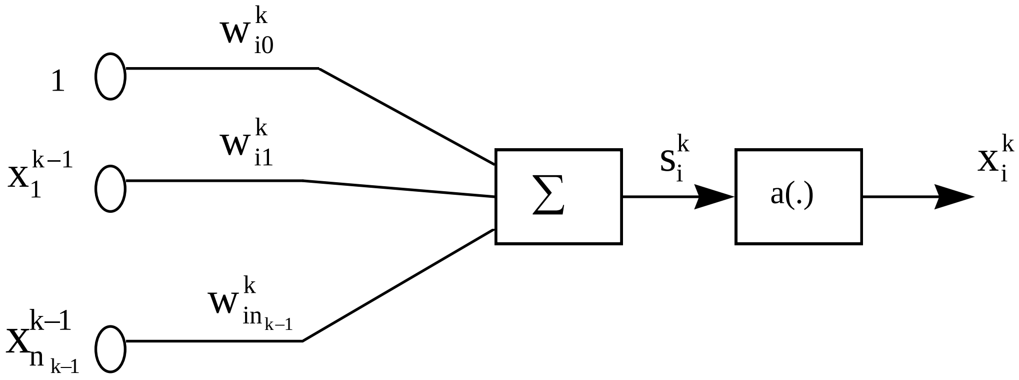

Figure 1: Artificial Neuron

$$s_i^k = \sum_{j=1}^{n_{k-1}} w_{ij}^k x_j^{k-1} + w_{i0}^k \qquad (1)$$

$$x_i^k = a(s_i^k), k = 1,2,...,l-1 \qquad (2)$$

with the activation function a(s) being typically taken as:

$$a(s)= 1/(1+ exp(-s)) \text{ or } a(s)=tanh(s) \qquad (3)$$

The inputs to the first hidden layer are $x_i^0 = x_i, i = 1,2,...,n$, the network input vector. For the neurons of the output layer $(k = l)$ it is sufficient to have and are frequently used zero threshold weights (w_{i0}^l) and identity activation functions:

$$\hat{y}_i \equiv x_i^l = \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^{l-1}, i = 1,2,...,m \qquad (4)$$

Feedforward artificial neural networks can be trained to uniformly and with the desired accuracy represent a nonlinear and continuous mapping (see, e.g., Zurada, 1992):

$$f \in C{:}x \in D \subset R^n \to y \in R^m \tag{5}$$

The theory already available (see, e.g., Hecht-Nielsen, 1990) guarantees that for the case of the Mutilayer Perceptron it is enough to have a neural network built with just one hidden layer, as illustrated in Fig. 2.

