RECURRENT AND FEEDFORWARD NEURAL NETWORKS APPLIED TO THE DATA ASSIMILATION IN CHAOTIC DYNAMICS

Fabrício P. Härter[1] and Haroldo F. de Campos Velho [1]

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 is analized. The Lorenz system under chaotic regime is used as a test problem. These four NNs were trainned for emulating a Kalman filter using cross validation scheme. Multilayer perceptron and Elman ANNs show better results. The results obtained encouraging the application of the ANNs as an assimilation tecnique.

Resumo Redes nurais artificiais é uma nova abordagem para assimilação de dados. 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 realimentação com validação cruzada. O perceptron de múltiplas camadas e a rede de Elman foram as que obtiveram os melhores desempenhos. Os resultados encorajam a aplicação de redes neurais como uma técnica assimilação.

Key-words: Recurrent neural networks, data assimilation, Lorenz system, cross validation.

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 desagreement between the output from the numerical prediction and the real atmosphere, in other words, the forecasting error has a direct relationship with the increasing of the integration time. NWP is an initial value problem, this implys that a better representation for the inital condition 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 obvervational 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 (Dayley, 1991). They have different strategies to combine the forecasting (*background*) and observations. From mathematical point of view, the assimilatin process can be represented by

$$\mathbf{x}^a = \mathbf{x}^f + \mathbf{W}\ p\left[\mathbf{y}^o - \mathbf{H}(\mathbf{x}^f)\right] \tag{1}$$

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

[1]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, haroldo]@lac.inpe.br

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. (2000a) (see also Nowosad et al. (2000b), 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-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 as representer of an unknown equation in the mathematical system equations of the model.

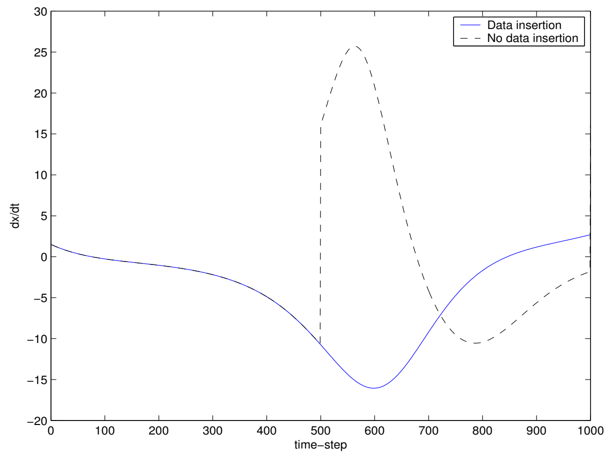


Figure 1: Shock in the numerical model after data insertion, without assimilation technique.

Differently from previous works using ANN for data assimilation, this paper deals with recurrent ANN. 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.

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.

Testing Model: Lorenz System

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

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

$$dY/d\tau = r\,X - Y - XZ \tag{3}$$

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

where $\tau \equiv \pi^2\, H^{-2}\,(1 + a^2)\,\kappa\, t$ is the non-dimensional 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 vicosity); $b \equiv 4(1+a^2)^{-1}$. 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.

Kalman Filter

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

$$\mathbf{w}^f_{n+1} = \mathbf{F}_n \mathbf{w}^f_n + \mu_n \tag{5}$$

$$\mathbf{z}^f_n = \mathbf{H}_n \mathbf{w}^f_n + \nu_n \tag{6}$$

where $\mathbf{F}^f_n$ is our mathematical model, μ_n is the stochatic forcing (modeling noise error). The observation system is modeled by matrix $\mathbf{H}_n$, 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 $\mathbf{w}_{n+1} = [X_{n+1}, Y_{n+1}, Z_{n+1}]^T$, and it is estimated through the recursion

$$\mathbf{w}^a_{n+1} = (\mathbf{I} - \mathbf{G}_{n+1}\mathbf{H}_{n+1})\mathbf{F}_n \mathbf{w}^a_n + \mathbf{G}_{n+1}\mathbf{z}^f_{n+1} \tag{7}$$

where $\mathbf{w}^a_{n+1}$ is the analysis value, $\mathbf{G}_n$ is the Kalman gain, computed from the minimization of the estimation error variance J_{n+1} (Jaswinski, 1970)

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

being $\mathbf{E}\{.\}$ the expected value. The algorithm of the *Linear Kalman Filter* (LKF) is shown in figure 2, where $\mathbf{Q}_n$ is the covariance of μ_n and $\mathbf{R}_n$ is the covariance of ν_n.

The assimilation is done through the sampling:

$$\mathbf{r}_{n+1} \equiv \mathbf{z}_{n+1} - \mathbf{z}^f_{n+1} = \mathbf{z}_{n+1} - \mathbf{H}_n \mathbf{w}^{\mathbf{f}}_{\mathbf{n+1}} \ . \tag{9}$$

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 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. 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 the network can be presented to new inputs for which it calculates the corresponding outputs, based on what it has learned. The back-propagation algorithm is used as the learning process for

