Time Series Forecasting with Neural Networks: A Case Study

By JULIAN FARAWAY
University of Michigan, USA
and CHRIS CHATFIELD [1]
University of Bath, UK

Research Report 95-06 of the Statistics Group, University of Bath.

December 19th, 1995

SUMMARY

Using the famous airline data as the main example, a variety of neural network (NN) models are fitted and the resulting forecasts are compared with those obtained from the (Box-Jenkins) seasonal ARIMA model called the airline model. The results suggest that there is plenty of scope for going badly wrong with NN models and that it is unwise to apply them blindly in 'black-box' mode. Rather the wise analyst needs to use traditional modelling skills to select a good NN model, for example in making a careful choice of input variables. The BIC criterion is recommended for comparing different NN models. Great care is also needed when fitting a NN model and using it to produce forecasts. Methods of examining the response surface implied by a NN model are examined as well as alternative procedures using Generalized Additive Models and Projection Pursuit Regression.

Keywords: ARIMA model; Box-Jenkins forecasting; Airline model; Generalized additive model; Projection pursuit regression

[1] *Address for Correspondence:* School of Mathematical Sciences, University of Bath, Bath BA2 7AY, UK. E-mail: cc@maths.bath.ac.uk

1 Introduction

Neural networks (hereafter abbreviated NNs) have been vigorously promoted in the computer science literature for tackling a wide variety of scientific problems. Recently, investigations have started to see how useful NNs are for tackling statistical questions in general (Ripley, 1993, 1996; Cheng and Titterington, 1994), and for time-series modelling and forecasting in particular (e.g. Hill et al., 1994, 1996; Gorr et al., 1994). Despite some impressive claims, the empirical results using NN models have been rather mixed — see for example Chatfield (1993) and the review in Section 3 of Hill et al. (1996) — though the results in Section 7 of the latter paper using the M-competition data do provide some evidence that NN models may be able to give more accurate forecasts on average over a diverse collection of time-series data than some alternative univariate forecasting procedures.

It is pertinent to ask whether the success of NN modelling depends on (a) the type of data (b) the skill of the analyst in selecting a suitable NN model and/or (c) the numerical methods used to fit the model and compute predictions. Experience for question (a) can be built up with large-scale forecasting competitions, while case studies can be used to address questions (b) and (c). This paper describes a case study which aims to do just that.

2 Box-Jenkins analysis of the airline data

The main time series used in this paper is the so-called airline data, made famous by the first edition (in 1970) of Box et al. (1994, Series G), though it was earlier given by Brown (1962). Fig. 1 shows that the data have an upward trend together with seasonal variation whose size is roughly proportional to the local mean level and hence is said to be *multiplicative*.

The standard Box-Jenkins analysis (e.g. Harvey, 1993; Box et al., 1994) generally incorporates taking a logarithmic transformation of the data in order to make the seasonal effect additive (see Fig. 1), and then taking one seasonal and one non-seasonal difference in order to make the series stationary. One seasonal and one non-seasonal moving average terms are then fitted. The resulting model is a special type of seasonal ARIMA model, often called the *airline model*. The model is of order (0,1,1) $\times$ $(0,1,1)_{12}$ in the usual notation (see for example Box et al., 1994, p. 333). This model is generally regarded as being the most suitable model for this particular set of data and will be used as the yardstick for NN models, though many other seasonal ARIMA models could be fitted which give measures of fit and forecast accuracy nearly as good as those for the airline model.

Most of the results reported in this paper are computed by fitting a model to the first 11 years of data and then making forecasts of the last 12 observations. The latter will be computed in two different ways. Firstly all forecasts will be made from time period 132 using the first 11 years data only. These forecasts will be called Multi-step (abbreviated MS) forecasts. Secondly, forecasts will be made one step (abbreviated 1S) ahead so that the observed data are brought in one at a time. For example the value at time 134 is

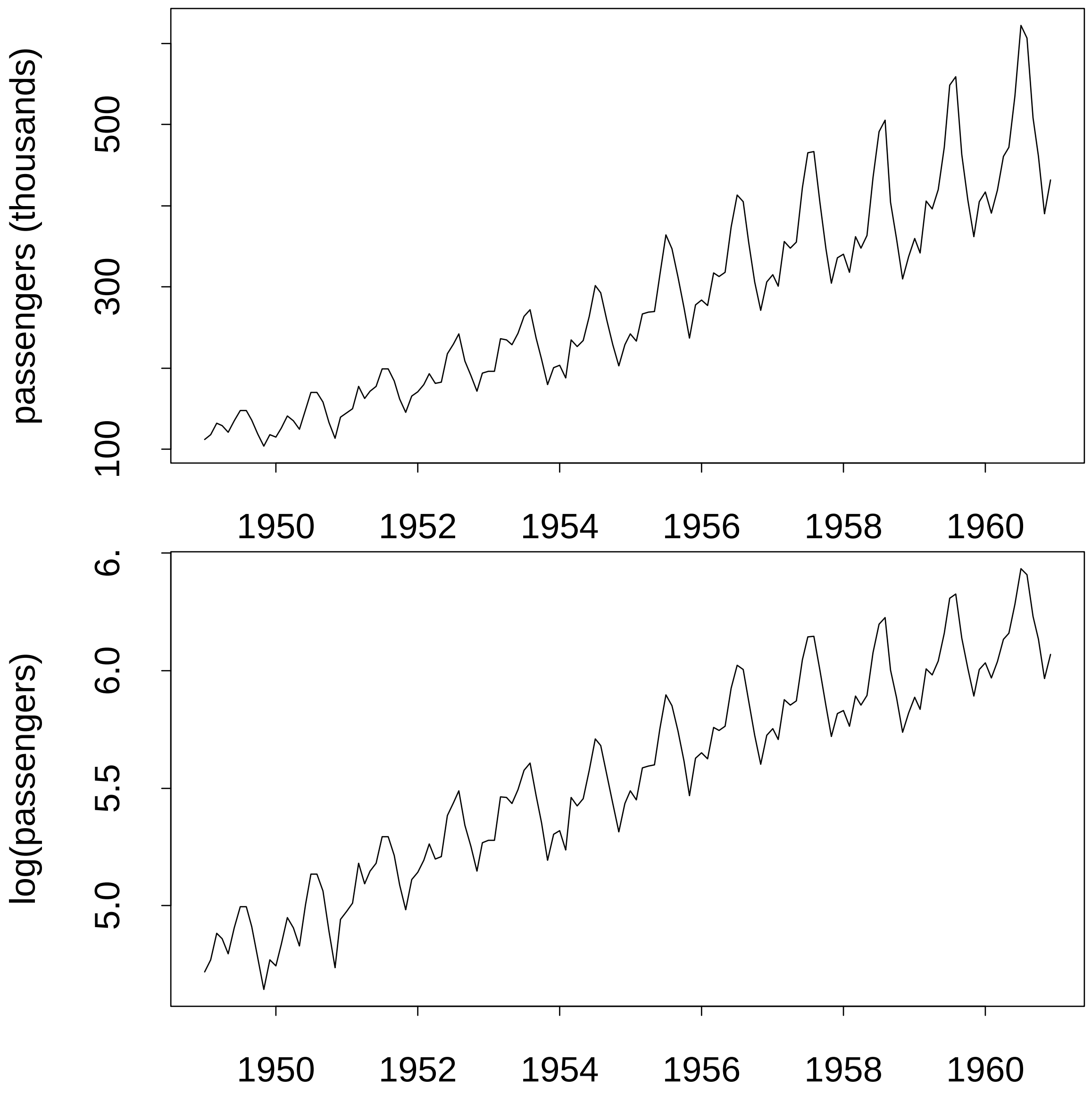

Figure 1: The airline data. Monthly totals of international airline passengers from January 1949 to December 1960. Upper graph - The raw data; Lower graph - the logarithms.

forecast using the observed value at time 133. The model is *not* refitted at each step when computing the one-step forecasts.

