Proceedings of the 8th WSEAS International Conference on APPLIED MATHEMATICS, Tenerife, Spain, December 16-18, 2005 (pp386-391)

Predicting time series with advanced hybrid systems

O. Valenzuela[1], I.Rojas[2], F.Rojas[2], H.Pomares[2], J.Gonzalez[2],LJ.Herrera[2], A.Guillen[2]
1: Department of Applied Mathematic
2: Department of Computer Architecture and Computer Technology
University of Granada (Spain)

Abstract.- Autoregressive moving average (ARMA) has been widely used to model processes that generate linear time-series. Recent research activities in forecasting with artificial neural networks (ANNs) suggest that ANNs can be a promising alternative to the traditional ARMA structure. These linear models and ANNs are often compared with mixed conclusions in terms of the superiority in forecasting performance. This study was designed: a) to investigate a hybrid methodology that combines ANN and ARMA models; b) to resolve one of the most important problems in time series using ARMA structure and Box-Jenkins methodology, the identification of the model. In this paper we present a new procedure to predict time series using paradigms as: fuzzy system, neural networks and evolutionary algorithm. Our goal is to obtain an expert system based on paradigms of artificial intelligence, so that the linear model can be identified automatically, without the necessity for a human expert to intervene. The obtained linear model will be combine with ANN, making and hybrid system that could outperform the forecasting result.

Keywords: Time series, ARMA models, fuzzy system, neural networks, hybrid system.

1. Introduction

For more than half century, the Box-Jenkins methodology using ARMA linear models have dominated many areas of time series forecasting. In 1970, Box and Jenkins (see Box et al.,describe the method of automatic ARMA model- 1994 [2]), made ARMA models popular by proposing a model building methodology comprising several stages: specification, estimation, diagnostic checking and forecasting.
The ARMA (p,q) model for a time series can be modelled as:

$$x_t = \sum_{i=1}^{p} \phi_i x_{t-i} - \sum_{j=1}^{q} \theta_j e_{t-j} + e_t \qquad (1)$$

Where e_t is a normal white noise process with zero mean an variance σ^2, and t=1, … T, being T the number of data of the time series. It is assumed that the autoregressive and moving average parameters satisfy the conditions for stationarity and invertibility, respectively.
The main advantage of ARMA forecasting is that it requires data on the time series in question only. The problem of estimating the order and the parameters of a model such ARMA is an active area of research [1][3][4][5]. Modelling linear and stationary time series, one frequently chooses the class of ARMA models because of its high performance and robustness. The selection of a particular ARMA model, however, is neither easy nor without implications for the goal of the analysis. Some disadvantages of ARMA forecasting are that:

- The result of the Box-Jenkins procedure depends greatly on the competence and experience of the investigators and is affected by a strong path-dependence [6]. Unfortunately, one frequently observes that different models have similar estimated correlation patterns, and the choice among competitive models can be quite arbitrary. Some concern derives also from comparative studies in which experts, asked to identify a number of series, frequently reach different conclusions [2][3][4].
- A further objection to the Box-Jenkins procedure for model selection is related to the time required to develop the identification model, which sometimes can be excessively high.
- It is not embedded within any underlying theoretical model or structural relationships. Furthermore, it is not possible to run policy simulations with ARMA models, unlike with structural models
- ARMA models are linear model, and in real-word, time series are rarely pure linear combination.

Although the application of neural network modelling to Box–Jenkins models encompasses

two sub-areas, that is, model identification (by an statistical feature extractor, perform a pattern classification) and forecasting (as a function approximator), the scope of this paper will be focused in both areas, but using and fuzzy expert system for automatic model identification.

2. Fuzzy Expert System: the Structure of the ARMA Model

In Section 1, we have mentioned several justifications for an automatic ARMA model building procedure. They can be summarized by saying that using the Box–Jenkins method on a large scale requires both expertise and time. Aiming to make the method available to people without that expertise, several software vendors have implemented automated time series forecasting methods, e.g. Mandrake [7], and the list in Tashman [8]. However, these approaches were not very successful and nowadays, most popular statistical package (SPSS, Statgraphics, Matlab, etc), require the intervention of the user. The goal of this section is present an automated way to obtain the structure of an ARIMA model, using an expert system working with fuzzy logic.
What we have to achieve is a program that, on the basis of a given number of rules, is capable of assigning weights to each so that the series that are analysed can be correctly identified.