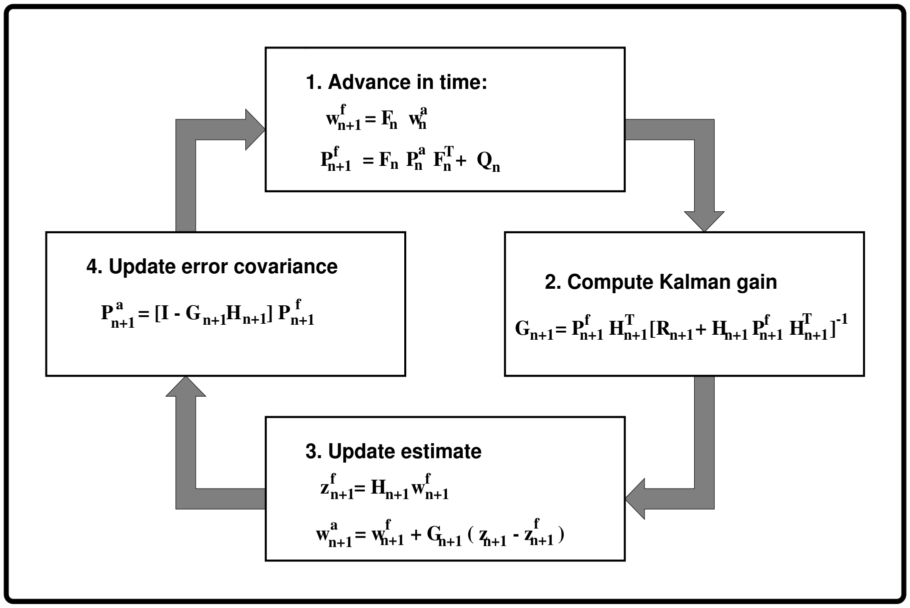


Figure 2: Schematic diagram of the linear Kalman filter.

all ANN topologies studied.

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

$$u_k = \sum_{j=1}^{m} w_{kj} \, x_j \ , \tag{10}$$

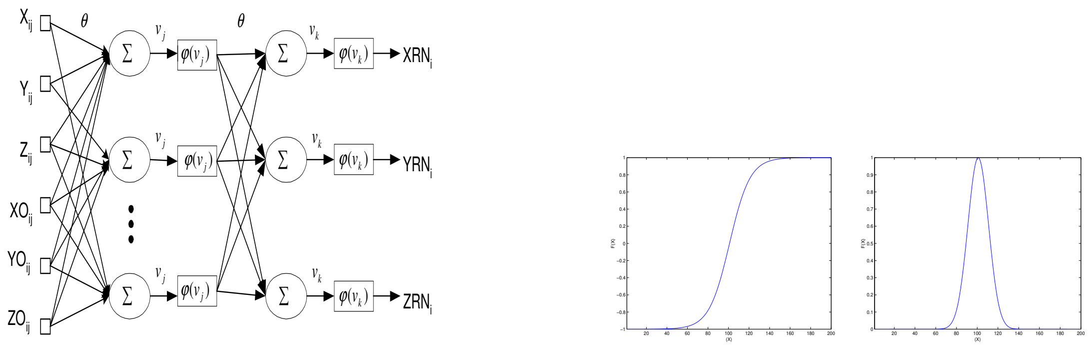
$$y_k = \varphi(u_k + b_k) \tag{11}$$

where x_1, ..., x_m are the inputs; w_{k1}, ..., w_{km} are the conection weights for the neuron-k, u_k is the linear output of the linear combinação combination amoung weighted inputs, b_k is the bias; $\varphi(.)$ is the activations 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) and fig:ANN-1(b1) depicts a backpropagation neural network, one hidden layer, and the activation function.

Radial basis function 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(b2).

Two recurrents ANNs are also investigated, Elman and Jordan NNs (Braga et al., 2000). Beyond of the standard units, such as input/output and hidden layers, recurrent NNs present the context units. Input and output units are in contact with the external environment, while the context

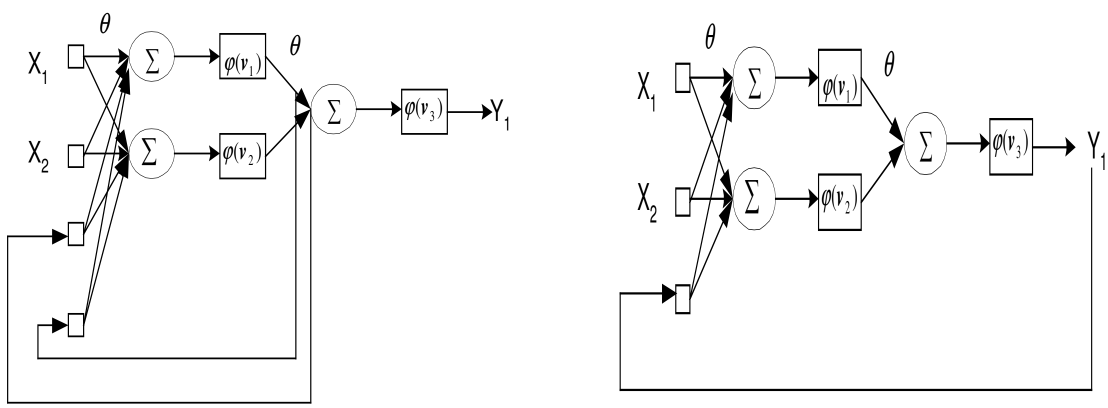


(a) **(b1)** **(b2)**

Figure 3: (a) Outline for the neural network; activation functions (b1) for the MLP-NN: $\tanh^1(x)$; (b2) for the RBF-NN: Gaussian distribution.

units are not. Input units are just buffer units, they do not change the inputs. The hidden layers present ativation functions, alterning the inputs and producing the outputs. Context units introduce a *memory* in the system, keeping the previous outputs as additional inputs (recurrency).