For each model fitted, using data up to time T, we computed the following statistics:

1. $S =$ the sum of squared residuals up to time T (the residuals are the within-sample one-step-ahead forecast errors).

2. $\hat{\sigma} = \sqrt{S/(n-p)}$ = estimate of residual standard deviation, where n denotes the number of *effective* observations used in fitting the model and p denotes the number of parameters fitted in the model. Thus when fitting the airline model to the airline data with $T = 132$, the value of n is (132-13) = 119 (since 13 observations are 'lost' by differencing), and the value of p is taken to be 2 (since there are non-seasonal and seasonal moving average parameters).

3. AIC = Akaike's information criterion $= n \ln(S/n) + 2p$

4. BIC = Bayesian information criterion $= n \ln(S/n) + p + p \ln n$

5. SS_{MS} = sum of squares of multi-step-ahead forecasts made from time $T+1$ to the end of the series. These are the out-of-sample (genuine ex-ante) forecasts.

6. SS_{1S} = sum of squares of one-step ahead (out-of-sample) forecasts.

For the airline model fitted to the airline data with $T = 132$, the MINITAB package (Release 9.1) gave the following values (after backtransforming all forecasts from the model for the logged data into the original units): (i) $S = 10789$; (ii) $\hat{\sigma} = 9.522$; (iii) AIC = 540.35; (iv) BIC = 547.91; (v) $SS_{MS} = 3910$; (vi) $SS_{1S} = 4328$. Note that the 1S forecasts happen to have slightly worse accuracy than the MS forecasts as will happen occasionally. Note that, if the airline model is fitted to the *raw* data, rather than to the logs, then the fit is about 20% worse ($S = 12920$) while the accuracy of forecasts suffers even more (e.g. $\mathrm{SS}_{MS} = 5230$ is 34% worse).

An alternative way to compute forecasts for the airline data, without taking logs, is to use the multiplicative version of Holt-Winters exponential smoothing. The Holt-Winters method typically gives forecasts which have comparable accuracy to those obtained from the Box-Jenkins approach, especially for series whose variation is dominated by trend and seasonal variation (see for example Chatfield and Yar (1988)), and we verified this for the airline data. It is worth noting that the multiplicative version of Holt-Winters is inherently non-linear in that the formula for a point forecast is not a linear function of past observations. Likewise the linear ARIMA model fitted to the logged data implies a non-linear model for the original data. This suggests that it should be worth trying an NN model applied to the original (as opposed to the logged) data to see if the non-linear flexibility of NN models can cope with multiplicative seasonality.

3 Neural networks

The following brief account of NNs, and how to fit them, is intended to make this paper as self-contained as possible. However the reader may find it helpful to read Ripley

(1993), Gorr et al. (1994, Section 2) and/or Chatfield (1996, Section 11.4). In addition it might also be helpful to read an introduction from a computer science perspective such as Hertz, Krogh and Palmer (1991) and Gershenfeld and Weigend (1994).

This paper restricts attention to one (popular) form of (artificial) NN called the feedforward NN with one hidden layer. In time-series forecasting, we wish to predict future observations using some function of past observations. One key point about NNs is that this function *need not be linear*, so that a NN can be thought of as a sort of non-linear regression model.

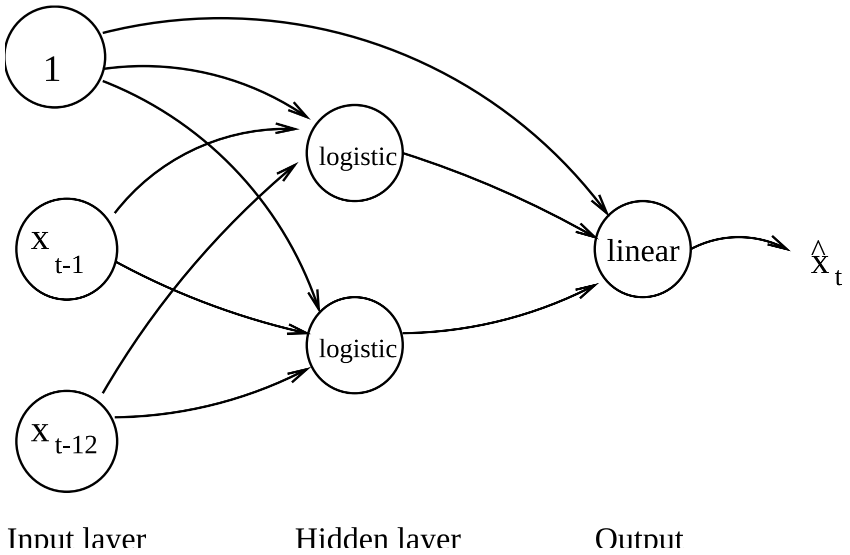

Figure 2: The architecture of a typical neural network for time-series forecasting with one hidden layer of 2 neurons. The output (the forecast) depends on the lagged values at times $(t-1)$ and $(t-12)$

Fig. 2 depicts a typical *architecture* as applied to time-series forecasting with monthly data. The value at time t is to be forecasted using the values at lags one and twelve. The latter are regarded as *inputs* while the forecast is the *output*. The illustrated example includes one hidden layer of two *neurons* (often called *nodes* or *processing units* or just *units*). In addition there is also a constant input term which for convenience may be taken as unity. Each input is connected to both the (hidden) neurons, and both neurons are connected to the output. There is also a direct connection from the constant input to the output. The "strength" of each connection is measured by a weight. A numerical value is calculated for each neuron in two stages. First a linear function of the inputs is found, say $\sum_{i=1}^{3} w_{ij}\, y_i$ where w_{ij} denotes the weight of the connection between input y_i and the jth neuron. The values of the inputs in our example are $y_1 =$ unity, $y_2 = x_{t-1}$ and $y_3 = x_{t-12}$. The linear sum, say ν_j, is then transformed by applying a function called an *activation function*, which is typically non-linear. A commonly used function is the *logistic function*, $z_j = 1/(1+e^{-\nu_j})$, which gives values in the range (0,1). In our example this leads to values z_1 and z_2 for the two neurons. A similar operation could then be applied to the values of z_1, z_2 and the constant input in order to get the predicted output. However note that the logistic function should generally not be used at the output stage in time-series forecasting because it constrains the forecast to lie in the interval (0,1). Instead a linear function of the neuron values is typically used, as in our case study. This means that the activation function at the output stage is just the identity function.

The introduction of a constant input unit connected to every neuron in the hidden layer and also to the output avoids the necessity of separately introducing what computer scientists call a *bias*, and what statisticians would call an intercept term, for each unit. Essentially the biases just become part of the set of weights (the parameters).

