Neural Networks, Vol. 5, pp. 961–970, 1992
Printed in the USA. All rights reserved.

0893-6080/92 $5.00 + .00

ORIGINAL CONTRIBUTION

Forecasting the Behavior of Multivariate Time Series Using Neural Networks

KANAD CHAKRABORTY, KISHAN MEHROTRA, CHILUKURI K. MOHAN, AND SANJAY RANKA

Syracuse University

(*Received* 13 *December* 1990; *revised and accepted* 6 *April* 1992)

Abstract—*This paper presents a neural network approach to multivariate time-series analysis. Real world observations of flour prices in three cities have been used as a benchmark in our experiments. Feedforward connectionist networks have been designed to model flour prices over the period from August 1972 to November 1980 for the cities of Buffalo, Minneapolis, and Kansas City. Remarkable success has been achieved in training the networks to learn the price curve for each of these cities and in making accurate price predictions. Our results show that the neural network approach is a leading contender with the statistical modeling approaches.*

Keywords—Neural networks, Back propagation, Multivariate time-series, Statistical models, Training, One-lag prediction, Multi-lag prediction, Combined modeling, Forecasting.

1. INTRODUCTION

Predicting the future is the prime motivation behind the search for laws that explain certain phenomena. As observed by Weigend, Huberman, & Rumelhart (1990), it hinges on two types of knowledge: knowledge of underlying laws, a very powerful and accurate means of prediction, and the discovery of strong empirical regularities in observations of a given system. However, there are problems with both approaches—discovery of laws underlying the behavior of a system is often a difficult task, and empirical regularities or periodicities are not always evident and can often be masked by noise.

Multivariate time-series analysis is an important statistical tool to study the behavior of time dependent data and forecast future values depending on the history of variations in the data. A time-series is a sequence of values measured over time, in discrete or continuous time units. By studying many related variables together, rather than by studying just one, a better understanding is often obtained. A multivariate time-series consists of sequences of values of several contemporaneous variables changing with time. An important case is when the variables being measured are significantly correlated, e.g., when similar attributes are being measured at different geographic locations. In forecasting new values for each variable, better prediction capabilities are available if variations in the other variables are also taken into account. Robust forecasting must rely on all available correlations and empirical interdependencies among different temporal sequences.

Many available techniques for time-series analysis assume linear relationships among variables (see Box and Jenkins, 1970). But in the real world, temporal variations in data do not exhibit simple regularities and are difficult to analyze and predict accurately. Linear recurrence relations and their combinations for describing the behavior of such data are often found to be inadequate. It seems necessary, therefore, that nonlinear models be used for the analysis of real-world temporal data. Tong (1983) describes some of the drawbacks of linear modeling for time series analyses. These include, e.g., their inability to explain sudden bursts of very large amplitudes at irregular time intervals. Discussion following the Tiao and Tsay (1989) paper also addresses some of the problems with linear models for multivariate time series. In order to accommodate for such inabilities, nonlinear statistical models, such as the threshold model and the bilinear model, have been suggested and discussed in Tong (1990), whereas Granger and Newbold (1986) suggest the use of nonlinear transformation of the original data before performing the "usual" linear modeling. Farmer and Sidorowich (1987) report a significantly better predic-

Requests for reprints should be sent to Kishan Mehrotra, 4-116 Center for Science and Technology, School of Computer and Information Science, Syracuse University, Syracuse, NY 13244.

tion for chaotic time series by using a local approximation approach involving use of nearest k neighbors with respect to the magnitudes of the values rather than with respect to time points to which the values belong. Despite considerable progress over the last decade, formulation of reasonable nonlinear models is an extremely difficult task because of simplifications made in the modeling stage, e.g., omitting parameters which are unknown or which do not seem to affect the observed data directly. Also, the relationships between known parameters and observed values can only be hypothesized with no simple laws governing their mutual behavior. See, for example, Saikkonen and Luukkonen (1991). Hence, we resort to a "neural network" approach for nonlinear modeling of multivariate time-series; in earlier work, we have successfully used this approach in analyzing univariate time-series (see Li, Mehrotra, Mohan, & Ranka, 1990).

Neural networks belong to the class of *data-driven* approaches, as opposed to model-driven approaches. The analysis depends on available data, with little rationalization about possible interactions. Relationships between variables, models, laws, and predictions are constructed post-facto after building a machine whose behavior simulates the data being studied. The process of constructing such a machine based on available data is addressed by certain general-purpose algorithms such as "back propagation" (see Rumelhart, Hinton, & Williams, 1986).

In this paper, we use neural networks to predict future values of possibly noisy multivariate time-series based on past histories. The particular data analyzed are monthly flour prices for Buffalo, Minneapolis, and Kansas City over a period of 100 months. For impartial evaluation of the prediction performance of the approach, data for different periods are used in the "training" (modeling) and "testing" (prediction) phases. The performance exceeded expectations and the root mean squared errors (in long-term, or multilag prediction) obtained using this approach are better than those obtained from the statistical model of Tiao and Tsay (1989) by at least one order of magnitude. We expect such results to be obtained in other applications of interdependent time-series as well.

Section 2 presents the architecture of the neural networks used for our analysis, the experiments performed and the training paradigm used. In Section 2.3, a model of statistical prediction is described and its performance compared in Section 3 with the network performance. Discussion and concluding remarks then follow.

As stated before, several other statistical approaches have been suggested for modeling and analysis of time-series data. We use the model proposed by Tiao and Tsay as a benchmark to judge the neural network performance. Our results indicate that the back propagation network approach provides a competitive alternative to the existing procedures for learning as well as forecasting of interdependent data.

2. METHODOLOGY

2.1. Neural Networks

Artificial neural networks are computing systems containing many simple nonlinear computing units or nodes interconnected by links. In a "feedforward" network, the units can be partitioned into layers, with links from each unit in the k^{th} layer being directed (only) to each unit in the $(k+1)^{st}$ layer. Inputs from the environment enter the first layer, and outputs from the network are manifested at the last layer. A $d-n-1$ network, shown in Figure 1, refers to a network with d inputs, n units in the intermediate "hidden" layer, and one unit in the output layer (see Weigend et al., 1990). A weight or "connection strength" is associated with each link, and a network "learns" or is trained by modifying these weights, thereby modifying the network function which maps inputs to outputs.

