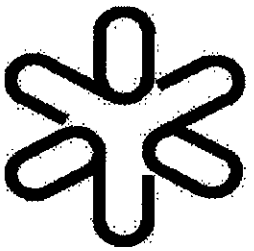

Instituto de Física
Universidade de São Paulo

Using Artificial Neural Networks to Forecast Chaotic Time Series

Oliveira, K. A.; Vannucci, A.; Silva, E. C. da

Instituto de Física, Universidade de São Paulo, SP, Brasil

Publicação IF - 1389/99

Using Artificial Neural Networks to Forecast Chaotic Time Series

Kenya Andrésia de Oliveira [1] , Álvaro Vannucci and Elton Cesar da Silva

Instituto de Física da Universidade de São Paulo, caixa postal: 66318, CEP: 05315-970, São Paulo, SP, Brazil

Abstract

Two-layer feedforward neural network was used in this work to forecast chaotic time series with very promising results, especially for the Lorenz system, as in comparison to others that had been previously published elsewhere. It was observed that the architecture *m:2m:m:1*, where m is the embedding dimension of the atractor of the dynamical system in consideration, is a very good initial guess for the process of finding the ideal architecture for the neural network, which is usually hard to achieve. The results we obtained with this particular type of series, and also with some others like Henon and Logistic maps, clearly indicate that there is an interplay between the architecture of a multilayer network and the embedding dimension m of the time series used. From the very good forecasting results we obtained it can be concluded that neural networks can be considered to be an important tool for making predictions of the time evolution of nonlinear systems.

Key words: Neural Networks; Predictions; Atractors

1 Introduction

In many situations in science and technology we usually face the necessity of predicting the future evolution of a system from past measurements of it. Mathematical models of physical systems are generally investigated by writing down the equations of motion and by trying to integrate them, forward in time, to predict the future state of the system. Mathematically speaking, this dynamics is described by the motion of a point $\vec{v}$, which represents the state of the system in a multi-dimensional space Γ. However, in nonlinear systems

[1] Corresponding author. Fax: +55-11-8187014; e-mail: kenya@if.usp.br

with many degrees of freedom, it is just impraticable to solve all the equations without making some sort of assumptions and simplifications. Dissipations can reduce the number of the effectively relevant degrees of freedom in apparently chaotic dynamical systems. Thus the motion of the system becomes confined to a subspace Γ_A, of Γ, known as *atractor* with lower dimension d (1; 2). Within this scenario, neural networks may be considered an important forecasting tool to be used in such situations.

As it is already well known, the human brain is superior to even the most powerful digital computer in many tasks and a good example is the prediction and controlling of complex systems without the necessity of having an underlying knowledge of the physics of the system: any child can ride a bicycle without knowing Newtonian Mechanics, for instance.

Thus, this is the real motivation for studying neural computation, which is based on the knowledge of neuroscience apart from attempting to be biologically realistic in all details. The main point to be considered is the essential property of the biological neurons from the viewpoint of information processing (3). In this respect, artificial neural network models can be constructed to imitate some aspects of the structure of the brain and the nervous system, and have been extensively used in a variety of areas such as Robotics, Medicine, Economics, Astronomy and, of course, in Physics. In this work we have concentrated our attention on predicting chaotic time series obtained from the solutions of Lorenz equations and by iterations from Henon and Logistic maps. However, the same procedure here described could be used for other rather complicated dynamical systems, such as turbulence in fluids (4), lasers (5), chemical reactions (6), and plasma physics (7), only to name a few.

In section 2 a simplified description of the basics of neural networks will be given (1; 2). In section 3 the forecasting of chaotic time series will be reported. Although some works on Lorenz system predictions have already been reported by using different approachs (8; 9), we will be showing that artificial neural networks with the particular architecture *m:2m:m:1*, where m is the embedding dimension of the atractor of the dynamics in focus, can yield much better results. We have already been using neural network mainly to forecast disruption instabilities in magnetically confined plasma with relative success (10), and much more promising results have also been obtained from this new architeture (11). Finally, in section 4, the conclusions of this work will be presented.