For the E-NN, the recuurency is done connecting the hidden layer with the input layer – see Figure 4(c). Similar topology is employed to the J-NN, Figure 4(d), but the output values are used in the recurrency.



(a) **(b)**

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

NUMERICAL EXPERIMENTS

Our numerical experiments are performed using the following features for Kalman filter:

$$\mathbf{Q}_n = 0.1\,\mathbf{I}\,; \qquad \mathbf{R}_n = 2\,\mathbf{I}\,; \qquad \mathbf{P}_0^f = \begin{cases} 10\,(\mathbf{w_0^f})_i^2 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} . \tag{12}$$

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 10^6 time-steps.

Table 1: Neural network parameters: one hidden layer.

	MLP	RBF	E-NN	J-NN
neurons	2	2	2	2
α_h	0.6	0.0	0.3	0.
α_X	0.6	0.0	0.9	0.
α_Y	0.6	0.0	0.9	0.
α_Z	0.6	0.0	0.8	0.
η_h	0.001	0.001	0.01	0.1
η_X	0.001	0.001	0.01	0.01
η_Y	0.001	0.001	0.01	0.1
η_Z	0.001	0.001	0.01	0.01

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.

During the learning process, the cost function (the square difference between the ANN output and the target data) is decreasing, but this does 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. Appropriated momemtum constant and learning ratio can avoid these local minima, and the cross validation is an alternative to choise a better set of the connection weights implying in a better generalization.

Cross validation consists to split 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. In our experiments, the number of iterations was fixed at 1000, and the cross validation was used to keep the best weight set for the generalization.

The training, cross validation, and estimation errors are computed as following ($n = X,\ Y,\ Z$ and average):

$$EMQ_n = \frac{1}{2000} \sum_{i=1}^{2000} \frac{1}{2} \left(X_i^{KF} - X_i^{NN} \right)^2 ; \tag{13}$$

$$EA_n = \frac{1}{333} \sum_{k=1}^{333} \sqrt{(X_k^{KF} - X_k^{NN})^2} ; \tag{14}$$

$$EE_n = \frac{1}{10^5} \sum_{k=1}^{10^5} \sqrt{\left(X_k^{\mathrm{Obs}} - X_k^{NN} \right)^2} . \tag{15}$$

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

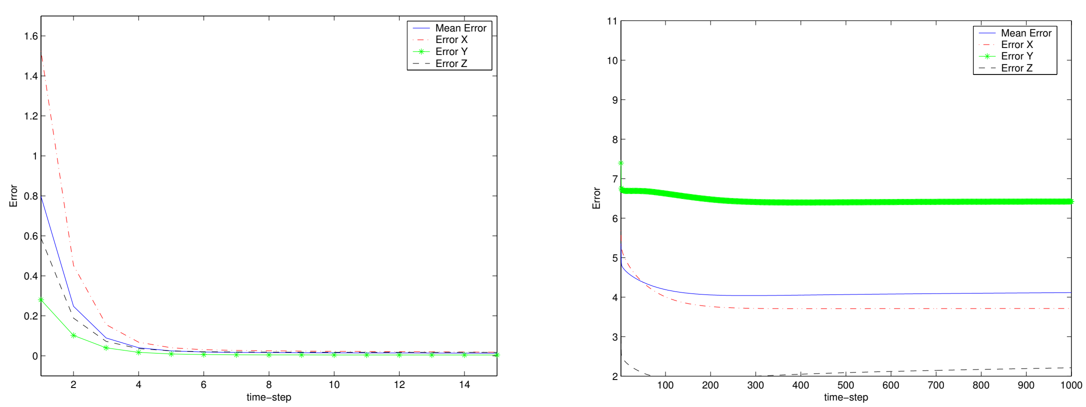
Table 2: Neural network parameters: two hidden layers.

	MLP (L1, L2)	RBF (L1, L2)	J-NN (L1, L2)
neurons	(2, 6)	(11, 2)	(6, 2)
α_h	0.5	0.1	0.0
α_X	0.5	0.1	0.0
α_Y	0.5	0.0	0.0
α_Z	0.5	0.1	0.0
η_h	0.001	0.1	0.01
η_X	0.001	0.1	0.001
η_Y	0.1	0.1	0.01
η_Z	0.001	0.1	0.01

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

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.



(a) (b)

Figure 5: Error for the Lorenz system: (a) during the training phase; (b) for the cross validation.

Figures 6(a)-6(c) depicted the last 10^3 time-steps of the integration, where the data assimilation is performed by the MLP with one hidden layer. In these figures, the solid line represents the observation. Clearly, the MLP-NN is effective to carry out the assimilation.

Two neurons are used for MLP-NN in the hidden layer, and the learning ratio are described in Table 1, reaching a minimum average error at 3.89 for the second epoch. The learning error dramatically decreases for the first two 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

