# Improving multi-step time series prediction with recurrent neural modelling

I.M. Galván, J.M. Alonso and P. Isasi
*Dep. of Computer Science. Universidad Carlos III, Av. de la Universidad 30, 28911 Leganés, Madrid, Spain*

**Abstract.** Multi-step prediction is a difficult task that has been attracted increasing the interest in recent years. It tries to achieve predictions several steps ahead into the future starting from information at time k. This paper is focused on the development of nonlinear neural models with the purpose of building long-term or multi-step time series prediction schemes. In these context, the most popular neural models are based on the traditional feedforward neural network. However, these kind of models may present some problems when a long-term prediction problem is formulated. In this paper, a neural model based on a partially recurrent neural network is proposed as an alternative. For the new model, a learning phase with the purpose of long-term prediction is imposed, which allows to obtain better predictions of time series in the future. This recurrent neural model has been applied to the logistic time series with the aim to predict the dynamic behaviour of the series in the future. Models based on feedforward neural networks have been also used and compared against the proposed model.

## 1. Introduction

The ability to forecast the behaviour of a system hinges, generally, on the knowledge of the laws underlying a given phenomenon. When this knowledge is expressed as a solvable equation, one can predict the behaviour along the future once the initial condition is given. However, phenomenological models are often unknown or extremely time consuming.

Nevertheless, it is also possible to predict the dynamic behaviour of the system along the future by extracting knowledge from the past. We are interested in time series processes which can be viewed as generalized nonlinear autoregressive models, also named NAR models. In this case, the time series behaviour can be captured by expressing the value x(k+1) as a function of the d previous values of the time series, x(k),...,x(k-d), that is:

$$x(k+1) = F(x(k), \ldots, x(k-d)) \quad (1)$$

where k is the time variable and F is some function defining a very large and general class of time series. This function can be very complex and its explicit form is usually unknown.

The standard prediction method involves approximating the function F in such a way that the model given by eq. 1 allows to predict or find the sequence x(k+1),x(k+2),x(k+3),... starting from the observed sequence at the current time k, x(k),...,x(k-d). In many time series applications, one-step prediction schemes are used to predict the next sample of data based on previous samples. However, one-step prediction may not provide enough information, specially in situations where a broader knowledge of the time series behaviour

can be very useful or in situations where it is desirable to anticipate the behaviour of the time series process.

The present study deals with long-term or multi-step prediction, i.e. how to achieve predictions several steps ahead into the future starting from information at time k. Hence, the goal is to approximate the function F such that the model given by eq. 1 can be used as a multi-step prediction scheme.

The neural models most widely used in time series applications are built up using multilayer feedforward neural networks [1-3]. In this paper, two classical neural models are reviewed analysing their advantages and disadvantages when they are used for multi-step prediction purpose. The first classical model reviewed consists in approximating the function F by a multi-layer feedforward network; after that, the output of the network is fed back into the input and the model is used to predict the behaviour of the time series along the interval [k+1,k+h+1], where h is a natural number named *prediction horizon*. In the second neural approach reviewed in this paper, a multilayer feedforward neural network is used to predict the time series value at instant k+h+1 from the information available at the current instant k. Both neural models may not produce efficient predictions along the future. The first model is not trained with a multi-step prediction purpose and the second model may not have enough information through the input to predict the future.

In order to solve the troubles of classical models for multi-step prediction, a new recurrent neural model is proposed in this paper. It is based on a partially recurrent neural network and consists of adding feedback connections from the output neuron to the input layer. In this case, the parameters of the model are determined to minimise the error along interval [k+1,k+h+1]. Thus, the model is trained with the purpose of long-term prediction and better predictions than classical feedforward neural models may be expected.

## 2. Classical neural models for multi-step prediction

Neural networks have proved their ability in NAR model prediction. The neural models most widely used in time series applications are based in feedforward neural networks with backpropagation learning algorithm. The main difference between them arises in the way of gathering the data from the time series and how to compute the learned network for prediction. Here, we review two models (Model1 and Model2) outlining theirs advantages and disadvantages for a multi-step prediction purpose.