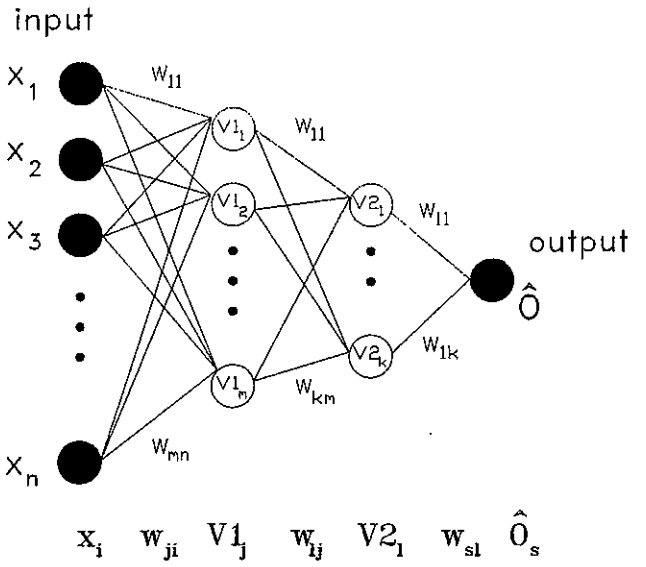

Fig. 1. Diagram of a feedforward neural network with two hidden layers and architecture *n:m:k:1.*

2 Neural Networks

Artificial neural networks are computer algorithms which simulate in a very simplified form the ability of brain neurons to process information. In general, a neural net is typically composed of interconnected *units* organized in layers which serve as model neurons. The function of the synapse, the structure responsable for storing information in the brain, is modelled by a modifiable weight which is associated to each connection between neural units located in adjacents layers (fig 1). Within each unit all the weighed input signals are summed up and an excitatory or inhibitory signal is then fired to the next layer of units, depending on whether the result of the sum has reached or not a certain threshold value, according to the specific activation function chosen. The ideal set of weights are obtained by *training* the neural net, that is, by feeding the net with known examples of the data to be used on it, and by minimizing the prediction error through the backpropagation algorithm with inertial terms (1; 2; 3). Starting from a given initial configuration for the m input units $\{x_1^\mu, x_2^\mu, ..., x_m^\mu\}$, the dynamics of the two-layer feedforward network is defined as follows:

Given the data training pattern μ, each hidden unit j in the first hidden layer receives a net input

$$\Upsilon_j^\mu = \sum_i W_{ji} x_i^\mu, \tag{1}$$

and produces the output

$$V1_j^\mu = \tanh(\Upsilon_j^\mu) = \tanh\left(\sum_i W_{ji} x_i^\mu\right), \tag{2}$$

where W_{ji} represents the connection weight between the *ith* input unit and the *jth* hidden unit in the first layer. Following the same procedure for the other units in the next layers, the final output is then given by:

$$\hat{O}_s^\mu = \sum_l W_{sl} \left\{ \tanh\left[\sum_j W_{lj} \tanh\left(\sum_i W_{ji} x_i^\mu\right)\right]\right\}, \tag{3}$$

where the *hyperbolic tangent* activation function was chosen for all hidden units, and the *linear function* for the final output unit.

Supervised feedforward networks learn from examples. The weights of the conections are determined by presenting the network with a set of actual input-output values (the training set) and comparing, by means of some *error* or *cost function* $E(W)$, the output of the network with the real value of the time series. For this work, the cost function was chosen to be the mean square error:

$$E(W) = \frac{1}{M} \sum_{\mu s}^{M} (O_s^\mu - \hat{O}_s^\mu)^2, \tag{4}$$

where M is the size of the training data set, $\hat{O}_s^\mu$ is the output data yielded by the unit of the last layer and O_s^μ is the actual data. The error is minimized by adjusting the weights according to the backpropagation algorithm, which corresponds to the gradient descent recipe with the inclusion of an inertial term to accelerate the convergence (2; 3; 12). In particular,

$$W_{qt}^{new} = \alpha W_{qt}^{old} - \eta \Delta W_{qt}, \tag{5}$$

where:

$\Delta W_{qt} = \frac{\partial E(W_{qt})}{\partial W_{qt}}$; q and t identify each connection

$0 < \eta \leq 1 \equiv$ learn rate, and

$0 < \alpha < 1 \equiv$ inertial term.

In order to characterize the precision of the training process, the *average relative prediction error variation* (ARV) and the *correlation coefficient* (ρ) are

therefore introduced (10; 12):

$$ARV = \frac{\sum_{\mu s}(O_s^\mu - \hat{O}_s^\mu)^2}{\sum_{\mu s}(O_s^\mu - \langle O_s^\mu \rangle)^2} \tag{6}$$

$$\rho = \frac{\sum_{\mu s}(O_s^\mu - \langle O_s^\mu \rangle)(\hat{O}_s^\mu - \langle \hat{O}_s^\mu \rangle)}{\sigma_O \sigma_{\hat{O}}} \tag{7}$$

where $\langle O_s^\mu \rangle$ is the mean value of the set of M actual outputs, and σ_O and $\sigma_{\hat{O}}$ are the standard deviations of the actual and the predicted outputs respectively.

