NEURAL NETWORK AS A NEW APPROACH FOR DATA ASSIMILATION

A.G. Nowosad [a] (DCM), H.F. de Campos Velho [a] (LAC), A. Rios Neto [b]

[a] Instituto Nacional de Pesquisas Espaciais (INPE)
Caixa Postal 515
12201-970 -- São José dos Campos (SP), BRAZIL,
E-mail: alex@met.inpe.br - haroldo@lac.inpe.br

[b] Instituto de Pesquisa e Desenvolvimento (IPD)
Universidade do Vale do Paraíba (UNIVAP)
Av. Shishima Hifumi, 2.911 - Urbanova
12245-720 -- São José dos Campos (SP), BRAZIL
E-mail: atair@univap.br

Abstract

Multilayer Perceptron Neural Networks are tested as a new method for data assimilation in DYNAMO meteorological model. The approach "emulates" the Kalman Filter data assimilation method avoiding recalculation of the gain matrix at each instant of assimilation. A new prodedure for training the networks is also presented, based on a modification in the backpropagation algorithm. An Adaptive Extended Kalman Filter was used to provide examples for network training.

Keywords: Data assimilation, neural networks, Kalman filter, nonlinear dynamics.

1. Introduction

The data assimilation process can be described as a procedure that uses observational data to improve the prediction made by an inaccurate mathematical model. For example, suppose a computational model where many properties are only expressed approximately, like turbulent fluxes. Typically, the assimilation process can be outlined as a two step process:

$$\text{Forecast step: } w_n^f = F\left[w_{n-1}^a\right] \qquad \text{Analysis step: } w_n^a = w_n^f + d_n ;$$

where w_n represents model state variable at n-th time step, $F[.]$ is the mathematical (forecast) model, superscripts f and a denote forecasted and analyzed values respectively, and d_n is the inovation.

Several methods of data assimilation have been developed for air quality problems (Zanneti, 1990), numerical weather prediction (Daley, 1991), and numerical oceanic simulation (Bennet, 1992). In the case of atmospheric continuous data assimilation there are many deterministic and probabilistic methods (Daley, 1991; Todling, 1997). Deterministic methods include *Dynamic Relaxation*, *Variational Methods* and *Laplace Transform*, whereas probabilistic methods
include *Optimal Interpolation* and *Kalman Filtering*. Dynamic Relaxation assumes the prediction model to be perfect, as does Laplace Transform. Variational Methods and Optimal Interpolation can be regarded as minimum-mean-square estimation of the atmosphere. In Kalman Filtering the analysis inovation d_n is computed as a linear function of the misfit between observation (denoted with superscript o) and forecast:

$$d_n = G_n\left(w_n^o - H_n w_n^f\right)$$

where G_n is a weighting (gain) matrix, w_n^o is the with error observed value of w_n and H_n is an observation matrix. An adaptive extended Kalman filter has been tested in strongly nonlinear dynamical systems for assimilation procedure: the Lorenz chaotic system, and DYNAMO meteorological model - a simplified version of the shallow water

equations (Nowosad et al, 1999). Kalman Filtering has the advantage of minimizing the error in the assimilation (Nowosad et al, 1999) *plus* propagating itself the error from one data insertion to the next. But it can be computationally too expensive for large systems (Mendel, 1971).

The goal of the present study is to test in DYNAMO meteorological model a new method to compute an assimilation function, where such function is implemented by an artificial neural network (ANN): $w_n^a = F_{ANN}\left(w_n^f, w_n^o\right)$ (Nowosad et al, 2000). With this intention, the possibilities of using Neural Networks for data assimilation are shown. Specifically, it is the objective of this work to examine whether Multilayer Perceptron Neural Networks can "emulate" the accuracy of Kalman Filtering with economy in computer time.

The multilayer perceptron has been applied to a wide variety of tasks in atmospheric sciences (Gardner et al, 1998). Applications include prediction of air-quality prediction and severe weather, modeling nonlinear transfer functions and classification of atmospheric circulation patterns.

In the next sections the paper is organized as follows: in section 2 there is a brief exposition of artificial neural networks (ANN); in section 3 is described the nonlinear system in which the new approach is tested, the DYNAMO meteorological model; in section 4 are shown the numerical results obtained using Multilayer Perceptrons; and section
5 contains a summary of the results and some comments on the new method.