For a NN model with one hidden level, the general prediction equation for computing a forecast of x_t (the output) using selected past observations, $x_{t-j_1}, \ldots, x_{t-j_k}$, as the inputs, may be written (rather messily) in the form:

$$\hat{x}_t = \phi_o\left(w_{co} + \sum_h w_{ho}\,\phi_h\Big(w_{ch} + \sum_i w_{ih}\,x_{t-j_i}\Big)\right) \qquad (1)$$

where $\{w_{ch}\}$ denote the weights for the connections between the constant input and the hidden neurons and w_{co} denotes the weight of the direct connection between the constant input and the output. The weights $\{w_{ih}\}$ and $\{w_{ho}\}$ denote the weights for the other connections between the inputs and the hidden neurons and between the neurons and the output respectively. One minor point to note is that the labels on the hidden neurons can be permuted without changing the model. The two functions ϕ_h and ϕ_o denote the the activation functions used at the hidden layer and at the output respectively. Throughout this case study ϕ_h is taken to be the logistic function whereas ϕ_o is taken to be the identity function in order to ensure, as noted earlier, that the forecasts are not restricted to the range (0,1). We use the notation $NN(j_1, \ldots, j_k; h)$ to denote the NN with lags $j_1, \ldots, j_k$ and h neurons (or units) in the one hidden layer. Thus Fig. 2 represents a $NN(1, 12; 2)$ model.

The weights to be used in the NN model are estimated from the data by minimizing the sum of squares of the within-sample one-step-ahead forecast errors, namely $S = \sum_t(\hat{x}_t - x_t)^2$, over the first part of the time series, called the *training set* in NN jargon. In our case study this is usually the first 11 years data. The minimization is no easy task as the objective function often has several local minima and the number of weights may be large. Various numerical algorithms have been proposed but we used some software written in the S-PLUS language by Brian Ripley (See Venables and Ripley, 1994). The training algorithm for selecting the weights may take several hundred iterations to converge, but may still converge to a local minimum. The starting values chosen for the weights can be crucial and it is advisable to try several different sets of starting values to see if consistent results are obtained. Note that other packages carry out the numerical fitting in different ways. For example a technique called *simulated annealing* can be used to try to avoid local minima but this requires the analyst to set numerical parameters with names like 'the cooling rate', and even then there is no guarantee that convergence to a global minimum will occur.

The last part of the time series, called the *test set*, is kept in reserve so that genuine out-of-sample (ex-ante) forecasts can be made and compared with the actual observations.

Equation (1) effectively gives a *one-step-ahead forecast* as it uses the actual observed values of all lagged variables as inputs. If *multi-step-ahead forecasts* are required, then it is possible to proceed in one of two ways. Firstly, one could construct a new architecture with several outputs, giving $\hat{x}_t, \hat{x}_{t+1}, \hat{x}_{t+2}, \ldots$, where each output would have separate weights for each connection to the neurons. Secondly, one could 'feed back' the one-step-ahead forecast to replace the lag-one value as one of the input variables and the same architecture could then be used to construct the two-step-ahead forecast, and so on. We adopted the latter iterative approach, as did Hill et al. (1996), because of its numerical simplicity and because it requires fewer weights to be estimated.

Even so, the number of parameters in a NN model is typically much larger than in traditional time-series models, and for a single-layer NN model is given by $p = (n_i + 2)n_h + 1$ where n_i denotes the number of input variables (excluding the constant) and n_h denotes the number of hidden neurons. For example the architecture in Fig. 2 (where n_i and n_h are both two) contains 9 connections and hence has 9 parameters (weights). Because of this large number, there is a real danger that the algorithm may "overtrain" the data and produce a spuriously good fit which does not lead to better forecasts. Some recent research (Ripley, 1995) has focussed on the need to penalize the fitting of extra parameters rather than just optimize goodness-of-fit, and the use of something like Akaike's information criterion (AIC) is needed to prevent the fitting of spurious parameters.

NN modelling is non-parametric in character and it has been suggested that the whole process can be completely automated on a computer "so that people with little knowledge of either forecasting or neural nets can prepare reasonable forecasts in a short space of time" (Hoptroff, 1993). This *black-box* character can be seen as an advantage or disadvantage. Certainly black boxes can sometimes give silly results and NN models obtained in this way are no exception. Thus Gershenfeld and Weigend (1994, p. 7) say that "there was a general failure of simplistic 'black-box' approaches – in all successful entries (in the Santa Fe competition), exploratory data analysis preceded the algorithm application". The results from our case study also demonstrate that a good NN model for time-series data must be selected by combining general traditional modelling skills allied with knowledge of time-series analysis and the particular problems involved in fitting NN models. Our case study will focus on: (i) the choice of input variables; (ii) the choice of the number of neurons in the hidden layer; (iii) the numerical procedure for estimating the weights including the choice of starting values; (iv) the criterion for selecting the 'best' model.

4 Fitting NN models to the airline data

The Appendix gives information on the software used to fit the NN models. We immediately ran into two practical problems. The untransformed airline data lies in the range 104 to 622. The default choice of the activation function at the output stage is the logistic function, but this constrains the output (the forecasts) to be in the range (0,1). Failure to specify the identity activation function gave ridiculous results. Furthermore the starting values used in the algorithm are out of scale with the input values so that the fitting

S	SS_{MS}	SS_{1S}
2.305	0.351	0.344
2.490	0.332	0.328
2.492	0.350	0.347
2.409	0.346	0.310
2.377	0.343	0.308
2.316	0.378	0.376
2.510	0.439	0.403

Table 1: Fit and forecast accuracy from seven local minima for the $NN(1,12;2)$ model

algorithm failed to converge in a sensible way. Thus we found it necessary to rescale the data, and dividing by 100 was found to work satisfactorily. All subsequent numbers, including forecasts, refer to these scaled data. We could have temporarily overcome both the above problems by dividing by 1000, rather than 100, since this would have ensured that all short-term forecast were in the range (0,1). However the series will clearly exceed the value 1000 within a year or two, and it is safer to use the identity activation function at the output stage in time-series forecasting. Both the above problems might have been expected from a careful reading of the manual and may not be present in other packages, but the need for attention to such details is clear.

Another serious problem arises because consecutive restarts of the fitting algorithm, with different random starting points for the weights, typically finds several different local minima (or even saddle points). For example, seven distinct local minima were found for the $NN(1,12;2)$ model and the resulting fit and forecast accuracy are given in Table 1. We checked that the Hessian is positive definite for all seven minima (though this can be difficult to do because the minima tend to be flat) to ensure that they do all represent local minima. As the algorithm often converges to a saddle point, it really is important to check the Hessian. Even though we restarted the algorithm many times from different starting points, there is no guarantee that the first model in Table 1 (which has the smallest S-value) gives the global minimum. All the models discussed below are the result of refitting the model at least 50 times from different random starting points and taking the best of the resulting minima.

It is unfortunate that the fitted weights are not stable across different minima. Table 2 shows the fitted weights for the second and seventh minima in Table 1 and large differences are apparent. In particular, note that the second neuron in the first model has much more weight to the output than the first neuron, whereas the two neurons in the second model have near equal weights. This instability suggests that it is generally unwise to try to interpret individual weights. Even at the global minimum, it is possible that the weights of the corresponding model could be changed substantially without changing the fit or the predictions very much (as in near multicollinear regression).

