Neural networks versus time series methods: a forecasting exercise*

Marcelo S. Portugal**

Summary: 1. Introduction; 2. ARIMA models; 3. Unobservable components models; 4. Artificial neural networks; 5. Empirical results; 6. Final remarks.

This paper presents an empirical exercise in economic forecast using traditional time series methods, such as ARIMA, and unobservable components models (UCM), and artificial neural networks (ANN). We use monthly gross industrial output data for the state of Rio Grande do Sul (Brazil) to perform a comparative exercise and access the relative performance of the different forecasting methods. The results show that ANN forecast more accurately than ARIMA models, but the comparison with UCM is not quite straightforward. The UCM is found to produce better one step ahead forecast than the ANN, but the performance of the ANN for larger forecasts horizons shows that, specially once a proper modeling methodology has been established, it may be a valuable tool to economic forecasting.

Este artigo apresenta um exercício empírico que tenta comparar a capacidade preditiva de um modelo de redes neurais artificiais (ANN) com modelos de série de tempo tradicionais, como ARIMA e modelos de decomposição em componentes não-observados (UCM). Para tanto, utilizamos a série de produção industrial para o estado do Rio Grande do Sul. Os resultados mostram que o modelo de ANN prevê melhor que o modelo ARIMA, mas é superado pelo UCM nas previsões um passo a frente. No entanto, o bom desempenho dos ANN em previsões com mais de um passo a frente indica que este método é um instrumento valioso de previsão.

1. Introduction

This paper is mainly an empirical exercise intended to compare the forecasting performance of artificial neural networks and traditional time series methods, such as ARIMA, and unobservable components (structural time series) models. Recent work that has used ANN and ARIMA models for forecasting purposes has indicated that ANN seems to produce more accurate forecasts.[1]

We argue that ARIMA models are not a proper benchmark for comparison with ANN. An artificial neural network is a biologically inspired attempt to model the human brain. The network is trained, like the human brain, to be able to recognize patterns and regularities in the data. A better benchmark could be the UCM. It has been shown in the forecasting literature that UCM generally perform better than ARIMA models. The estimation of an UCM, in both classical or Bayesian fashions, based in the Kalman filter, is more similar to the estimation of ANN, since it also learns from its forecast errors.

* Paper presented at the 14th International Symposium on Forecasting, Stockholm School of Economics, Stockholm, Sweden, June 12-15, 1994. Paper received in Aug. 1994 and approved in Apr. 1995.

** Financial support of Fapergs (grant number 93.0719.1) and CNPq (grant number 301445\92-3) is gratefully acknowledge. I also would like to thank the research assistantship of Frederico A. C. N. Pinto (CNPq/UFRGS), Rafael J. Rocha (CNPq/UFRGS), and Erik Sasdelli Camarano, Pedro Valls Pereira and two anonymous referees for their comments to a previous version.

[1] See, for example, Souza and Zandonade (1993) and Refenes (1991).

This paper has other five sections. In the next three sections we will briefly discuss the basic principles and methodology of ARIMA models, unobservable components models and artificial neural networks. Section five presents and compares the empirical results for the three models using gross industrial output data for the state of Rio Grande do Sul (Brazil). Finally, in the last section, some conclusions and remarks are presented.

2. ARIMA models

The ARIMA methodology was proposed by Box and Jenkins (1976) and it is now a quite popular tool in economic forecasting. The basic idea is that a stationary time series can be modeled as having both an autoregressive (AR) and a moving average (MA) component. Non-stationary integrated series can also be handled in the ARIMA framework, but it has to be reduced to stationarity beforehand by differencing the data. The multiplicative ARIMA representation can be written as

$$\Phi_P(L)\phi_p(L)(1-L^s)^D(1-L)^d y_t = \delta + \Theta_Q(L)\theta_q(L)\varepsilon_t$$

where $\Phi_P(L)$, $\phi_p(L)$, $\Theta_Q(L)$ and $\theta_q(L)$ are polynomials in the lag operator (L), d and D are the number of consecutive and seasonal differences needed to make the series stationary, p, P, q and Q are the degrees of the autoregressive and moving average polynomials and ε_t is a normally distributed random error with zero mean and constant variance. The model is multiplicative in the sense that the observed data result from the successive filtering of a random noise (ε_t) through the non-seasonal filter $\phi_p(L)$ and then the seasonal filter $\Phi_P(L)$.

The modeling procedure can be divided in three parts. In the first stage, the order of differencing and the degrees of the AR and MA polynomials are determined using both the estimated autocorrelation and partial autocorrelation functions. In the second stage, the parameters (ϕ, Φ, θ *and* Θ) are estimated and several tests are performed to assure that the residuals are white noise. Likelihood methods are generally used for estimation purposes, but if the model has only an AR part, least squares estimation will be appropriate. Finally, in the third stage, the best and most parsimonious model is used to obtain the forecasts. The likelihood function for the non-seasonal stationary model can be written as

$$L(\phi, \theta, \sigma^2 / y_t) = (2\pi)^{-T/2}(\sigma^2)^{-T/2} \exp[(-1/2\sigma^2)\Sigma e_t^2]$$

3. Unobservable components models

The literature on unobservable components models (UCM) has grown quite significantly in the last few years, due mainly to the introduction of the Kalman filter to econometrics. The Kalman filter has helped to make these models operational, giving an easy way to estimate the time varying unobservable components. Such models have been developed in both classical and Bayesian statistics frameworks. The main difference between these two approaches

is the estimation of the variances or hyperparameters. In the Bayesian models the observational and evolution variances are obtained by variance learning and discount factors. In the classical models, on the other hand, the hyperparameters are estimated by maximum likelihood.[2] In this paper we are going to follow the classical approach.[3]

In this framework, a time series is modeled by decomposition into its basic forming elements. That is, it is decomposed by trend (μ_t), cycle (ψ_t), seasonal (γ_t) and irregular (ε_t) components.

$$y_t = \mu_t + \gamma_t + \psi_t + \varepsilon_t$$

The complete model should specify also the behavior of each individual component. A complete treatment of unobserved components models is beyond the scope of this paper. Therefore, we will concentrate in briefly discuss the case of the "basic structural time series model".[4] In basic structural time series model only three components are used since the cycle component is omitted. The trend is modeled as a random walk with a time varying drift, while the drift itself is a random walk. The seasonal component is modeled by a combination of sine and cosine waves. The model can be expressed as:

$$y_t = \mu_t + \gamma_t + \varepsilon_t \tag{1}$$

$$\mu_t = \mu_{t-1} + \beta_{t-1} + \eta_t \tag{2}$$

$$\beta_t = \beta_{t-1} + \zeta_t \tag{3}$$

$$\gamma_t = \sum_{j=1}^{s/2} \gamma'_{jt}$$

where γ'_{jt} is formed by

$$\begin{bmatrix} \gamma_{jt} \\ \gamma^*_{jt} \end{bmatrix} = \begin{bmatrix} \cos\lambda_j & \sin\lambda_j \\ -\sin\lambda_j & \cos\lambda_j \end{bmatrix} \begin{bmatrix} \gamma_{jt-1} \\ \gamma^*_{jt-1} \end{bmatrix} + \begin{bmatrix} \omega_{jt} \\ \omega^*_{jt} \end{bmatrix} \tag{4}$$

and ε_t, η_t and ζ_t are all normally distributed errors with variances σ^2_ε, σ^2_η and σ^2_ζ, ω_t and ω^*_t are also normally distributed errors with a common variance σ^2_ω, λ_j is the frequency and γ^*_{jt} appears by construction.