We use such $d-n-1$ networks to learn and then predict the behavior of multivariate time-series. The hidden and output nodes realize nonlinear functions of the form $(1+\exp(-\sum_{i=1}^{m} w_i x_i + \theta))^{-1}$, where w_i's denote real-valued weights of edges incident on a node, θ denotes the adjustable "threshold" for that node, and m denotes the number of inputs to the node from the previous layer. The training algorithm is detailed in Section 2.2.

2.2. Procedure for Training the Networks

We use the error back propagation algorithm of Rumelhart et al. (1986) to train the networks, using mean squared error (MSE) over the training samples as the objective function. From the given p-variate time-series $S = \{\langle v_1(t), \ldots, v_p(t)\rangle : 1 \le t \le N\}$, we obtain two sets: a training set $S_{\text{train}} = \{\langle v_1(t), \ldots, v_p(t)\rangle : 1 \le t \le T\}$, and a test set $S_{\text{test}} = \{\langle v_1(t), \ldots, v_p(t)\rangle : T < t \le N\}$. A set P_{train} of $(d+1)$-tuples (the first d

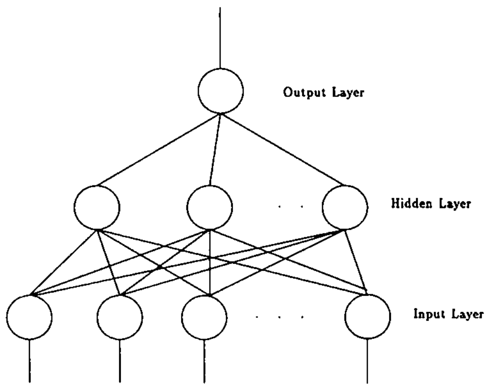

FIGURE 1. A feedforward neural net with one hidden layer.

components representing inputs and the last component representing the desired outputs) and a set P_{test} of d-tuples (each component representing an input) are then created from S_{train} and S_{test}, respectively. The MSE(training) is calculated as $\sum_{k=1}^{M} (u^k - \hat{u}_k)^2/M$, where u_k and $\hat{u}_k$, $1 \le k \le M$ are the desired and the actual network outputs, respectively, for each of the M training patterns. The MSE(test) is similarly defined for the test patterns.

In each step in the training phase, a d-tuple (recent history) of normalized input data is presented to the network. The network is asked to predict the next value in the time sequence. The error between the value predicted (by the network) and the value actually observed (known data) is then measured and propagated backwards along the feedforward connections. The weights of links between units and node thresholds are modified to various extents, using a technique which apportions "blame" for the error to various nodes and links as prescribed by the back propagation algorithm. If the MSE exceeds some small predetermined value, a new "epoch" (cycle of presentations of all training inputs) is started after termination of the current epoch. After training the network, its performance on test samples is measured in terms of MSE(test), the mean squared error on the test samples alone.

The parameters of the back propagation algorithm are the "learning rate" and "momentum," which roughly describe the relative importance given to the current and past error-values in modifying connection strengths. For better performance in our experiments, we found that it was best to use a small learning rate in training the network. In all training cases, we chose a learning rate of 0.3 and an associated momentum term of 0.6. The number of epochs varied between 25,000 to 50,000 in all cases.

2.3. Experiments

In our experiments, we analyzed a trivariate time-series $X_T = \{(x_t, y_t, z_t) : t = 1, 2, \ldots, T\}$, where T ranges up to 100. The data used are logarithms of the indices of monthly flour prices for Buffalo (x_t), Minneapolis (y_t), and Kansas City (z_t), over the period from August 1972 to November 1980, obtained from Tiao and Tsay (1989). In all cases, we train the network over a certain part of our data, and once training is completed, test the network over the remaining data, i.e., make the network predict the so-called "future" values.

Both "one-lag" and "multi-lag" output predictions for the test samples are done with the given models. In one-lag prediction, we forecast flour prices of each year based only on actual past values. In multi-lag prediction, on the other hand, we append the predicted values to our input database and use these values also to predict future values. For instance, if the network is used

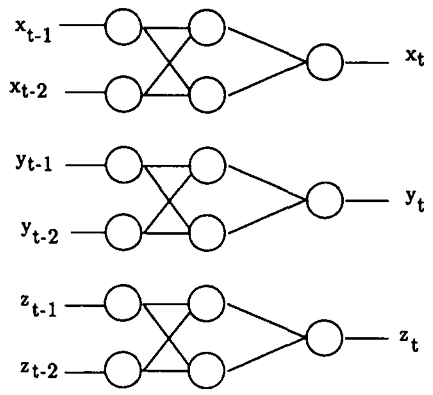

FIGURE 2. Separate architectures schema.

to predict a value n_6 from observed input data $i_1, \ldots i_5$, then the next network prediction n_7 is made using inputs $i_2, \ldots i_5, n_6$, and the subsequent network prediction n_8 is made using inputs i_3, i_4, i_5, n_6, n_7. With one-lag prediction, on the other hand, the prediction at the eighth instant is made using only the actual input data values i_3, i_4, i_5, i_6, i_7. The following three sets of experiments were performed in this study.