We now compared various NN models having different input (lagged) variables and different architectures. All models had one hidden layer and were fitted using the first 11 years data. They were then used to predict the last 12 observations. When several models are compared, having different numbers of parameters, the residual mean square

w_{c1}	w_{11}	w_{21}	w_{c2}	w_{12}	w_{22}	w_{co}	w_{1o}	w_{2o}
-27.61	4.19	4.62	-0.86	0.05	0.23	-4.62	-0.14	16.10
-4.71	0.01	0.74	-0.93	0.20	0.53	-1.40	6.73	5.32

Table 2: Comparison of weights from two different local minima. The notation is defined in Section 3.

lags	no. of hidden neurons	no. of pars.	S	$\hat{\sigma}$	AIC	BIC	Predictions SS_{MS}	SS_{1S}
1,2,3,4	2	13	7.738	0.245	-379.4	-328.1	58.52	1.027
1 – 13	2	31	0.726	0.091	-544.8	-427.6	1.078	0.709
1 – 13	4	61	0.264	0.067	-605.1	-374.6	4.116	1.122
1,12	2	9	2.305	0.144	-456.3	-422.2	0.351	0.344
1,12	4	17	2.164	0.145	-447.7	-383.5	0.376	0.443
1,12	10	41	1.774	0.150	-423.7	-268.4	0.508	0.592
1,2,12	2	11	2.173	0.141	-459.4	-417.7	0.339	0.291
1,2,12	4	21	1.914	0.139	-454.6	-375.1	6.820	1.032
1,2,12,13	2	13	0.993	0.097	-543.5	-494.4	0.374	0.519
1,2,12,13	4	25	0.814	0.093	-543.1	-448.7	0.339	0.517
1,12,13	1	6	1.180	0.102	-537.1	-514.4	0.334	0.504
1,12,13	2	11	1.031	0.098	-543.1	-501.4	0.329	0.503
1,12,13	4	21	0.845	0.093	-546.8	-467.4	0.538	0.621
BJ		2	1.079	0.095	-555.7	-546.1	0.391	0.432

Table 3: Comparison of various NN models together with the corresponding values for the Box-Jenkins airline model

over the fit period will not provide a fair comparison but will be biased towards models with many parameters. Model selection criteria such as the AIC and BIC, defined in Section 2, provide a fairer comparison (see for example Priestley, 1981). For NN models, the number of parameters, p, is the number of weights, while n, the number of effective observations, depends on the maximum lag. The results for a range of models are shown in Table 3. The symbols are defined in Section 2. Table 3 also includes the corresponding values for the Box-Jenkins airline model, as already fitted in Section 2, but note that the numbers have been suitably scaled to make them comparable to the NN values fitted to the data divided by 100. In particular, note that the scaled AIC and BIC values have been obtained by subtracting 119 ln(10000).

The $NN(1-4;2)$ model is the sort of model which might be tried by someone without any training in time series – clearly its fit is poor and its predictive performance, particularly in the long term, is awful. Someone with slightly more experience, realising this is annual data, might use all lags up to lag 12, or indeed up to 13 or even more if they were feeling extravagant. These models, with lots of parameters, achieve small S-values

	Lag												
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0.26	-0.06	0.02	-0.01	0.05	-0.05	0.06	0.01	0.06	-0.09	-0.03	0.21	-0.20
2	2.70	-3.47	2.53	-0.69	-1.39	-0.36	2.52	4.27	2.16	-4.25	-3.05	2.43	-3.31

Table 4: Weights in the $NN(1-13;2)$ model for the connections from the 13 lagged values to the two hidden neurons

which might lead the naive to suppose that they had found a good model, especially in view of the AIC values. However the BIC values tell a different story and the predictive performance is poor. With some knowledge of time series, it might seem reasonable to include lag 12 without all the intervening lags as inputs and these models do give reasonable predictive performance provided not too many hidden neurons are used. It is arguable that only prior knowledge of the Box-Jenkins analysis would suggest the use of the lag 1,2,12,13 or lag 1,12,13 models which generally lead to better forecasts and BIC values.

If the model is chosen on the basis of minimizing the AIC or $\hat{\sigma}$ (where the latter is comparable to adjusted R^2), then the $NN(1-13;4)$ model will be selected leading to poor predictions. Of course, experienced statisticians would guess intuitively that the use of so many inputs cannot lead to good results and so would likely never fit such a model in the first place. In contrast the BIC criterion picks the $NN(1,12,13;1)$ model which is quite plausible and does give sensible results. Clearly, in this context, the use of AIC does not do enough to penalize extra parameters. Even so, notice that the best NN model does not give such a good BIC value as the Box-Jenkins model though its forecasts are comparable.

A natural question to ask is whether the better models found above (i.e. ones that include lags 1, 12 and maybe 13, but excluding intervening lags) could have been discovered without the use of Box-Jenkins? In order to answer this question, consider Table 4 which shows the weights of the connections between the 13 lagged values and the two neurons in the $NN(1-13;2)$ model. Notice that the weights connecting the lagged values to the first hidden neuron show small values for lags $2-11$. This does suggest dropping the intermediate lags. Admittedly for connections to the second hidden neuron, lags $2-11$ have larger weights, but the weights from the two hidden neurons to the output unit are 16.04 and -0.90 respectively, so that second hidden neuron has a much lower effect on the output. Thus, in this case, an examination of the weights does turn out to be useful. However, in view of the instability in estimating the weights mentioned earlier, this may be the exception rather than the rule and our experience in Chatfield and Faraway (1996) suggests the former.

Why does having too many hidden neurons result in poor performance? Consider the $NN(1,2,12)$ model with either 2 or 4 hidden neurons. The prediction surface for this varies over the three lags so it is not possible to plot it directly but we can view it along specified directions. In Fig. 3 the predicted one-step response is plotted along the line $x_{t-1} = x_{t-2} = 0.87x_{t-12} + 0.07$, chosen because the data is dense in this direction. Notice how the $NN(1,2,12;4)$ model contorts to achieve a better fit to the middle range of the data but that this causes the prediction to be pulled down for higher values.