---

[2] See Meinhold and Singpurwalla (1983) and West and Harrison (1989) for a discussion of the Bayesian approach.

[3] For a detailed presentation and discussion of this model see Harvey (1987 and 1989).

[4] The trend plus cycle model is discussed in Portugal (1993).

To simplify matters let us suppose that we are dealing with quarterly data, so that $s = 4$. Therefore, the frequency is $\lambda_j = 2\pi j/4$ for $j = 1$ e 2, which gives $\lambda_j = \pi, \pi/2$. Using some basic trigonometric results we can rewrite (4) as:

$$\begin{aligned} \gamma_{1t} &= \gamma^{*}_{1t-1} + \omega_{1t} \\ \gamma^{*}_{1t} &= -\gamma_{1\,t-1} + \omega^{*}_{1t} \\ \gamma_{2t} &= -\gamma_{2t-1} + \omega_{2t} \\ \gamma^{*}_{2t} &= -\gamma^{*}_{2t-1} + \omega^{*}_{2t} \end{aligned} \tag{5}$$

To be able to apply the Kalman filter to equations (1), (2), (3) and (5) we need first to cast this model in the state space framework. This can be easily done, as shown in equations (6) and (7):

$$y_t = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \end{bmatrix} \alpha_t + \varepsilon_t \tag{6}$$

$$\alpha_t = \begin{bmatrix} \mu_t \\ \beta_t \\ \gamma_{1t} \\ \gamma^{*}_{1t} \\ \gamma_{2t} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mu_{t-1} \\ \beta_{t-1} \\ \gamma_{1\,t-1} \\ \gamma^{*}_{1\,t-1} \\ \gamma_{2\,t-1} \end{bmatrix} + \begin{bmatrix} \eta_t \\ \zeta_t \\ \omega_{1\,t} \\ \omega^{*}_{1\,t} \\ \omega_{2\,t} \end{bmatrix} \tag{7}$$

In this specification we have a component to represent the level of the series (μ_t) and other to represent the trend itself or slope coefficient (β_t). Therefore, series with no trend will have $\beta_t = 0$ while in the case of a constant trend we have $\beta_t = \beta$. Equations (6) and (7) are just a particular state space representation. In general form a state space model can be written as:

$$Y_t = X_t \alpha_t + \varepsilon_t \tag{8}$$

$$\alpha_t = M_t \alpha_{t-1} + S_t c_t + R_t u_t \tag{9}$$

where,

α_t = state vector;

Y_t = measurement vector;

c_t = input vector;

X_t, M_t, S_t, R_t = known matrices with appropriate dimensions.

We also assume that:

$$E(\varepsilon_i, \varepsilon_j) = \delta_{ij}\,\sigma^2 H_t \qquad i, j = 1, \ldots, T$$

$$E(u_i, u_j) = \delta_{ij}\,\sigma^2 Q_t \qquad i, j = 1, \ldots, T$$

$$E(\varepsilon_t, u_t) = 0 \qquad t = 1, \ldots, T$$

$$E(\alpha_0, \varepsilon_t') = 0 \qquad t = 1, \ldots, T$$

$$E(\alpha_0, u_t') = 0 \qquad t = 1, \ldots, T$$

where δ_{ij} is the Kronecker delta. These conditions just state that ε_t and u_t have zero mean and covariance matrices $\sigma^2 H_t$ and $\sigma^2 Q_t$, respectively, are non-autocorrelated, non-correlated between themselves and non-correlated with the state vector at time zero. The errors are also assumed to be normal, although this is not crucial.

The Kalman filter is a learning algorithm that includes the prediction equations (10) and (11) and the updating equations (12) and (13). The basic idea behind the Kalman filter is that estimation is performed recursively in two steps. In the first step the "best"[5] estimate at time t, using all the information available up to time $t-1$, is obtained. Then, this estimate is updated using the new information that becomes available at time t. Note that the filter gain is used to improve the state vector forecasts. Every time the state vector is such that predicted and actual Y_t are different, the error is incorporated into the new state vector estimate to make it more accurate. In other words, equation (12) says that the expected value of the state vector, given all the information up to time t, can be divided into two components: the expected value of this vector, given all information up to $t-1$, plus a coefficient times the forecast error of Y_t.

$$a_{t/t-1} = M_t\, a_{t-1} + S_t\, c_t \tag{10}$$

$$\Sigma_{t/t-1} = M_t\, \Sigma_{t-1} M_t' + R_t Q_t R_t' \tag{11}$$

$$a_{t/t} = a_{t/t-1} + K_t\,(Y_t - X_t\, a_{t/t-1}) \tag{12}$$

$$\Sigma_{t/t} = \Sigma_{t/t-1} - \Sigma_{t/t-1} X_t' F_t^{-1} X_t \Sigma_{t/t-1} \tag{13}$$

$$K_t = \Sigma_{t/t-1} X_t' F_t^{-1}$$

where K_t is the filter gain and $F_t = X \Sigma_{t/t-1} X_t' + H_t$.

The unobservable components model has several advantages in relation to the ARIMA models. Since each of the components is time varying, structural changes in the trend or seasonal components can be more easily dealt with by this kind of model. Moreover, it is more informative, allowing a direct interpretation of its components. Finally, as Harvey (1989) has

[5] This estimate is the best in the sense that it minimizes the mean square error.

shown, each linear unobservable components model has a reduced or canonical form that is an ARIMA, implying generally, since it was derived from a structural model, in some restrictions to the parameter space.