Model1 consists of approximating the function F appearing in eq. 1 by a multilayer feedforward neural network as follows:

$$\tilde{x}(k+1) = \tilde{F}(x(k),\ldots,x(k-d),W_1) \tag{2}$$

where $W_1$ is the parameter set of the model, which is obtained using the backpropagation algorithm [4]. The update of the parameter set is based on the local difference between the measured and predicted values, i.e.:

$$e(k+1) = \frac{1}{2}\cdot(x(k+1)-\tilde{x}(k+1))^2 \tag{3}$$

When the model given by eq. 2 has to predict the behaviour of time series in the future, i.e. along the interval [k+1, k+h+1], its structure has to be modified. Therefore, the

predictive network output must be fed back as an input for the next step prediction and all the remaining input neuron values are shifted back one unit, i.e.,

$$\tilde{x}(k+1) = \tilde{F}(x(k),\ldots,x(k-d), W_1) \tag{4}$$

$$\tilde{x}(k+2) = \tilde{F}(\tilde{x}(k+1), x(k),\ldots,x(k-d+1), W_1) \tag{5}$$

...

$$\tilde{x}(k+h+1) = \tilde{F}(\tilde{x}(k+h),\ldots\tilde{x}(k+1), x(k),\ldots,x(k-d+h), W_1) \tag{6}$$

The main disadvantage of Model1 in the context of multi-step prediction is that the parameter set has been obtained with the purpose of one-step prediction, i.e. to minimise the local errors given by eq. 3. During the training phase, the model captures the relation between the actual observations of the original time series, x(k),...,x(k-d) and the next sampling time, x(k+1). However, when the model is acting as a multi-step prediction scheme (see eqs. 4-6) a group of the input neurons receives the earlier approximated values, $\tilde{x}(k+h),\ldots\tilde{x}(k+1), x(k),\ldots,x(k-d+h)$. This fact may produce a non desired behaviour of the model because errors occurred at some instant are propagated and magnified to future sampling times. This effect decreases the capability of these neural models to predict the future.

An alternative neural model structure to be considered (Model2) consists of using a multilayer feedforward network in order to predict, directly, the time series value at instant k+h+1 from the information available at the current instant k, x(k),...,x(k-d), instead of using the immediate d previous values as in Model1 (see eq. 2). In this case, the nonlinear model becomes as follows:

$$\tilde{x}(k+h+1) = \tilde{F}(x(k),\ldots,x(k-d), W_2) \tag{7}$$

where h is the prediction horizon. The set of parameter $W_2$ is updated using the backpropagation algorithm, following the negative gradient direction of the error measured at instant k+h+1:

$$e(k+h+1) = \frac{1}{2}\cdot(x(k+h+1) - \tilde{x}(k+h+1))^2 \tag{8}$$

This second neural approach provides directly the prediction of the time series at instant k+h+1 from the information at instant k (see eq. 7). The inputs to the network, when the model is used to predict some steps in the future, are the real measured time series values. Now, the predictions of the network are not fed back into the input, as in the previous model (Model1). Thus, the problem concerning the propagation of errors disappears when Model2 is used as a nonlinear multi-step prediction scheme.

A disadvantage of Model2 is relative to the model structure. The inputs to the model may not own enough information about the time series in order to predict the instant k+h+1. This is, the input vector, $x(k),\ldots,x(k-d)$, can be very distant in time from the prediction horizon, k+h+1, and may not have any relation with that instant. In this case, Model2 should not be used for multi-step prediction. This structure has only sense when there exists a relation between the information available at the current instant and the prediction horizon. In many real cases, it is not possible to know previously the relationship between the data in the series, and therefore whether the model is appropriate for the prediction problem.

## 3. Recurrent neural model for multi-step prediction

The recurrent neural model proposed in this paper is presented as an alternative to classical neural models when the goal is to predict the future behaviour of time series in some prediction horizons. Basically, the Recurrent Model consists of imposing a special learning phase with the purpose of long-term prediction.

The Recurrent Model is based on a partially recurrent neural network [5]. The network consists of adding feedback connections to a multilayer feedforward neural network from the output neuron to the input layer. The number of recurrent connections depends on the prediction horizon value. If the horizon is h, the input layer of the network is formed by a group of h neurons that memorize previous network outputs; generally, these neurons are called *context neurons*. The remaining neurons in the input layer receive the original or measured time series data (see fig. 1). When the prediction horizon, h, is higher than the number of external input neurons, d+1, all input neurons of the network are context neurons and no measured time series value is fed into the network.