1. **Separate Modeling:** Each univariate time-series $x_T = \{x_t : t = 1, 2, \ldots, T\}$, $y_T = \{y_t : t = 1, 2, \ldots, T\}$, and $z_T = \{z_t : t = 1, 2, \ldots, T\}$, was analyzed separately, without utilizing their interdependencies. For example, only the values of $x_1, \ldots, x_k$ were used to predict x_{k+1}. A separate neural network was used for each of the three series, as illustrated in Figure 2, and trained with 90 input data values ranging from August 1972 to January 1980. The training phase was followed by output prediction for the next 10 time points (for February 1980 to November 1980) using the weights and thresholds generated during training. These predictions were compared with the test data set

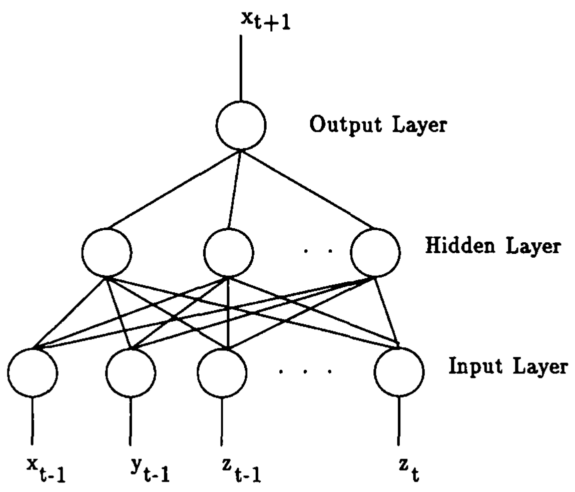

FIGURE 3. Combined architectures schema.

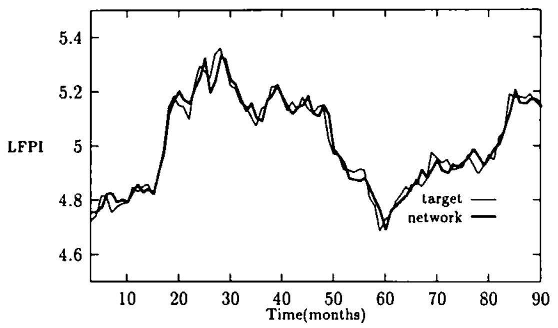

FIGURE 4. 6-6-1 network modeling: Buffalo (training).

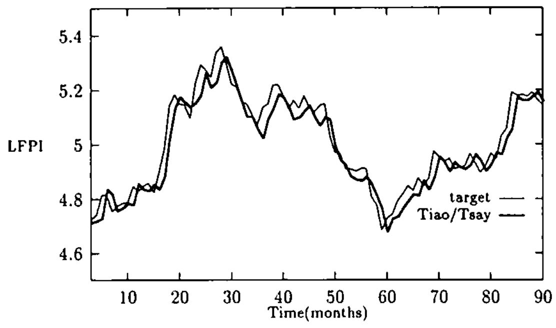

FIGURE 6. Tiao & Tsay's model: Buffalo (90 months).

to judge the performance of the network in terms of MSE(test). Experiments were performed with 2-2-1, 4-4-1, 6-6-1, and 8-8-1 networks. These experiments were done in order to compare with the combined modeling approach, described below.

2. **Combined Modeling:** We obtained a considerably improved performance using (for each series) information from all three series, instead of treating each series individually. Two kinds of experiments were performed using this approach. The first kind is illustrated in Figure 3 in which x_{t+1} is shown as being learned/predicted using six preceding values (recorded at time points t and $t-1$ only) from all the three series, i.e., $(x_t, y_t, z_t, x_{t-1}, y_{t-1}, z_{t-1})$. Similar experiments were performed to predict y_{t+1} and z_{t+1} also. This method determines the value of a variable at any time t using strictly past data for all the variables in each training input; it does *not* utilize contemporaneous data for the other variables. We experimented with 6-6-1 networks of this kind and the results are shown in Table 2. We experimented with a variety of 6-h-1 networks and obtained the best results with 6-6-1 networks. The reason for choosing 6 input nodes is mentioned in Section 3.

However, for the data studied, there was an implicit ordering among the three series: x_t values were available before y_t values, and y_t values were available before z_t values, and (naturally) all these were available before x_{t+1} values. This observation led to our second kind of combined modeling approach. In this approach, each training input pattern consists of "past" data items by the above criterion. For instance, in the $d-n-1$ feedforward network used to predict y_t, if $d = 8$, the chosen input values would be $x_t, z_{t-1}, y_{t-1}, x_{t-1}, z_{t-2}, x_{t-2}, y_{t-2}, z_{t-3}$. In both cases, the training set consisted of the first 90 items of trivariate data and the test set consisted of the remaining 10. The results shown in Figures 4 through 30 compare performances of both a 6-6-1 network (the first approach) and an 8-8-1 network (the second approach) with that of Tiao and Tsay's model. As before, we have explained our choice of an 8-h-1 network in Section 3. The best performances were obtained with $h = 8$ for such networks that were experimented with. In all the graphs shown, the y-axis is labeled "LFPI," an abbreviation for "Logarithms of monthly Flour Price Indices."

3. **Statistical Model:** Tiao and Tsay (1989) developed a statistical model of prediction which involves computation of the overall order of the temporal process with the help of normalized criterion and root tables, followed by estimation of unknown parameters. For this example, it was found that a trivariate ARMA(1,1) model or AR(2) model would be appropriate for the data. Subsequently, Tiao and Tsay (1989) modeled the data as described below.

First, each trivariate input vector $\mathbf{z}_t$ has to be transformed by premultiplication with a 3×3 matrix $\mathbf{T}$, called the transformation matrix. The transformed data conforms to a trivariate equation of the form

$$(\mathbf{I} - \Phi_1 B)\mathrm{y}_t = \mathbf{c} + (\mathbf{I} - \Phi_1 B)\mathbf{a}_t$$

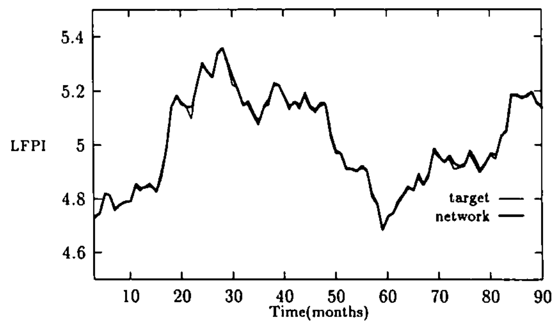

FIGURE 5. 8-8-1 network modeling: Buffalo (training).