2. Multilayer Perceptrons

An artificial neural network (ANN) is an arrangement of units characterized by:
1. a large number of very simple neuron-like processing units;
2. a large number of weighted connections between the units, where the knowledge of a network is stored;
3. highly parallel, distributed control.

The processing element (unit) in an ANN is a linear combiner with multiple weighted inputs, followed by an activation function. There are several different architectures of ANN's, most of which directly depend on the learning strategy adopted. It is not the aim of the paper to present an overview on ANN. Instead, a brief description of the ANN used is focused: the multilayer Perceptron with backpropagation learning (Haykin, 1994).

The Multilayer Perceptron with backpropagation learning, also called the backpropagation neural network, is a feedforward network composed of an input layer, an output layer, and a number of hidden layers for extracting high order statistics from the input data (Haykin, 1994, page 19). In order to make the network more flexible to solve nonlinear problems, the activation functions for the hidden layer are sigmoid functions.

Mathematically, a perceptron network simply maps input vectors of real values onto output vector of real values. The connections have associated weights that are adjusted during learning process, thus changing the performance of the network.

2.1 Learning Process: Backpropagation Algorithms

There are two distinct phases in the usage of an ANN: the training phase (learning process) and the running phase (activation of the network). In the training phase, the weights are adjusted for the best performance of the network in establishing the mapping of many input-output vector pairs. Once trained, the weights are fixed and new inputs can be presented to the network for it to compute corresponding outputs, based on what it has learned.

The training phase of a multilayer perceptron is controlled by a supervised learning algorithm, which differs from unsupervised learning. The main difference is that the latter uses only information contained in the input data, whereas the former requires both input and output (desired) data, which permits the calculation of the error of the network as the difference between the calculated output and the desired vector. Adjustment of the network's weights is conducted
by backpropagating such error through the network. This adjustment is called *Backpropagation Algorithm*. The weight change rule is a development of the Perceptron learning rule. Weights are changed by an amount proportional to the error at that unit, times the output of the unit feeding into the weight. This is the essence of the so-called *delta rule*. The training phase in batch mode is nextly described in more detail.

Training in batch mode, which uses *all* examples at the same time, searches a set of weights q and biases m that minimizes the total squared error

$$e_m = \sum_{k=1}^{N} \left\| F_{ANN}(X_k, q, m, m) - F(X_k) \right\|_2$$

where m is the number of the iteration, N is the number of examples in the training set, X_k is the input vector of example k, q and m are the weights and biases of the network F_{ANN}, is the approximation and F is the desired output value.

Initialization of weights and biases can be done by randomly choosing them or by, for example, the *Nguyen-Widrow method* (Nguyen et al, 1990), which is the case here.The *Nguyen-Widrow method* linearizes the network and finds a linear interpolation initial approximation for the desired function.

In the following algorithm y_{jk} is the input to neuron j for each example k and Δ_{ik} is the gradient of the error at neuron i for example k. An adaptive backpropagation algorithm in batch mode proceeds like this (Demuth et al, 1994):

0. Start with learning rate h_0, momentum constant $a = 0.9$, and momentum $a_0 = 0$;
1. Calculate outputs of network and total e_m;
2. If $e_m < e$ stop;
3. At iteration m the backpropagation algorithm calculates:
   3.1 The error gradient of each neuron i for each example k: Δ_{ik};
   3.2 New tentative weights and biases using, for each neuron i, its inputs j and each example k
      3.2.1 $\Delta(q\,y)_{ij} = \sum_k \Delta_{ik} y_{jk}$
      3.2.2 $\Delta q'_{ij}(m) = a_m \cdot \Delta q_{ij}(m-1) + (1-a_m) \cdot h_m \cdot \Delta(q\,y)_{ij}$
      3.2.3 $\Delta m'_i(m) = a_m \cdot \Delta m_i(m-1) + (1-a_m) \cdot h_m \cdot \left(\sum_k \Delta_{ik} \right)$;
4. If $e_m > 1.04 \cdot e_{m-1}$ then $h_{m+1} = 0.7 \cdot h_m$ and $a_{m+1} = 0$, go to 1;
- Else
   4.1 if $e_m < e_{m-1}$ then $h_{m+1} = 1.05 \cdot h_m$ and $a_{m+1} = a$;
   4.2 Accept tentative weights and biases $q_{ij} = q'_{ij}$ and $m_i = m'_i$
   4.3 go to 1.