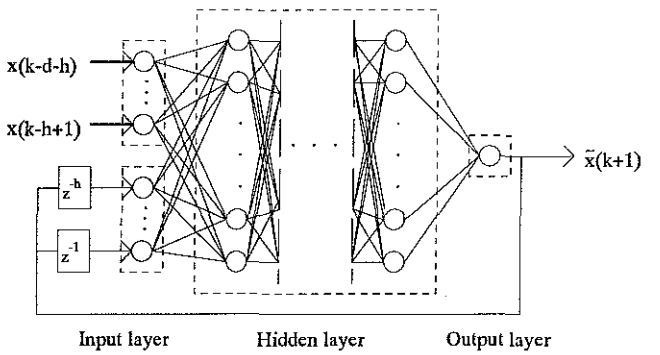


Figure 1. Partially recurrent neural network

In the context of multi-step time series prediction, the training procedure of the partially recurrent neural network is carried out as follows:

At each instant k+1, starting with k=d,

Step 1. The neurons in the input layer receive the measured sequence $x(k),\ldots,x(k-d)$. Hence, in the first step the number of context neurons is zero and the network output is given by:

$$\tilde{x}(k+1) = \tilde{F}(x(k),\ldots,x(k-d), W_3) \tag{9}$$

Step 2. The number of context neurons is increased by one unit; this neuron memorizes the previously calculated output of the network, $\tilde{x}(k+1)$. Thus, the prediction at instant k+2 is given by:

$$\tilde{x}(k+2) = \tilde{F}(\tilde{x}(k+1), x(k),\ldots,x(k-d+1), W_3) \tag{10}$$

Step 3. Step 2 is repeated until h context neurons are achieved. When the instant k+h+1 is reached, the output of the Recurrent Model is given by:

$$\tilde{x}(k+h+1) = \tilde{F}(\tilde{x}(k+h),\ldots\tilde{x}(k+1), x(k),\ldots, x(k-d+h), W_3) \quad (11)$$

Step 4. At this moment, the parameter set of the Recurrent Model, $W_3$, is updated. In order to impose a training phase with the purpose of long-term prediction, the learning is based on the sum of the local errors along the prediction horizon, i.e. along the interval [k+1, k+h+1]. Hence, the parameter set $W_3$ is updated following the negative gradient direction of the error function given by:

$$e(k+1) = \frac{1}{2}\cdot\sum_{i=1}^{h}(x(k+i+1) - \tilde{x}(k+i+1))^2 \quad (12)$$

Since the internal structure of the partially recurrent network is like a feedforward neural network, the training can be realised using the traditional backpropagation algorithm, although other extensions of this algorithm should be feasible [6-7].

Step 5. At this point the time variable k is increased by one unit and the procedure returns to step 1. The procedure finalises when the instant k=N-h is reached, where N stands for the number of patterns.

The structure of the Recurrent Model (eqs. 9-11) is identical to the structure of Model1 when it is used for prediction (eqs. 4-6). However, there exists an important difference between them: the way to obtain the parameter sets of the models. This is, the learning procedure of the system.

As it was said before, the parameter set $W_1$ of Model1 is obtained training a multilayer feedforward network and remains fixed during the prediction phase. This means that the parameter set $W_1$ is updated using the local error measured at each instant (eq. 3). When the Recurrent Model is used, the update of the parameters at each instant is based on the measured error along the prediction interval [k+1, k+h+1]. Thus, the set of parameters $W_3$ has been determined to minimize the error in the future (eq.12). In consequence, the Recurrent Model is trained in such a way that it acts as a multi-step prediction scheme as opposed to Model1 (eq. 2) which is trained to predict exclusively the next sampling time (one-step prediction scheme).

Due to the recurrent structure of the proposed model, errors occurred at the same instant are propagated into the next sampling time as usual in Model1. However, in the Recurrent Model the propagated errors are reduced during the training phase because the learning is carried out using the predicted output at earlier time steps. Thus, the errors are corrected and better predictions in the future may be expected.

## 4. Experimental verification

The simulations have been conducted and applied to the map of the form:

$$x(k+1) = \lambda\cdot x(k)\cdot(1 - x(k)) \quad (13)$$

with $\lambda = 3.97$ and $x(0)=0.5$. This map describes a strongly chaotic time series which is called logistic time series.

Two different structures of NAR models have been used to predict the logistic time series. From equation 11, it follows that the logistic map at instant k+1 depends on the value at instant k. Hence, the first experiment has been realised using the following equation:

$$\textit{Experiment 1}:\ x(k+1) = F(x(k)) \tag{14}$$

As the ultimate goal in this paper is to predict the future in an horizon greater than one, it is suitable to consider NAR models that own more information about the past behaviour of the time series. Thus, a second experiment has been carried out, which is based on the following structure:

$$\textit{Experiment 2}:\ x(k+1) = F(x(k), x(k-1), x(k-2)) \tag{15}$$

In order to train the neural models, data of the logistic time series from t=0 to t=100 are used. A different data set corresponding to t=100 through t=500 has also been used as test patterns.