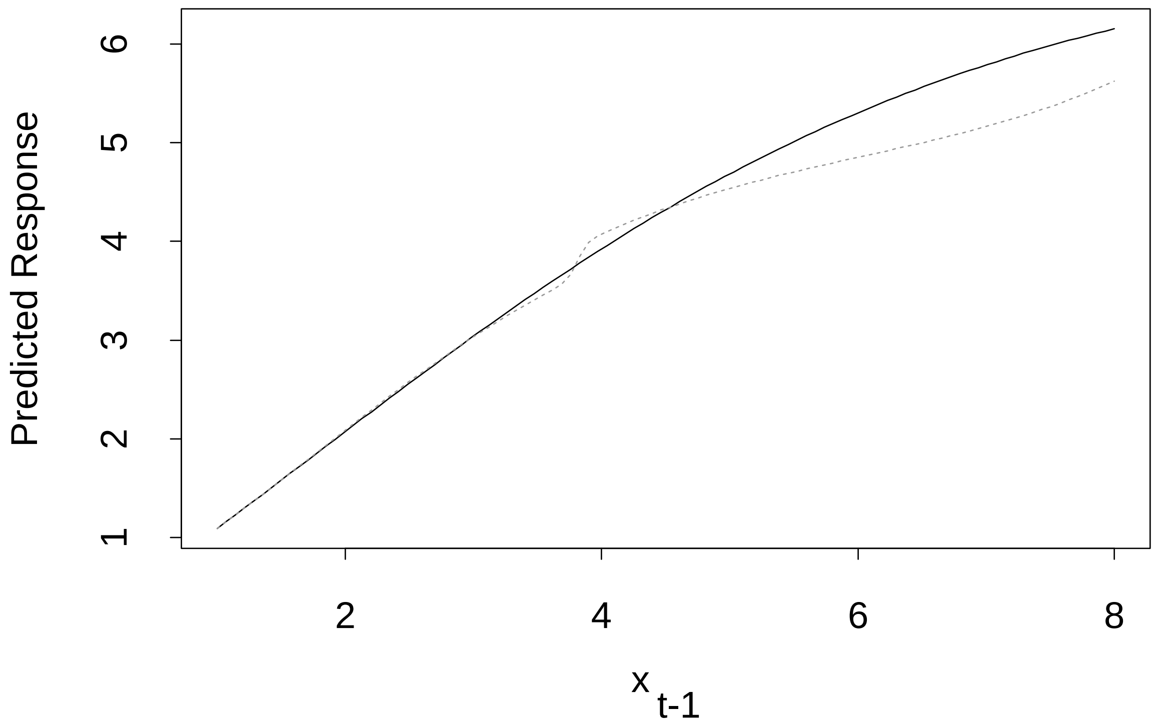

Figure 3: Predicted response for $NN(1, 2, 12; 2)$ (solid line) and $NN(1, 2, 12; 4)$ (dotted line) models along the direction $x_{t-1} = x_{t-2} = 0.87x_{t-12} + 0.07$

Some other NN models were considered that were not the result of a pure neural network approach since they involve some intelligent pre-examination of the data, as a sensible statistician might naturally carry out. Fig. 1 reveals that the variation increases with the level which suggests transforming the data in order to stabilize the variance. Here the log transformation seems most appropriate, and three plausible NN models were explored:

1. A $NN(1, 12, 13; 2)$ model (for the logs).
2. Remove the linear trend (from the logs) and then remove the seasonal trend by subtracting the monthly averages. Then apply a $NN(1-4; 2)$ model.
3. Apply $\nabla\nabla_{12}$ differencing (to the logs) and then use a $NN(1-4; 2)$ model.

The results are shown in Table 5. Fitted values and predictions were transformed back to the original scale so that the entries in the table are comparable to those appearing in Table 3. Note the similarity in the fits between the $NN(1, 12, 13; 2)$ models for the untransformed and the log-transformed data (in contrast to what was found for the Box-Jenkins models). A possible explanation is the local linearity of the log transformation —

	no. of					Predictions	
Model	pars	S	$\hat{\sigma}$	AIC	BIC	SS_{MS}	SS_{1S}
1	11	1.043	0.098	-541.7	-500.1	0.438	0.633
2	13	0.959	0.091	-600.4	-550.3	0.659	0.565
3	13	1.142	0.105	-504.4	-455.7	4.35	0.674

Table 5: Results for three NN models fitted to the logged data

the ratio of the maximum to minimum values for the whole series is about 5 but since the current fit depends only on the values at lags 12 and lag 13 then it is the ratio $x_t/x_{t-12or13}$ which matters. The maximum value of this ratio is 1.4 so the curvature introduced by the log transform is not a big factor.

An interesting question is whether taking logs should count as an extra parameter. In model 2, the fit criteria look very good but one might argue that the detrending and deseasonalizing represent $(2 + 12) = 14$ more parameters so that the true number of parameters is really 27 and not 13. Recalculating AIC, BIC and $\hat{\sigma}$ on this basis gives -572.4, -468.4 and 0.097 respectively, which makes the model less persuasive.

Model 3 (a $NN(1-4; 2)$ model for $\nabla\nabla_{12}$ differences of the logs) gives inferior fit criteria and much worse multi-step forecasts compared with model 2 (the detrended, deseasonalized alternative). One reason for this is that there are only 115 observations used in the former model but 128 in the latter, but there is a major question as to what should be the true number of parameters in these models. More to the point, model 3 is not only worse than the airline model (see Table 3) but even gives worse forecasts than just using the 'naive' model $\nabla\nabla_{12}\log x_t$. Forecasts from the latter give $SS_{MS} = 2.16$ and $SS_{1S} = 0.683$. It appears that NN models cannot capture moving-average-type structure and that putting more units into a hidden layer makes things worse rather than better.

A final point to notice about all the models considered in this section is that the within-sample (fit) estimate of the error standard deviation (i.e. $\hat{\sigma}$), is typically much less than the prediction error standard deviation, namely $\sqrt{SS_{1S}/12}$. The best values of $\hat{\sigma}$ are around 0.1 , while the best estimate of of the prediction error standard deviation is 0.15 (for the $NN(1, 2, 12; 2)$ model) but is more typically around 0.2 . This is a common phenomenon in time-series forecasting where within-sample fit always appears better than out-of-sample prediction. The over-optimism caused by choosing the best of many fitted models and then behaving as if the selected model were known to be true has been well documented in work on model uncertainty (e.g. Chatfield, 1995).

5 Alternative ways of looking into the 'black box'

One major criticism levelled at NN models is that they provide a "black-box" approach which may produce satisfactory predictions but generally provides little insight into the structure of the data. It can be very hard to interpret a NN model. So let us see if we can look inside the black box. When there are only two inputs, it is possible to plot the prediction surface as in Fig. 4 for the $NN(1, 12; 2)$ model. Notice that the surface is non-linear. However if attention is focussed on the region where most of the data occur (which is around the main diagonal) then the surface is close to linear.