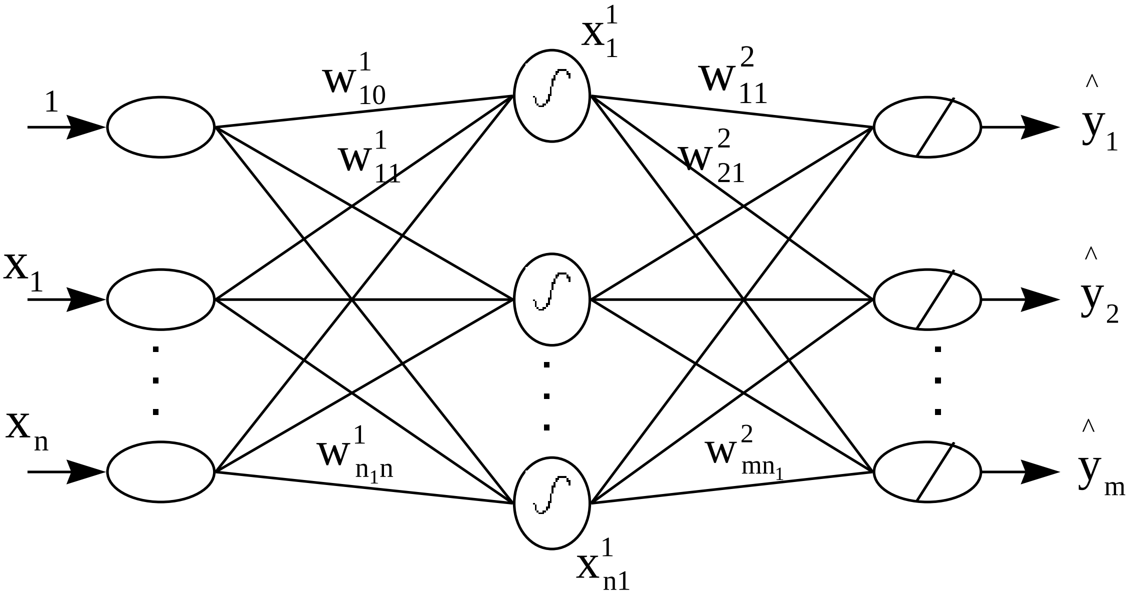

Figure 2: One Hidden Layer Perceptron

The training of a feedforward network is usually done by supervised learning, from mapping data sets:

$$\{(x(t), y(t)){:}y(t) = f(x(t)), t = 1,2,\ldots, L\} \tag{6}$$

adjusting (estimating) weight parameters to approximately fit the artificial neural net correspondent computational model to this data of input-output patterns.

The processing by the trained artificial neural net of the input data $x(t)$, to produce outputs $\hat{y}(t)$, can be viewed and treated as a parameterized mapping:

$$\hat{y}(t) = \hat{f}(x(t), w) \tag{7}$$

where w is the vector of weight parameters. In the case of the perceptron neural net with one hidden layer and hyperbolic tangent activation function (Fig.2), Eq. (7) is expressed as:

$$\hat{y}_i(t) = \sum_{j=1}^{n_1} w_{ij}^2 (\tanh[\sum_{k=1}^{n} w_{jk}^1 x_k(t) + w_{j0}^1]) \tag{8}$$

This capacity of feedforward artificial neural nets of representing nonlinear mappings can be used to approximately model dynamic systems of the type:

$$\dot{x} = f(x, u) \tag{9}$$

as long as $f(.)$ is invariant in time (see, e.g., Chen and Billings, 1992). To do so, one has to implicitly assume that it is possible to consider a system as in Eq.(9) as if it was approximated by a discrete model like:

$$x(t + \Delta t) = \hat{f}(x(t), x(t - \Delta t), ..., x(t - n_x \Delta t); u(t), u(t - \Delta t), ..., u(t - n_u \Delta t), w) \tag{10}$$

which is of the type of Eq.(7) and where $n_x, n_u, \Delta t$ are to be adjusted together with the neural net architecture and size, depending upon the problem treated and desired accuracy. Notice that Δt can be treated as an extra component of the input to the neural net. What feedforward neural nets do in this case is to learn the mapping of Eq. (10).

At this point is opportune to remember the similar situation that occurs when numerical integrators are used and dynamic systems as in Eq.(9) are treated as discretized approximations, like in Eq.(10).

SUPERVISED TRAINING: OPTIMAL LINEAR ESTIMATION PROCEDURE

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

$$J(w) = 1/2[(w-\overline{w})^T \overline{P}^{-1}(w-\overline{w}) + \sum_{t=1}^{L}(y(t) - \hat{f}(x(t),w))^T R^{-1}(t)(y(t) - \hat{f}(x(t),w))] \qquad (11)$$

given the input-output data {x(t),y(t): t=1,2,….,L}, an a priori estimate $\overline{w}$, and the weight matrices $\overline{P}^{-1}, R^{-1}(T)$.

In the proposed solution (Rios Neto, 1994), a linear perturbation is adopted to approximate the functional of Eq.(11) in a typical ith iteration, imposing the condition that:

$$\alpha(i)[y(t) - \overline{y}(t,i)] \cong \hat{f}_w(x(t),\overline{w}(i))[w(i) - \overline{w}(i)] \qquad (12)$$

where, $i = 1,2,...,I; \overline{w}(i)$ is the a priori estimate of w coming from the previous iteration, starting with $\overline{w}(1) = \overline{w}; \overline{y}(t,i) = \hat{f}(x(t),\overline{w}(i));$ $\hat{f}_w(x(t),\overline{w}(i))$ is the matrix of first partial derivatives with respect to w ; and $0\langle\alpha(i)\leq 1$ is a parameter to be adjusted in order to guarantee the hypothesis of linear perturbation. The resulting approximation of $J(w)$ in Eq.(11J) is then:

$$J(w(i)) = 1/2[(w(i)-\overline{w})^T\overline{P}^{-1}(w(i)-\overline{w})+\sum_{t=1}^{L}(z(t,i)-H(t,i)w(i))^T R^{-1}(t)(z(t,i)-H(t,i)w(i))] \quad (13)$$

where the following compact notation was adopted:

$$z(t,i) \doteq \alpha(i)[y(t)-\overline{y}(t,i)]+\hat{f}_w(x(t),\overline{w}(i))\overline{w}(i) \quad (14)$$

$$H(t,i) \doteq \hat{f}_w(x(t),\overline{w}(i)) \quad (15)$$

The solution of minimizing the functional of Eq.(13) is formally equivalent (see, e.g., Jazwinski, 1970) to the following stochastic linear estimation problem:

$$\overline{w} = w(i)+\overline{e} \quad (16)$$

$$z(t,i) = H(t,i)w(i)+v(t) \quad (17)$$

$$E[\overline{e}] = 0, E[\overline{e}\overline{e}^T] = \overline{P} \quad (18)$$

$$E[v(t)] = 0, E[v(t)v^T(t)] = R(t) \quad (19)$$

with $\overline{e}$ and $v(t)$ not correlated and taken to have Gaussian distributions.

PROPOSED OFF LINE SOLUTION

Following closely Rios Neto (1994), a Kalman filtering algorithm is proposed for an off line batch solution of problem of Eqs. (16) to (19) :

$$\hat{w}(i) = \overline{w} + K(i)[z(i) - H(i)\overline{w}] \tag{20}$$

$$K(i) = \overline{P}H^T(i)[H(i)\overline{P}H^T(i) + R]^{-1} \tag{21}$$

$$\overline{w}(i+1) = \hat{w}(i), \alpha(i) \leftarrow \alpha(i+1) \tag{22}$$

where all the values of $t = 1,2,...,L$ were considered to define the extended vector $z(i)$, matrix $H(i)$ and the error vector *v* with covariance matrix *R*. The off line solution after iterations *i=1,2,…,I* is given by*:*

$$\hat{w} = \hat{w}(I),\ P(I) = [I - K(I)H(I)]\overline{P} \tag{23}$$

If $\alpha(i)$ is sufficiently small to disregard high order terms in the linearization and enough redundancy exists in the training data, then unless of bad numerical behavior of the Kalman filtering algorithm (Bierman, 1997) theory guarantees that *P(I)* is an approximation for the estimation error covariance matrix, that is:

$$P(i) \cong E[(w - \hat{w})(w - \hat{w})^T]$$

For this off-line solution the following remarks apply:

(i) since the natural situation is to have components of the error *v* uncorrelated , then *R* is diagonal and the recursive Kalman filtering algorithm can be used to process the vector *z(I)* componentwize , avoiding the need of matrix inversion in Eq.(21) (Jazwinski, 1970);

(ii) if it happens that a new data set of pairs {x(t),y(t)} is to be considered for network training, one only has to consider the recursive nature of Kalman filtering algorithm and take as new a priori information:

$$\overline{w} \leftarrow \hat{w} \qquad and \qquad \overline{P} \leftarrow P(I);$$

(iii) though the backpropagation rule can be used to calculate the gradients in matrix H(i) (Chandran, 1994), the algorithm presented does not attain parallel processing;

(iv) due to Kalman filtering typical behavior one should consider adopting some kind of factorization (Bierman, 1977) and or adaptive technique (Jazwinski, 1970; Rios Neto and Kuga, 1985) to avoid either numerical divergence or loosing the capacity of having the learning distributed to all data.

SIMPLIFIED SOLUTIONS: PARALELL PROCESSING

Algorithms with the structure of a Kalman filtering, which coincides with that of a recursive least squares, can be simplified to produce versions which preserve the local parallel processing capability of artificial neural networks (Chen and Billings, 1992). Exploring this possibility the author in a previous paper (Rios Neto, 1995) proposed approximated versions of the off line stochastic optimal parameter estimation algorithm (Eqs. (20) to (23)) and showed that even for the most simplified version of the stochastic optimal linear estimation, Kalman filtering algorithm leads to a local parallel processing algorithm still more general and sophisticated than the usual Backpropagation.