The capability of neural models to predict the future has been evaluated using the following error function, also called prediction error:

$$E = \frac{1}{2N} \cdot \sum_{k=0}^{N-h} (x(k+h+1) - \tilde{x}(k+h+1))^2 \tag{16}$$

where h is the prediction horizon and N is the number of patterns. In this work, four predictions horizons have been used as test cases, h=0, h=1, h=2 and h=3.

For each experiment (eqs. 14 and 15), the three previously described models (Model1, Model2 and Recurrent Model) have been tested, which implies several run simulations. To test Model1, the parameters have been determined to approximate the functional F in eqs. 14 and 15 using multilayer feedforward neural networks with 10 units in the hidden layer; after that, the respective models have been used to predict the logistic time series for several prediction horizons, h=0, h=1, h=2 and h=3. In the second approach (Model2), four neural networks with 10 hidden units have been trained for each experiment. The training is carried out to approximate the logistic time series at instants k+1, k+2, k+3, k+4 (see eq. 7). Finally, the functional F in eqs. 14 and 15 have also been approximated using the recurrent neural network and the learning algorithm presented in section 3. The recurrent neural networks have 10 hidden units and the number of context neurons is varying form 0 to 3. The prediction errors for each experiment and for each model approach over the training data set are presented in Table 1.

| | ***Experiment 1.* Input: (x(k))** | | | ***Experiment 2.* Input: (x(k),x(k-1),x(k-2))** | | |
|---|---|---|---|---|---|---|
| **Prediction horizons** | **Model 1** | **Model 2** | **Recurrent Model** | **Model 1** | **Model 2** | **Recurrent Model** |
| **h=0** | 0.00125 | 0.00125 | 0.00125 | 0.00089 | 0.00089 | 0.00089 |
| **h=1** | 0.00823 | 0.00188 | 0.00544 | 0.00790 | 0.00154 | 0.00310 |
| **h=2** | 0.04149 | 0.04817 | 0.01489 | 0.03925 | 0.03445 | 0.00571 |
| **h=3** | 0.09592 | 0.09805 | 0.08135 | 0.05510 | 0.06756 | 0.00701 |

Table 1. Prediction Errors over the training data set

In this table it can be observed that there are not apparent differences when learning the training set between Model1 and Model2 as the prediction horizon increases (h=2 and h=3). However, the Recurrent Model has a considerable accuracy on the training set for these prediction horizons. When the prediction horizon is fixed to two sampling times, Model2 and Recurrent Model provide the best approximations.

In some cases, an excessive adaptation to the training set affects the generalization ability of the models. In order to validate this assumption, the prediction errors have been evaluated over the test set. The results are shown in Table 2.

| | *Experiment 1.* **Input: (x(k))** | | | *Experiment 2.* **Input: (x(k),x(k-1),x(k-2))** | | |
|---|---|---|---|---|---|---|
| **Prediction horizons** | **Model 1** | **Model 2** | **Recurrent Model** | **Model 1** | **Model 2** | **Recurrent Model** |
| **h=0** | 0,00154 | 0,00154 | 0,00154 | 0,00152 | 0,00152 | 0,00152 |
| **h=1** | 0,01006 | 0,00347 | 0,00586 | 0,00904 | 0,00358 | 0,00464 |
| **h=2** | 0,04667 | 0,07866 | 0,01809 | 0,04807 | 0,04961 | 0,00784 |
| **h=3** | 0,11900 | 0,09876 | 0,09592 | 0,07827 | 0,10371 | 0,01123 |

Table 2. Prediction Errors over the test data set

In some cases it seems that the superiority of the Recurrent Model is not relevant, for instance in *Experiment 1* with h=3 (see Tables 1 and 2). This could be due to the quadratic error is not always the best way to measure the predictive capability of a system (compare in the following graph the predicted values for Model 1 and Model 2 which have nevertheless similar quadratic errors).

In order to see the predictive behaviour of neural models some graphical results are presented. In figure 2 the predictions of the logistic time series provided by the first experiment for the prediction horizon h=3 are shown. The predictions for h=3 provided by the second experiment are shown in figure 3.