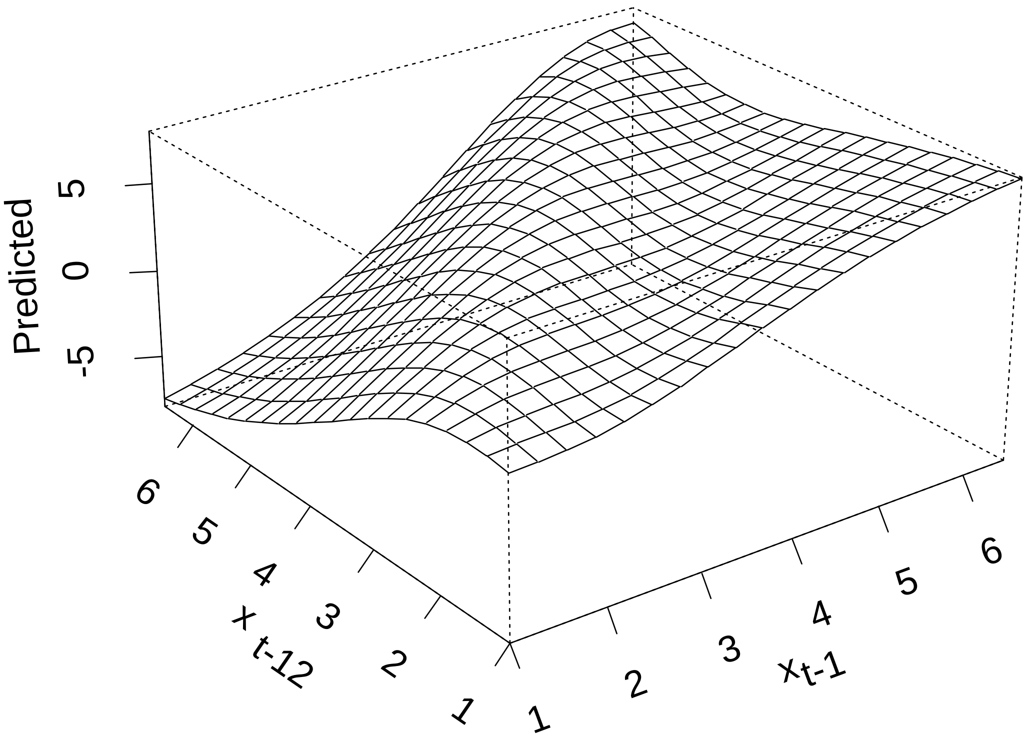

Figure 4: Prediction surface for the $NN(1,12;2)$ model (in scaled units)

When there are more than 2 inputs, it is not possible to plot the prediction surface so easily and we can only view the predictions along specified directions. For the $NN(1,2,12,13;2)$ model, Fig. 5 shows the predicted response along the direction of the first principal component of $\{x_{t-1}, x_{t-2}, x_{t-12}, x_{t-13}\}$ as x_{t-1} varies. Most of the data lies within a distance ± 0.2 in the direction of the second principal component where we see that the prediction surface is close to linear. This happens because the weighted sum of the inputs into the hidden neurons, that have some weight in the output, happen to fall in the central, nearly linear, region of the logistic activation function. Thus these plots (which are not usually considered in NN analysis), demonstrate that the prediction

	no. of					Predictions	
lags	pars.	S	$\hat{\sigma}$	AIC	BIC	SS_{MS}	SS_{1S}
1,12,13	13	1.136	0.104	-527.5	-478.4	0.431	0.553

Table 6: GAM results

surface is approximately linear in the region of interest. Such plots will not necessarily be useful for all data and all NN models but they are valuable here.

We now go on to consider some alternative modern statistical methods which can give similar results more directly. *Generalized Additive Models (GAM)* (Hastie and Tibshirani, 1990) can be used to represent the predicted response in terms of a sum of functions of the chosen inputs. Choosing lags 1, 2, 12 and 13 for example, this may be written:

$$\hat{x}_t = f_1(x_{t-1}) + f_2(x_{t-2}) + f_{12}(x_{t-12}) + f_{13}(x_{t-13})$$

where the the functions f_i are estimated nonparametrically. Using S-Plus software, the GAM was fitted as described in Chambers and Hastie (1992), where the functions f_i were estimated using splines. The estimated functions along with ± 2 standard error pointwise confidence bands are shown in Fig. 6.

The fitted function for x_{t-2} is not significantly different from constant zero so we have the immediate message that this variable may be excluded from the model. In addition, the other fitted functions are all very close to linear. If x_{t-2} is excluded the refitted GAM gives a very similar fit. The GAM can be used to make predictions when splines are used for the fitting since these fits can be extrapolated. Of course, it is dangerous to extrapolate too far but the same problem would arise with a NN model. The results are shown in Table 6 and it can be seen that the results are comparable to the best NN models obtained earlier.

We have tried GAMs on other time series and have found it to be a useful exploratory technique. In particular when applied to the famous sunspots data (e.g. Chatfield, 1996, Fig. 11.1; de Groot and Würtz, 1991), which nearly everyone agrees is *not* linear, the functions were indeed found not to be linear but to be approximately two straight lines with a bend in the middle indicating that a threshold model might be appropriate.

An alternative approach is provided by *Projection Pursuit Regression (PPR)* (see Friedman and Stuetzle, 1981). Given inputs $\mathbf{x} = (x_1, \ldots, x_p)^T$ and output y, then the PPR model is

$$\hat{y} = \bar{y} + \sum_{m=1}^{M_0} \beta_m \phi_m(\boldsymbol{\alpha_m}^T \mathbf{x})$$

where $\boldsymbol{\alpha_m}$ denotes a vector of constants of appropriate length, β_m is a constant and the functions ϕ_m are estimated nonparametrically and are scaled to have mean zero and unit variance. Note that, in comparison with a NN model, the "activation" functions ϕ_m are estimated from the data rather than given some pre-specified form such as the logistic.

Setting $(x_{t-1}, x_{t-2}, x_{t-12}, x_{t-13})$ as the input and x_t as the output, we first determined that the best choice of M_0 is 1 as larger values improve the fit to a negligible extent. The function ϕ_1 turns out to be virtually linear with $\boldsymbol{\alpha_1}$ = (0.504, -0.008, 0.681, -0.531).