To better fulfill the purposes of this paper, in what follows one summarizes these results previously proposed by the author. Examining the equation bellow:

$$\alpha(i)[y(t)-\bar{y}(t,i)]=\hat{f}_w(x(t),\bar{w}(i))[w(i)-\bar{w}(i)]+v(t) \tag{24}$$

which is equivalent to Eq.(17), it can be seen that in the ith iteration the input-output data set can be locally processed to get an estimate of the vector of weights $w_{kl}(i)$ of the lth neuron in the kth layer if one considers that:

(i) for connection weight parameters $w_a(i)$ of neural net layers forward or after the one where processing is being done there are already available the estimated values $\hat{w}_a(i)$ and associated error $e_a(i)$ of known distribution;

(ii) for parameters $w_s(i)$, correspondent to weights of connections do the other neurons in the same layer, there is an a priori estimate $\bar{w}_s(i)$ with error $\bar{e}_s(i)$ of known distribution which can be taken as an approximation for w_s;

(iii) for the weight parameters $w_e(i)$ correspondent to connections to the neurons in earlier layers there is also an a priori estimate $\bar{w}_e(i)$ with error $\bar{e}_e(i)$ of known distribution which can be taken as an approximation for $w_e(i)$.

With the previous assumptions, the problem of getting an estimate for the vector of weights $w_{kl}(i)$ of the lth neuron in the kth layer is reduced to the local estimation problem in the ith iteration and for *t=1,2,…,L* :

$$\bar{w}_{kl}(i)=w_{kl}(i)+\bar{e}_{kl} \tag{25}$$

$$\alpha(i)[y(t)-\bar{y}(t,i)]-\hat{f}_{w_a}(x(t),\bar{w}(i))[\hat{w}_a(i)-\bar{w}_a(i)]=\hat{f}_{w_{kl}}(x(t),\bar{w}(i))[w_{kl}(i)-\bar{w}_{kl}(i)]+(\hat{f}_{w_a}(x(t),\bar{w}(i))e_a(i)$$

$$+\hat{f}_{w_s}(x(t),\overline{w}(i))\overline{e}_s(i)+\hat{f}_{w_e}(x(t),\overline{w}(i))\overline{e}_e(i)+v(t)) \tag{26}$$

Or in a more compact notation and for all the data corresponding to *t=1,2,...,L*:

$$\overline{w}_{kl}(i)=w_{kl}(i)+\overline{e}_{kl} \tag{27}$$

$$\overline{z}_{kl}(i)=H_{kl}(i)w_{kl}(i)+\overline{v}_{kl}(i) \tag{28}$$

where $\overline{e}_{kl}$ is the correspondent partition of $\overline{e}$ in Eq.(16) and since in an ith iteration this problem can be locally and recursively solved with the Kalman filtering algorithm, starting with $\overline{P}$ diagonal, there results that the components of the errors $e_a(i),\ \overline{e}_s(i)$ *and* $\overline{e}_e(i)$ associated to parameters of different neurons are not correlated.

Further approximations can be done to produce simpler local parallel processing algorithms by: (iv) disregarding the off diagonal terms of the covariance matrix $\overline{R}_{kl}(i)$ of the error $\overline{v}_{kl}(i)$ (Eq.(28)) allows to process $\overline{z}_{kl}(i)$ componentwize in Eq. (28), thus avoiding the need of matrix inversion; taking this approximation corresponds to consider $e_a(i),\overline{e}_s(i)$ *and* $\overline{e}_e(i)$ in Eq. (26) not correlated to each other and to $v(t)$, $R(t)$ diagonal, and to disregard off diagonal terms of the covariance matrices of:

$$\hat{f}_{w_a}(x(t),\overline{w}(i))e_a(i),\quad \hat{f}_{w_s}(x(t),\overline{w}(i))\overline{e}_s(i),\quad \hat{f}_{w_e}(x(t),\overline{w}(i))\overline{e}_e(i);$$