For this reason genetic algorithms are used to assign weights to the various rules, and this is the novel aspect of the current project. To date, and as observed above, trial-and-error methods have been applied. Fundamentally, a visual examination has been made of the estimated autocorrelation function (ACF) and the estimated partial autocorrelation function (PACF)., from which relevant conclusions are drawn. This technique, naturally, requires a great deal of skill and long practice. On the basis of this visual examination, the various models possible are identified. An estimate is made of the Φ and θ coefficients, and a decision is taken regarding which of the estimated models best fits the series, using mathematical tests. If there are two models fitted equally well by the series, the simpler one is chosen.

2.1. Initial rules utilized

Before the learning program starts to test the rules fulfilled by each series, a number of simple tasks must be performed:

1. *Obtain the estimated ACF and PACF coefficients and the error criterion.*

2. *Check that the series is not white noise.* If 95% of the samples fall within the error criterion, the series is white noise, the model is classified (0,d,0) (where d is the differentiation) and no analysis may be undertaken.

3. *Exponential fit of the first terms of the ACF and of the PACF.* The program attempts to determine whether the shape of the ACF and of the PACF is similar to that of any of the theoretical shapes of the various models described above. To do this, an exponential fit is carried out on the first terms of the ACF and of the PACF, fitting them to an exp(-βx) curve. By these means, values of ß are obtained for the ACF and the PACF, and these values will be used, together with those of the correlation coefficient, to help identify the model.

4. *Determine the spikes in the coefficients of autocorrelation and of partial autocorrelation.* In fact, if the series is (for example) of the AR(1) type, it will present a spike in the PACF. This is what is determined in this stage of the procedure. As the program is unaware of the type of series presented, it checks the number of ACF and PACF coefficients that are 70% above the error criterion.

5. *Estimate Φ and θ.* When this step is reached, we still do not know the series type, but it is possible to estimate Φ and θ for various known types of series, such that we perform an estimation of coefficients for the following models:
AR(1) → Estimating the value of Φ_1
AR(2) → Estimating the values of Φ_1 and Φ_2.
MA(1) → Estimating the value of θ_1
MA(2) → Estimating the values of θ_1 and θ_2.
ARMA(1,1)→ Estimating the values of Φ_1 and θ_1.

6. *Test changes of sign in the ACF and the PACF.* This is performed in order to determine which model is best fitted.

2.2. Rules of the expert system

The expert system that is created consists of a total of 35 fuzzy rules. Due to space limitations imposed on the present paper, we can only discuss

a few of these, together with the justification for their inclusion.

Rule 1. *If the PACF fall more abruptly than the ACF, then the model is AR(p), where p is the PACF number immediately above the error criterion.*

This rule is suggested by the shape of the AR models. In such a model, the ACF fall smoothly, while the PACF fall abruptly. The number of PACF above the error criterion will be the order of the AR model. To determine this, the program uses the previously-obtained ß values of the exponential fit. The exponential presenting the most abrupt change is the one with the highest absolute ß value, such that if the ß calculated for the PACF is greater than that for the ACF, the model will be AR. To determine the order, we examine the number of spikes in the PACF. As the series are not ideal, but have a component of white noise, we only take into consideration the PACF that are 70% above the error criterion.

Rule 2. *If the ACF fall more abruptly than the PACF, then the model is MA(q), where q is the number of ACF above the error criterion.*

This rule is the inverse of the previous one, and the explanation is analogous. As before, the series is not an ideal one, and so to determine the ACF we take into consideration the ACF that are 70% above the error criterion.

Rule 21. *If the estimated Φ_1 and Φ_2 coefficients fulfil the stationarity rules, then the series corresponds to the AR(2) model.*

The stationarity conditions for this type of series are:

$$\Phi_1 + \Phi_2 < 1$$
$$\Phi_2 - \Phi_1 < 1$$
$$-1 < \Phi_2 < 1$$

The coefficients must fulfil the three requirements simultaneously; otherwise, the series will not be valid. As stated above, the system thus created is made up of 40 rules, each of which is assigned a value or relative weight. The last five rules tries to combine or identify mixed model when both AR and MA models are plausible.