2.2 A Modification of the Adaptive Backpropagation Algorithm

A modification of the Adaptive Backpropagation Algorithm is proposed. In step **3.2.3** of the algorithm the formula for calculating $\Delta m'_i$ is changed to

$$\Delta m'_i(m) = a_m \cdot \Delta m_i(m-1) + (1-a_m) \cdot h_m^2 \cdot \left(\sum_k \Delta_{ik} \right)$$

This modification tends to make the rate of change in biases m slower than the rate of change in weights q, because usually $h < 1$. In the numerical results (section 4) this change lead us to obtain a good assimilation function.

3 DYNAMO Meteorologic Model

The assimilation scheme presented in section 2 will be tested in the DYNAMO model for the atmosphere. The one-dimensional computational model based on shallow-water approach was derived by P. Lynch (Lynch, 1984) to simulate large-scale (synoptic scale) atmospheric movements. Model DYNAMO assumes periodic boundary conditions for all perturbed dependent variables, constant density □ and that the horizontal scale of movements is much greater than the vertical scale.

The physical equations of the model are:

$$\frac{du}{dt} - fv + \frac{\partial f}{\partial x} = 0$$
$$\frac{dv}{dt} + fu + \frac{\partial f}{\partial y} = 0$$
$$\frac{\partial f}{\partial t} + f\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) = 0$$

where u is longitudinal speed (x-coordinate), v is zonal speed (y-coordinate) and $f = \int_{h_0}^{h_s} g dz$ is geopotencial height.

Using perturbation method these equations are manipulated to produce a one-dimensional version of the shallow-water equations whose prognostic variables are $z = \partial v / \partial x$ and $d = \partial u / \partial x$, given by the curl and by the divergence of the motion equations, respectively.

After transforming the resulting equations to a nondimensional form and applying finite difference operators, the equations of the computational method are obtained:

$$\frac{dW}{dt} + AW = -\left[1/2\ B\ X + R_O\ N_W(W, X)\right]$$

where

$$W = M\ X$$
$$W = \left[z_1 \quad d_1 \quad f_1 \ldots z_{N_x} \quad d_{N_x} \quad f_{N_x}\right]^T$$
$$X = \left[v_{-1/2} \quad u_{-1/2} \quad f_1 \ldots v_{N_x - 1/2} \quad v_{N_x - 1/2} \quad f_{N_x}\right]^T$$
$$N_W(W, X) = -\frac{1}{2\Delta x}\ C\ D(X)\ R\ W$$
$$N_x = 20 \quad ,$$

and $R_O = 10^{-1}$ is the Rossby number, $f = f_0 + b\,y$ with $f_0 = 10^{-4}$ and $b = 1.6 \times 10^{-11}$ is coriolis force for b-plane approximation. All matrices appearing in the equation above are shown in (Campos Velho, 1997).

4 Numerical Results

The performance of the assimilation method is now examined. The test is made using the DYNAMO model quoted in section 3.

It is necessary explain which data will be used to test the assimilation methods. Observations of the real atmosphere will be simulated by a vector

$$Z_n = W_n + u_n$$

where u has as components zero-mean gaussian white noises artificially generated and whose magnitude is small (0.1%) compared to the magnitude of W, representing the disturbances in the data assimilation process. The disturbances are generated as variations of u, v and f transformed into variations of z, d and f through $u = \Delta W = M\,\Delta X$.