(v) disregarding the information on the level of accuracy in previous knowledge of $w_a(i),\ w_s(i)$ *and* $w_e(i)$, and taking these values as if they were:

$$w_a(i) = \overline{w}_a(i),\ w_s(i) = \overline{w}_s(i),\ w_e(i) = \overline{w}(i) \tag{29}$$

what implies a further simplified version of Eq. (26):

$$\alpha(i)[y(t) - \overline{y}(t,i)] = \hat{f}_{w_{kl}}(x(t), \overline{w}(i))[w_{kl}(i) - \overline{w}_{kl}(i)] + v(t) \tag{30}$$

and which combined with Eq.(27) results in the simplified estimation problem for the data corresponding to *t=1,2,...,L*

$$\overline{w}_{kl}(i) = w_{kl}(i) + \overline{e}_{kl} \tag{31}$$

$$z_{kl}(i) = H_{kl}(i) w_{kl}(i) + v \tag{32}$$

This last simplified version is still more sophisticated than the usual Backpropagation algorithm, and can be shown to be the result of application of Newton's method to the functional of Eq.(13) when the approximation of Eq. (29) is considered.

ADAPTIVE SOLUTION: DISTRIBUTED LEARNING OF NEURAL WEIGHTS

The problem with least squares type of estimators, and with Kalman filtering in particular, is that due to both algorithm bad numerical behavior and observation model errors, divergence usually occurs as many data sets are processed. This is due to the fact that the algorithm "learns too well the wrong information" (Jazwinski,1970) loosing capacity of keeping learning as new data are processed. What happens is an excessive and unrealistic decrease in the estimated dispersions of the errors in the calculated estimates. This corresponds to the situation of having the matrix of estimated covariances of the errors en the estimates with eigenvalues too close to zero.

To avoid this ill behavior and to try to keep a distributed and as much as possible uniform capacity of learning, it is common to use forgetting factor type techniques or more effective adaptive state estimation techniques like the one proposed by Jazwinski(1970) and modified by Rios Neto and Kuga(1985).

To apply an adaptive procedure based on a criterion of statistical consistency to balance a priori information priority with that of new learning information, the neurons connection weights parameters in the problem of neural net supervised training need to be modeled as random walk processes. Thus, in the ith iteration and for *t=1,2,...,L-1* :

$$w(i,t+1) = w(i,t) + \eta(t) \tag{32}$$

$$E[\eta(t)] = 0, \quad E[\eta(t)\eta^T(\tau)] = Q(T)\delta_{t\tau} \tag{33}$$

where $\delta_{t\tau}$ is the Kronecker symbol and for the n_w weight parameters:

$$Q(t) = diag[q_j(t): j = 1,2,...,n_w] \tag{34}$$

With this modeling approximation for the neural weights, learning from the th input-output data pattern is transformed in the estimation problem:

$$\overline{w}(i,t) = w(i,t) + \overline{e}(i,t) \tag{35}$$

$$z(t,i) = H(t,i)w(i,t) + v(t) \tag{36}$$

starting with $\overline{e}(i,1) = \overline{e}, \quad \overline{w}(i,1) = \overline{w}(i)$ and for *t=1,2,...,L.*.

To propagate estimates from t to t+1 Kalman filter predictor is used considering the dynamics of Eq.(32) :

$$\overline{w}(i,t+1)=\hat{w}(i,t) \tag{37}$$

$$\overline{P}(i,t+1)=P(i,t)+Q(t) \tag{38}$$

where $\overline{P}(i,t+1)=E[\overline{e}(i,t+1)\overline{e}^{T}(i,t+1)]$ *and* $P(i,t)$ is given by the filtering algorithm:

$$P(i,t)=[I-K(i,t)H(t,i)]\overline{P}(i,t) \tag{39}$$

where $\overline{P}(i,t)$ starts with $\overline{P}(i,1)=E[\overline{e}\overline{e}^{T}]=\overline{P}$.

The adaptation is done by adjusting the noise $\eta(t)$ dispersion, such as to keep statistical consistency and to attain distributed learning:

$$\beta E[v_j^2(t+1)]=H_j(t+1,i)[P(i,t)+Q(t)]H_j^{\,T}(t+1,i), \qquad 1\le\beta\ge\beta_{lmx} \tag{40}$$

where *j=1,2,…,m* and β is to be adjusted close to 1 in order to have distributed learning.

