Revista Brasileira de Meteorologia, v.20, n.3, 411-420, 2005

RECURRENT AND FEEDFORWARD NEURAL NETWORKS TRAINED WITH CROSS VALIDATION SCHEME APPLIED TO THE DATA ASSIMILATION IN CHAOTIC DYNAMICS

FABRÍCIO PEREIRA HÄRTER e HAROLDO FRAGA DE CAMPOS VELHO

Instituto Nacional de Pesquisas Espaciais (INPE)
Av. dos Astronautas, 1.758 - Jd. Granja - CEP 12227-010 - São José dos Campos/SP - Brasil - Fone: (12) 3945-6547
E-mail: fabricio@inmet.gov.br, haroldo@lac.inpe.br

Recebido Agosto 2004 - Aceito Julho 2005

ABSTRACT

Artificial Neural network (ANN) is a new approach for data assimilation process. The performance of two feedforward (multilayer perceptron and radial basis function), and two recurrent (Elman and Jordan) ANNs are analyzed. The Lorenz system under chaotic regime is used as a test problem. These four ANNs were trained for emulating a Kalman filter using cross validation scheme. Multilayer Perceptron and Elman ANNs show better results among ANNs tested. The results obtained encouraging the application of the ANNs as an assimilation technique.
Keywords: cross validation, data assimilation, Lorenz system, recurrent neural networks.

RESUMO: REDES NEURAIS RECORRENTES E *FEEDFORWARD* TREINADAS COM CORRELAÇÃO CRUZADA APLICADAS A ASSIMILAÇÃO DE DADOS EM DINÂMICA NÃO-LINEAR

Rede Neural Artificial é uma nova abordagem para assimilação de dados. Neste artigo é analisado o desempenho de duas redes feedforward (perceptron de múltiplas camadas e função de base radial) e duas redes recorrentes (Elman e Jordan). O sistema de Lorenz sob regime caótico é usado como um problema teste. As redes foram treinadas para emular um filtro de Kalman, usando a técnica de validação cruzada. O perceptron de múltiplas camadas e a rede de Elman apresentaram melhor desempenho entre as redes testadas. Os resultados encorajam a investigação de redes neurais como uma técnica para assimilação de dados em previsão de tempo operacional.
Palavras-chave: Assimilação de dados, redes neurais recorrentes, sistema de Lorenz, validação cruzada.

1. INTRODUCTION

After 1950 the numerical weather prediction (NWP) became an operational procedure. Essentially, NWP consists on the time integration of the Navier-Stokes equation using numerical procedures. Therefore, after some time-steps, there is a disagreement due to small uncertainties in the specification of the Initial Conditions (IC). In other words, sensitive dependence on the IC causes the forecasting error to grow exponentially fast the integration time, (Grebogi et al., 1987). NWP is an initial value problem, this imply that a better representation for the IC will produce a better prediction. The problem for estimating the initial condition is so complex and important that it becomes a science called Data Assimilation, (Kalnay, 2003).

The insertion of the noise observational data into an inaccurate computer model does not allow a good prediction (see Figure 1). It is necessary to apply some data assimilation technique.

Many methods have been developed for data assimilation (Daley, 1991). They have different strategies to combine the forecasting (background) and observations. From mathematical point of view, the assimilation process can be represented by

$$x^a = x^f + W\,p[y^o - H(x^f)] \qquad (1)$$

where x_a is the value of the analysis; x_f is the forecasting (from the mathematical model); W is the weighting matrix, generally computed from the covariance matrix of the prediction errors from forecasting and observation; yo denotes the observation; H represents the observation system; $\{y^o - H(x^f)\}$ is the inovation; and p[.] is a discrepancy function.

The use of ANN for data assimilation is a very recent issue. ANNs were suggested as a possible technique for data assimilation by Hsieh and Tang (1998), but the first implementation of the ANN as a new approach for data assimilation was employed by Nowosad et al. (2000) (see also Vijaykumar et al. (2002), Campos Velho et al. (2002)). The ANN has also be used in the works of Liaqat et al. (2003) and Tang and Hsieh (2001) for data assimilation. The technique developed by Nowosad et al. (2000) is quite different from the latter two works, where the ANN is used to represent unknown equation in the mathematical system equations of the model.

Differently from previous works using ANN for data assimilation, this paper deals with recurrent NN. In addition, the cross validation is used as learning process. The Lorenz system under chaotic dynamics is applied as test model for assimilating noise data. Four ANNs are considered: Multi-Layer Perceptron (MLP), Radial Base Function (RBF), Elman Neural Network (E-NN), and Jordan Neural Network (J-NN). The E-NN and J-NN are known as recurrent or time-delay ANNs.

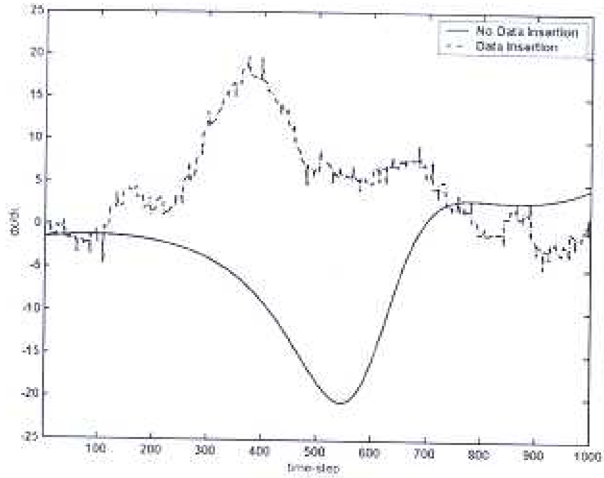


Figure 1: Shock in the numerical model after data insertion, without assimilation technique.

2. METHODOLOGY

This Section presents the model used for testing the assimilation schemes, a brief overview on Kalman filter, and a description of the ANNs employed in this work.

2.1. Testing Model: Lorenz System

Lorenz (1963) was looking for the periodic solutions of the Saltzman's model (Saltzman, 1962), considering a spectral Fourier decomposition and taking into account only low order terms. Lorenz obtained the following system of nonlinear coupled ordinary differential equations

$$dX/d\tau = -\sigma\,(X - Y) \tag{2}$$

$$dY/d\tau = rX - Y - XZ \tag{3}$$

$$dZ/d\tau = XY - bZ \tag{4}$$

where $\tau \equiv \pi^2 H^{-2}(1+a^2)\kappa\, t$ is the nondimensional time, being H, a, κ and t respectively the layer height, thermal conductivity, wave number (diameter of the Rayleigh-Bérnard cell), and time; $\sigma \equiv \kappa^{-1}\nu$ is the Prandtl number (ν is the kinematic viscosity); $b \equiv 4(1+a^2)^{-1}$. The parameter $r = R/R_c \propto \Delta T$ is the Rayleigh number (T is the temperature), and R_c is the critical Rayleigh number.