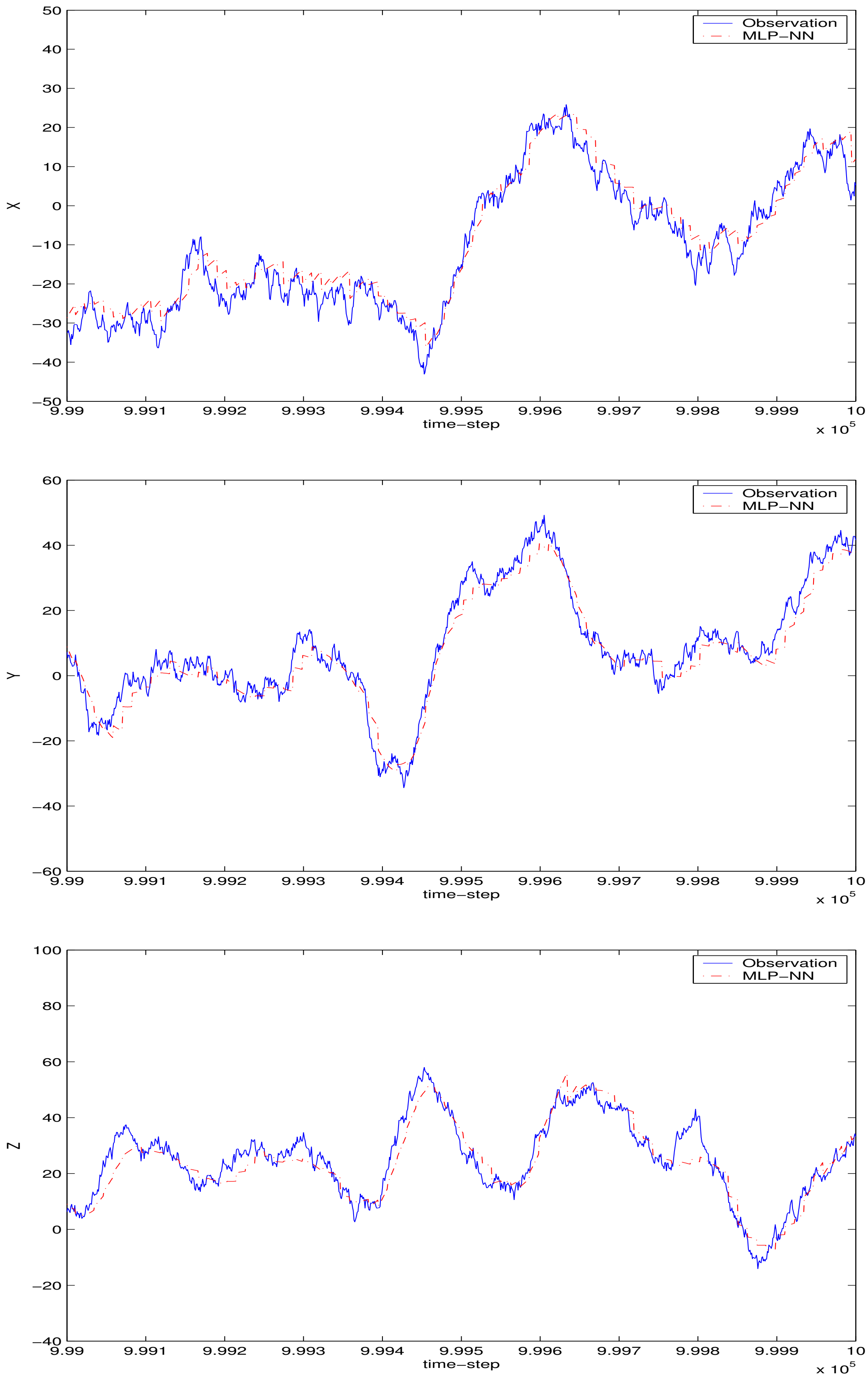
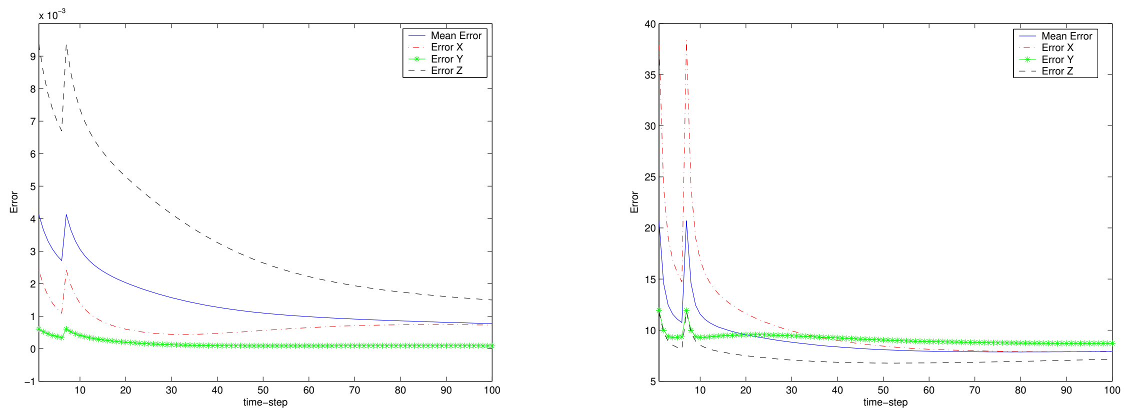


Figure 6: Data assimilation for the Lorenz system using MLP-NN: (a) component-X, (b) component-Y, (c) component-Z.

weight set for activation (or generalization) of the ANN.

Assimilation: Radial Base Function (RBF)

The same experiment applied to the LMP-NN is carried out for the RBF-NN, with one hidden layer. The iteration errors are shown in Figures 7(a)-(b) and 8(a)-(c). The best architecture of this NN is obtained with two neurons in the hidden layer, using learning ratio and momentum constant listed in Table 1. 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 7: Error for the Lorenz system: (a) during the training phase; (b) for the cross validation.