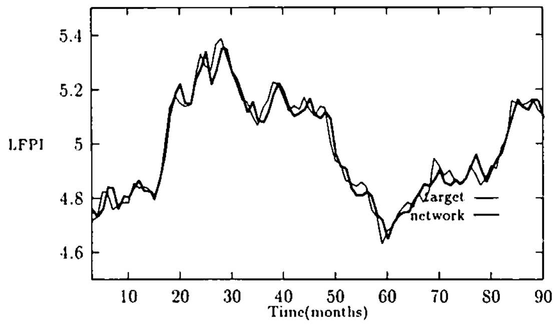

FIGURE 7. 6-6-1 network modeling: Minneapolis (training).

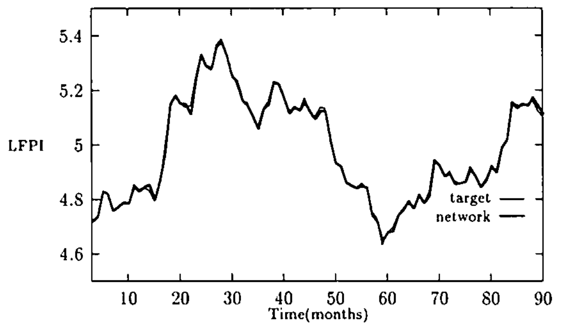

FIGURE 8. 8-8-1 network modeling: Minneapolis (training).

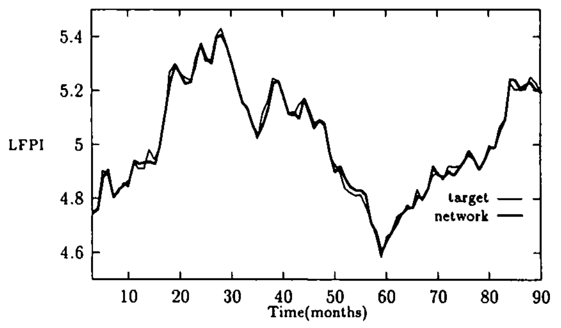

FIGURE 11. 8-8-1 network modeling: Kansas City (training).

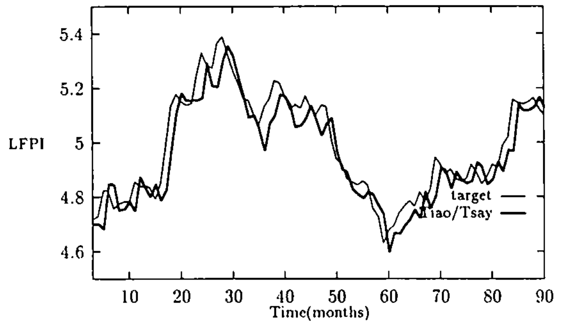

FIGURE 9. Tiao & Tsay's model: Minneapolis (90 months).

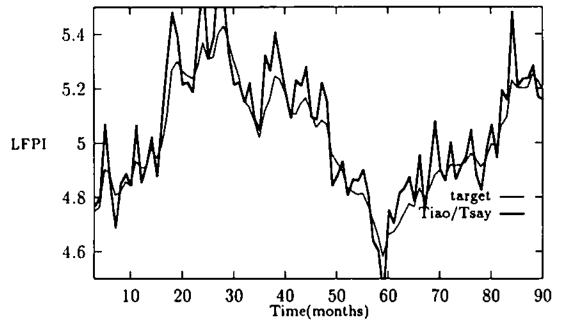

FIGURE 12. Tiao & Tsay's model: Kansas City (90 months).

where the transformed series $\mathbf{y}_t = \mathbf{T}\mathbf{z}_t$ is a 3×1 (trivariate) column vector, B represents the backshift operator (i.e., $B\mathbf{y}_t = \mathbf{y}_{t-1}$), and the 3×1 column vectors $\mathbf{a}_t$ comprise the error components of the model. The matrix coefficients $(\mathbf{I} - \Phi_1 B)$ and $(\mathbf{I} - \Theta_1 B)$ represent the autoregressive and moving average components, respectively. The estimated values of the 3×3 matrices Φ_1, Θ_1, and the 3×1 vector $\mathbf{c}$ are given in Tiao and Tsay (1989).

In the above trivariate model, the mean squared errors are obtained from $\mathbf{a}_t$'s, after premultiplying each such vector by the inverse of the transformation matrix $\mathbf{T}$. The manner of computing the $\mathbf{a}_t$ vectors is as follows. We initialize $\mathbf{a}_1$ to zero; and for $t = 1, 2 \ldots 89$, compute $\mathbf{y}_{t+1}$ by the recipe of the model by using known values of $\mathbf{y}_t$, $\mathbf{a}_t$. The error vector $\mathbf{a}_t$ is obtained at each step by taking the difference between the computed and the actual values of $\mathbf{y}_t$, for $t = 2, 3 \ldots 90$. One-lag prediction is merely a continuation of the above process for $t = 91, 92 \ldots$, and multi-lag prediction is performed in a similar fashion but using estimated values of $\mathbf{y}_t$ for $t = 91, 92, \ldots, 99$ instead of actual values $\mathbf{y}_{91}, \ldots, \mathbf{y}_{99}$.

3. COMPARATIVE ANALYSIS OF EXPERIMENTAL RESULTS

The mean squared errors obtained for three different models of separate modeling networks are presented in Table 1. The values correspond to the mean squared errors observed for (a) the first 90 univariate data items, which correspond to the training data, (b) one-lag, and (c) multi-lag predictions over the remaining 10 univariate data items for each of the three time-series. It is interesting to observe that the training performance

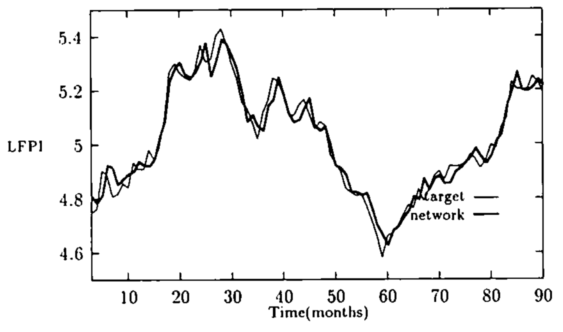

FIGURE 10. 6-6-1 network modeling: Kansas City (training).