2.2. Kalman Filter

Starting from a prediction model (subscripts n denotes discrete time-step, and superscripts f represents the forecasting value) and an observation system:

$$w^f_{n+1} = F_n w^f_n + \mu_n \tag{5}$$

$$z_n = H w^f_n + \nu_n \tag{6}$$

where F^f_n is our mathematical model, μ_n is the stochastic forcing (modeling noise error). The observation system is modeled by matrix H, and ν_n is the noise associated to the observation. The typical gaussianity, zero-mean and ortogonality hypotheses for the noises are adopted. The state vector is defined as $w_{n+1} = [X_{n+1}, Y_{n+1}, Z_{n+1}]^T$ and it is estimated through the recursion

$$w^a_{n+1} = (I - G_{n+1} H_{n+1}) F_n w^a_n + G_{n+1} z^f_{n+1} \tag{7}$$

where w^a_{n+1} is the analysis value, G_n is the Kalman gain, computed from the minimization of the estimation error variance J_n+1, (Jaswinski, 1970)

$$J_{n+1} = E\{(w^a_{n+1} - w^f_{n+1})^T (w^a_{n+1} - w^f_{n+1})\} \tag{8}$$

being E{.} the expected value. The algorithm of the Linear Kalman Filter (LKF) is shown in Figure 2, where Q_n is the covariance of μ_n and R_n is the covariance of ν_n.

The assimilation is done through the sampling:

$$r_{n+1} \equiv z_{n+1} - z^f_{n+1} = z_{n+1} - H_n w^f_{n+1} \tag{9}$$

2.3. Artificial Neural Networks

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

The processing element (unit) in an ANN has 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. Two distinct phases can be devised while using an ANN: the training phase (learning process) and the run phase (activation of the network). The training phase consists of adjusting the weights 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 network for which it calculates the corresponding outputs, based on what it had learned. The back-propagation algorithm is used as the learning process for MLP studied in this paper, (Haykin, 2001).

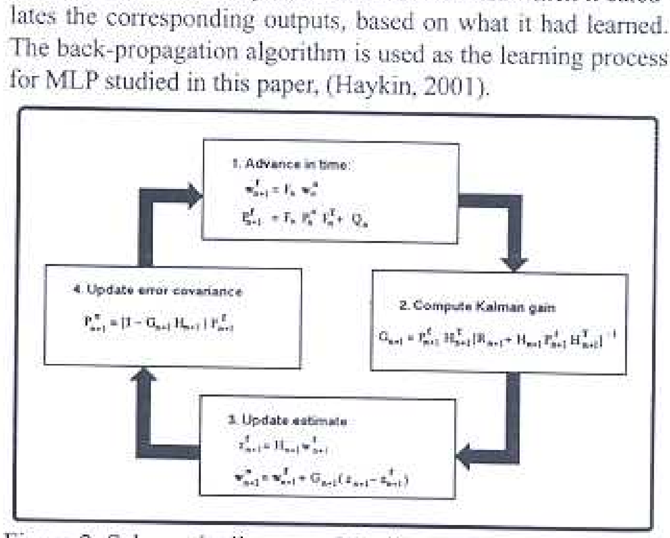


Figure 2: Schematic diagram of the linear Kalman filter.

The k-th neuron can be described by the two coupled equations

$$u_k = \sum_{j=1}^{m} \theta_{kj} w_j, \qquad (10)$$

$$y_k = \varphi\ (u_k + b_k) \qquad (11)$$

where w1, ..., wn are the inputs; θ_{k1}, ..., θ_{km} are weights for the neuron-*k*, u_k is the linear output of the linear combination among weighted inputs, b_k is the bias; $\varphi(.)$ is the activation function, and y_k is the neuron output.

Multilayer perceptrons with backpropagation learning algorithm, commonly referred to as backpropagation neural networks are feedforward networks composed of an input layer, an output layer, and a number of hidden layers, whose aim is to extract high order statistics from the input data (Haykin, 2001). Figure 3(a)-(c) depicts a backpropagation neural network, one hidden layer, and the activation functions.

RBF networks are feedforward networks with only one hidden layer. They have been developed for data interpolation in multidimensional space. RBF nets can also learn arbitrary mappings. The primary difference between a backpropagation with one hidden layer and an RBF network is in the hidden layer units. RBF hidden layer units have a receptive field, which has a center, that is, a particular input value at which they have a maximal output. Their output tails off as the input moves away from this point. The most used function in an RBF network is a Gaussian distribution – Figure 3(c).

Two recurrents ANNs are also investigated, Elman and Jordan ANNs (Braga et al., 2000). Beyond of the standard units, such as input/output and hidden layers, recurrent ANNs present the context units. Input and output units are in contact with the external environment, while the context units are not. Input units are just buffer units, they do not change the inputs. The hidden layers present activation functions, alternating the inputs and producing the outputs. Context units introduce a memory in the system, keeping the previous outputs as additional inputs (recurrent).

For the E-NN, the recurrent neurons is done connecting the hidden layer with the input layer -see Figure 4(a). Similar topology is employed to the J-NN, Figure 4(b), but the output values are used in the recurrent.