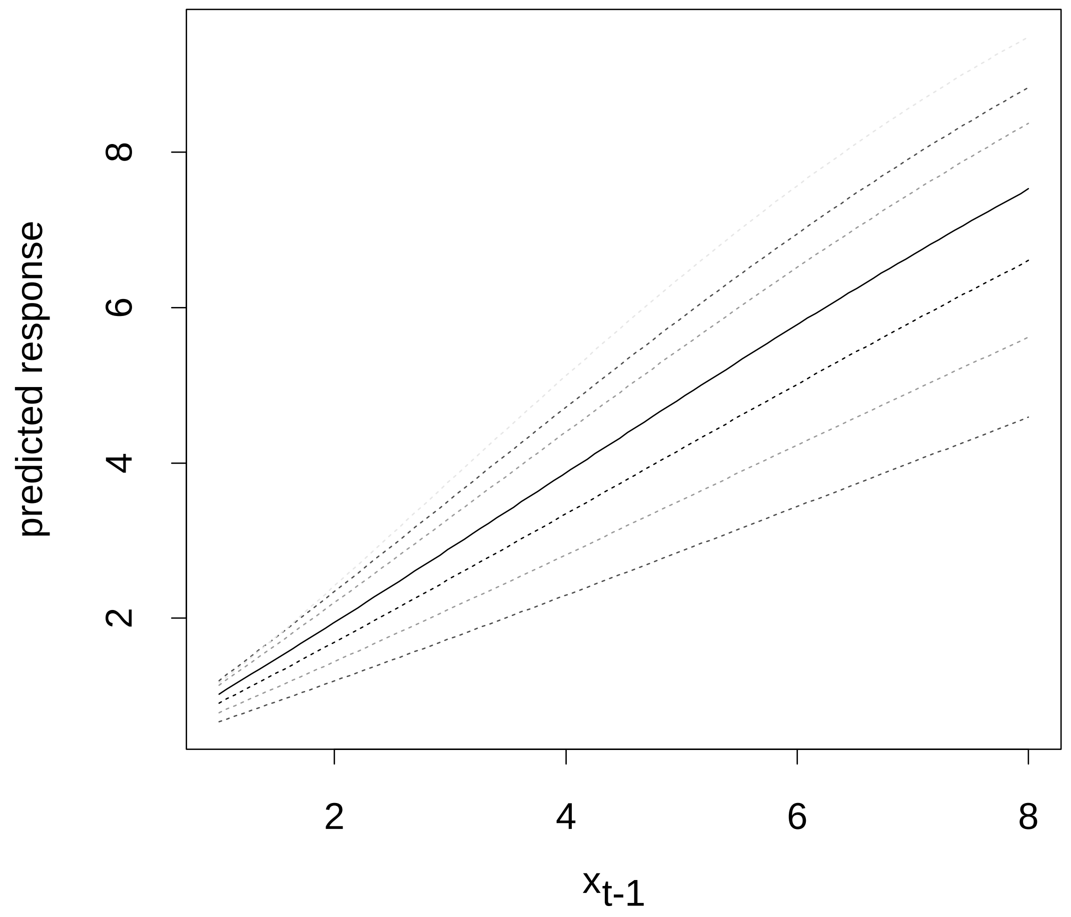

Figure 5: Predicted response (in scaled units) for the $NN(1, 2, 12, 13; 2)$ model along the direction of the first principal component of $\{x_{t-1}, x_{t-2}, x_{t-12}, x_{t-13}\}$ as x_{t-1} varies (solid line). Also shown is the predicted response along the first principal component perturbed by $\pm 0.1, \pm 0.2$ and ± 0.3 in the direction of the second principal component

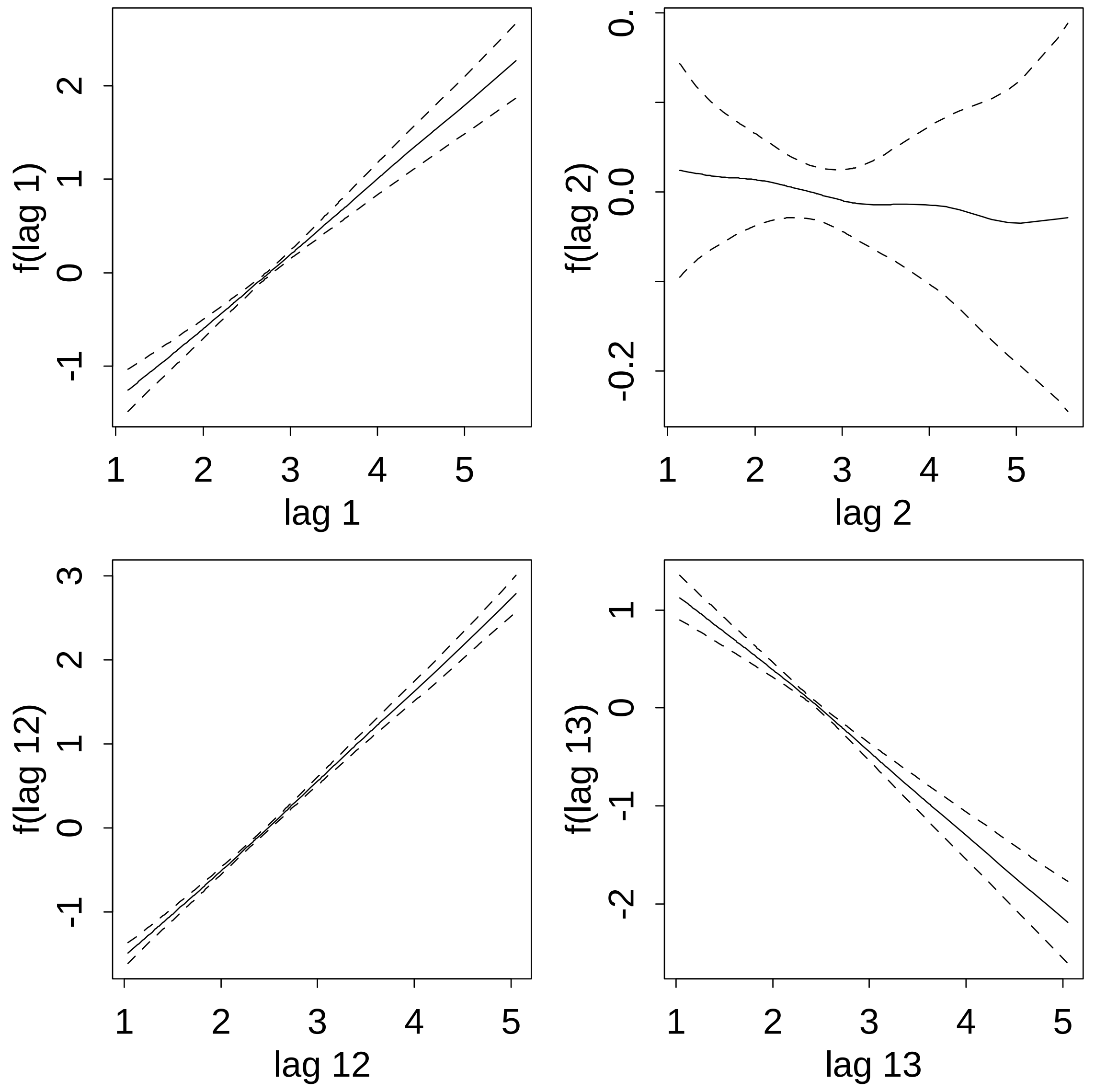

Figure 6: The four fitted additive functions for the terms of a GAM, corresponding to lags 1,2,12 and 13, are shown with solid lines. The ± 2 standard error confidence bands are shown as dotted lines

no. of					Predictions	
pars.	S	$\hat{\sigma}$	AIC	BIC	SS_{MS}	SS_{1S}
4	1.181	0.101	-540.9	-525.8	0.322	0.507

Table 7: Results for the linear regression model

The small coefficient for x_{t-2} implies again that this term is not needed and the PPR model obtained is very similar to the earlier GAM model. It is rather more difficult to make predictions using a PPR model since the "activation function" is estimated using the "Supersmoother" method which does not lend itself readily to extrapolation. This could be modified in principal but would involve rewriting the software. The fitted Sum of Squares (S) for the PPR model is 1.145 which is comparable to the value for the GAM model.

It may seem strange that the GAM and PPR models give "activation" functions which are approximately linear, in contrast to the logistic function for the NN model. In fact the logistic function is close to linear in its midrange. Hence the similarity.

All this suggests that we could use a simple linear regression model with $\{x_{t-1}, x_{t-12}, x_{t-13}\}$ as explanatory variables. The fitted linear regression equation is

$$\hat{x}_t = 0.0322 + 0.7824x_{t-1} + 1.0720x_{t-12} - 0.8394x_{t-13}$$

Note that the coefficients are almost proportional to the corresponding ones obtained by PPR. Predictions are easily made using a linear regression model and the results are shown in Table 7. Comparing with Table 3, we see the forecasts are excellent and the model gives the smallest value for the BIC yet seen!

6 Results for a different time period

It is interesting to see if there is any qualitative change in the results if we try to predict from a different part of the seasonal cycle over a different time period. We extended the test set to the last 18 monthly observations (so that there were six less observations in the training set) and tried various NN models. The results are shown in Table 8.