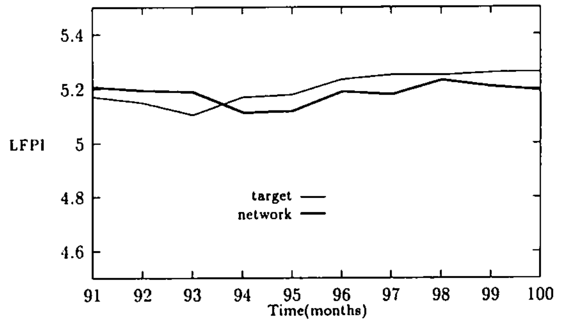

FIGURE 13. 6-6-1 network prediction, one-lag (Buffalo).

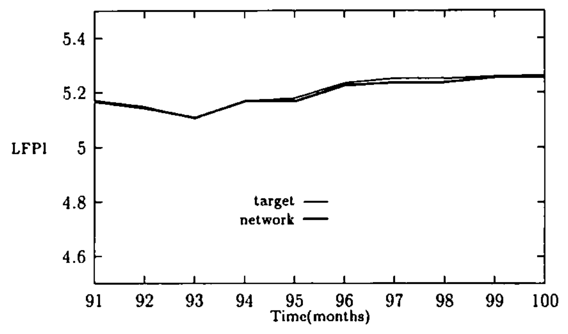

FIGURE 14. 8-8-1 network prediction, one-lag (Buffalo).

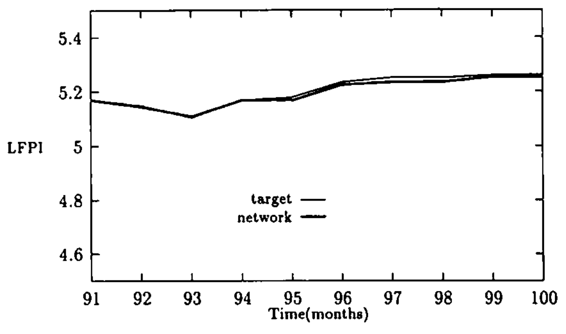

FIGURE 17. 8-8-1 network prediction, multi-lag (Buffalo).

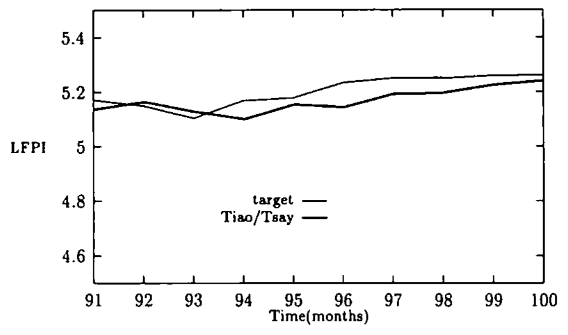

FIGURE 15. Tiao & Tsay's model, one-lag (Buffalo).

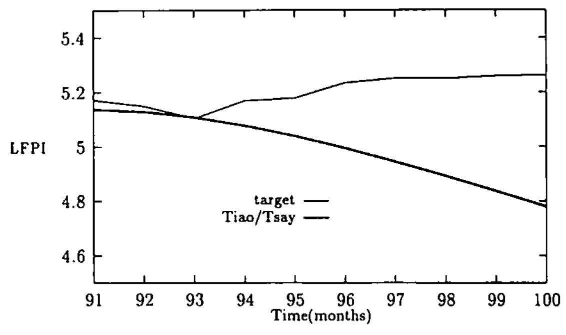

FIGURE 18. Tiao & Tsay's model, multi-lag (Buffalo).

improves whereas one-lag and multi-lag performances deteriorate (in terms of MSE(training) and MSE(test), respectively), as the size of the networks is increased. This suggests that the 4-4-1 and 6-6-1 networks are oversized for the given univariate data and a 2-2-1 network is more suitable for prediction. Thus, every data item in each time-series considered individually is found to be strongly dependent on the past two values only, a fact which agrees with Tiao and Tsay's AR(2) model. This observation also led us to experiment with 6-*h*-1 and 8-*h*-1 networks for the two combined modeling approaches, respectively. Since combined modeling uses trivariate data in our example, the choice of 6 for the number of inputs in the first case is obvious. For the second combined modeling approach, we uniformly chose an 8-*h*-1 network for modeling each city because 8 is the least number of network inputs which ensures that there are two data items belonging to strictly past time points for each city.

Considerably improved performance, as measured in terms of mean squared errors in training and testing, resulted by following the combined modeling approach for the given data. We believe that separate modeling gives poorer results than combined modeling for two reasons. First, since the three series have a high pairwise positive correlation, each series carries information valuable not only for prediction of its own future values but also for those of the other two series. Second, the combined modeling training set contains three times as many observations as are available for each single modeling training set.

The mean squared errors and coefficients of variation for three different sets of experiments are listed in Table 2. The values correspond, respectively, to the

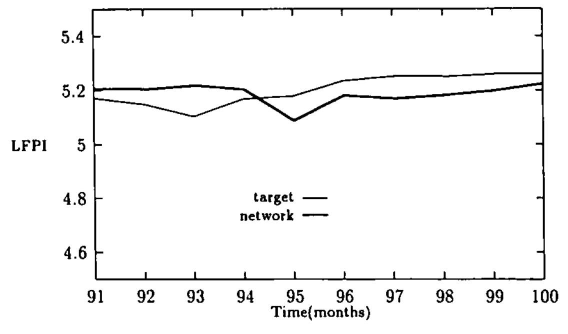

FIGURE 16. 6-6-1 network prediction, multi-lag (Buffalo).

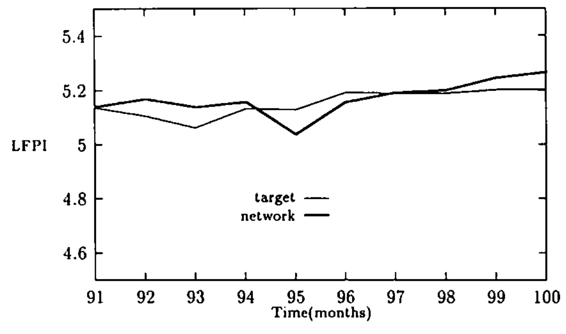

FIGURE 19. 6-6-1 network prediction, one-lag (Minneapolis).