The training of the network was carried out by using one arbitrarily chosen temporal series out of a given dynamical system. In order to obtain the best set of weights $\{W\}$ we monitored the performance of the net by using a second temporal series of the same system which is different from the first one. For each *training epoch* (certain number of passes through the training set) a resulting set of weights are recorded and the value of E (eq. 4), ARV (eq. 6) and ρ (eq. 7) are then calculated as an evaluation of the neural network's performance. The best results of the training process are those which give the lowest value of the E, the smallest value of the ARV and the biggest ρ.

3 Prediction of Chaotic Time Series

According to *Takens* (13), there exists a smooth function of at most $2d+1$ past measurements of a temporal series that allows the correct prediction of its future value, and the prediction is just as good as the one it would be obtained if we had been able to solve the complete system with its all degrees of freedom (2; 13). What the theorem of Takens does not provide is the explicit form of the function which would contain the desired extrapolation and it is in this context that neural networks can be successfully used. By setting the delay coordinates of the temporal series $x(t)$: $\vec{v}_D = (x(t), x(t-\tau), \ldots, x(t-(m-1)\tau))$ as the input pattern $\{x_i^\mu\}$ in equation 1, and choosing $x(t+\Lambda)$ as the known target O_s^μ in equation 4, the network can be trained to predict the future state of the system at a time Λ, which corresponds to a certain number of iterations, or time steps. Mathematically:

$$\{x(t), x(t-\tau), x(t-2\tau), ..., x(t-(m-1)\tau)\} \quad \longrightarrow \quad x(t+\Lambda), \tag{8}$$

where τ is the time delay.

In order to investigate the capability of neural networks in predicting the future state of the chaotic systems, which is the main purpose of this work,

the Lorenz system was initially used. The time series was obtained from the numerical solutions yielded by the three ordinary differential equations (14):

$$\begin{aligned} \dot{x} &= \sigma x + \sigma y \\ \dot{y} &= -xz + rx - y \\ \dot{z} &= xy - bz \end{aligned} \tag{9}$$

For this work, the time series chosen to feed the neural network with were the ones related to the solutions in x with $\sigma = 3.0$ (*Prandtl number*), $r = 26.5$ (*Rayleigh number*) and $b = 1.0$.

Initially, an extensive set of tests were carried out by using many different architectures and also different conditions for training the net and forecasting. For the training process, the series (1,0,0) [2] was used, and the validation process was carried out by using the (0,2,0) series with the following parameters: $\eta = 0.001$, $\alpha = 0.0$ and $\tau = 6$, over 100 training epochs. Finally, the prediciton process was done over the series (1,1,1).

Basically, two different approaches, based on different types of architecture, were chosen to test the prediction power of neural networks. Firstly, we tried using the same architecture (*15:9:3:1*) that had been successfully used to forecast plasma disruptions in tokamaks (10). However, the neural network did not allow good results to the chaotic Lorenz series with this type of architecture. Some other different architectures were also tried but the best results were obtained when the embedding dimensiom m of the dynamical system (in the proprotion *m:2m:m:1*) was taken into account. Therefore, as an initial guess, we started using the architecture *3:6:3:1*, since the embedding dimension for this type of dynamical system is known to be 3 (2; 14).

With this type of architecture we were then able to successfully predict the future state of the system for a wide range of different time steps. For up to time step 15 ($\Lambda = 0.3$), for example, the result of the forecasting process is practically perfect when compared to the real data, as shown in figure 2-a. For time steps up to 25 ($\Lambda = 0.5$) the match of the neural network output with the actual data can also be considered to be very good, as shown in figure 2-b. Although the fitting is not so perfect, in this case, the chaotic pattern of the Lorenz series was still very well predicted by the neural network, and our results can be considered much better than the ones reported elsewhere (8; 9). *Koga*, for instance, obtained good predictions only up to 3 time steps, when neural networks were used with a type of architecture different from ours. On the other hand, *Diambra* and *Plastino* were able to obtain good predictions

[2] The set (x_0,y_0,z_0) represents the initial conditions chosen to solve the system given by equation 9.