2.3. Using an Evolutionary Algorithm to optimize the parameters of the fuzzy system

A genetic algorithm is used to determine the weight assigned to each rule. The limits of the weights range from 0 to 1. The most complex task is the creation of the fitness function, which must perform the following tasks:

1. Using the learning program, evaluate various series of known types, obtaining one or more results for each.
2. Calculate the distances of the possible models obtained for each series from the real model, and store a distance for each series.
3. Sum the distances thus obtained. The fitness function seeks to minimise the distance. As a maximisation algorithm must be applied, the function to be maximised is then:

$$F = \frac{1}{\sum_i d_i + \varepsilon} \qquad (2)$$

where ε is a constant required to avoid an infinite result with zero distances, and where d_i is the distance from the model obtained for the series i to the real model. The distance used is the Euclidean distance, i.e. given two vectors v1=(x, y, z) and v2=(a, b, c), the distance between them is:

$$d(v1, v2) = \sqrt{(x-a)^2 + (y-b)^2 + (z-c)^2} \qquad (3)$$

We analysed a large quantity of real series used as benchmarks in the prediction of time series (http://www.secondmoment.org/time_series.php)

The results obtained for the simulations, except in the case of the mixed series, were highly promising, obtaining for AR(p) models classification errors of 1.2 % and for MA(q) model classification errors of 12.5 % for a total of 1500 time series. Therefore, this ARMA(p,q) structure will be used in conjunction with the Neural Network.

3. Using Simultaneously ARMA and Neural Network Models

Most of the time series can be descomposed in a linear and nonlinear component in the following way:

$$Y_t = L_t + NL_t \qquad (4)$$

being L_t the linear component and NL_t denotes the nonlinear component. The linear component can be captured using the ARMA system developed in Section 2. However, the error in the forecasting (the residual), should be produced

because the nonlinear component can not be represented by the ARMA model. Therefore, the different between the time series forecasting using the ARMA model and the real time series, will be the input to the neural network. It is important to note, that in order to obtain the linear model, the first step is to determine if a natural log transformation of the series is necessary. This step is accomplished using the likelihood ratio. It is necessary to take into account that the likelihood is greater for the transformed series than the raw series when:

$$\log\left(\frac{1}{n}\sum_i\left(\log(Y_i)-\frac{1}{n}\sum_i\log(Y)\right)^2\right) > \left(\log\left(\frac{1}{n}\sum_i(Y_i-\bar{Y})^2\right)+\frac{2}{n}\sum_i\log(Y_i)\right) \quad (5)$$

If the inequality is fulfilled, it is necessary to take the log transform. Therefore, the proposed methodology of the hybrid system consists of two steps. In the first step, an ARMA model is used to analyze the linear component of the time series and in the second step a neural network model is developed to model the residuals from the ARMA model. To design the ANN, a fully connected Multilayer Perceptron (MLP) with two hidden layers and one output layer has been used. This class of network consists of an input layer, a number of hidden layers and an output layer. For a MLP with one hidden layer, the output of the system N_t given selected past observations, Y_{t-d1}, Y_{t-d2},…, Y_{t-dk}, at lags (d1, d2, …,dk), and h neuron in the hidden layer is obtained by:

$$\hat{Y}_t = \phi_0\left(\sum_h w_{h0}\phi_h\left(\sum_i w_{ih}x_{t-di}\right)\right) \quad (6)$$

Being w_{ih} and w_{h0} the weights for the synapses between the inputs and the hidden neurons and between the hidden neurons and the output. The two functions ϕ_h and ϕ_0 denote the activation function, that of course, could be a nonlinear mappings from the inputs to hidden nodes and from hidden nodes to the output, respectively. The logistic function, expressed as:

$$\phi_h(x) = \frac{1}{1+e^{-x}} \quad (7)$$

Is often used in the hidden layer, and the identify function for the output layer (ϕ_0).To configure the structure of the multilayer perceptron, we need to describe the number of hidden layers, the number of hidden neurons in each layer, and the types of activation functions for each hidden neurons. Another important task of ANN modeling of a time series is the selection of the number of lagged observations, i.e. the dimension of the input vector. This is perhaps the most important parameter to be estimated in an ANN model because it plays a major role in determining the (nonlinear) autocorrelation structure of the time series. However, there is no theory that can be used to guide the selection of the number of input. To solve this problem, we have used the input proposed for the linear model, in order to identify and forecast the residual.