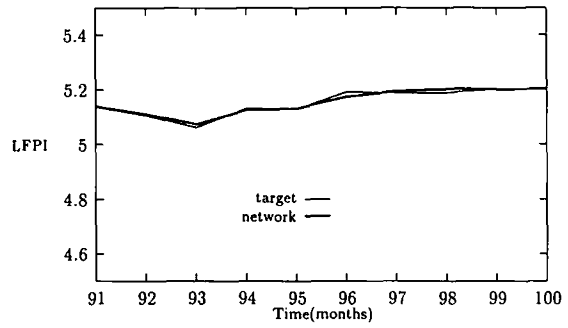

FIGURE 20. 8-8-1 network prediction, one-lag (Minneapolis).

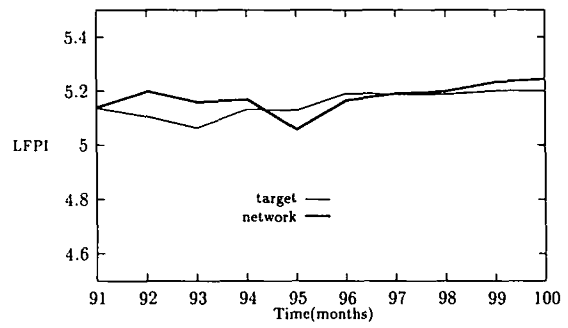

FIGURE 22. 6-6-1 network prediction, multi-lag (Minneapolis).

mean squared errors and coefficients of variation observed for (a) the first 90 trivariate data items, which correspond to the training data for the combined modeling networks, (b) one-lag, and (c) multi-lag predictions of the combined modeling network and statistical models. Tiao and Tsay's model is seen to produce greater mean squared errors in training and prediction than the combined modeling networks in all but one case (one-lag prediction for Buffalo, for which Tiao and Tsay's model outperforms the 6-6-1 model.) A word of caution is in order at this point. The model of Tiao and Tsay was derived using all 100 trivariate observations. For comparison with our network errors, we obtained two sets of MSE values from their model. In this sense, the neural network is in a slightly disadvantageous position because the network modeling is based on the first 90 trivariate observations only. Yet this method is a simple and desirable way to perform comparisons between the two approaches.

We also notice that the 8-8-1 combined modeling network, which assumes ordering among the three series, performs considerably better than the 6-6-1 network, which utilizes strictly past data in both modeling and prediction. This proves that for the example studied, contemporaneous values of the two other variables have a great role to play in determining the behavior of a variable at any time instant. This fact indicates that there is a strong pairwise correlation between the data for the three cities.

The performance of the neural networks did not vary much for different choices of input sizes in the training and prediction phases of our experiments and the results shown in Table 2 are fairly representative. Since back propagation is a gradient-descent optimization technique, training a network may result in its sinking into a local minimum of the MSE that may be far removed from the global one (see Tesauro & Janssens, 1988). Hence, to ensure that the MSE(training) was not trapped in such a local minimum, several training runs were conducted for each network. The MSE values achieved in these runs were found to be extremely small quantities that differed only slightly from one another.

A small value of MSE(training) does not necessarily imply a small value of MSE(test) as well. However, our experimental results generally indicate that if the neural network performs well on the training samples, it also has good generalization capabilities, i.e., MSE(test) is also quite small. This is to be expected if the training and test samples are drawn from the same distribution and the network is not overtrained. In most of our experiments, the MSE(training) and MSE(test) values for the network were highly correlated.

4. DISCUSSION

Most statistical models for learning and predicting time-series are based only on linear recurrences. Though computationally inexpensive, such functions do not often accurately represent temporal variations. Nonlinear functions, on the other hand, are more useful for tracing temporal sequences. This is probably the main reason for the significantly better performance of the neural

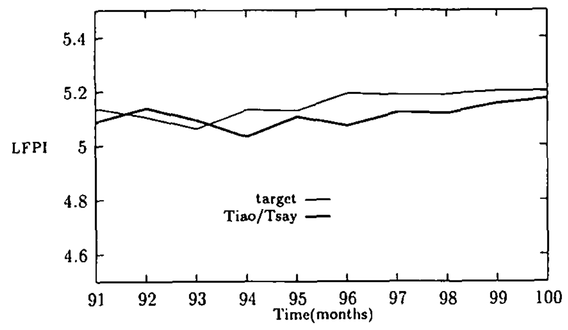

FIGURE 21. Tiao & Tsay's model, one-lag (Minneapolis).

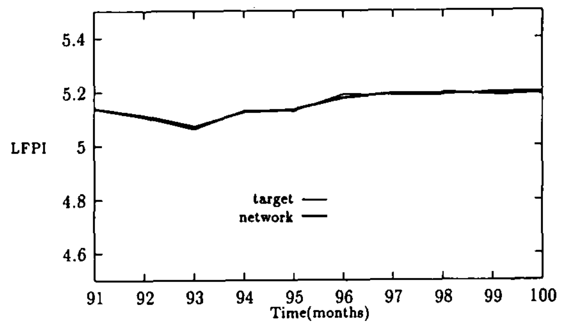

FIGURE 23. 8-8-1 network prediction, multi-lag (Minneapolis).

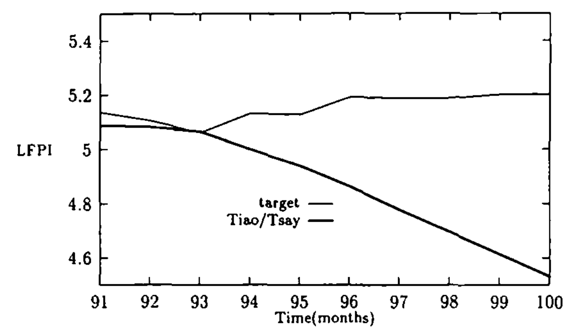

FIGURE 24. Tiao & Tsay's model, multi-lag (Minneapolis).