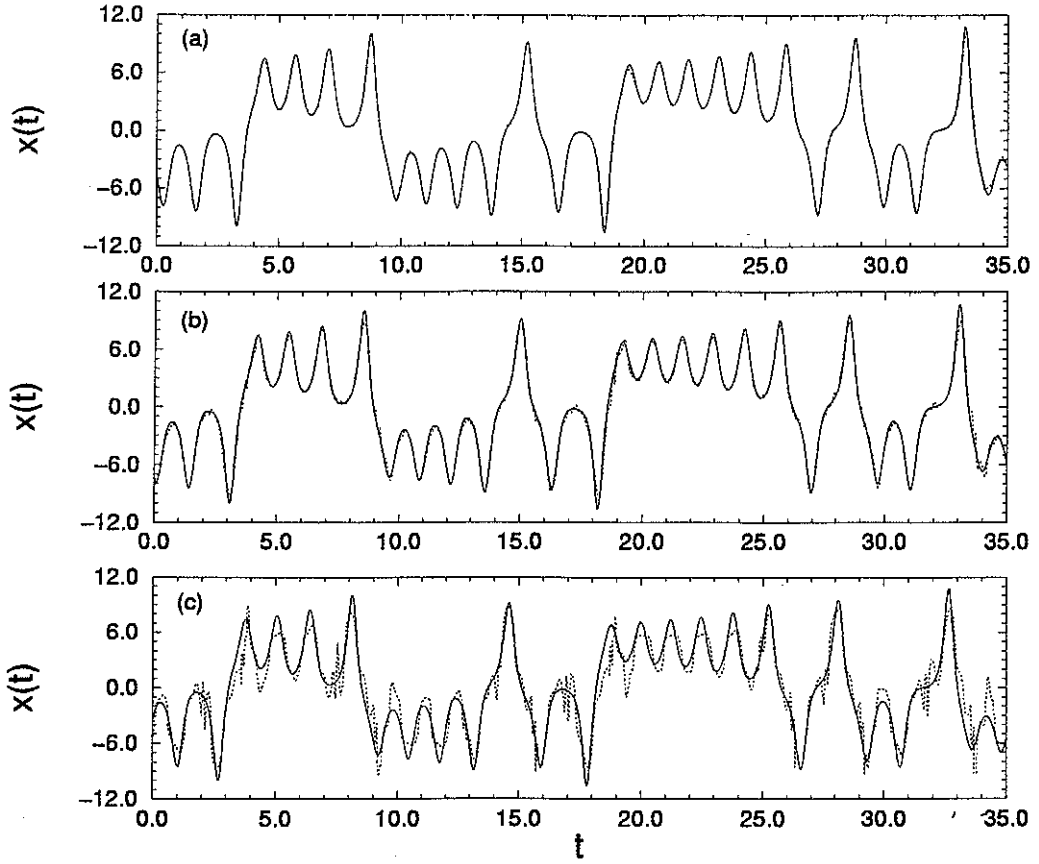

Fig. 2. Prediction of the Lorenz series using the architecture *3:6:3:1* (dotted line) as compared with the real data (solid line) for (a) 15 time steps, (b) 25 time steps and (c) 45 time steps. As it can be observed, the prediction is pratically perfect for 15 time steps, which indicates the network has successfully learned the dynamic of the system.

for up to 10 time steps, by using another method of time series predictions.

As it could be observed from our simulations, if the predictions is carried out for time steps greater than 25, the output data from the net start deviating from the real data but, the pattern of the Lorenz series is still reasonably well predicted by the net, as exemplified in figure 2-c for 45 time steps ($\Lambda = 0.9$).

By increasing the time steps prediction even further, no good results are obtained. However, by slightly changing the architecture used, i.e, by increasing the m number of neural units in the input and hidden layers we were able to improve these results a little. Figure 3-b, for instance, shows the forecasting results obtained with the architecture *4:8:4:1* for 45 time steps. The match between the two sets of data can be considered slightly better in this case than the results obtained with the previous architecture *3:6:3:1* (fig. 3-a). In short, from these results we can conclude that neural networks can be effectivelly used to predict with success the chaotic behavior of Lorenz series.