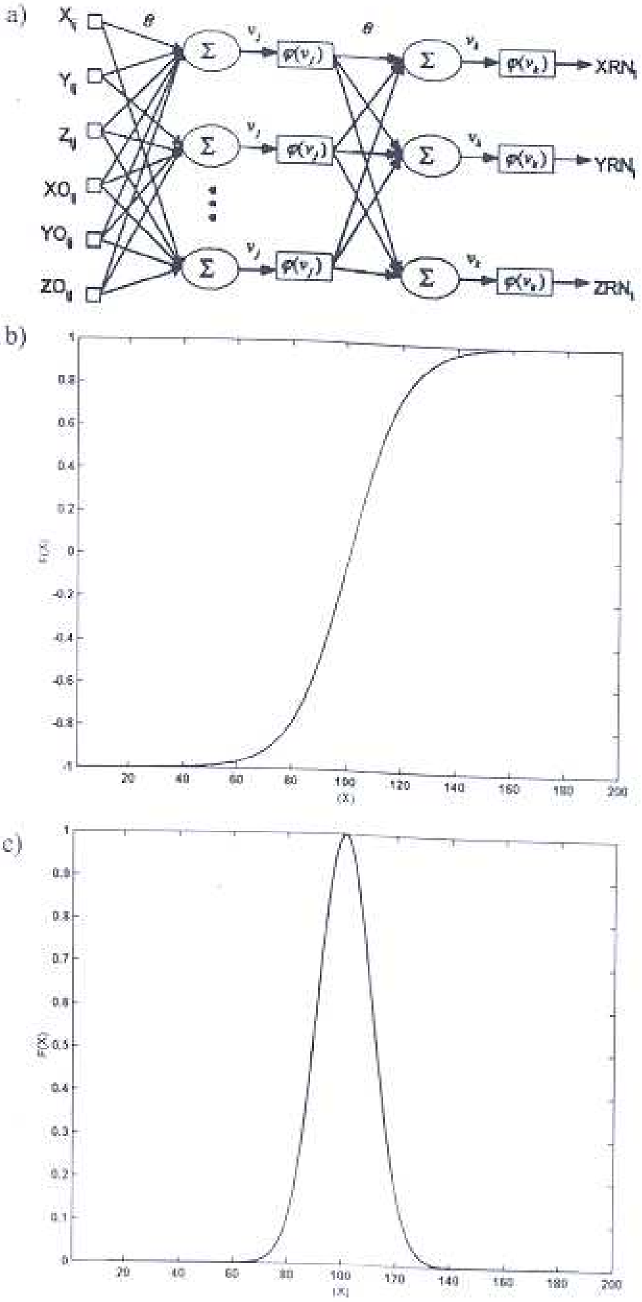


Figure 3: (a) Outline for neural network; activation functions (b) for the MLP: tanh-1; (c) for the RBF: Gaussian distribution.

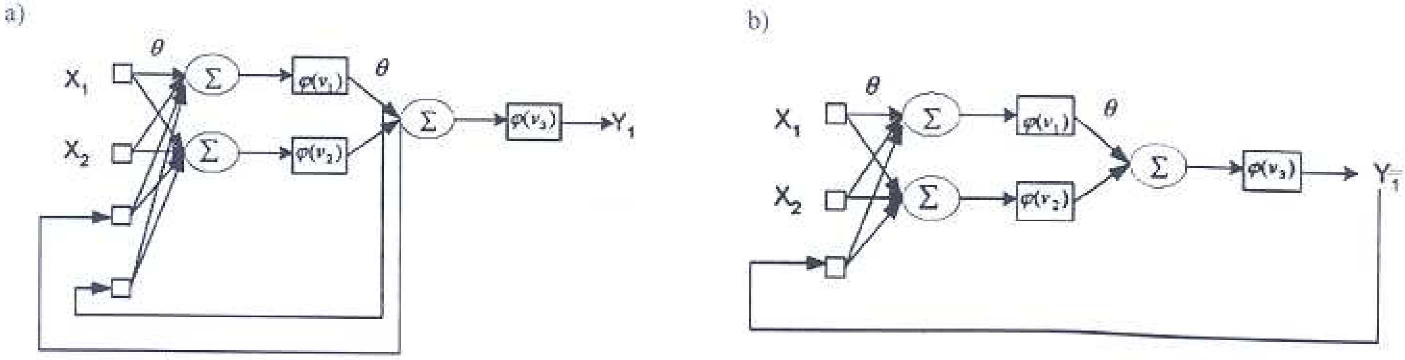


Figure 4: Outline for recurrent ANNs: (a) Elman; (b) Jordan.

3. NUMERICAL EXPERIMENTS

Our numerical experiments are performed using the following features for Kalman filter: $Q_n=0.1I$, $R_n=2I$,

$$P_0^f = \begin{cases} 10(w_0^f)_i^2 & for \quad i=j \\ 0 & for \quad i \neq j \end{cases}$$

The Lorenz system was integrated using a first order predictor-corrector scheme, with $\Delta t=10^{-3}$. The data insertion is done at each 12 time-steps. For training data set, 2000 data are considered, and 333 data are used for cross validation. After the training, the model is integrated for 106 time-steps.

In order to test different architectures of ANNs, for emulating a Kalman filter in data assimilation, one uses momentum constant for hidden layer α_h and α_n ($n= X, Y, Z$) for the output layer. Similar feature is used for learning rates: η_h for hidden layer, and η_n ($n= X, Y, Z$) for the output layer. The numerical values for these parameters are shown in Table 1.

Neural networks with two hidden layers were also considered. The parameters employed for this topology is presented in Table 2.

	MLP	RBF	E-NN	J-NN
neurons	2	40	2	2
α_h	0.6	---	0.51	0.4
α_X	0.6	---	---	0.4
α_Y	0.6	---	---	0.4
α_Z	0.6	---	---	0.4
η_h	0.001	0.0001	0.00001	0.0001
η_X	0.001	0.0001	0.00001	0.0001
η_Y	0.001	0.0001	0.00001	0.0001
η_Z	0.001	0.0001	0.00001	0.001

Table 1: Neural network parameters: one hidden layer.

	MLP (L1,L2)	RBF (L1,L2)	J-NN (L1,L2)
neurons	(2,6)	(11,2)	(6,2)
α_h	0.6	---	0.4
α_X	0.6	---	0.4
α_Y	0.6	---	0.4
α_Z	0.6	---	0.4
η_h	0.001	0.0001	0.0001
η_X	0.001	0.0001	0.0001
η_Y	0.001	0.0001	0.0001
η_Z	0.001	0.0001	0.001

Table 2: Neural network parameters: two hidden layers.

During the learning process, the cost function (the square difference between the ANN output and the target data) is decreasing, but this does not mean that the ANN will have an effective generalization. On the other hand, the minimization can fall in a local minimum of the error surface. Appropriate momentum constant and learning rate can avoid these local minimum, and the cross validation is an alternative to choose a better set of connection weights implying in a better generalization.

Cross validation consists of splitting the target data set into two sub-sets, one for training and another one for validation. For each iteration, the connection weights are tested with the second data set to evaluate the ANN skill for generalization. For the training phase, the first data set containing 2000 pairs of input/output vector is considered (observation and forecasting vector in input layer and KF output as a target vector). After training, cross validation is performed with the vector with the second data set of 333 vector pairs. The data sets for the cross validation phase are obtained from the previous simulation using KF for data assimilation every 12 time-steps. After the choice of the best weight set, the Lorenz system is integrated by 106 time-steps considering data assimilation at each 12 time-steps. The training (for each epoch), cross validation (for each epoch), and estimation errors (for each time-step) are computed as following (n = X, Y, Z and average):

$$EMQ_n = \frac{1}{2000}\sum_{i=1}^{2000}\frac{1}{2}(X_i^{KF} - X_i^{NN})^2; \quad (12)$$

$$EA_n = \frac{1}{333}\sum_{j=1}^{333}\sqrt{(X_j^{KF} - X_j^{NN})^2}; \quad (13)$$

$$EE_n = \frac{1}{10^3}\sum_{k=1}^{10^3}\sqrt{(X_k^{Obs} - X_k^{NN})^2}; \quad (14)$$

Results show below are obtained with only one hidden layer. However, some experiments were also performed considering two hidden layers for MLP, FBR, and J-NN, looking for a better performance. However, the results are pretty similar to those obtained with one hidden layer.

Therefore, results with two hidden layers will not be commented.