4. Simulation Results

Once the forecasting methodology has been established, one can proceed with intensive experimental studies. In this section, we provide two numerical examples to evaluate the advantages and the effectiveness of the proposed approach. These include Lorenz attractor, and Eriel.

4.1 Lorenz attractor time series

The Lorenz attractor time series was generated by solving the Lorenz equations:

$$\begin{aligned}\frac{dx_1(t)}{dt} &= \sigma(x_2(t)-x_1(t))\\ \frac{dx_2(t)}{dt} &= \rho\cdot x_1(t)-x_2(t)-x_1(t)x_3(t)\\ \frac{dx_3(t)}{dt} &= -x_3(t)\beta+x_1(t)x_2(t)\end{aligned} \quad (8)$$

where the parameters are set at the standard values σ=10, ρ=28 and β=8/3. Solutions to this system of three differential equations exhibit the sensitive dependence on initial conditions which is characteristic of chaotic dynamics. In realistic situations, knowledge of the true state of a system can be done only in finite precision. In such cases, sensitivity to initial conditions rules out long-term prediction. On the other hand, short-term prediction is possible to the extent that the current position can be estimated and that the dynamics can be approximated. A long trajectory of the Lorenz attractor (1500 points) was generated using a differential equation solver (Runge-Kutta method) with step size of 0.05 to create a univariate time series (x_1(t)). The data was split into 2 parts: 1127 points were used for training and the remaining 361 for assessing the generalization capability of the network.

Figure 1 shows a characterization of this time series (its histogram and its phase diagram).

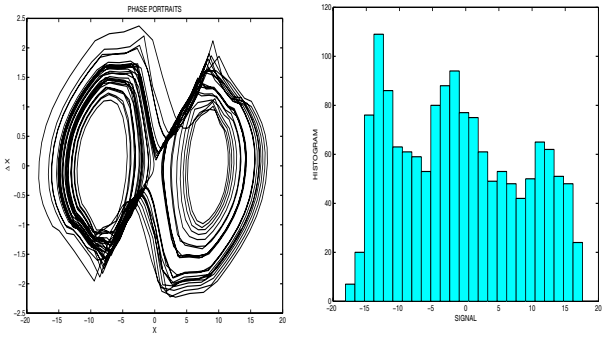

Figure 1 The characterization of the Lorenz time series: a) the phase diagram b) histogram

Figure 2 shows the Lorenz time series forecasting, the predicted and desired values, dashed and continuous lines, respectively, using only ANN.

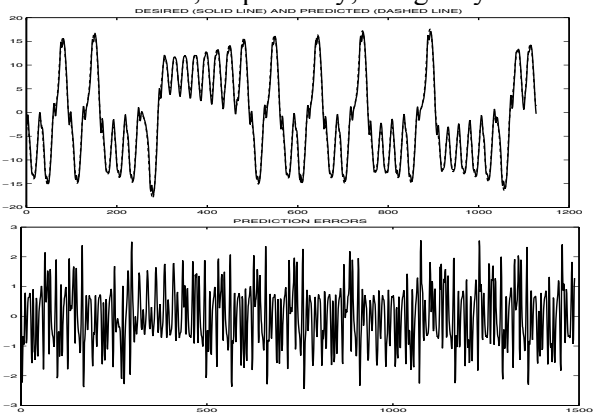

Figure 2 Lorenz time series (prediction step=1): a) result of the original and predicted Lorenz time series using only ANN b) prediction error