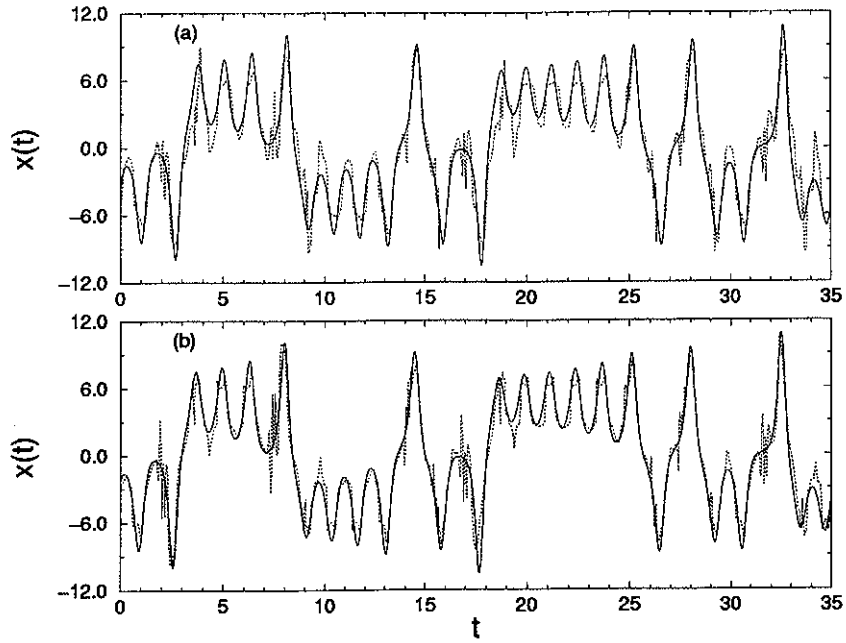

Fig. 3. Prediction of the Lorenz series using (a) the architecture *3:6:3:1* ($ARV = 0.07$, $\rho = 0.96$) as compared with (b) the architecture *4:8:4:1* ($ARV = 0.05$, $\rho = 0.98$). In both figures, the solid line represents the x component of Lorenz system and the dotted line the predictions of the neural network for 45 time steps. As it can be observed, the predictions in (b) are slightly better than those obtained in (a).

As an easy way to evaluate the prediction performance of the neural network we plotted the ARV and ρ parameters against the number of time steps predictions. The results obtained are shown in figures 4-a and 4-b, respectively, for the architecture *3:6:3:1*. From these figures it is observed that the degradation of the prediction power of the neural networks tends to exponentially increase as the number of time steps increases, as one should expect. It is important to note, however, that the prediction results can be considered excellent up to 15 time steps. For bigger values than that, the fitting between the actual data and the results predicted by the net are not so perfect, but the neural networks still predict reasonably well the general chaotic pattern of the Lorenz series.

As it can be observed in figure 5 the best forecasting results are obtained when the *m:2m:m:1* type of architecture is used.

Finally, the Henon and Logistic maps were also used as further evaluations of the forecasting power of neural networks, with the particular set of units (*m:2m:m:1*) chosen as an initial guess for the ideal architecture. As it is well known, the Henon map is described by the following equations (14):

$$\begin{aligned} x_{n+1} &= a - x_n^2 + by_n \\ y_{n+1} &= x_n \end{aligned} \tag{10}$$

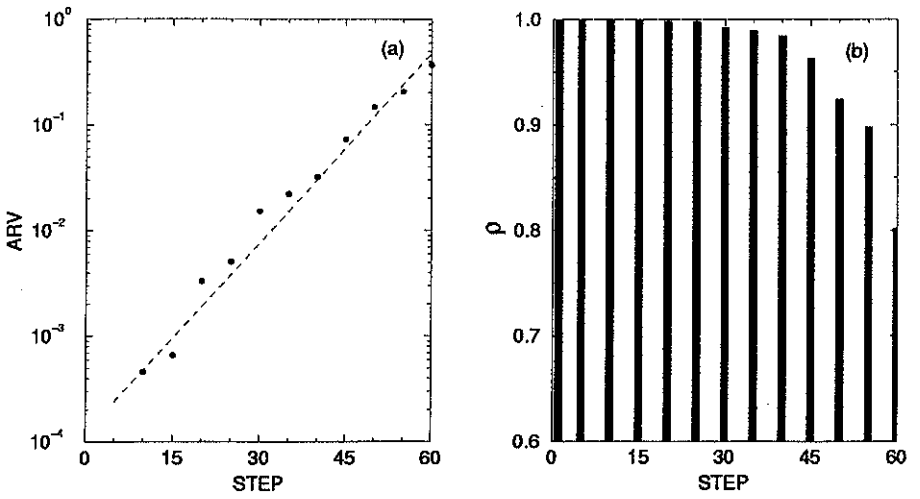

Fig. 4. Plots showing (a) the predicting error ARV and (b) correlation coefficient for the Lorenz system. As noted, the performance of the network tends to degrade exponentially as the time step increases, and the result is very good up to 15 time steps prediction.