3.1. Assimilation: Multi-layer Perceptron (MLP)

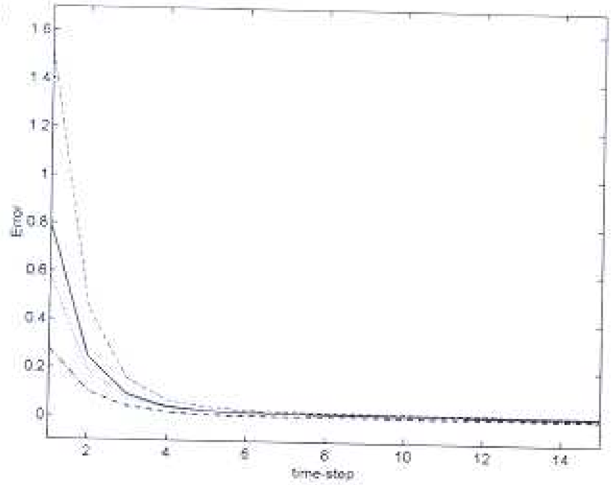
Figures 5(a) shows the training error curves obtained during the learning phase, and 5(b) displays the generalization error. The mean error ([error-X + error-Y + error-Z]/3) is also shown. After the choice of the best weight set, the Lorenz system is integrated considering data assimilation at each 12 time-steps.

Two neurons are used for MLP in the hidden layer, and the learning ratio are described in Table 1, reaching a minimum EA average error at 4.106 for the 144-th epoch. The learning error dramatically decreases for the first four iterations, and it continues decreasing slowing for the rest of iteration process - see Figure 5(a). This shows how the cross validation works, selecting the best weight set for activation (or generalization) of the ANN.

Figures 6(a)-(c) depicted the last 103 time-steps of the integration, where the data assimilation is performed by the MLP with one hidden layer. In these figures, the blue line represents the observation and the red line represents the MLP. Clearly, the MLP is effective to carry out the assimilation.

3.2. Assimilation: Radial Base Function (RBF)

The same experiment applied to the MLP is carried out for the RBF, with one hidden layer. The iteration errors are shown in Figures 7(a)-(b) and forecasting performance in Figures 8(a)-(c). The best architecture of this ANN is obtained with 40 neurons in the hidden layer, using learning ratio listed in Table 1, without momentum constant. The smallest activation error was obtained at the first epoch. The training error is very small at second and twelve epochs, but the estimation error for the cross validation data set grows exponentially, except for the component-Y.

a)


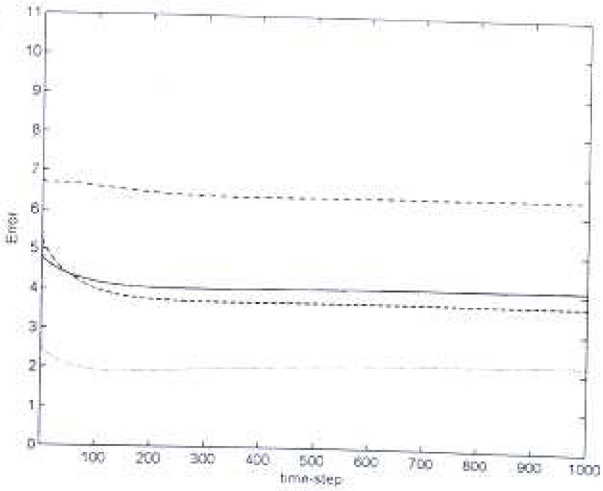
b)


Figure 5: Error for Lorenz system, solid (mean error), dashed (error X), dashdot (error Y), dotted (error Z): (a) during the training phase; (b) for the cross validation.

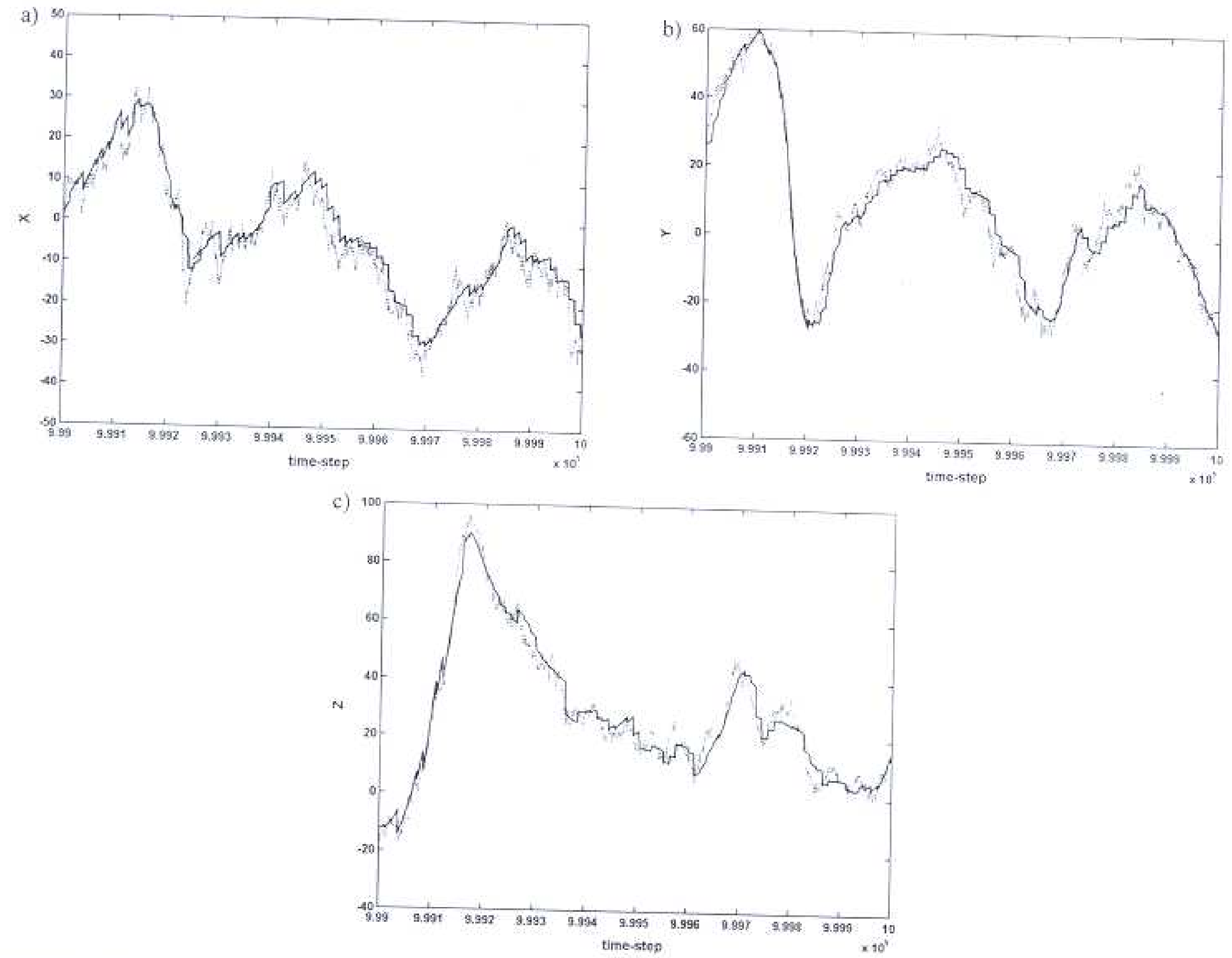


Figure 6: Data assimilation for the Lorenz system using MLP, solid (MLP), dotted (observation), components: (a) - X, (b) - Y, (c) - Z.