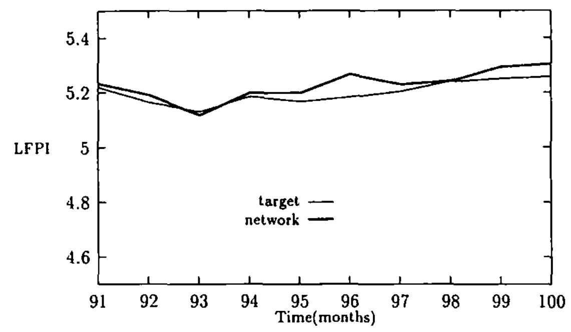

FIGURE 26. 8-8-1 network prediction, one-lag (Kansas City).

network approach (with nonlinearities at each node) as compared to statistical modeling.

In any nontrivial time-series, new values depend not only on the immediately prior value but also on many preceding values. Using too few inputs can result in inadequate modeling, whereas too many inputs can excessively complicate the model. In the context of neural networks, too many inputs would imply slower training and slower convergence, and may in fact worsen the generalization capabilities (applicability to test cases) of the network. Weigend et al. (1990) have a rule of thumb for determining the number of weights in the network as a function of the number of training samples. But this rule was found to be too restrictive for the data set of 100 patterns we worked with, and hence, had to be disregarded.

Different types of connectionist models have been proposed for learning temporal variations of data. It has generally been held in the past that recurrent networks are more suitable for learning temporal data. There were two reasons why recurrent networks were not used for modeling the trivariate data on flour-prices—we observed experimentally that unfolding them into simple feedforward networks would cause worse training and output predictions than single hidden layer feedforward nets; the network would become inherently slower because of a much greater amount of computation involved. An unfolded version of a recurrent network is an approximation of it and implementing an exact recurrent network is a computationally expensive task. The main reason for this is that the units in hidden layers must be made to iterate among themselves until their outputs converge, and there is no way of knowing a priori how many iterations it would take before all the hidden units have stable outputs. Simple feedforward nets are less computationally intensive and in many applications give good performance in less time.

A potential objection to the claim of improved performance using the neural network approach, in comparison to the statistical approach, is that neural networks are more complex and have many more parameters (weights and thresholds): Would a more complex statistical model perform equally well? The answer is essentially methodological. Often, the real-world phenomena being modeled are so complex that it is impossible to theorize and generate statistical models. A large investment of experts' domain-specific research studies must precede the formulation of an adequate model for each separate phenomenon. When a large number of parameters are involved, it is difficult to predict data even when the laws which govern their behavior are known, e.g., in the gravitational interactions between a large number of bodies. With neural networks, on the contrary, an essentially similar architecture can be quickly modified and trained for a variety of different phenomena. The procedure is data-driven rather than model-driven and gives good results in many cases despite the unavailability of a good theory/model underlying observed phenomena.

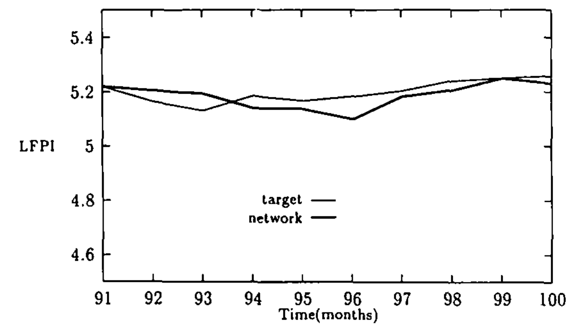

FIGURE 25. 6-6-1 network prediction, one-lag (Kansas City).

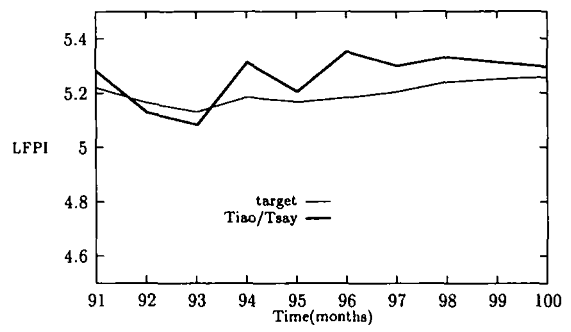

FIGURE 27. Tiao & Tsay's model, one-lag (Kansas City).

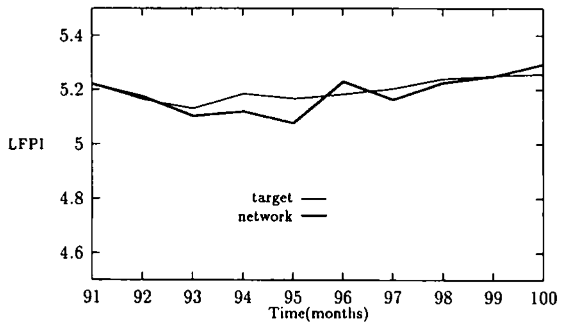

FIGURE 28. 6-6-1 network prediction, multi-lag (Kansas City).

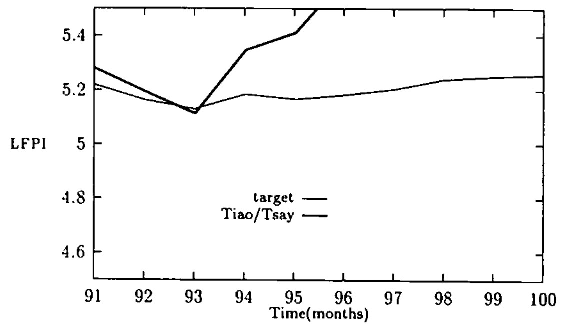

FIGURE 30. Tiao & Tsay's model, multi-lag (Kansas City).