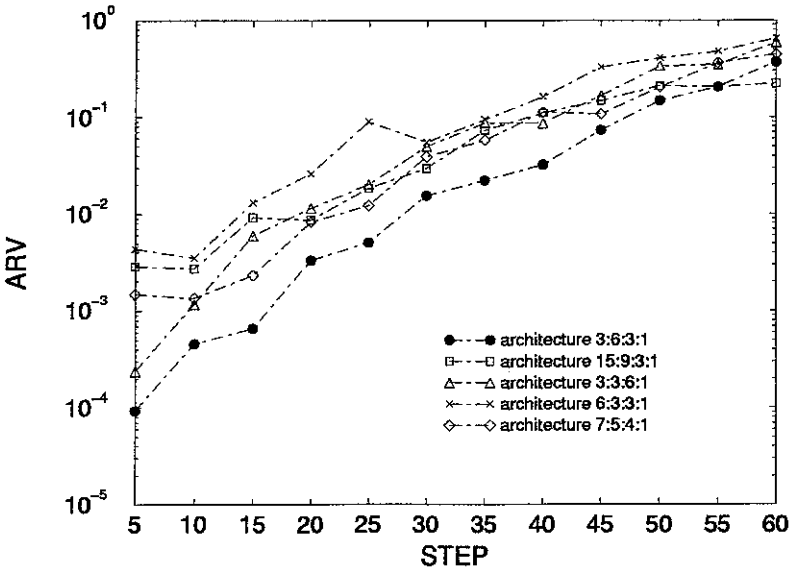

Fig. 5. Different architectures used to forecast series from Lorenz system. As noted, the best configuration is *3:6:3:1*.

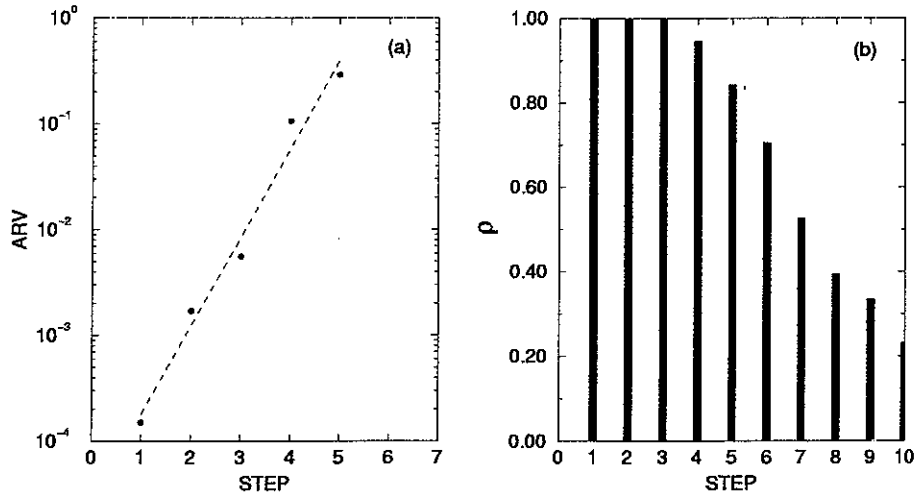

Fig. 6. Plots showing (a) the predicting error ARV and (b) correlation coefficient for Henon map. As noted, the result is very good for up to 3 time steps prediction.

The chosen time series used was obtained by iterating the map in relation to y with the following parameters: $a = 1.4$ and $b = 0.3$. Since the *embedding dimension* for this chaotic system is known to be 2 (*m=2*), the initial architecture was chosen to be *2:4:2:1* (14).

The sequencial data used to feed the net and perform the training was obtained from the initial conditions (0.5, 1.0), chosen at will. The validation process was carried out for the series (0.6, 1.2) and the net training parameters were $\eta = 0.001$, $\alpha = 0.0$ and $\tau = 1$, over 100 training epochs. Finally, the series $(-0.5, 1.0)$ was also arbitrarially chosen and used in the prediction process.

With this type of architecture we were able to predict the future state of the system, with a large amount of success, only for up to 3 iterations. This result is in accordance with the ARV and ρ parameters shown in figure 6. The rather short time step predictions obtained in this case should not be a surprise, since this type of system is known to be very chaotic (Lyapunov coeficient: $k = 0.42$) (14). Different types of architectures, with different relations between the number of neural units allocated for different layers (other than *m:2m:m:1*), were also tried out but without good results. When the architecture *6:12:6:1* was chosen and the inertial term was changed to 0.9, the results could be improved, as it can be seen in figure 7.