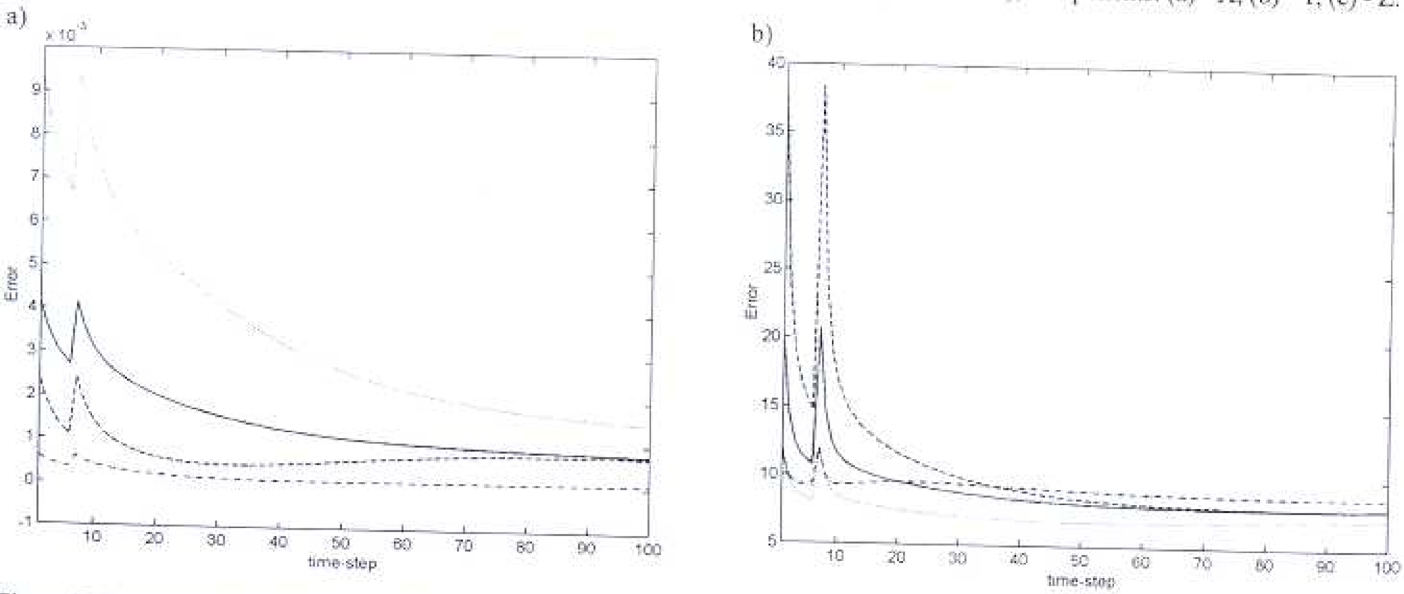


Figure 7: Error for Lorenz system, solid (mean error), dashed (error X), dashdot (error Y), dotted (error Z): (a) during the training phase; (b) for the cross validation.

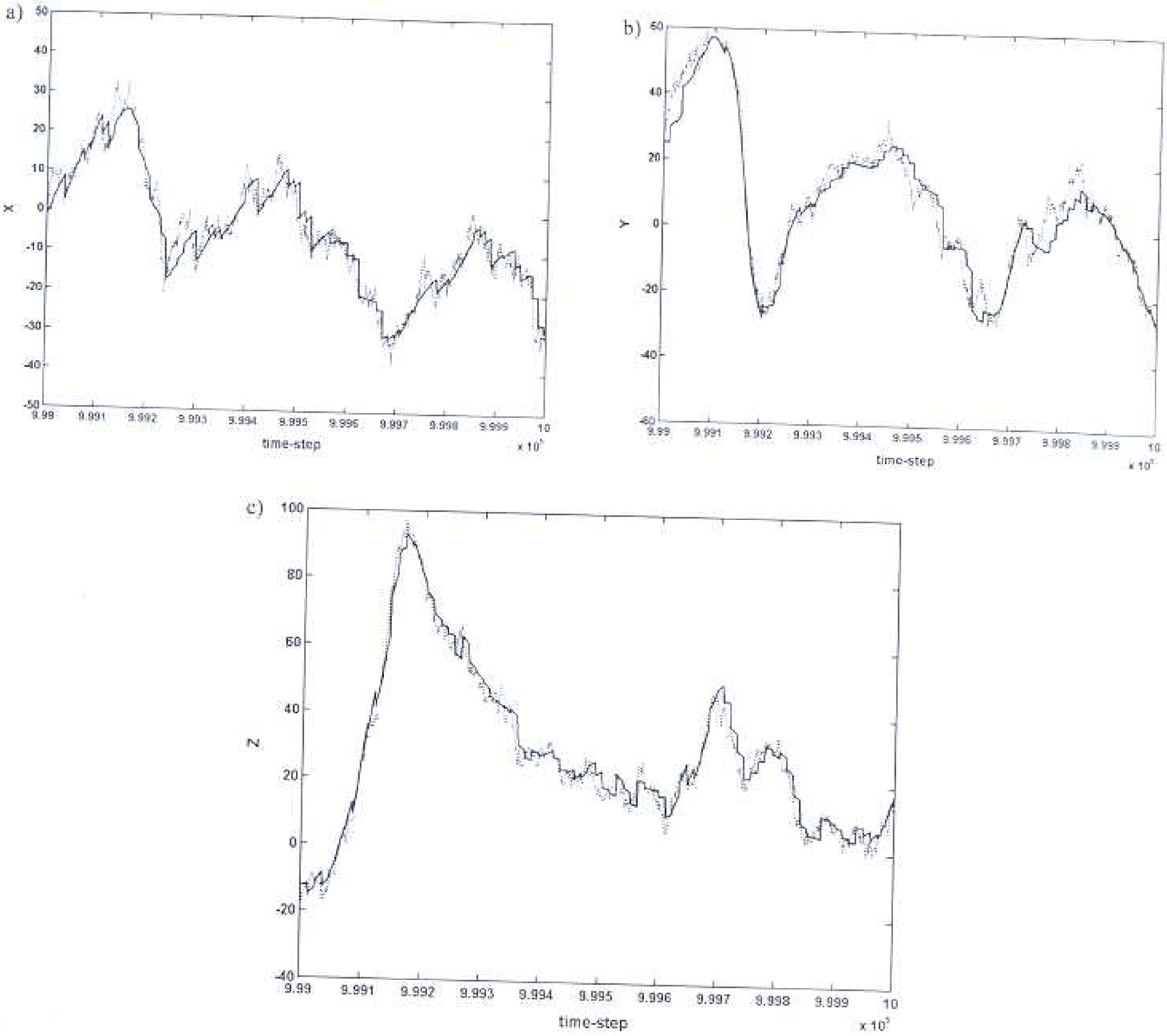


Figure 8: Data assimilation for the Lorenz system using RBF, solid (RBF), dotted (observation), components: (a) - X, (b) - Y, (c) - Z.