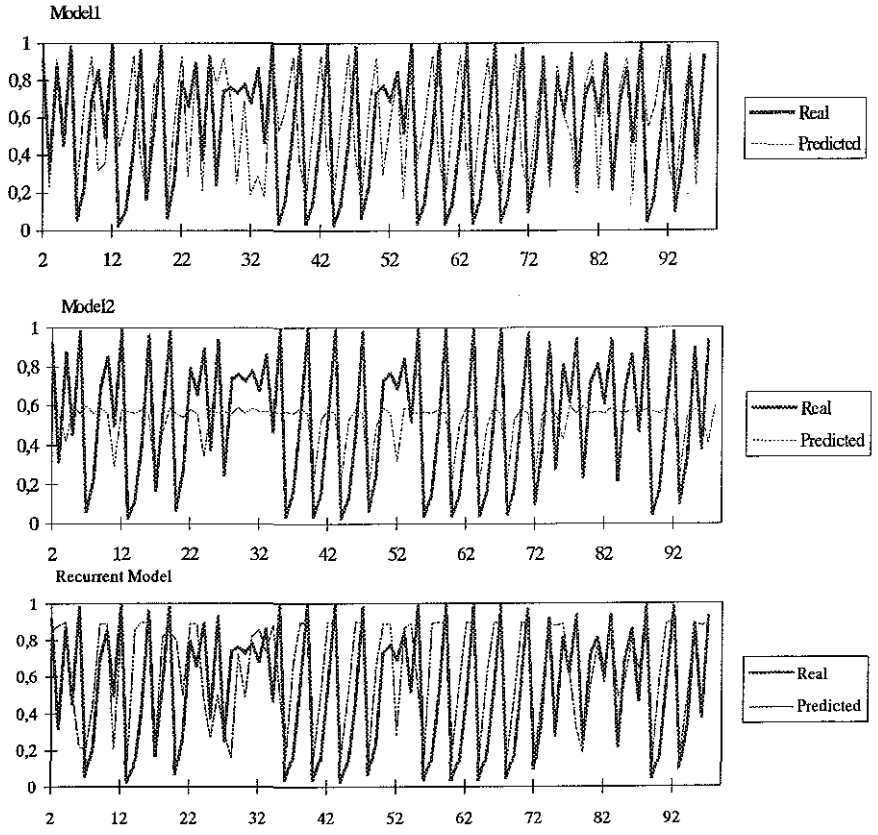


Figure 2. Predictions of logistic time series for h=3. *Experiment 1.*

As it is possible to observe in figure 2, Model2 does not capture the temporal behaviour of the logistic time series. Model1 and the Recurrent Model do not provide the most appropriate predictions, however, their performance is higher, being more accurate the Recurrent Model.

The superiority of the Recurrent Model for large prediction horizons is more evident for the *Experiment 2*, as shown in figure 3. In this case, the predictions provided by Model1 and Model2 are very poor, while the Recurrent Model is able to obtain convenient long-time predictions.