The spatial profile of the disturbance in each component of the vector is given by

$$\Delta f(x) = (1 - \cos(x')), \quad \Delta V(x) = \frac{d\Delta f(x')}{dx}, \quad \Delta U(x) = -\Delta V(x),$$

where

$$x' = \begin{cases} 2p\ \dfrac{x}{L/6} - \dfrac{L}{2}, & x \in \left[\dfrac{L}{2} - \dfrac{L}{12}, \dfrac{L}{2} + \dfrac{L}{12}\right] \\ 0, & x \notin \left[\dfrac{L}{2} - \dfrac{L}{12}, \dfrac{L}{2} + \dfrac{L}{12}\right] \end{cases},$$

$L = 10 \quad m \quad .$

The result obtained inserting these disturbances *without filtering* in a simulation of the atmosphere using DYNAMO *without initialization* every $4\ s$ is illustrated through the evolution of u at the middle-point of the grid in figure 1. In this figure the dashed line represents the true atmosphere and the solid line the result of simulation with disturbance.

In the next step the Adaptive Kalman Filter described in (Nowosad et al, 1999) was used to provide the examples for a network composed of

- an input layer with inputs $U^o, V^o, f^o, U^f, V^f, f^f$;
- *one* hidden layer having 80 neurons with activation function $f(x) = \tanh(x)$;
- an output layer having 60 neurons with activation function $f(x) = x$ and outputs U^a, V^a, f^a.

Instead of assimilating the data at every $4\ s$ the filter received data at every $0.01\ s$. The network was trained by attaching to its input and output layers respectively the pairs (x_n^i, w_n^a) formed by the vectors

$$x_n^i = \left[\hat{U}_n^o \quad \hat{V}_n^o \quad \hat{f}_n^o \quad \hat{U}_n^f \quad \hat{V}_n^f \quad \hat{f}_n^f\right]^T, \quad w_n^a = \left[\hat{U}_n^a \quad \hat{V}_n^a \quad \hat{f}_n^a\right]^T \quad .$$

The vectors are normalized with scaled so that the input and output signals are of the same order of magnitude. Using as reference

$$U_{\max} = 0.05, \quad V_{\max} = 2.00, \quad f_{\max} = 1.50,$$

respectively the maximum absolute values of U_n^a, V_n^a and f_n^a given by the Kalman filter at the mid-grid point $i = N/2$ for all n, scaling factors

$$U^* = a \cdot U_{\max}, \quad V^* = a \cdot V_{\max}, \quad f^* = a \cdot f_{\max},$$

were defined using a determined experimentally for satisfactory result. In this case $a = 2.60$ and all signals were in the range $[-1,1]$. Thus

$$U^* = 0.1300, \qquad V^* = 5.200, \qquad f^* = 3.900,$$
$$\hat{U}_i^f(n) = U_i^f(n)/U^*, \quad \hat{V}_i^f(n) = V_i^f(n)/V^*, \quad \hat{f}_i^f(n) = f_i^f(n)/f^*,$$
$$\hat{U}_i^o(n) = U_n^o(n)/U^*, \quad \hat{V}_i^o(n) = V_i^o(n)/V^*, \quad \hat{f}_i^o(n) = f_i^o(n)/f^*,$$
$$\hat{U}_i^a(n) = U_n^a(n)/U^* \quad \hat{V}_i^a(n) = V_i^a(n)/V^*, \quad \hat{f}_i^a(n) = f_i^a(n)/f^*,$$

The network was trained with the adaptive backpropagation algorithm (section 2.2) using 100 examples of the Kalman filter taken at every $\Delta t = 0.4\ s$ and $h_0 = 10^{-4}$ until $e_m < 0.09$. This experiment will be denominated *experiment 1*. Attempts to train multilayer perceptrons with more hidden layers using the adaptive backpropagation algorithm were unsuccessful.

To test the new assimilation method one attaches to the input of the trained network the vectors

$$x_n^i = \left[U_n^o/U^* \quad V_n^o/V^* \quad f_n^o/f^* \quad U_n^f/U^* \quad V_n^f/V^* \quad f_n^f/f^*\right]^T$$

and reads at the output layer the vectors

$$\hat{w}_n^a = \left[\hat{w}_1^a(n) \quad \hat{w}_2^a(n) \quad \hat{w}_3^a(n)\right]^T .$$

U_n^f, V_n^f and f_n^f are predicted by DYNAMO and U_n^o, V_n^o and f_n^o are observations. The assimilation output will actually be

$$\tilde{U}_n^a = U^* \cdot \hat{w}_1^a(n), \qquad \tilde{V}_n^a = V^* \cdot \hat{w}_2^a(n), \quad \tilde{f}_n^a = f^* \cdot \hat{w}_3^a(n) \quad .$$

The results inserting data at each $\Delta t = 4\ s$ can be seen in figure 2. The dashed lines represent the true signal, the solid lines the predicted signal. The assimilation was acceptable.