3.3. Assimilation: Recurrents NNs

Here, the results for E-NN (recurrency from the hidden layer to input layer) and J-NN (recurrency from the output to input layer). The idea to investigate the use of recurrent ANNs is to verify if the imbedding a memory in the ANN could improve the assimilation for a longer period of the time integration.

Figures 9(a)-(b) plot the error computed from the training data set for the E-NN. In the Figures 10(a)-(c) the last 1000 time-steps of the whole integration (with 106 time-steps) is shown for this ANN.

Two neurons are used in the hidden layer, with learning ratio and momentum constant as given in Table 1. The average error EAaverage=4.091 is reached at the second weight set from the training set. From Figure 9(a), one can be noted that the training error decreases abruptly after the second epoch.

The E-NN also produces a good assimilation, as seen in Figures 10(a)-(c), giving EEX=3.840, EE_Y=3.469, EE_Z=3.612 and EEaverage=3.612 for components X, Y, Z, and the average error, respectively.

The assimilation with J-NN is efficient too, and there is no big difference related to other ANNs - see Figures 12(a)-(c). The errors for the X, Y, Z, and the average are EE_X=2.134, EE_Y=4.609, EE_Z=2.388, EEaverage=3.044 Figures 11(a)-(b) show that the training errors decrease fast for the first epochs, while cross validation errors increase. For this ANN, the best answer from the J-NN is obtained at the first epoch.

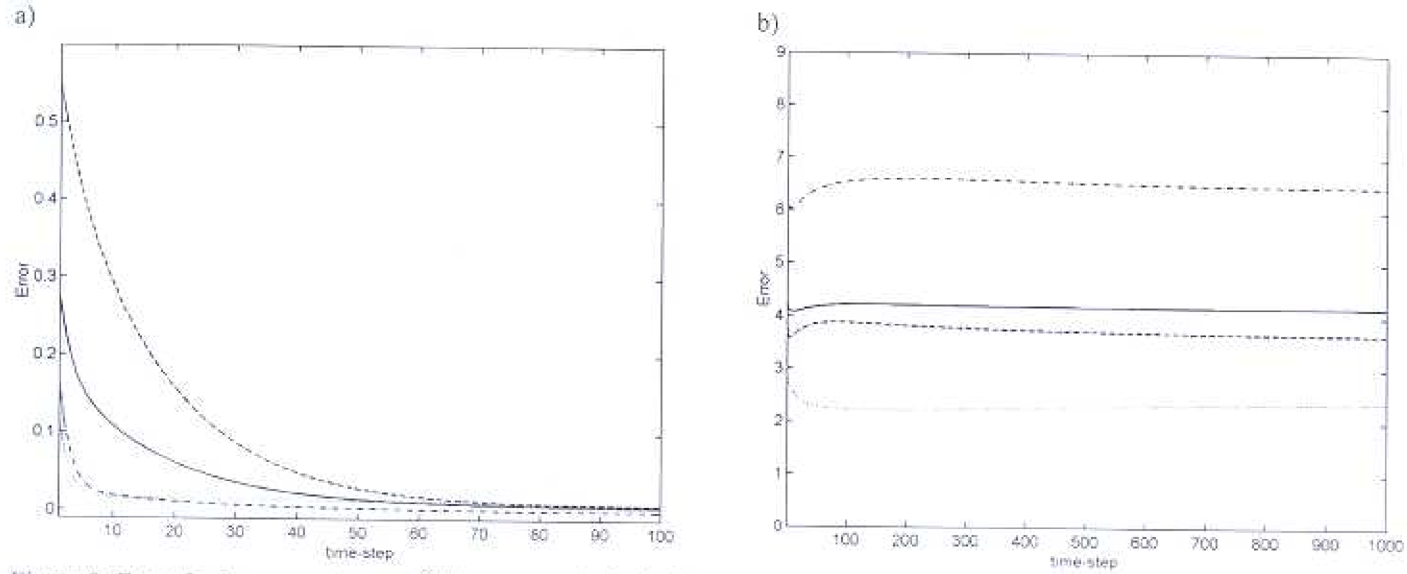


Figure 9: Error for Lorenz system, solid (mean error), dashed (error X), dashdot (error Y), dotted (error Z): (a) during the training phase; (b) for the cross validation.