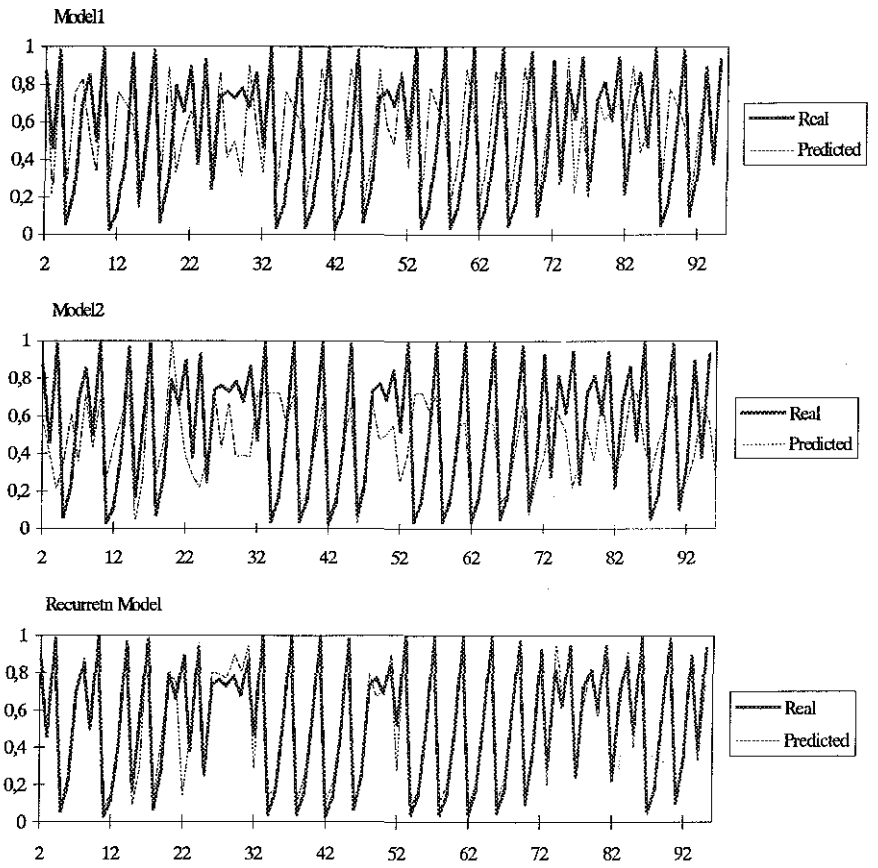


Figure 3. Predictions of logistic time series for h=3. *Experiment 2*.

## 5. Discussion and conclusions

From the experimental results we can conclude that the second structure (*Experiment 2*) is more adequate to predict the future of the logistic time series because the model has more information about the behaviour of the time series through the extended number of input neurons. The three approaches studied in this work provide better approximations when this structure is used (see Table1 and Table2).

The *Experiment 1* is able to predict the future when short prediction horizons are defined. However, when the prediction horizon is increased, the performance of this structure decreases. When the prediction horizon is fixed to four sampling times, classical neural models (Model1 and Model2) and Recurrent Model do not provide appropriate predictions. This is due to the fact that these models do not own enough information about the time series. Hence, in multi-step prediction, the number of inputs have an important significance on the quality of predictions.

Assuming that the structure of NAR model has enough information through the input in order to predict the future, the immediate question that arises concerns the choice of the neural approach to be used. The results presented in the previous section show that models based on the recurrent neural network provide better approximations than models built up with multilayer feedforward neural networks. The recurrent neural model has been trained with the purpose of multi-step prediction which seems to be a better approach.

Furthermore, it is pointed out that the improvement of recurrent neural models over classical ones is more relevant when the prediction horizon is increased. For short prediction horizons (h=1), the performance of the second neural approach (Model2) is slightly better that the Recurrent Model. In this case, Model2 is able to provide appropriate predictions because the inputs of the network, (x(k), x(k-1), x(k-2)), own enough information to approximate the time series value at instant k+2. However, when the prediction horizon is increased, the best approximations are provided by the Recurrent Model and its superiority is more evident.

In consequence, for short prediction horizons Model2 may be more suitable because it supplies acceptable predictions and is easier to train. However, if the prediction horizon increases, the most convenient performance is provided by the Recurrent Model.

## References


[1] A.S. Weiggend, B.A. Huberman and D.E. Rumelhart: Predicting the future: A Connectionist Approach. Technical Report: PARC-SSL-90-20, April, 1990.

[2] M. Cottrell, B. Girard, Y. Girard, M. Mangeas and C. Muller: Neural Modeling for Time Series: A Statistical Stepwise Method for Weight Elimination. *IEEE Transations on Neural Networks*, **6**, Nº 6, (1995) 1355-1363.

[3] Hee-Yeal Yu and Sung-Yang Bang: An Improved Time Series Prediction by Applying the Layer-by-Layer Learning Method to FIR Neural Networks. *Neural Networks*, **10**, Nº 9 (1997) 1717-1729.

[4] D. Rumelhart, G. Hinton and R.J. Williams: Learning Internal Representations by Error Propagation, in *Parallel Distributed Processing*, MIT Press, Cambridge 1986.

[5] I.M Galván and J.M. Zaldívar: Application of recurrent neural networks in batch reactors. Part I. NARMA Modelling of the dynamic behaviour of the heat transfer fluid temperature. *Chemical Engineering and Processing*, **36** (1997) 505-518.

[6] K. S. Narendra and K. Parthasarathy: Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Trans. on Neural Networks*, **2** (1991) 252-262.

[7] D. Barrios, I.M. Galván, P. Isasi and J. Ríos: "NARMA recurrent neural modelling for controlling chemical batch reactors". Technical Report. UC3M-TR-CS-97-3. Department of Computer Science, Universidad Carlos III de Madrid, 1998.