4. Artificial neural networks

Neural networks, expert systems and fuzzy logic are the three composing branches of artificial intelligence. An artificial neural network (ANN) is basically a non-parametric attempt to model the human brain.[6] ANN act like a human brain, trying to recognize regularities and patterns in the data. They can learn from experience and generalize based on their previous knowledge. Although biologically inspired, artificial neural networks (ANN) have found applications in many different fields, specially for forecasting purposes.[7]

The human brain is formed by over a billion neurons that are connected in a large network that is responsible for what we normally call thought. The initial attempts to build a thinking machine were not very successful. The pioneers of artificial intelligence have argued that they did not need to know the way the brain works any more than the first aviators needed to build airplans with flapping wings. The idea that one can bypass nature worked well in aviation but not in artificial intelligence. When it comes to thinking it seems we have to do it nature's way. To build an artificial mind you must first build an artificial brain.

The human brain with its billions of neurons connected together in thousands of ways is not like a computer. Experiences which come in through the senses trigger electrical signals that pass among the neurons. Patterns and regularities perceived by the senses are then remembered as similar patterns of neural discharges. In the brain, experiences are recorded as strength of connections among neurons. A new experience coming in through the senses changes the strength of the connections. This pattern is then subsequently remembered. An ANN is just an attempt to imitate how the brain's network of neurons learns.

Each neuron, individually, functions in a quite simple fashion. It receives signals from other cells through connection points (synapses), averages them and if the average over a short period of time is greater than a certain value the neuron produces another signal that is passed on to other cells. Note that there is a high degree of connectivity, since a neuron can generate signals that will activate or inhibit thousands of other neuron. As Wasserman (1989) points out "it is the high degree of connectivity rather than the functional complexity of the neuron itself that gives the neuron its computational power".[8]

An artificial neuron acts exactly like a human one. As shown in figure 1 the artificial neuron receives a set of inputs or signals (x), calculates a weighted average of them (z) and then uses some activation function (F) to produce an output (y).[9]

---

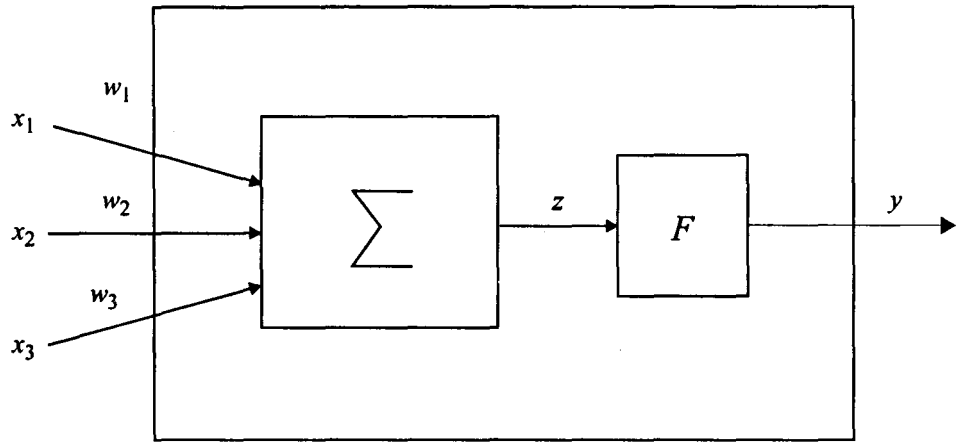
[6] A general introduction to neural computing can be found in Alexander and Morton (1990).

[7] See for example Refenes (1991) for an application to economics and Tohma and Imai (1993) for an application to hydrology.

[8] See Wasserman (1989: 194).

[9] Note that the artificial neuron is somehow "static'. In the biological neuron the averages are calculated over a period of time, whereas in the artificial neuron there is no such dynamic element. In the later the average is done when all inputs have been received.

Figure 1

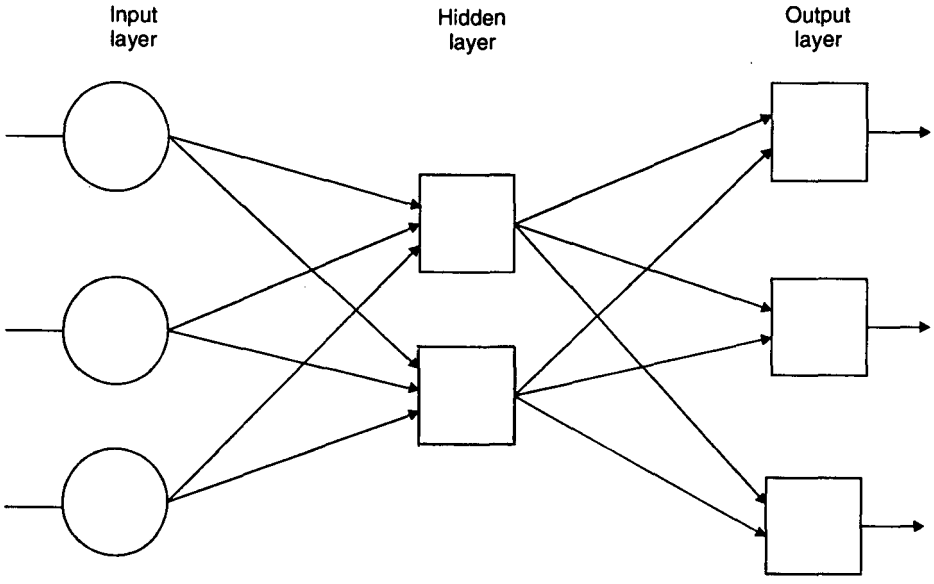


where $z = \sum_{i=1}^{3} x_i w_i$.

The activation function can be a step function, a logistic or even a simple linear function. The most commonly used activation function is

$$F(z) = \frac{1}{1+e^{-z}}$$

Figure 2



In an ANN we have a certain number of neurons arranged in layers. Figure 2 shows an ANN with three neurons in the output layer and two neurons in the hidden layer. The input layer has no neurons, it is used only to distribute the original inputs.[10] That is, actually, a feed-forward or non-recurrent network, since all the connections go in the same direction. It is, nevertheless, possible to have a recurrent network where the output of a layer is fed back as input to a previous layer.[11] Note, also, that figure 2 displays all possible connections among the processing elements. In general, some of the weights (or synaptic strength in the case of biological neurons) may be zero, indicating that a particular connection does not exist. Since the computational power of neural networks comes from their connectivity one should prefer multilayer networks to single layers ones (perceptrons).[12]

The main question now is how one can find the set of weights for the ANN and to achieve this goal we need to use some training algorithm. In time series forecasting we are going to use only supervised training, that is, there is a set of targets we want to obtain and the ANN is trained in such way as to minimize the difference between the network output and the target. Therefore, training is the process of weight adjustment that tries to obtain a desirable outcome. In other words, one wants the set of weights that produces least squares residuals. The most common training algorithm used in the ANN literature is called Back-propagation. It uses a type of gradient descent method, following the slope of the error surface downwards towards its minimum. If the network is linear in its output and has no hidden layers the error surface will have a bowl type shape and there will be only a global minimum. On the other hand, as it is normally the case, if there are hidden layers in the network the error surface may contain several local minimums and the steepest descent method may be trapped in one of them.