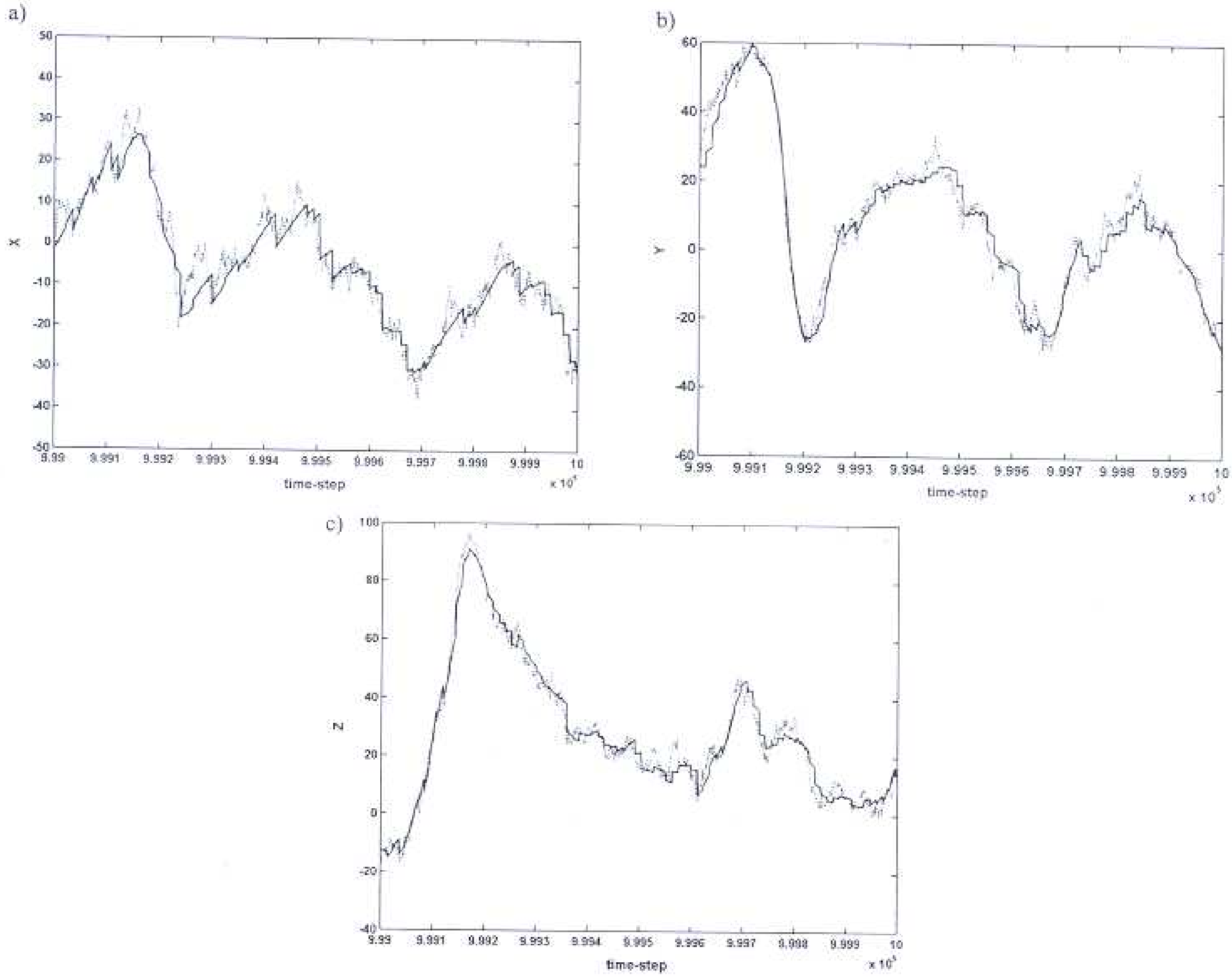


Figure 10: Data assimilation for the Lorenz system using E-NN, solid (E-NN), dotted (observation), components: (a) - X, (b) - Y, (c) - Z

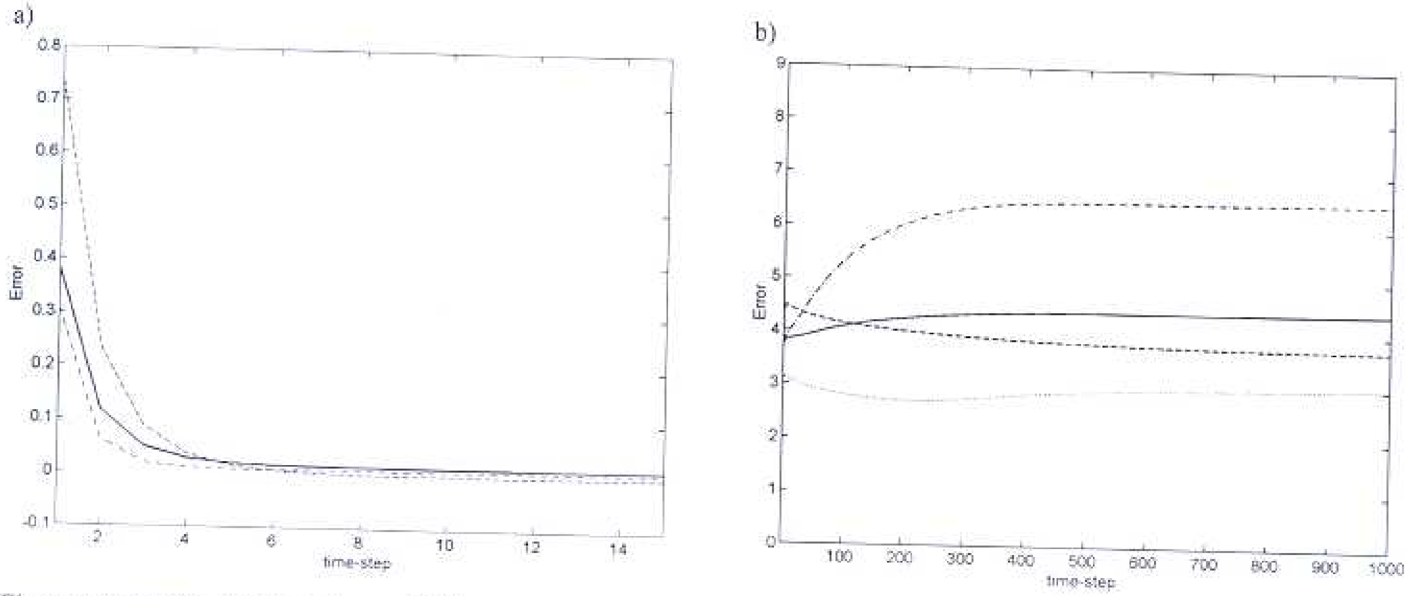


Figure 11: Error for Lorenz system, solid (mean error), dashed (error X), dashdot (error Y), dotted (error Z): (a) during the training phase; (b) for the cross validation.