On the other hand, the time series used for the forecasting process related to the Logistic map were constructed by iterating the equation:

$$x_{n+1} = ax_n(1 - x_n) \tag{11}$$

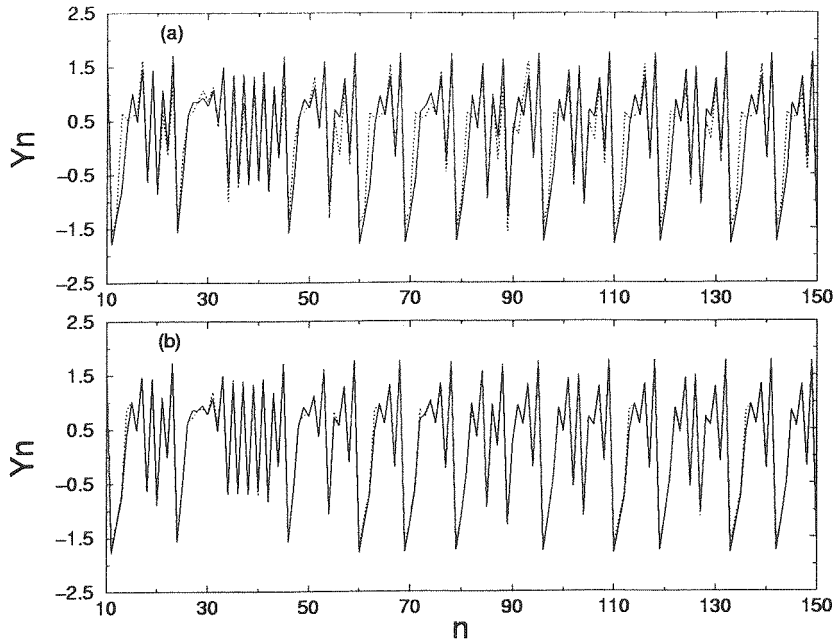

Fig. 7. Results obtained for the y component of the Henon map by using two different architectures: (a) *2:4:2:1* and (b) *6:12:6:1*. In these figures, the solid line corresponds to the Henon map and the dotted line is the prediction of the neural network corresponding to 3 iterations.

for which two different values for the parameters a were chosen: $a = 3.7$ ($k = 0.35$) and $a = 4.0$ ($k = 0.69$) (14). The architecture used in both situations was *4:8:4:1*.

Very good forecasting results were obtained up to 3 time steps for $a = 3.7$ and up to 2 time steps for $a = 4.0$, as shown in figures 8-a and 8-b, respectively. Identically good forecasting results were also obtained for other series constructed with different initial conditions, without the necessity of training the net again; that is, once the net has learned the dynamic of the system and the best set of weights is obtained, the neural net is able to yield the same quality of results, for any given initial conditions. This is the significant advantage over other methods usually used in chaotic time series prediction as the *local approximation* (15).

One way to evaluate in advance the capability of the neural net to make predictions in time for a given series can be done by analysing the *autocorrelation* of the actual data. In figure 9 some of the autocorrelations obtained for the time series discussed in this article are plotted. It is clear from these figures that long time steps predictions (~ 40) can only be expected for Lorenz system.

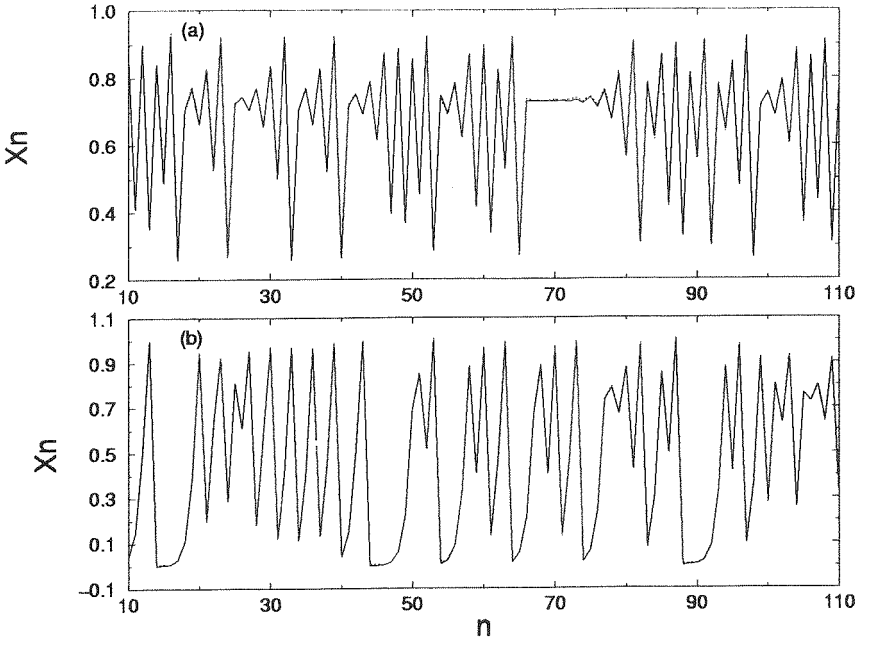

Fig. 8. Forecasting results (dashed line) obtained for the Logistic map (solid line) for two different values of a: (a) $a = 3.7$, with 3 time steps prediction and (b) $a = 4.0$, with 2 time steps prediction.