Equations (14) to (16) present the Backpropagation algorithm. The subscripts i and j are used to identify a particular weight and k to identify the layer, while the superscripts denote the step of adjustment. To start the process small random numbers are set as weights and then equations (14) to (16) are used to adjust them.

$$\Delta w_{ij,k}^{n+1} = \eta \, \delta_{j,k} \, y_{i,k-1} \tag{14}$$

$$w_{ij,k}^{n+1} = w_{ij,k}^{n} + \Delta w_{ij,k}^{n+1} \tag{15}$$

$$\delta_{j,k} = \left(\frac{dy}{dz}\right)_{i,k} (y_{i,k} - y_{i,k}^{T})' \tag{16}$$

for $i = 1, 2, ..., I$; $j = 1, 2, ..., J$ and $k = K$

where

$w_{ij,k}^{n+1}$ is the weight connecting neuron i in layer $k-1$ to neuron j in layer k at step $n+1$;

---

[10] To distinguish between processing and distribution elements we use different geometric shapes.

[11] For simplicity we have also not included any connections between among in the same layer.

[12] Nevertheless, to guarantee a higher power for the multilayer networks a non-linear activation function has to be used. A linear multilayer network is equivalent to a single layer one.

η is the training rate coefficient (usually between 0.01 and 1.0);

$y_{i,k}$ is the output of neuron i in layer k;

$y^T_{i,k}$ is the target value for $y_{i,k}$.

Note that the algorithm above can only be applied to the last layer ($k = K$) of neurons, since it requires the knowledge of the target output value. For the hidden layers there is no target output and, therefore, $\delta_{i,k}$ has to be obtained in another fashion. Backpropagation derives its mane from the fact that it propagates the value of $\delta_{i,k}$ backwards through out the network. For the hidden layers, equation (16) has to be modified as

$$\delta_{j,k-1} = \left(\frac{dy}{dz}\right)_{i,k} \left(\sum_j \delta_{j,k} w_{ij,k}\right) \tag{16a}$$

for $k = 1, 2, ..., K$.

To achieve a faster adjustment in the Backpropagation algorithm it has been suggested[13] to include a momentum constant. The momentum method is intended to help network to follow the bottom narrow gullies, if it exists, in the error surface, avoiding jumping rapidly from side to side. It consists in adding an autoregressive feature to the algorithm, so that it has now a "memory" of previous adjustments. Therefore, one needs just to modify slightly the algorithm to add a new term that is proportional to the previous period weight adjustment. We then obtain:

$$\Delta w^{n+1}_{ij,k} = \eta\,(\delta_{j,k}\, y_{i,k-1}) + \alpha\,(\Delta w^{n}_{ij,k}) \tag{17}$$

$$w^{n+1}_{ij,k} = w^{n}_{ij,k} + \Delta w^{n+1}_{ij,k} \tag{18}$$

where α is the momentum constant (normally around 0.9).

5. Empirical results

In this section we present the empirical results for the three models proposed in the early part of the paper. The series we are going to use to perform the comparative forecasting exercises is the gross industrial output of the state of Rio Grande do Sul (Brazil). This series, shown in figure 3, consists of monthly data from January 1981 to July 1993. One can readily see that the series is stationary and seasonal.[14] For all the three models the sample estimation period was from January 1981 to December 1992, leaving the next seven consecutive observations to be used for out of sample forecasting comparison. We have generated a seven steps ahead forecast and also a set of seven one step ahead forecasts.[15]

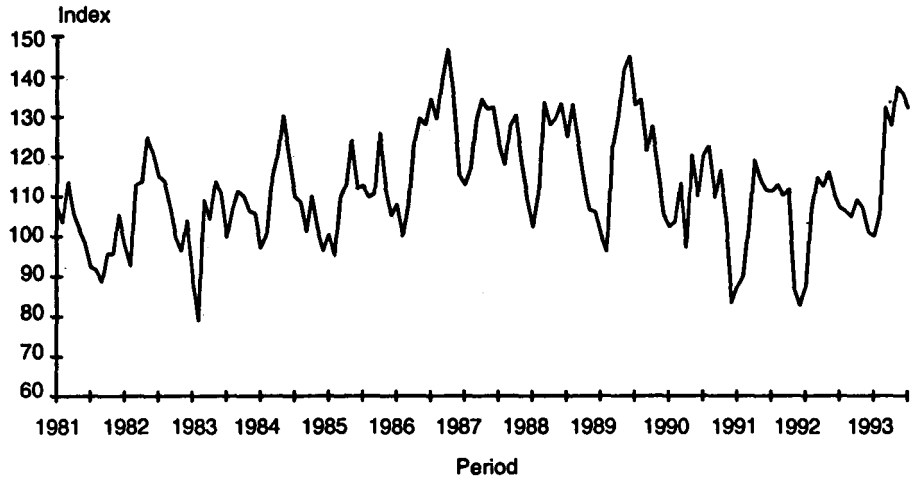
---

[13] See Rumelhart, Hinton and Williams (1986).

[14] The unit root test performed confirms the hypothesis of stationarity.

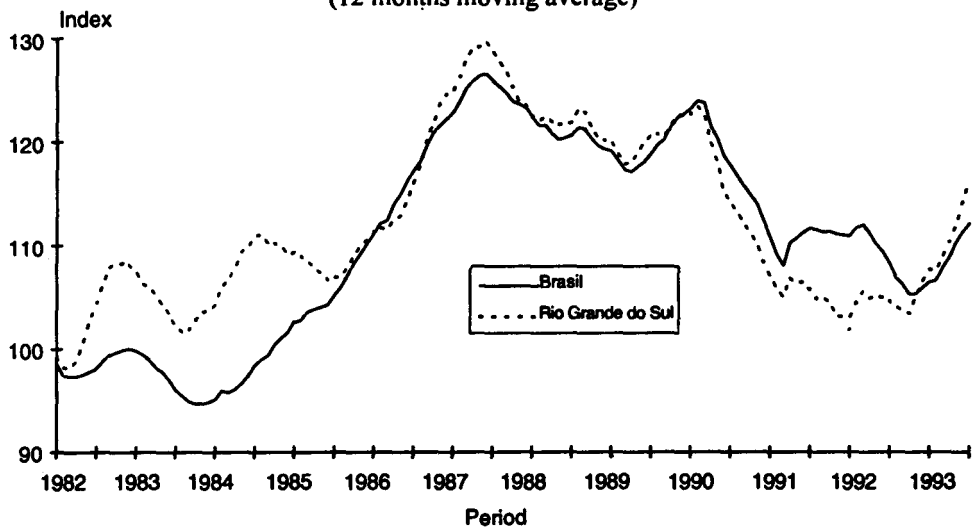
[15] The estimations in this section were performed using the softwares Forecast Master Plus, Stamp and Neural Works Professional.