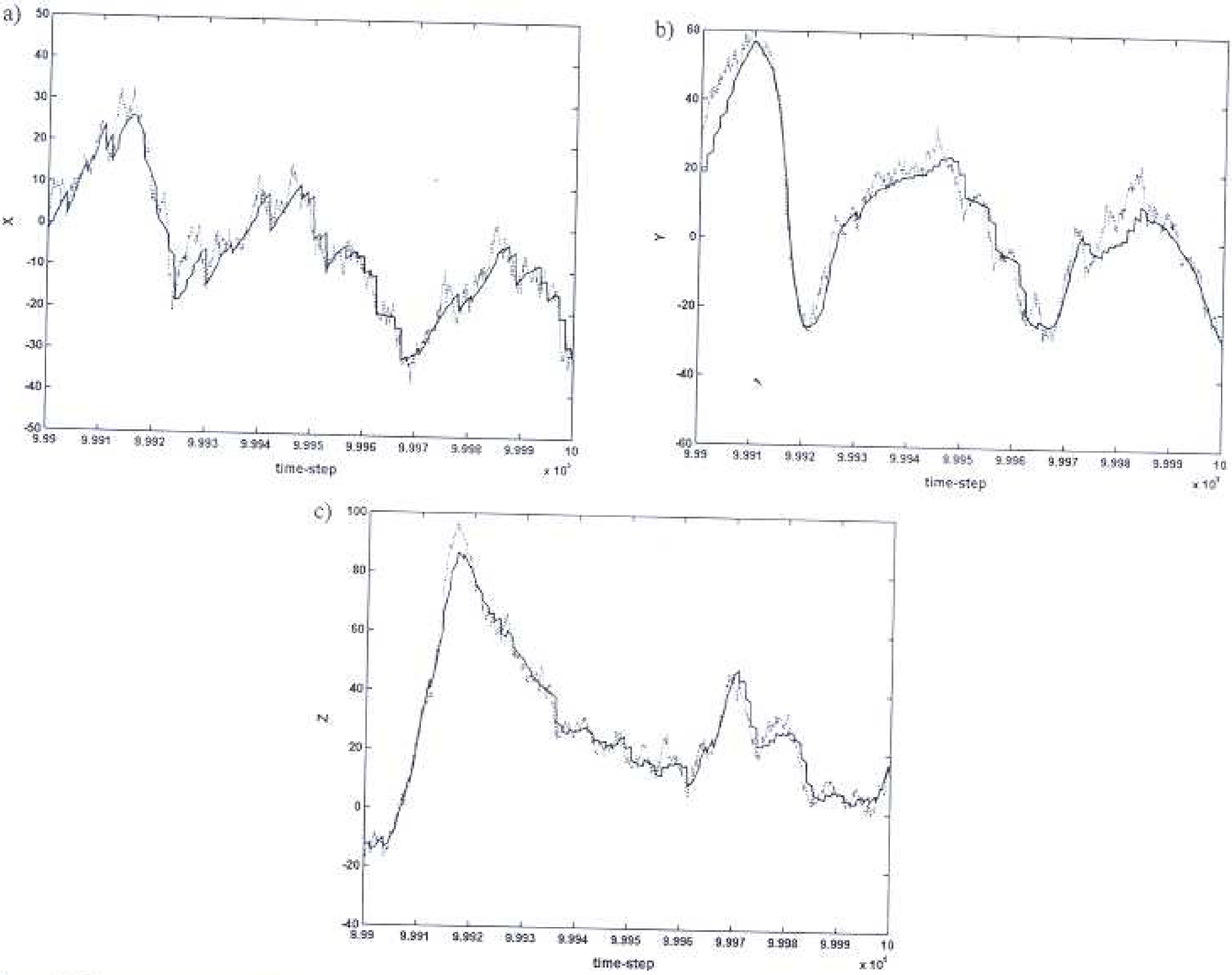


Figure 12: Data assimilation for the Lorenz system using J-NN, solid (J-NN), dotted (observation), components: (a) - X, (b) - Y, (c) - Z.

4. CONCLUSIONS

In this work the technique of the ANNs was tested for application to the data assimilation in chaotic dynamics. The efficiency of the recurrent ANNs (Elman and Jordan), was compared with the feed-forward ANNs (multi-layer perceptron and radial base function). These ANNs were trained using cross validation scheme. The learning with cross validation allows a complete knowledge of the error surface. From the knowledge of the authors, this is the first time that cross validation was employed in this application.

Neural networks are a nice alternative for data assimilation, because, after the training phase, the ANNs present a lower computational complexity than the Kalman filter and the variational approach. There are also other additional advantages, such as: ANNs are intrinsically parallel procedures, and a hardware implementation (neuro-computers) is also possible.

All ANNs employed were effective for data assimilation. It was not noted any improvement considering the recurrent ANNs used, related to the two feed-forward ANNs employed. The cross correlation is a good strategy to choose the best weight set. The best weight set means that we are not only looking for the weight set that learn from the patterns, but also the ANN that gives a best estimation for a data out from the training set.

5. ACKNOWLEDGEMENTS

The authors want to thank to the FAPESP and CNPq, Brazilian agencies for research support.

6. REFERENCES

BRAGA, A.; LUDERMIR, T.; CARVALHO, A. Redes neurais artificiais, teoria e aplicações. Rio de Janeiro: LTC - Livros Técnicos e Científicos Editora, 2000. 262 p.

CAMPOS VELHO, H.; VIJAYKUMAR, N.; STEPHANY S.; PRETO A.; NOWOSAD, A. A neural network implementation for data assimilation using MPI, applications of high-performance computing in engineering. Southampton: WIT, 280 p., Section 5, p.211-220, 2002.

DALEY, R. Atmospheric data analysis. Cambridge University Press, 1991. 457 p.

GREBOGI, C.; OTT, E.; YORKE, J. Chaos, strange attractors, and fractal basin boundaries in nonlinear dynamics. Science, Whasington, v. 238, n. 632, p. 585-718, 1987.

HAYKIN, S. Redes neurais artificiais: princípios e práticas. Porto Alegre: Bookman, 2001. 900 p.

HSIEH, W.; TANG, B. Applying neural network models top prediction and data analysis in meteorology and oceanography. Bulletin of the American Meteorological Society, Boston, v. 79, n. 9, p. 1855-1870, 1998.

JAZSWINSKI, A. Stochastic processes and filtering theory. New York: Academic, 1970. 376 p.

KALNAY, E. Atmospheric modeling, data assimilation and predictability. Cambridge University Press, 2003. 341 p.

LIAQAT, A.; FUKUHARA, M.; TAKEDA, T. Applying a neural collocation method to an incompletely known dynamical system via weak constraint data assimilation. Monthly Weather Review, Boston, v. 131, n. 8, p. 1697-1714, 2003.

LORENZ, E. Deterministic nonperiodic flow. Journal of the Atmospheric Sciences, Boston, v. 20, n. 2, p. 130-141, 1963.

NOWOSAD, A.; CAMPOS VELHO, H.; RIOS NETO, A. Neural network as a new approach for data assimilation. In: BRAZILIAN CONGRESS ON METEOROLOGY, 11., 2000, Rio de Janeiro. Anais... Rio de Janeiro: SBMET, 2000. p. 3078-3086.

NOWOSAD, A.; RIOS NETO, A., CAMPOS VELHO, H. Data Assimilation In Chaotic Dynamics Using Neural Networks. In: THIRD INTERNATIONAL CONFERENCE ON NONLINEAR DYNAMICS, CHAOS, CONTROL, AND THEIR APPLICATIONS IN ENGINEERING SCIENCES, 6., 2000, Campos do Jordão. Anais... São Paulo: Control, 2000. p. 212-221.

SALTZMAN, B. Finite amplitude free convection as an initial value problem. Journal of the Atmospheric Sciences, Boston, v. 19, n. 4, p.329-341, 1962.

TANG, Y.; HSIEH, W. Coupling neural network to dynamical systems via variational data assimilation. Monthly Weather Review, Boston, v. 129, n. 4, p. 818-834, 2001.

VIJAYKUMAR, N.; STEPHANY, S.; CAMPOS VELHO, H.; PRETO, A.; NOWOSA, A. Optimized neural network code for data assimilation. In: BRAZILIAN CONGRESS ON METEOROLOGY, 12., 2002, Foz do Iguaçu. Anais... Paraná: SBMET, 2002. p. 3841-3849.