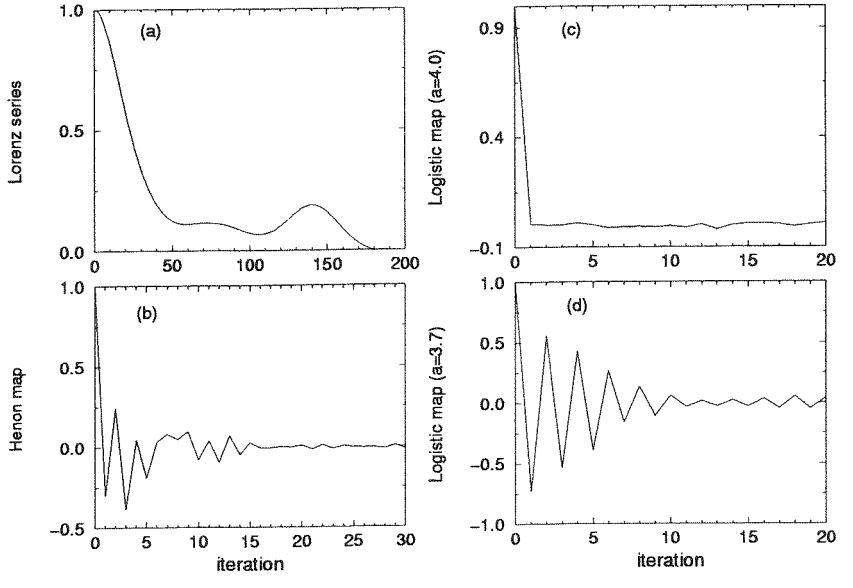

Fig. 9. Autocorrelation curves of the chaotic series used in this work.

4 Conclusions

It has been shown in this work that neural network can be successfully used to predict the future state of nonlinear systems as the ones described by the Lorenz equations and Henon and Logistic maps. For all these cases, it has been verified that the architecture *m:2m:m:1* is an initial good guess to construct the neural network architecture for predicting the future state of chaotic temporal series. The reasonably good results obtained here clearly suggests that there is an interplay between the architecture of a multilayer network with the embedding dimension m of the time series that is under investigation. This conclusion can be very useful if we consider that, when dealing with neural networks, a large number of training data sets and interactions are generally required to find the right architecture and, therefore, reliable and faster convergence process for the neural networks are mostly welcome. The prediction approach used herein has shown to have considerable advantages in terms of quality of the results and flexibility over the more conventional methods reported in other published articles. Thus, the results obtained here are encouraging and neural networks can be considered an important tool for making predictions in time of nonlinear systems.

References

[1] R. Rojas, *Neural Networks, A systematic Introduction*, Springer (1996).

[2] J. Hertz, A. Krogh and R.G. Palmer, *Introdution to the Theory of Neural Computation*, Lecture Notes volume I, Addison-Wesley Publishing Company (1991).

[3] G.E. Hilton, *Sci. Am.* September (1992) 105.

[4] D. Ruelle, F. Takens, Comm. Math. Phys. **20** (1971) 167.

[5] H. Haken, Phys. Lett. **A53** (1975) 77.

[6] K. Tomita et al., J. Stat. Phys. **21** (1979) 65.

[7] D. Russell et al., Phys. Rev. Lett. **45** (1980) 1175.

[8] J. Koga, *Notes on Basic Science*, Jaeri-Japan **3** (2) July (1996) 37.

[9] L. Diambra, A. Plastino, *Phys. Lett. A* **216** (1996) 278.

[10] A. Vannucci, K.A.de Oliveira, T. Tajima, *Nuc. Fus.* **39** (2) (1999) 255.

[11] K.A.de Oliveira, A. Vannucci, E.C.da Silva, in preparation.

[12] J.V. Hernandez, A. Vannucci, et al, *Nuc. Fus.* **36** (8) (1996) 1009.

[13] F.Takens, *Lectures Notes in Mathematics*, Springer-Verlag **898** (1981).

[14] K.T. Alligood, T.D. Sauer and J.A. Yorke, *CHAOS - An Introduction to Dynamical Systems*, Springer (1996).

[15] J. D. Farmer and J. J. Sidorowich, Phys. Rev. Lett. **59** (8) (1987) 845.