Figure 3
Gross industrial output — RS



It can be seen from figure 4 that the behavior of the gross industrial output of Rio Grande do Sul (RS) has mainly followed the same pattern as the Brazilian one. Nevertheless, for the period 1982 to 1985 the performance of the RS industry was significantly better than the rest of the country. The industrial output of RS was on average some 10% higher than the rest of the country for this period. This better performance may be due to the behavior of the export sector. It is generally agreed that after 1983 Brazil had a export-led recovery. In the beginning of 1983 the share of the state of Rio Grande do Sul in the total Brazilian exports was at its highest level in the 1980s at over 12%. On the other hand, the RS industry was more affected for the early 1990s recession. During 1990/91 the RS industrial output fell more rapidly than in the rest of the country.

Figure 4
Gross industrial output — Brazil & RS
(12 months moving average)



The best ARIMA model was estimated using the natural logarithm of the original series and the results are shown below. Since the series is stationary, there was no need to difference the data. The estimated model has a constant, one autoregressive parameter and one seasonal autoregressive parameter as well [ARIMA (1, 0, 0)(1,0 0)].

$$(1-0.7053L)(1-0.5463L^{12})y_t = 0.6296+\varepsilon_t$$

$$Q(18) = 21.89 \quad R^2 = 0.676 \quad \sigma = 0.1211$$

Table 1
ARIMA model forecasts

Period	Actual	One step ahead	Percentage error	Seven steps ahead	Percentage error
93:1	99.70	102.44	-2.75	102.44	-2.75
93:2	105.33	111.12	-5.50	112.67	-6.97
93:3	131.95	111.40	15.57	116.26	11.89
93:4	127.39	125.41	1.56	114.25	10.31
93:5	136.98	125.59	8.32	115.74	15.50
93:6	135.56	127.06	6.26	122.24	17.20
93:7	131.84	126.78	3.84	110.13	16.46

The forecasts based on the ARIMA model do not seem very good. The Box and Ljung statistics, that is used to check for residual auto-correlation, show no evidence of non-white noise residuals. The seven steps ahead forecast shows a small forecast error for the first step and larger errors afterwards. This is specially so after March when the percentage forecast error is always double figures. March was a difficult month to forecast since it was quite atypical. Even though the seasonal factors show that March is a month of high industrial output, its growth in March 1993 was quite extraordinary. In that month the Rio Grande do Sul industrial output increased by 25.3% and remained high afterwards. As one would expected, for the set of one step ahead forecasts the model performs much better, although there is still a large forecast error in March.[16]

For the estimation of the unobservable components model we started by trying to estimate a general basic structural time series model. It may be argued, by looking at figure 3, that the slope coefficient should be fixed or even be set equal to zero. Nevertheless, it is better to start with a more general specification than apply a incorrect constraint that might have serious consequences. The initial results show that $\sigma_\varsigma^2 = 0$, confirming, therefore, that a fixed slope would be appropriate. The estimated slope in the fixed slope model is indeed close to zero (-0.00395). The seasonal pattern is also reasonably constant over the sample period. The seasonal factor shows a seasonal reduction in the industrial output for the period November/ February, with May having the highest factor. The estimation of the fixed slope basic structural time was performed using a time domain exact maximum likelihood method and the selected results are shown on page 622.

[16] In all the tables in the paper the "seven steps ahead forecasts" are actually a sequence of one, two, three, four, five, six and seven sets ahead forecasts.

Estimated hyperparameters

Estimate	Parameter	Standard error	*t*-ratio
0.0022168	σ^2 (level)	0.0005943	3.7301
0.0000023	σ^2 (seasonal)	0.0000	Missing
0.0009572	σ^2 (irregular)	0.0003955	2.4200

Estimated state vector

Estimate	State	RMSE	*t*-ratio
4.7739	Level	0.5680	8.4049
-0.0865	Harmonic	0.0150	-5.7728
-0.0195	Harmonic	0.0151	-1.2917
-0.0260	Harmonic	0.0111	-2.3442
-0.0402	Harmonic	0.0111	-3.6095
-0.0092858	Harmonic	0.0099151	-0.9365
-0.087441	Harmonic	0.0099798	-0.8762
-0.0039964	Harmonic	0.0094331	-0.4237
-0.0009803	Harmonic	0.0094953	-0.1032
0.0038284	Harmonic	0.0092274	0.4149
0.0200	Harmonic	0.0092953	2.1551
-0.0005954	Harmonic	0.0077289	-0.0770
-0.0003394	Fix trend	0.0039512	-0.0859
Prediction error variance	.0049489	$Q(6) = 9.181$	$Q(26) = 33.60$

Figure 5
One step ahead forecasts

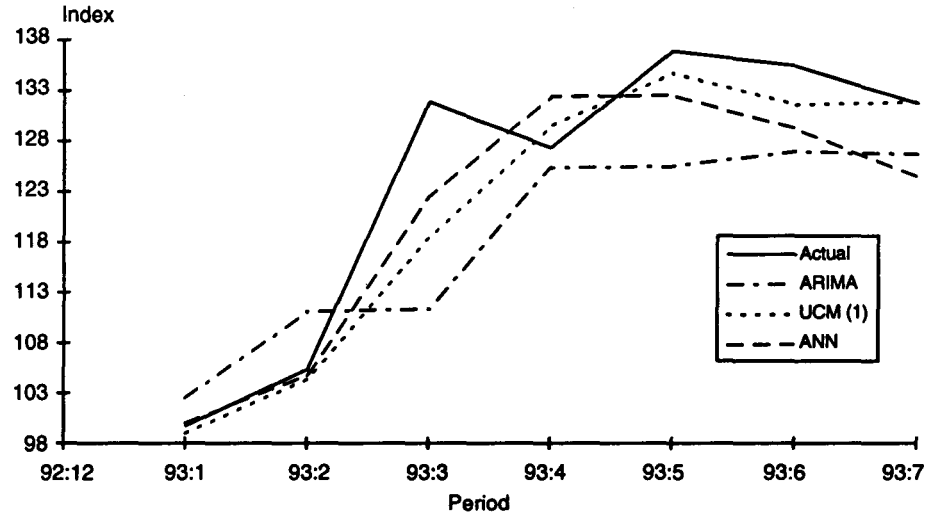


Table 2
Seasonal factors

Jan.	Feb.	Mar.	Apr.	May	June	July	Ago.	Sept.	Oct.	Nov.	Dec.
0.88	0.92	1.03	1.05	1.10	1.06	1.04	1.05	1.01	1.05	0.96	0.88