Figure 3 shows the prediction of the Lorenz attractor using the proposed methodology. As they are practically identical, the difference can only be seen on a finer scale (Figure 3.b). The error indices, the root mean square error and the correlation coefficient, for this simulation were 0.176 and 0.998 for the model using only ANN and 0.0876 and 0.9996 for the proposed algorithm. It is important to note that other approaches appeared in the bibliography, for example T.Iokibe et.al [10] obtain an RMSE of 0.244, J.S.R.Jang et. al [9] an RMSE of 0.143, using fuzzy and neuro-fuzzy systems. **¡Error! No se encuentra el origen de la referencia.** shows the results of correlating one prediction steps ahead. Figure 5 and Figure 6 show the results of predicting the time series of the Lorenz attractor where the prediction steps change (RMSE and correlation coefficient). As expected, the greater the prediction step, the lower the correlation coefficient and greater the RMSE.

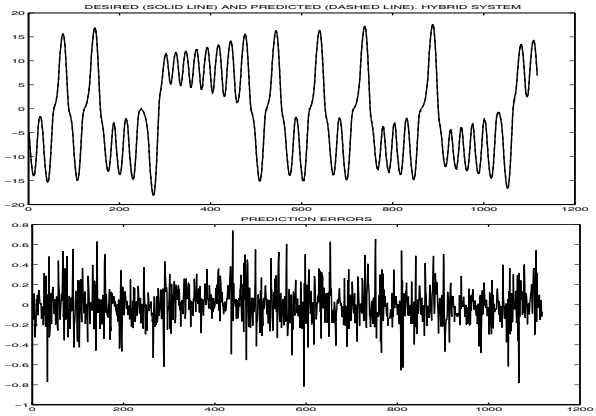

Figure 3 Lorenz time series (prediction step=1): a) result of the original and predicted Lorenz time series using proposed methodology (which are indistinguishable) b) prediction error

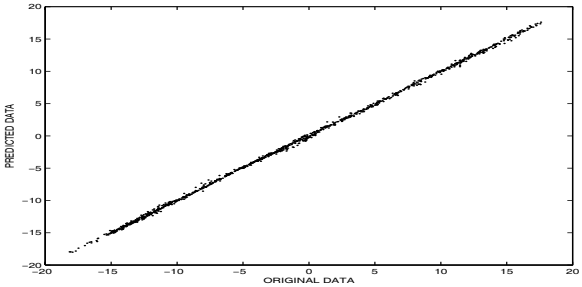

Figure 4 Correlation for the Lorenz time series using the proposed methodology

Figure 5 Result of prediction the Lorenz attractor (change of RMSE by prediction step)

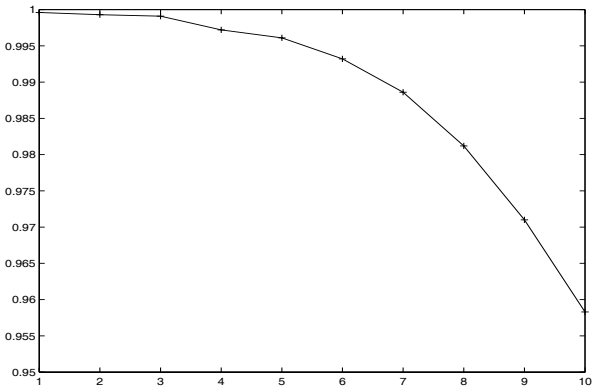

Figure 6 Result of prediction the Lorenz attractor (change of correlation by prediction step)

4.2 Monthly Lake Erie Levels

We used a natural phenomenon, the Monthly Lake Erie Levels from 1921 to 1970. The first 400 data pairs of the series were used as training data, while the remaining 200 where used to validate the model identified.

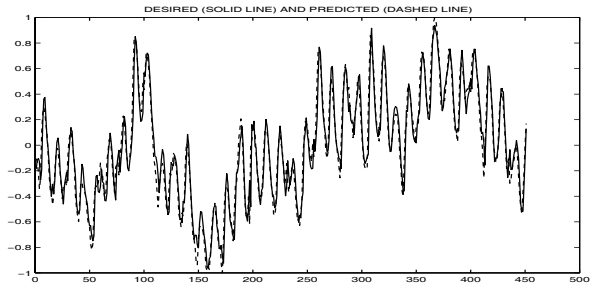

Figure 7 Eriel time series (prediction step=1): a) result of the original and predicted Eriel time series using only ANN b) prediction error

Figure 7 shows the predicted and desired values (dashed and continuous lines, respectively) using only ANN and Figure 8 shows the forecasting using the hybrid method.