After this first result the modified backpropagation algorithm (section 2.2) was tested on a perceptron with *two* hidden layers. The AKF was used to provide the examples for a network composed now of

- an input layer with inputs U^o, V^o, f^o, U^f, V^f, f^f;
- a *first* hidden layer having 50 neurons with activation function $f(x) = \tanh(x)$;
- a *second* hidden layer having 50 neurons with activation function $f(x) = \tanh(x)$;
- an output layer having 60 neurons with activation $f(x) = x$ and outputs U^a, V^a, f^a.

As in experiment 1 vectors were normalized with scale using $a = 1.00$. Thus

$$U^* = 0.05, \quad V^* = 2.00, \quad f^* = 1.50,$$

From here on the normalization (with scale) and unnormalization follow the same rule as previously explained.

The network was trained with the modified adaptive backpropagation algorithm (section 2.2) using 500 examples of the AKF taken at every $\Delta t = 0.08\ s$ and $h_0 = 10^{-3}$ until $e_m < 0.6$. The following restriction had to be made at the output layer (see step **3.2.1** of the algorithm) to ensure numerical accuracy:

$$\Delta(q\ y)' = \sum \Delta_{ik}\, y_{jk}$$

$$\Delta(q\ y)_{ij} = \begin{cases} \Delta(q\ y)'_{ij}\ , & when \quad \left|\Delta(q\ y)'_{ij}\right| \geq 10^{-6} \\ 0, & otherwise \end{cases}$$

This experiment will be denominated *experiment 2*.

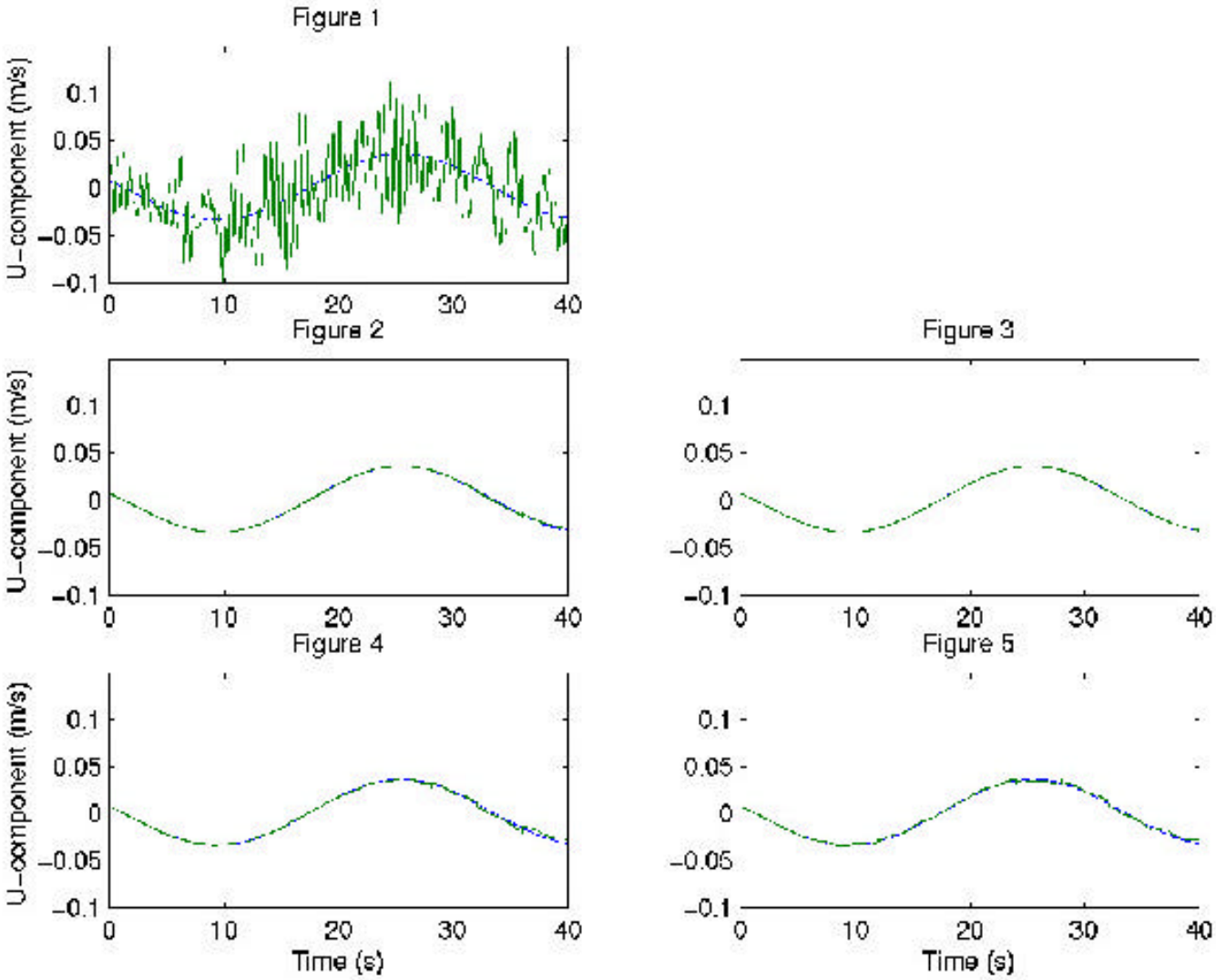
The results inserting data at each $\Delta t = 4\ s$ can be seen in figure 3. The dashed lines represent the true signal, the solid lines the predicted signal. The assimilation was very good.