Figures 6 to 9 show the components obtained after smoothing the data. The seasonal component is roughly constant over time. It shows a small reduction in seasonality by the end of the sample and a change in the seasonal pattern for the middle of the year. The exogenous component is just the fixed slope coefficient. It shows a small long term decline in the gross industrial output of Rio Grande do Sul. This result is not surprising since the 1980s and the beginning of the 1990s were characterized for several periods of strong recessions followed by partial recoveries.

The forecasting performance of the unobservable components model shows a substantial improvement when compared with the ARIMA model. The percentage errors for the seven steps ahead forecasts are substantially lower than the ones obtained by the ARIMA model. The highest forecast error (11.66%) is in March. More significant, nevertheless, is the capability of the unobservable components model to returning to low percentage errors after March 1993. The UCM is able to adjust its forecasts to the substantial increase in production after March. The one step ahead errors show a 10.32% error in March, but once this large error is fed back into the model it is able to adjust itself and start producing again very accurate forecasts. As we have mentioned before, the Kalman filter is capable of learning by incorporating the one step ahead forecast error in the updating equation for the state vector.[17]

Table 3
Unobservable components model (1st version) forecasts

Period	Actual	One step ahead	Percentage error	Seven steps ahead	Percentage error
93:1	99.70	98.94	0.77	98.94	0.77
93:2	105.33	104.37	0.91	103.65	1.60
93:3	131.95	118.33	10.32	116.56	11.66
93:4	127.39	129.56	-1.70	118.20	7.22
93:5	136.98	134.80	1.59	124.07	9.42
93:6	135.56	131.72	2.83	119.93	11.53
93:7	131.84	132.01	-0.13	117.42	10.94

[17] See equation (12).

Figure 6
Trend component

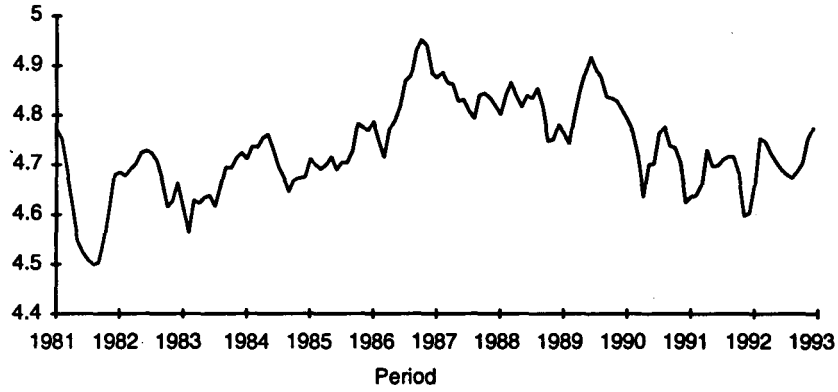


Figure 7
Seasonal component

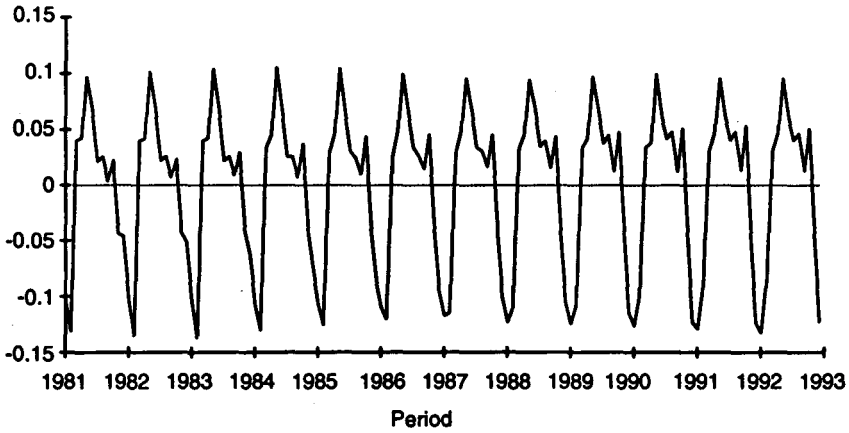


Figure 8
Exogenous component (fixed slope)

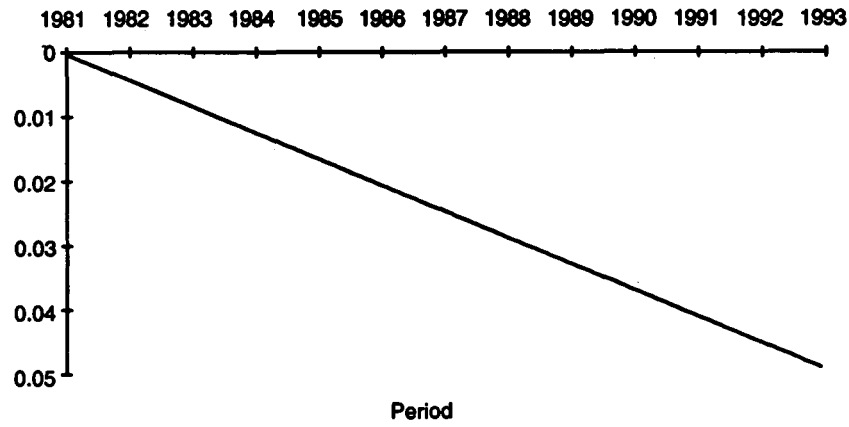
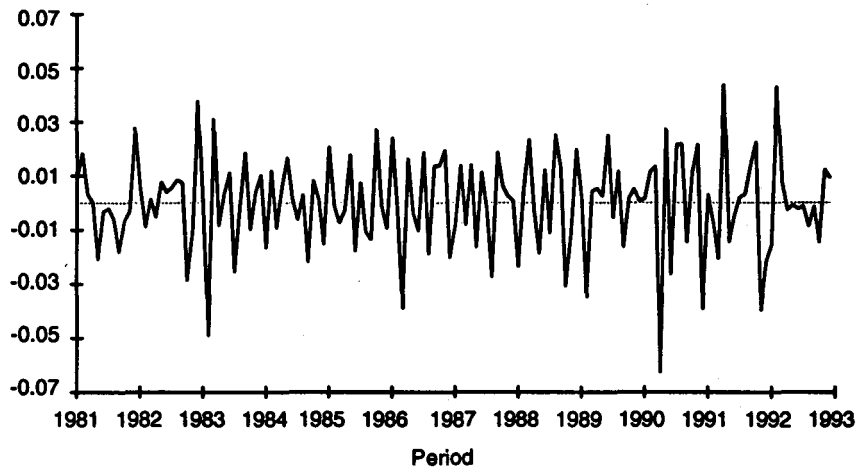