This adaptive condition leads to the associated observation like condition , after some algebraic manipulations and adoption of a compact notation:

$$z^{q}(t+1,i,\beta)=H^{q}(t+1,i)q(t) \tag{41}$$

In order to use the same Kalman filtering algorithm the following associated estimation problem is considered:

$$0 = q(t) + \bar{e}^q \tag{42}$$

$$z^q(t+1,i,\beta) = H^q(t+1,i)q(t) + v^q(t+1) \tag{43}$$

$$E[\bar{e}^q] = 0, \qquad E[\bar{e}^q\bar{e}^{q^T}] = I_{n_w} \tag{44}$$

$$E[v^q(t+1)] = 0, \qquad E[v^q(t+1)v^q(t+1)] = R^q(t+1) = 0 \tag{45}$$

which is a problem with exact observations that can be processed with Kalman filtering as long as one takes $R^q(t+1)$ in the limit as being zero (Freitas Pinto and Rios Neto, 1990). The solution gives a $\hat{q}(t)$ which is closest to zero in magnitude. Whenever a $\hat{q}_k(t)$ component is less than zero it is disregarded and taken to be zero , since the condition of positivity has to be observed.

CONCLUSIONS

Possibilities of results and past experience already existent in stochastic optimal linear parameter estimation were explored adapting Kalman filtering type of algorithms for feedforward neural networks supervised training. Full non parallel processing algorithms suitable for off line use as well as simplified parallel processing algorithms suitable for on line use which allows to stochastically treat the accuracy of training data were developed. Exploring past experience with state noise estimation in stochastic state observers, an automatic and adaptive approach was proposed which is expected to prevent these Kalman filtering based neural net weight estimators of loosing the capacity of distributing the extraction of information to all training data.

There is no reason for not expecting in neural nets training the same behavior stochastic optimal linear estimation algorithms have had in other applications of systems identification. The versions developed in this paper are all more sophisticated and realistic than the usual Backpropagation algorithm. The price to be paid is more numerical complexity. This should be not a serious limitation for off line applications. For on line, real time applications, the computational resources already available for parallel processing may be enough to make competitive the use of the simplified versions, specially in mechanical systems where typical times of response are not so small.

REFERENCES

Bierman,G.J., 1977, “Factorization Methods for Discrete Sequential Estimation”, Academic Press, U.S.A.

Chandran, P.S., 1994, “Comments on Comparative Analysis of Backpropagation and the Extended K alman Filter for Training Multilayer Perceptrons”, IEEE Transactions on Pattern Analysis and Machine Inteligence, 16(8), pp.862-863.

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

Chen, S. and Billings, S.A., 1992, “Neural Networks for Nonlinear Dynamic System Modelling and Identification”, Int. J. Control, 56(2), pp. 319-346.

Freitas Pinto, R.L.U. and Rios Neto, A., 1990, “An Optimal Linear Estimation Approach to Solve Systems of Linear Algebraic Equations”, J. Computational and Applied Mathematics, 33, pp. 261-268.

Iiguni,Y. and Sakai, H., 1992, “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.

Jazwinski, A.H., 1970, “Stochastic Processes and Filtering Theory”, Academic Press, N.Y., U.S.A.

Lange, F., 1995, “ 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.

Rios Neto, A., 1994, “ Stochastic Parameter Estimation Neural Nets Supervised Learning Approach”, Proceedings. First Brazilian Congress in Neural Networks, Itajubá, Minas Gerais, Brasil, pp. 62-65.

Rios Neto, A.,1995, “ Kalman Filtering Stochastic Optimal Estimation Algorithm and Usual Backpropagation in Neural Nets Training”, Proceedings of Second Brazilian Congress in Neural Networks, Curitiba, Parana, Brasil, pp. 139-144.

Rios Neto,A. and Kuga, H.K., 1985, “ Kalman Filtering State Noise Adaptive Estimation”, Proceedings of Second IASTED Int. Conference in Telecom and Control, TELECON’85, Rio de Janeiro, Brasil, pp. 210-213.

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

Singhal,S. and Wu, L., 1989, “ Training Multilayer Perceptrons with the Extended Kalman Algorithm”, In Advances in Neural Information Processing Systems, VI, Morgan Kaufman Pub. Inc., pp. 136-140.

Watanabe, K., Fukuda, T. and Tzafestas, S.G.,1991, “ Learning Algorithms of Layered Neural Networks Via Extended Kalman Filters”, Int. J. System Science, 22(4), pp. 753-768.

Zurada,J.M., 1992,” Introduction to Artificial Neural System”, West Pub. Co.