Finally a robustness test was made with the assimilation method. The parameters R_O, R_F e R_b of the model for the *true* signal ("the atmosphere") were altered using zero-mean white gaussian fluctuations

$$R'_O(n) = [1 + m_O(n)] \cdot R_O\ , \quad R'_F(n) = [1 + m_F(n)] \cdot R_F\ , \quad R'_b(n) = [1 + m_b(n)] \cdot R_b\ ,$$
$$E\{m_O(n)\} = E\{m_F(n)\} = E\{m_b(n)\} = 0$$
$$E\{m_O^2(n)\} = E\{m_F^2(n)\} = E\{m_b^2(n)\} = 1.6 \times 10^{-3}\ ,$$

where $E\{\cdot\}$ is temporal expected value, $R_F = 10^2$, $R_b = 1.6 \times 10^{-1}$, and both true signal and the initial condition of the filter's model are initialized. This test is the closest to operational data assimilation, because the atmosphere is not truly a dynamic system of time-invariant parameters as assumed in DYNAMO model and, on the other side, the initial condition of the assimilating system will usually be initialized.

The results obtained at the mid-grid point are shown in figures 4 and 5. The Neural Network assimilation with one hidden layer and adaptive backpropagation algorithm had discontinuity problems at some instants of data insertion, as seen in figure 4. After $t \geq 20\ s$ there was significant problem, but only after $t \geq 30\ s$ did it start becoming serious. The Neural Network assimilation with two hidden layers and modified adaptive backpropagation algorithm also had discontinuity problems at some instants of data insertion, as seen in figure 5. For $8 \leq t \leq 17\ s$ and then after $t \geq 24\ s$ there was significant problem, but only after $t \geq 35\ s$ did it start becoming serious. But in all cases the errors were much smaller than those of figure 1.



5 Conclusions

Multilayer Perceptron Neural Networks are used for data assimilation in a nonlinear dynamic system, the shallow-water model for the atmosphere DYNAMO.This aproach emulated Kalman Filter data assimilation methods avoiding recalculation of the gain matrix at each instant of assimilation.

Considering computational cost, suppose that a network with $m_0 = 2 \cdot m_3$ inputs, L hidden layers of m_1 neurons and an output layer with m_3 neurons has been trained to emulate a Kalman Filter. The assimilation function implemented by the network has complexity of order (Nowosad et al, 2000)

$$m_1 \cdot O(m_0) + L \cdot m_1 \cdot O(m_1) + m_3 \cdot O(m_1).$$

If $O(m_1) = O(m_3)$ and $L << m_3$ then the algorithm to calculate the output of the assimilation function $w_n^a = F_{ANN}(w_n^f, w_n^o)$ will have complexity $O(m_3^2)$.