Figure 9
Irregular component



It may be argued that the forecasts in table 3 are not really comparable to those on table 2, since the unobservable components model used is non-stationary, while the ARIMA process is stationary. In this sense we may not be making a fair comparison. Moreover, it may also be argued, by comparison with the Brazilian gross industrial output series, that a cyclical component should also be included. Therefore, we estimate a second version of the unobservable components that includes both the trend and a stationary cycle.[18] The results are shown

[18] See Marwald, Moreira e Pereira (1989).

in table 4. The simple inspection of tables 2, 3 and 4, and the MAE and RMSE presented in table 6, show that the basic structural model presents the best forecasting performance among the alternatives considered. Therefore, we shall use the results in table 3 to compare with the ANN.

Table 4
Unobservable components model (2nd version) forecasts

Period	Actual	One step ahead	Percentage error	Seven steps ahead	Percentage error
93:1	99.70	98.01	1.70	98.01	1.70
93:2	105.33	101.71	3.44	100.40	4.68
93:3	131.95	116.11	12.00	112.00	15.12
93:4	127.39	126.59	0.64	112.60	11.62
93:5	136.98	130.58	4.67	117.54	14.19
93:6	135.56	127.39	6.03	113.11	16.56
93:7	131.84	127.12	3.58	110.16	16.45

One of the main problems with ANN is to determine the number of layers and the number of neurons in each layer. We have used the results of the ARIMA model to provide some insights for the architecture of the network. Since the data is seasonal and autoregressive of order one, we have used 12 neurons to represent the seasonal factors and one neuron to represent the one period lagged value of the series (y_{t-1}). The "seasonal neurons" consist of a binary sequence that has value of one for the month in question and zero for all the others. That is, January neuron input is a vector (1 0 0 0 0 0 0 0 0 0 0 0), the February neuron input is a vector (0 1 0 0 0 0 0 0 0 0 0 0), and so on up to the December neuron which has an input vector of (0 0 0 0 0 0 0 0 0 0 0 1). The network has also one extra neuron with a constant input value to represent the constant term.

Besides these neurons in the input layer, the network has other four neurons in the first hidden layer, two neurons in the second hidden layer and one neuron in the output layer. Using Souza and Zandonade (1993) notation we have an ANN (14,4,2,1). The basic architecture of the network can be seen in figure 10.

Table 5
Artificial neural network forecasts

Period	Actual	One step ahead	Percentage error	Seven steps ahead	Percentage error
93.1	99.70	99.96	-0.26	99.96	-0.26
93.2	105.33	104.77	0.53	100.83	4.27
93.3	131.95	122.40	7.80	118.75	10.00
93.4	127.39	132.51	-3.86	129.18	-1.41
93.5	136.98	132.61	3.30	131.81	3.77
93.6	135.56	129.32	4.83	129.76	4.28
93.7	131.84	124.50	5.90	129.59	1.71

The results for the one and seven steps ahead forecasts for the ANN are presented in table 5. The network was trained for 15,000 times using a learning rate and a momentum coefficient of 0.9 and 0.6 for the first 5,000 times and 0.9 and 0.9 for the remaining 10,000 times. The results for the seven steps ahead forecast are very good. The one step ahead forecasts manage to improve even further the accuracy up to March. Nevertheless, once the observation for March is included, the performance of the network deteriorates considerably. Con-

trary to what happened to the UCM the network seems unable to adjust itself to the large increase in the industrial output in that month.

Figure 10

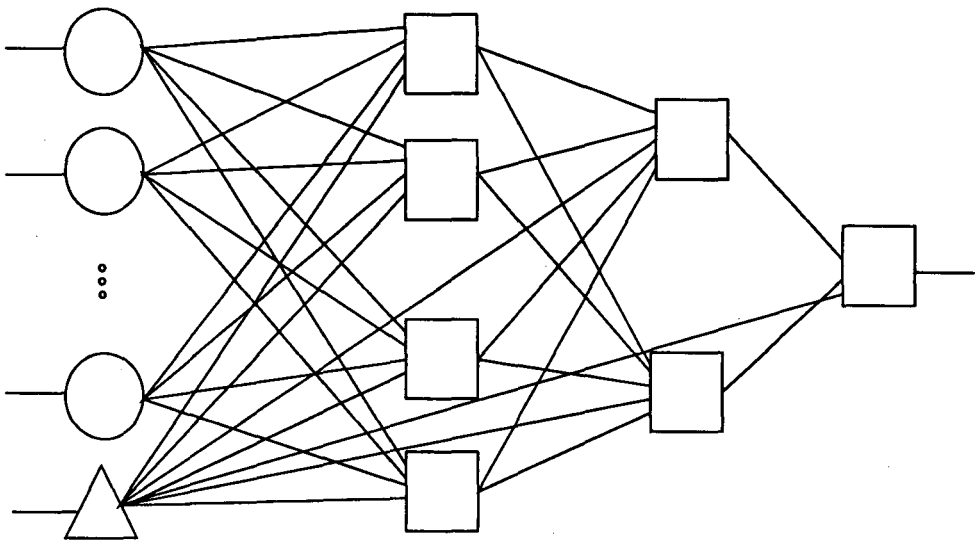
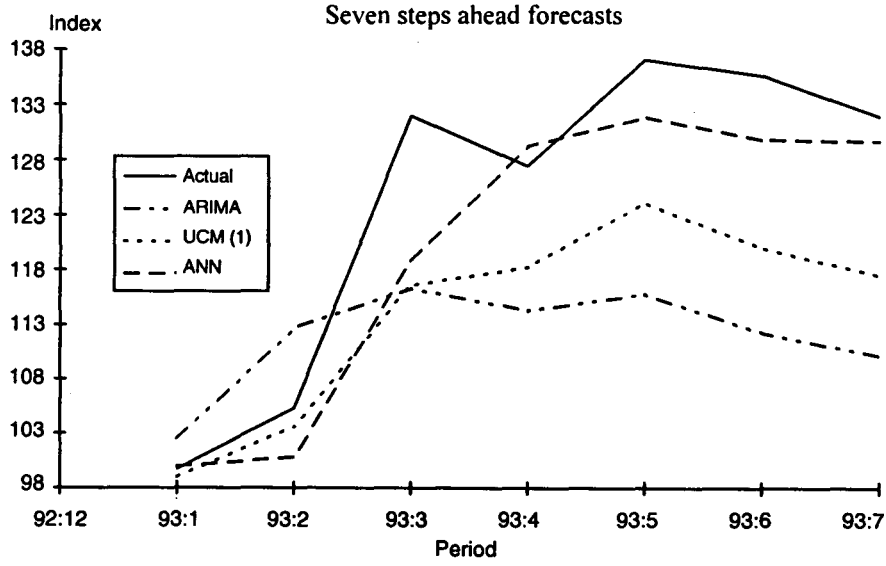