Figures 8(a)–8(c) show that the RBF also presents a good performance, with $EE_X = 3.477$, $EE_Y = 5.350$, $EE_Z = 3.962$ and $EE_{\text{average}} = 4.263$ for components X, Y, Z, and the average error, respectively.

Assimilation: Recurrents NNs

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

Figures 9(a)–9(b) plot the error computed from the training data set for the E-NN. In the Figures 11(a)–11(c) the last 1000 time-steps of the whole integration (with 10^6 time-steps) is shown for this NN.

Two neurons are used in the hidden layer, with learning ratio and momentum constant as given in Table 1. The average error $EE_{average} = 3.83$ 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 11(a)–11(c), giving $EE_X = 4.13$, $EE_Y = 3.63$, $EE_Z = 3.74$ and $EE_{\text{average}} = 3.84$ for components X, Y, Z, and the average

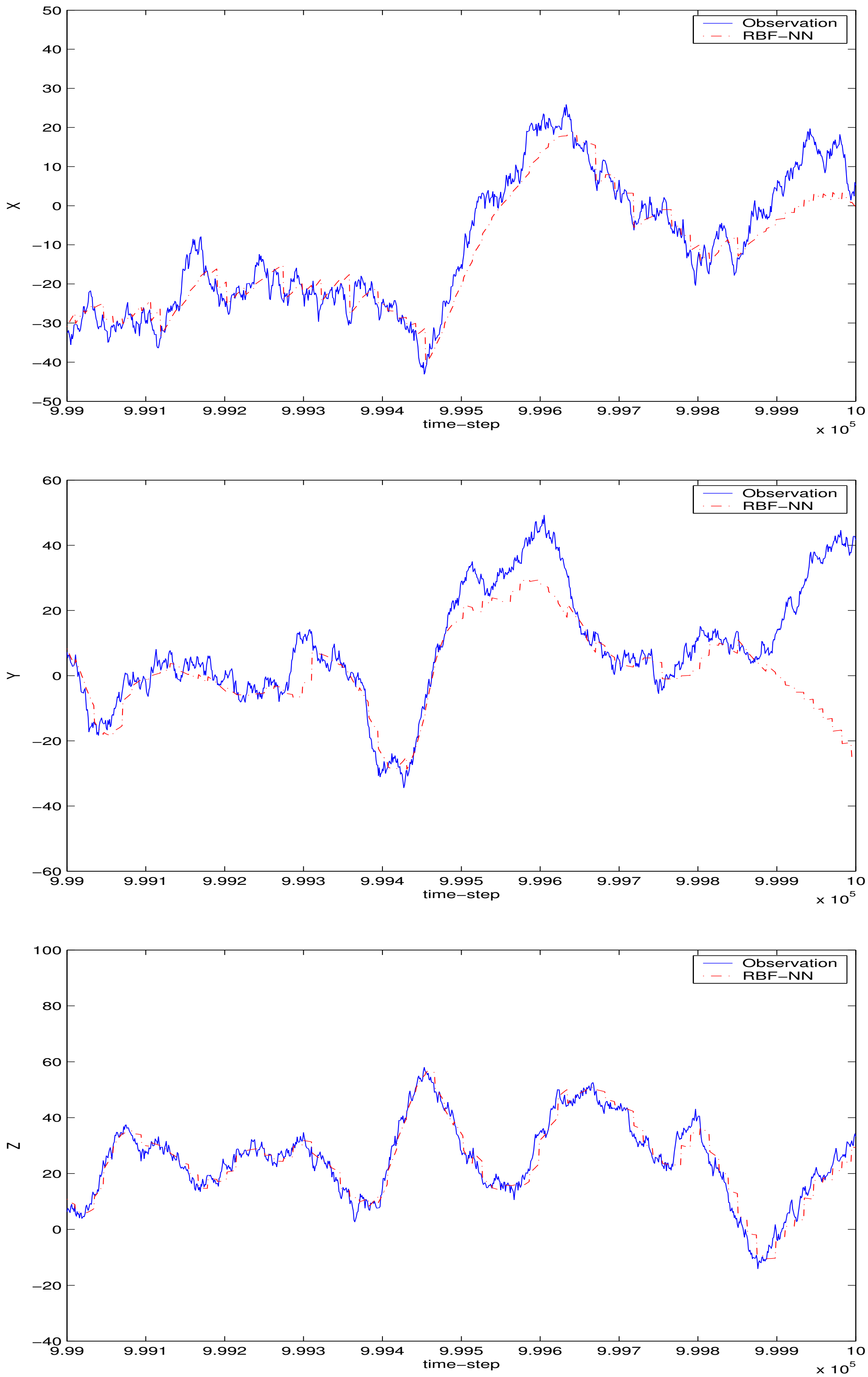
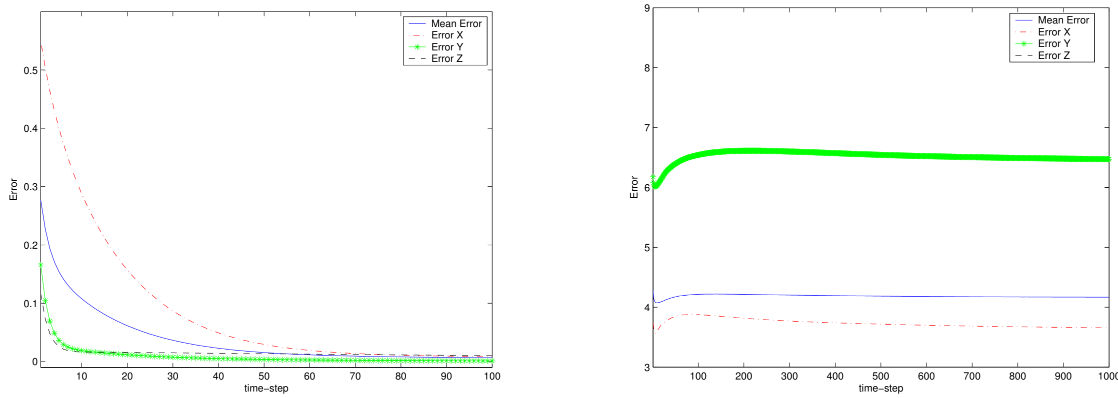


Figure 8: Data assimilation for the Lorenz system using RBF-NN: (a) component-X, (b) component-Y, (c) component-Z.