On the other hand, the complexity of a standard Kalman Filter with m_3 state variables and m_3 observables is $O(m_3^3)$ due to the matrix products at every step (Mendel, 1971).

Modification of an adaptive backpropagation algorithm for training multilayer perceptrons was also proposed. This modification consisted in practically decreasing the speed of adjustment of biases by using the square of the learning rate in place of the learning rate itself.

Two tests were made of data assimilation using this new approach in DYNAMO, a numeric model for the atmosphere using the shallow-water approach. Firstly two multilayer perceptrons with the backpropagation training algorithms were tried. Using straightlyforwardly DYNAMO to simulate a stationnary atmosphere the multilayer perceptron with one hidden layer and trained using adaptive backpropagation training algorithm performed acceptably. The multilayer perceptron with two hidden layers trained using modified adaptive backpropagation algorithm performed well.

Secondly robustness was evaluated in networks altering the programming code of DYNAMO to simulate a nonstationnary atmosphere, because it is a real issue in Numerical Weather Prediction. When the atmosphere was nonstationary the two multilayer perceptrons performed acceptably.

Finally, it should be pointed out that Kalman filters provided the training sets for the networks, but other assimilation methods can also be used to train multilayer perceptrons.

6 Acknowledgements

This research has been partially supported by a FAPESP grant.

REFERENCES

A.F. Bennet (1992): *Inverse Methods in Physical Oceanography*, Cambridge University Press.

H.F. Campos Velho, J.C.R. Claeyssen (1997): "A Comprehensive Analysis of a Barotropic Limited Area Model Using the Nonmodal Matrix Technique", *Brazilian Journal of Meteorology*, **12**(2), pp. 41-50.

R. Daley (1991): *Atmospheric Data Analysis*, Cambridge University Press, Cambridge, EUA.

H. Demuth, M. Beale (1994): *Neural Network Toolbox User's Guide (For Use with MATLAB)*, The MathWorks, Inc.,MA, USA.

M. W. Gardner, S. R. Dorling (1998): "Artificial Neural Networks (The Multilayer Perceptron) - A Review of Applications in the Atmospheric Sciences", *Atmospheric Environment*, 32(14/15), pp. 2627-2636.

S. Haykin (1994): *Neural Networks: A Comprehensive Foubdation*, Macmillan, New York.

P. Lynch (1984): *DYNAMO: A One-Dimensional Primitive Equation Model*, Technical Note 44, Irish Meteorological Service, Ireland.

J. M. Mendel (1971): "Computational Requirements for a Discrete Kalman Filter", *IEEE Transactions on Automatic Control*, AC-16 (6), pp. 748-758.

D. Nguyen, B. Widrow (1990): "Improving the Learning Speed of 2-Layer Neural Networks by Choosing Initial Values of the Adaptive Weights", *International Joint Conference on Neural Networks*, San Diego, California, June 17-21, 1990, v3, IEEE Neural Networks Council.

A.G. Nowosad, A. Rios Neto, H.F. Campos Velho (1999): "Data Assimilation Using an Adaptative Kalman Filter and Laplace Transform", *Workshop on Physics of the Planetary Boundary Layer and Dispersion Process Modelling*}, 23-26 November, Santa Maria (RS), Brasil - Proceedings will be published as special issue of the journal Hybrid Methods in Engineering.

A. G. Nowosad, A. Rios Neto, H. F. Campos Velho (2000): "Data Assimilation in Chaotic Dynamics Using Neural Networks", accepted for presentation at International Conference on Nonlinear Dynamics, Chaos, Control and Their Applications in Engineering Sciences - ICONNE 2000, at Campos do Jordão, SP, Brasil.

R. Todling (1997): *Estimation Theory and Foundations of Data Assimilation*, Course notes, National Laboratory of Scientific Computation (LNCC), 22-29 September, Rio de Janeiro (RJ), Brasil.

P. Zannetti (1990): *Air Pollution Modeling*, Computational Mechanics Publications, UK.