Table 6
Comparative results

	ARIMA	UCM (1)	UCM (2)	ANN
MAE	8.00	3.38	5.89	4.78
RMSE	9.96	5.49	7.54	5.73

Figure 11
Seven steps ahead forecasts



To help comparing the three approaches we calculate the mean absolute error (MAE) and the root mean square error (RMSE). The results show the UCM as the preferred model followed by the ANN. Nevertheless, it is noteworthy that the ANN is forecasting with more accuracy up to March and that for the seven steps ahead forecasts it is the best model. It should be pointed out though, that while the ANN is a non-linear method, both UCM and ARIMA are linear models, what makes the comparison not straightforward.

6. Final remarks

In this paper we have tried to offer an empirical comparative evaluation of the performance of artificial neural network to the problem of economic time series forecast. Most of the recent literature has focused on comparing ANN and ARIMA models and the results overwhelmingly favor the ANN. We argue that a better benchmark for the ANN are the unobservable components models based in the Kalman filter since these models also involve a learning process. The results show that the UCM come out as the preferred model for forecast the Rio Grande do Sul industrial output on the base of the MAE and RMSE. One should be careful not to generalize this result, since it was obtained from one single forecasting exercise and also because the ANN produced the best seven steps ahead forecast and very accurate one step ahead forecasts up to March. These results leave no doubt that further research is worthwhile to try to establish a more concrete conclusion.

The main problems with ANN seem to be their lack of explanation capabilities and of a proper building methodology to define the network architecture. At present, the ANN modeling process is basically empirical. There are no established modeling methodology or tests for network selection. On the other hand, in contrast with expert systems, it is impossible to know how the network has arrived to a particular result. Moreover, since there are no statistical considerations involved in the ANN modeling, it can only supply point forecasts.

References

Alexander, I. & Morton, H. *An introduction to neural computing*. London, Chapman & Hall, 1990.

Box, G. E. P. & Jenkins, G. M. *Time series analysis, forecasting and control*. San Francisco, Holden-Day, 1976.

Harvey, A. C. Applications of the Kalman filter in econometrics. In: Bewley, T. F. (ed.). *Advances in econometrics — Fifth World Congress*. Cambridge, Cambridge University Press, 1987. v. 1.

———. *Forecasting structural time series models and the Kalman filter*. Cambridge, Cambridge University Press, 1989.

Marwald, R. A.; Moreira, A. R. B. & Pereira, P. L. V. Previsão da produção industrial: indicadores antecedentes e modelos de série temporal. *Pesquisa e Planejamento Econômico*, *19*:233-53, 1989.

Meinhold, R. J. & Singpurwalla, N. D. Understanding the Kalman filter. *The American Statistician*, *37*:123-7, 1983.

Portugal, M. S. Measures of capacity utilization: Brazil 1920-1988. *Análise Econômica*, *11* (19):89-102, 1993.

Refenes, A. N. Constructive learning and its application to currency exchange rate forecasting. In: Turban, E. & Trippi, R. (eds.). *Neural networks applications in investment and finance services*. New York, Probus, 1991.

Rumelhart, D. E.; Hinton, G. E. & Williams, R. J. Learning internal representations by error propagation. *Parallel distributed processing*. Cambridge, MIT Press, 1986. v.1, pp. 318-62.

Souza, R. C. & Zandonade, E. Forecasting via neural networks: comparative study. Department of Electrical Engineering, Catholic University of Rio de Janeiro (PUC-RJ), 1993. mimeo.

Tohma, S. & Imai, S. Flood runoff forecasting using fuzzified neural networks. In: McAller, M. & Jakman, A. (eds.). *Proceedings of the International Congress on Modelling and Simulation*. Perth, Australia, University of Western Australia, 1993. v. I, pp. 361-6.

Wasserman, P. D. *Neural computing: theory and practice*. New York, Van Nostrand Reinhold, 1989.

West, M. & Harrison, J. *Bayseian forecasting and dynamic models*. New York, Springer Verlag, 1989.

Data appendix

Industrial Production Index — Rio Grande do Sul (base 1981 = 100)

Months \ Years	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993
Jan.	107.86	97.78	89.76	96.97	100.38	107.86	112.76	102.13	100.39	102.24	87.11	86.98	99.70
Feb.	103.53	92.67	78.95	100.03	95.20	99.99	116.79	110.42	96.05	103.45	89.67	106.16	105.33
Mar.	113.61	113.01	108.94	115.01	109.41	107.74	129.00	133.36	122.18	112.94	102.34	114.40	131.95
Apr.	105.72	113.55	104.26	120.41	112.79	123.22	134.11	127.71	129.43	96.92	118.75	112.18	127.39
May	101.73	124.77	113.52	130.19	123.83	129.46	131.71	129.31	141.34	120.15	113.85	115.89	136.98
June	98.37	120.89	110.80	119.49	111.85	127.79	132.20	132.97	144.79	109.72	111.28	110.19	135.56
July	92.43	115.00	99.71	109.83	112.63	134.23	123.08	124.79	132.68	120.07	110.96	106.92	131.84
Aug.	91.49	113.76	106.47	108.58	109.77	129.30	117.80	132.80	134.10	122.32	112.61	105.99	
Sept.	88.66	107.44	111.20	101.15	110.44	139.18	127.50	123.21	121.15	109.48	109.91	104.62	
Oct.	95.70	99.74	110.02	109.90	125.67	146.53	130.10	113.13	127.35	116.18	111.49	108.88	
Nov.	95.54	96.25	106.15	101.58	111.82	134.40	118.11	106.35	116.34	103.98	86.97	106.90	
Dec.	105.36	103.75	105.58	96.31	105.05	115.07	119.04	105.90	105.51	83.19	82.45	100.67	

Source: IBGE.