(a) (b)

Figure 9: Error for the Lorenz system: (a) during the training phase; (b) for the cross validation.

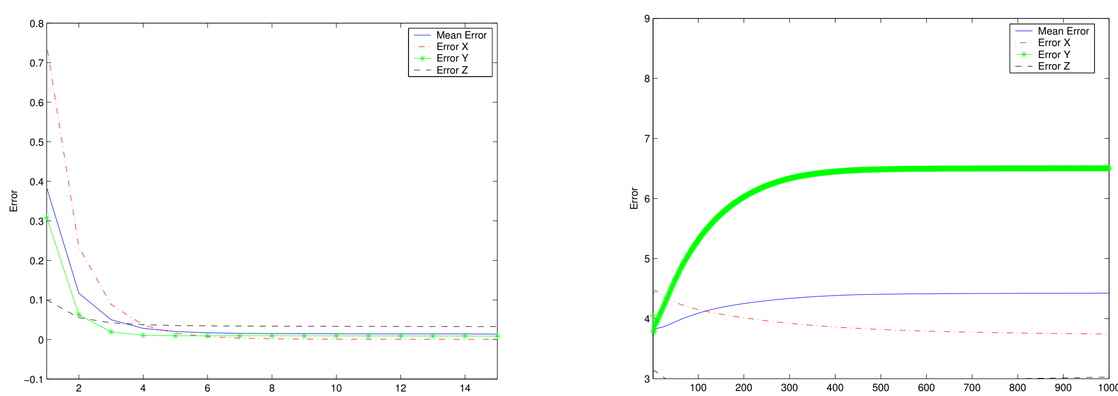
error, respectively.

The assimilation with J-NN is efficient too, and there is no big difference related to other NNs – see Figures 12(a)–12(c). The errors for the X, Y, Z, and the *average* are $EE_X = 3.95$, $EE_Y = 4,99$, $EE_Z = 4.62$, $EE_{\text{average}} = 4.52$. Figures 10(a)–10(b) show that the training errors decrease fast for the first epochs, while cross validation errors increase. For this NN, the best answer from the J-NN is obtained at the first epoch.

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 NNs (Elman and Jordan), was compared with the feed-forward NNs (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 alernative for data assimilation, because, after the training phase,



(a) (b)

Figure 10: Error for the Lorenz system: (a) during the training phase; (b) for the cross validation.