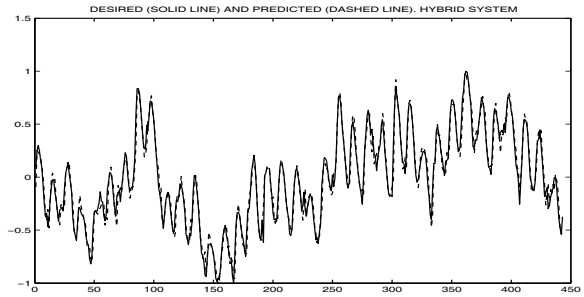

Figure 8 Eriel time series (prediction step=1): a) result of the original and predicted Eriel time series using proposed methodology (which are indistinguishable)

The error indices (RMSE and correlation coefficient) for both simulations are respectively [0.1045, 0.937] and [0.0820, 0.963].

5. Conclusion

In this paper, a hybrid methodology is proposed for time series forecasting that combines linear ARMA model and non-linear model such as artificial neural network. It is important to note, that obtaining the structure of the ARMA model is a problem itself, and in order to develop an automatic system without the direct intervention of a human expert in the Box-Jenkins methodology, a fuzzy expert system was design to acquire the structure of the ARMA model. The justification for automatic ARMA modelling is the following: (a) the method for building an ARMA model is somewhat complex and requires a deep knowledge of the method; (b) consequently, building an ARMA model is often a difficult task for the user, requiring training in statistical analysis, a good knowledge of the field of application, and the availability of an easy to use but versatile specialized computer program; (c) the number of series to be analyzed is often large. It is worth mentioning that the automatic system presented, using real data set and synthetic time series generated by differential equation, indicate that the proposed hybrid methodology improve the error indexes achieved by either of the models used separately

Acknowledgement

This work has been partially supported by the Spanish Project TIN2004-01419.

References:

[1] Adnan Al-Smadi, Ahmad Alshamali, "Fitting ARMA models to linear non-Gaussian processes using higher order statistics" Signal Processing 82 (2002) 1789 – 1793

[2] G.E.P. Box, G.M Jenkins, G.C. Reinsel, "Time Series Analysis: Forecasting and Control" Prentice-Hall, Englewood Cliffs, New Jersey 1994.

[3] W-S. Chan, "A comparison of some of pattern identification methods for order determination of mixed ARMA models", Statistics and Probability Letters, vol.42, pp.69-79, 1999.

[4] R.C. Souza, A.C Neto, "A bootstrap simulation study in *ARMA(p,q)* structures", Journal of Forecasting 1996;15: 343-53.

[5] R.Tsay, G.C.Tiao, "Consistent Estimates of Autoregressive Parameters and Extended Sample Autocorrelation Function for Stationary and Nonstationary ARMA Models", J.American Statistical Association, March 1984, vol.79, n.385, pp.84-96.

[6] A.S.Weigend, N.A. Gershenfeld, "*Time Series Prediction"*. Addison-Wesley, Publishing Company, 1994.

[7] Azencott, R. "Modelisation ARIMA automatique: le systeme MANDRAKE". Cahiers du Centre d 'Etudes de Recherche Operationnelle 32, 229, 1990

[8] Tashman, L. J. "Out-of sample tests of forecasting accuracy: a tutorial and review". International Journal of Forecasting 16, pp. 437–450, 2000

[9] J.S.R.Jang, C.T.Sun, E.Mizutani, "Neuro-Fuzzy and soft computing", Prentice Hall, ISBN 0-13-261066-3, 1997.

[10] T.Iokibe, Y.Fujimoto, M.Kanke, S.Suzuki, "Short-Term prediction of chaotic time series by local fuzzy reconstruction method", Journal of Intelligent and Fuzzy Systems, vol.5, pp.3-21, 1997.