We now evaluate the neural network approach with respect to the following criteria for a good model suggested in the literature, see Harvey (1989):

1. *Parsimony:* The neural network does contain a large number of parameters and is, hence, not parsimonious. However, the method of training does not impose any external biases and networks started with different random weights successfully converged to approximate the time-series very well.
2. *Data coherence:* The neural network model provides a very good fit with the data, as shown by the low MSE values for the training samples.
3. *Consistency with prior knowledge:* No explicit theory was constructed using the neural networks, hence, this criterion is largely irrelevant. In the best neural network model, the assumption that flour prices become known in a fixed order ($x_1, y_1, z_1, x_2, \ldots$) is consistent with the information that the data are available slightly earlier for some cities than for others. So the network modeling can easily accommodate contemporaneous data which become known in a fixed predetermined order.
4. *Data admissibility:* The values in a time-series predicted by the neural networks are always close to the immediately preceding values and do not violate any obvious definitional or reasonable constraints.
5. *Structural stability:* The neural networks seem to satisfy this criterion because they give a good fit for test data, which are outside the set of training samples. Multiple training runs started with changed random initial weights for the same network architecture produced remarkably consistent MSE values after training. In all our experiments, the MSE values after training were very close to zero.
6. *Encompassing:* The results obtained using the neural networks are better than those obtained using an alternative statistical model, indicating that the network exhibits the potential of a competitive alternative tool for analysis and especially, prediction, of multivariate time-series. However, no theory is directly suggested by the neural networks developed so far. The design of forecasting models utilizing the parameters of the trained neural networks is currently under way.

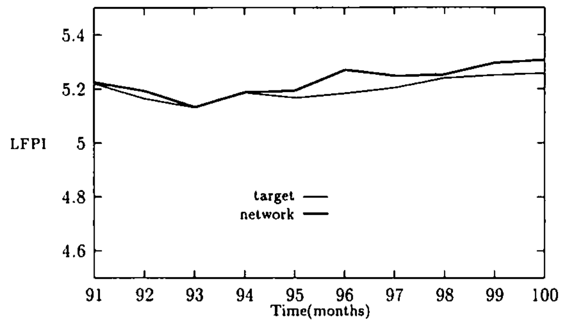

FIGURE 29. 8-8-1 network prediction, multi-lag (Kansas City).

4.1. Conclusions

We have presented a neural network approach to multivariate time-series analysis. In our experiments, real world observations of flour prices in three cities have been used to train and test the predictive power of feedforward neural networks. Remarkable success has been achieved in training the networks to learn the price curve for each of these cities, and thereby to make accurate price predictions. Our results show that the neural network approach leads to better predictions

TABLE 1
Mean-Squared Errors for Separate Modeling & Prediction $\times 10^3$

		Buffalo	Minneapolis	Kansas City
Network		MSE	MSE	MSE
2-2-1	Training	3.398	3.174	3.526
	One-lag	4.441	4.169	4.318
	Multi-lag	4.483	5.003	5.909
4-4-1	Training	3.352	3.076	3.383
	One-lag	9.787	8.047	3.370
	Multi-lag	10.317	9.069	6.483
6-6-1	Training	2.774	2.835	1.633
	One-lag	12.102	9.655	8.872
	Multi-lag	17.646	14.909	13.776

TABLE 2
Mean-Squared Errors and Coeffs. of Variation for Combined vs. Tiao/Tsay's Modeling/Prediction × 10^3

		Buffalo		Minneapolis		Kansas City	
Model		MSE	CV	MSE	CV	MSE	CV
6-6-1	Training	1.446	7.573	1.554	7.889	1.799	8.437
Network	One-lag	3.101	11.091	3.169	11.265	2.067	9.044
	Multi-lag	3.770	12.229	3.244	11.389	2.975	10.850
8-8-1	Training	0.103	2.021	0.090	1.898	0.383	3.892
Network	One-lag	0.087	1.857	0.072	1.697	1.353	7.316
	Multi-lag	0.107	2.059	0.070	1.674	1.521	7.757
Tiao/Tsay	Training	2.549	10.054	5.097	14.285	8.645	18.493
	One-lag	2.373	9.701	4.168	12.917	7.497	17.222
	Multi-lag	72.346	53.564	137.534	74.204	233.413	96.096

than a well-known autoregressive moving average model given by Tiao and Tsay (1989).

We obtained a very close fit during the training phase and the networks we developed consistently outperformed statistical models during the prediction phase.

We are currently exploring the combination of statistical and neural approaches for time-series analyses. We expect that model-based statistical preprocessing can further improve the performance or help in obtaining faster convergence of neural networks in the task of time-series predictions.

REFERENCES

Box, G. E. P., & Jenkins, G. M. (1970). *Time series analysis: Forecasting and control.* San Francisco, CA: Holden-Day.

Farmer, J. D., & Sidorowich, J. J. (1987). Predicting chaotic time series. *Physical Review Letters,* **59,** 845–848.

Granger, C. W. J., & Newbold, P. (1986). *Forecasting economic time series* (2nd ed.). Orlando, FL: Academic Press.

Harvey, A. C. (1989). *Forecasting, time series models and the Kalman Filter.* U.K.: Cambridge University Press.

Li, M., Mehrotra, K., Mohan, C. K., & Ranka, S. (1990). Sunspot numbers forecasting using neural networks. *Proceedings of the IEEE Symposium on Intelligent Control,* **1,** 524–529.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, & J. L. McClelland (Eds.), *Parallel distributed processing* (pp. 318–362). Cambridge, MA: MIT Press.

Saikkonen, P., & Luukkonen, R. (1991). Power properties of a time series linearity test against some simple bilinear alternatives. *Statistica Sinica,* **1,** 453–464.

Tesauro, G., & Janssens, B. (1988). Scaling relationships in back-propagation learning. *Complex Systems,* **2,** 39–44.

Tiao, G. C., & Tsay, R. S. (1989). Model specification in multivariate time series. *Journal of the Royal Statistical Society,* **B 51,** 157–213.

Tong, H. (1983). *Threshold models in non-linear time series analysis.* Lecture Notes in Statistics, **21,** New York: Springer-Verlag.

Tong, H. (1990). *Non-linear time series: A dynamical system approach.* Oxford: Oxford University Press.

Weigend, A. S., Huberman, B. A., & Rumelhart, D. E. (1990). Predicting the future: A connectionist approach. *International Journal of Neural Systems,* **1,** 193–209.