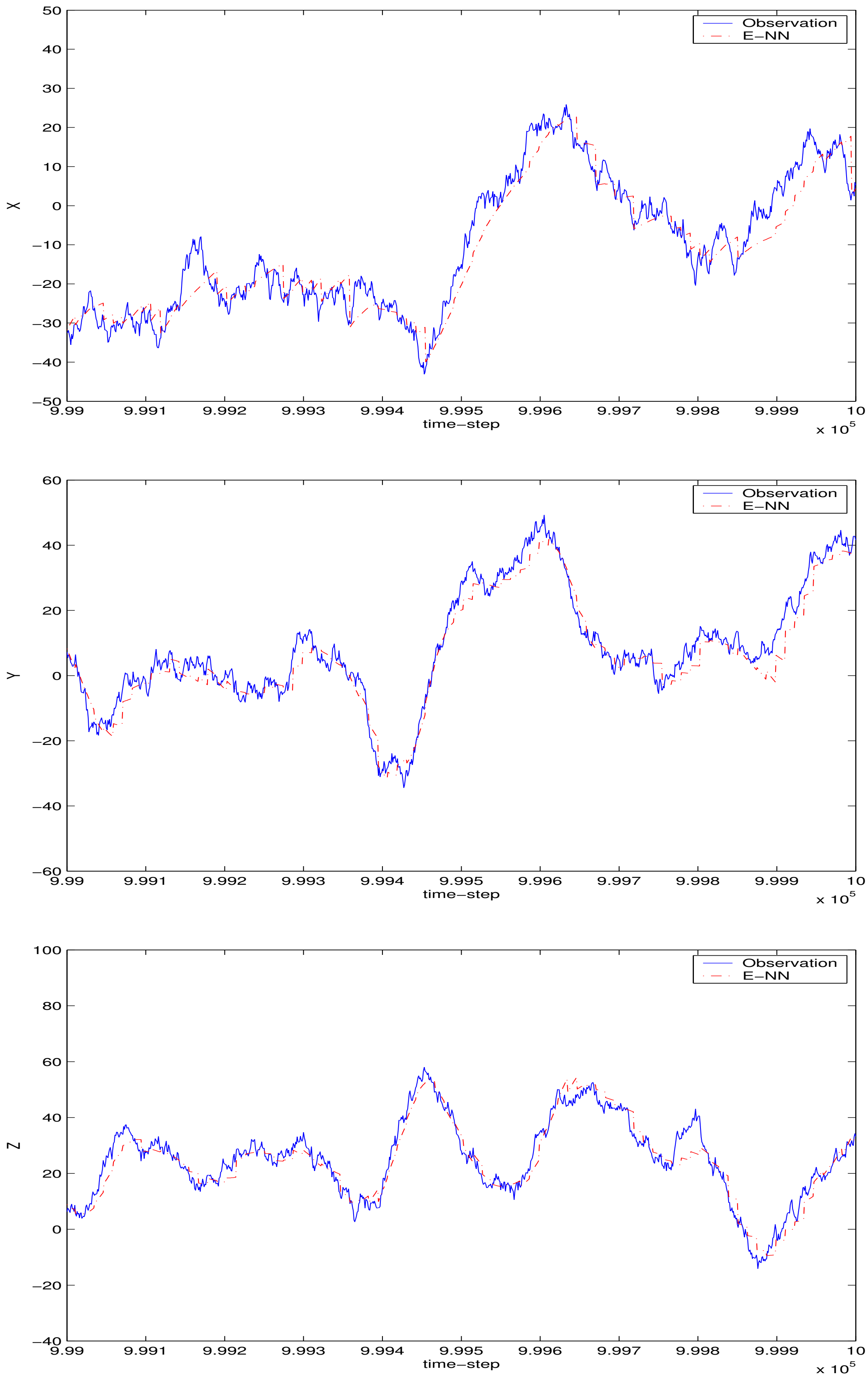


Figure 11: Data assimilation for the Lorenz system using E-NN: (a) component-X, (b) component-Y, (c) component-Z.

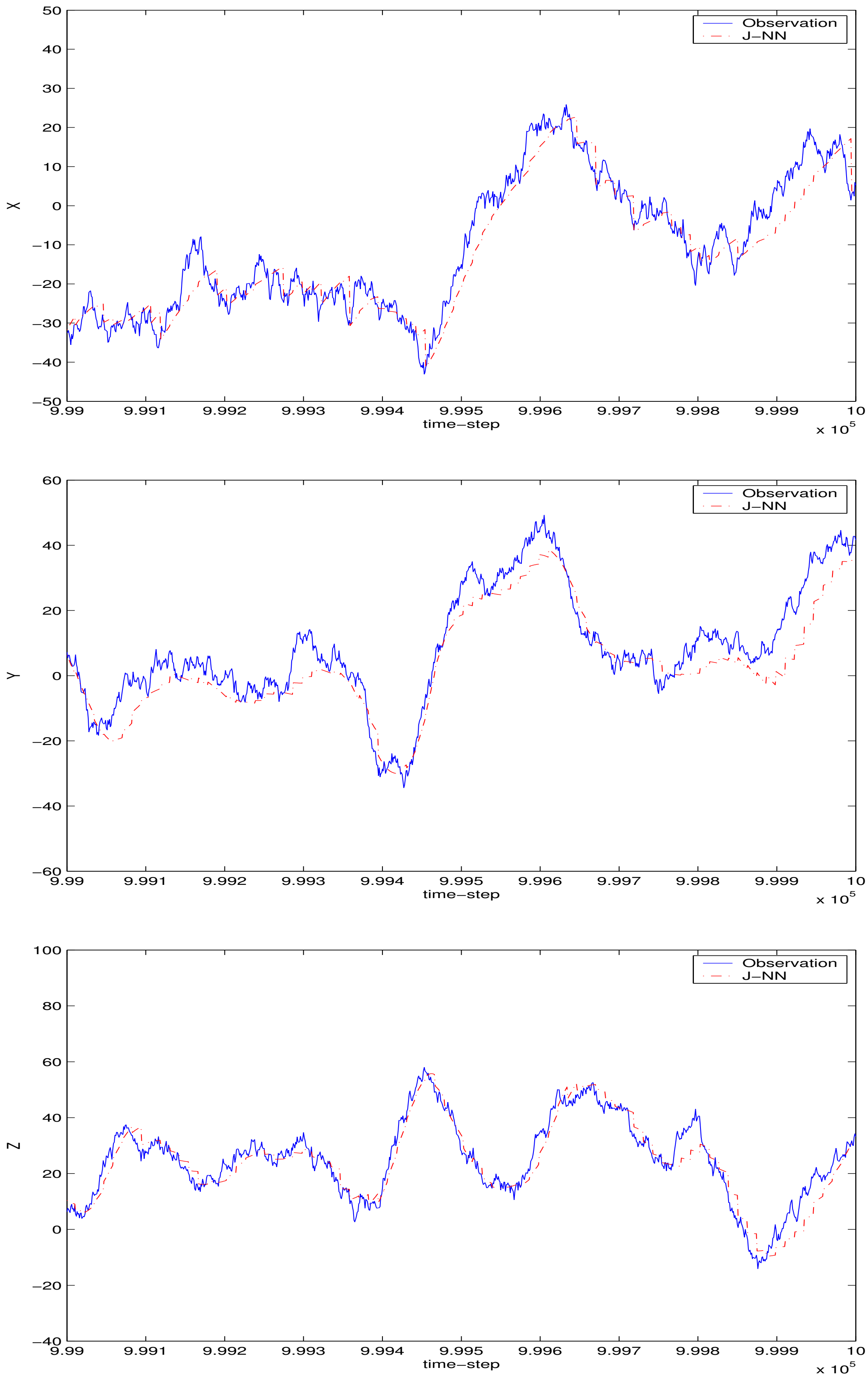


Figure 12: Data assimilation for the Lorenz system using J-NN: (a) component-X, (b) component-Y, (c) component-Z.

the ANNs present a lower computational complexity than the Kalman filter and the variational approach. There are also other additional advantages, such as: NNs are intrinsicly parallel procedures, and a hardware implementation (neuro-computers) is also possible.

All NNs employed were effetive for data assimilation. It was not noted any improvement considering the recurrent NNs used, related to the two feed-forward NNs employed. The cross correlation is a good strategy to choise 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 NN that gives a best estimation for a data out from the training set.

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

References

[1] Bejerknes, V. Das problem derwettervorhersage, betrachtet vom stnpunkt der mechanik und der Physik. Meteor. Zeits., 21:1-7, 1904.

[2] Bennet, A. F. Inverse Methods in Physical Oceanography, Cambridge University Press, 1992.

[3] Braga, A.P., Ludermir, T.B., Carvalho, A.C.P.L.F., Redes Neurais Artificiais, Teoria e Aplicações, 2000.

[4] Campos Velho, H. F.; Vijaykumar, N.L.; Stephany S.; Preto A.P.; Nowosad A.G. 2002: A Neural Network Implementation for Data Assimilation using MPI, Applications of High-Performance Computing in Engineering (Edited by C.A. Brebbia, P. Melli, A. Zanasi), WIT Press, Southampton (UK), 280 pp., Section 5, pp. 211-220, ISBN 1-85312-924-0.

[5] Charney, J. G.Dynamical forecasting by numerical process. Compendium of meteorology. American meteorological society, Boston, MA, 1951.

[6] Daley, R.Atmospheric data analysis1ed, 1st ed., Cambridge University Press,USA, 1991.

[7] Gardner, M.W. Dorling, S.R. Artificial Neural Networks (The Multilayer Perpectron) - A Review of Applications in the Atmospheric Sciences. *Atmospheric Environment*, **32**(14/15), p. 2627-2636, 1988.

[8] Girosi, F., Poggio, T. Networks and the best approximation property. Biological Cibernetic, p.169-176. 1990.

[9] Haykin, S. Redes Neurais: Princípios e prática, 2a ed, Bookman, Porto Alegre, 2001.

[10] Hsieh, W.W., Tang, B., 1998. Applying Neural Network Models top Prediction and Data Analysis in Meteorology and Oceanography. Bulletin of the American Meteorological Society. 79(9), 1855-1870.

[11] Jazswinski, A.H., Stochastic Processes and Filtering Theory. Academic Press, New York, USA, 1970.

[12] Kalnay, E. Atmospherica modeling, data assimilation and predictability, Cambridge, University Press, United Kingdom, 2003.

[13] Kolmogorov, A. On Representation of Continuous Functions. Dokl. Acad. Nauk. URSS, 114. 1957.

[14] Kovacs, Z. Redes Neurais Artificiais: Princípios e Aplicações, Collegium Cógnitio, Rio de Janeiro Brasil, 1996.

[15] Liaqat, A., Fukuhara, M., Takeda, T., 2003. Applying a neural Collocation Method to an Imcompletely Known Dynamical System via Weak Constraint Data Assimilation. Monthly Weather Riview. 131, 1697-1714.

[16] Lorenz, E. Deterministic Nonperiodic Flow. Journal of the Atmospheric Sciences. 20:130-141, 1963.

[17] Lynch, P.; Huang, X. Initialization of HIRLAM Model Using a Digital Filter. Monthly Weather Review, v. 120, n. 6, p. 1019-1034, Jun. 1992.

[18] Machenhauer, B. On the dynamics of gravity oscillations in a shallow water model with applications to normal mode initialization. Contribution Atmospheric Physics, v. 50, n. 1, p. 253-271. 1977.

[19] Miller, R.N., Guil, M., Gauthiez, F. Advanced Data Assimilation in Strongly Nonlinear System. Journal of the Atmospheric Sciences. 51(8):1037-1056.

[20] Nowosad, A. G., Campos Velho, H.F. de, Rios Neto, A, 2000a: Neural Network as a New Approach for Data Assimilation, Brazilian Congress on Meteorology, 16-20 October, Rio de Janeiro (RJ), Brasil, Proceedings in CD-ROM (paper code PT00002), pp. 3078-3086.

[21] Nowosad, A.G., Campos Velho, H.F., Rios Neto A.: Neural Network as a New Approach for Data Assimilation, Brazilian Congress on Meteorology, 16-20 October, Rio de Janeiro (RJ), Brasil, Proceedings in CD-ROM (paper code PT00002), pp. 3078-3086, 2000b.

[22] Richardson, L. F. Weather prediction by numerical process. Cambridge University press, Cambridge, 1922.

[23] Saltzman, B.. Finite Amplitude Freee Convection as an Initial Calue Problem. Journal of the Atmospheric Sciences. 19:329-341, 1962.

[24] Tang, B., Hsieh, W.W., 2001. Coupling Neural Network to Dynamical Systems via Variational Data Assimilation. Montlhy Weather Review. 79(9), 1855-1870.

[25] Vijaykumar, N.L., Stephany, S., Campos Velho, H.F., Preto, A.J. Nowosad, A.G.: Optimized Neural Network Code for Data Assimilation, Brazilian Congress on Meteorology, 04-07 August, Foz do Iguacu (PR), Brazil, pp. 3841-3849, 2002.