The NN model with the best BIC is $NN(1, 12, 13; 1)$ as it was in Table 3 and gives reasonable 1-step forecasts considering that the measure of prediction is now a sum of squares over 18 observations (so that the above figures should be discounted by one third to compare them with the 1-step forecasts of the last 12 observations). The multistep forecasts are much worse than before, partly because they involve going over one year ahead. The $NN(1, 12; 2)$ model actually gives better MS forecasts but it is worrying that none of the NN models gives such a good BIC or such good forecasts as the airline model. In other respects the results are qualitatively similar to those in Table 3. NN models with higher numbers of parameters tend to give better fits but worse forecasts. The use of AIC is not enough to penalize additional parameters and BIC is recommended.

lags	no. of hidden neurons	no. of pars.	S	$\hat{\sigma}$	AIC	BIC	Predictions SS_{MS}	SS_{1S}
1,12	1	5	2.383	0.147	-430.9	-412.2	1.729	0.958
1,12	2	9	2.149	0.143	-434.7	-401.1	1.331	0.992
1,12	4	17	1.815	0.137	-438.0	-374.4	3.275	3.296
1,12,13	1	6	1.090	0.101	-512.5	-490.1	2.591	0.664
1,12,13	2	11	0.903	0.094	-523.7	-482.7	2.050	0.600
1,12,13	4	21	0.688	0.086	-534.5	-456.2	4.929	2.016
1,2,12,13	1	7	1.082	0.101	-511.3	-485.2	2.980	0.671
1,2,12,13	2	13	0.906	0.095	-519.4	-471.0	18.10	1.484
1,2,12,13	4	25	0.644	0.086	-533.8	-440.7	5.241	4.183
BJ		2	1.048	0.097	-524.9	-519.5	0.612	0.470

Table 8: Results for NN models fitted to first 126 observations

7 Discussion

The airline data used in this paper was one of three series used in a case study paper by Tang et al. (1991). They found that NN models gave comparable forecasts to those from the Box-Jenkins approach, but unfortunately give few details on how the NN models were actually fitted and do not for example say what activation function was used or if the data were transformed (logged). Judging by the graphs and tables, we guess that the authors have scaled the data by dividing by 1000 so that all forecasts lie in the range (0,1). This means that they could have used the logistic function at the output stage. If, as stated in their paper, the authors really did set the number of hidden neurons to equal the number of inputs (which could be as high as 24), the number of weights will have sometimes exceeded the number of observations. The paper is written from the point of view of computer scientists and, for us, raised more questions than it answered.

We did carry out a second analysis using the sales data from Chatfield and Prothero (1973) and the results are reported in detail in Chatfield and Faraway (1996). Broadly speaking, the results regarding the difficulties in fitting NN models were qualitatively similar.

We do not claim that analysing one or two time series can tell us much, if anything, about the general comparative forecasting ability of NN and Box-Jenkins models. The airline data is for example quite different to the time series used in the Santa Fe competition, which were much longer (several thousand observations) and in several cases more clearly non-linear. There NN models seem generally more appropriate, though, for the one financial time series (exchange rate data), there was a "crucial difference between training and test set performance" (Gershenfeld and Weigend, 1994, p. 40). Out-of-sample predictions from the fitted NN model were no better than those from a random walk! All we can say from our experience is that NN models are not clearly better than alternatives, and that, in Table 8, they actually appear inferior to Box-Jenkins.

We realise that NN models can be elaborated in various ways, for example by allowing skip-layer connections, extra hidden layers, feedback connections, and weight decay, but the price of extra versatility is to increase the potential for going astray. With such short series, we think there are already more than enough choices to be made.

In conclusion we suggest that our case study does allow us to make the following general points:

1. It can be dangerous to adopt a a 'black-box' approach to NN modelling. Great care is need to choose (i) an appropriate set of input variables; (ii) an appropriate architecture; (iii) appropriate activation functions (which need not be the same at the hidden layers as at the output unit(s); (iv) an appropriate numerical procedure for fitting a NN model. In particular it is necessary to choose sensible starting values for the weights for the training algorithm and it may be necessary to scale the data first.

2. Adding extra hidden units increases the number of parameters in a NN model. This may lead to an improvement in fit but may well lead to a deterioration in out-of-sample predictions. Our results suggest that, when comparing models, the use of AIC is not enough to penalize the addition of extra parameters. Rather the use of BIC is indicated.

3. There is plenty of scope for going for going badly wrong with NN modelling (as there is for many other sophisticated statistical techniques).

4. The use of Generalized Additive Models and Projection Pursuit Regression can be useful in exploring a set of data in order to decide what model is appropriate.

Acknowledgement

This work was carried out while the first author was visiting the University of Bath.

References

Box, G. E. P. , G. M. Jenkins, G. M. and Reinsel, G. C. (1994) *Time Series Analysis, Forecasting and Control* (3rd ed.). Englewood Cliffs, N.J.: Prentice-Hall.

Brown, R. G. (1962) *Smoothing, Forecasting and Prediction of Discrete Time Series.* Englewood Cliffs, N.J.: Prentice-Hall.

Chambers, J. and Hastie, T. (1992) *Statistical Models in S.* Pacific Grove, CA: Wadsworth & Brooks/Cole.

Chatfield, C. (1993) Neural networks: Forecasting breakthrough or passing fad? *Int. J. of Forecasting*, **9**, 1–3.

Chatfield, C. (1995) Model uncertainty, data mining and statistical inference (with discussion). *J. R. Statist. Soc. A*, **158**, 419–466.

Chatfield, C. (1996) *Time Series Analysis* (5th edn.) London: Chapman & Hall.

Chatfield, C. and Faraway, J. (1996) Forecasting sales data with neural nets: A case study. *Recherche et Applications en Marketing*, (to appear).

Chatfield, C. and Prothero, D. L. (1973) Box-Jenkins seasonal forecasting: problems in a case study (with discussion). *J. R. Statist. Soc. A*, **36**, 295–336.

Chatfield, C. and Yar, M. (1988) Holt-Winters forecasting: some practical issues. *The Statistician*, **37**, 129–140.

Cheng, B. and Titterington, M. (1994) Neural networks: a review from a statistical perspective (with discussion). *Statistical Science*, **9**, 2–54.

de Groot, C. and Würtz, D. (1991) Analysis of univariate time series with connectionist nets: A case study of two classical examples. *Neurocomputing*, **3**, 177–192.

Friedman, J. and Stuetzle, W. (1981) Projection pursuit regression. *J. Am. Statist. Ass.*, **76**, 817–823.

Gershenfeld, N. A. and Weigend, A. S. (1994) The future of time series: learning and understanding. In *Time Series Prediction* (eds A.S. Weigend and N.A. Gershenfeld), pp. 1–70. Reading, MA: Addison-Wesley,

Gorr, W. L., Nagin, D. and Szczypula, J. (1994) Comparative study of artificial neural network and statistical models for predicting student grade point averages. *Int. J. of Forecasting*, **10**, 17–34.

Harvey, A. (1993) *Time Series Models* (2nd ed.). Hemel Hempstead, U.K.: Harvester Wheatsheaf.

Hastie, T. and Tibshirani, R. (1990) *Generalized Additive Models.* London: Chapman Hall.

Hertz, J., Krogh, A. and Palmer, R. (1991) *Introduction to the Theory of Neural Computation.* Redwood City, CA: Addison Wesley.

Hill, T., Marquez, L., O'Connor, M. and Remus, W. (1994) Artificial neural network models for forecasting and decision making. *Int. J. of Forecasting*, **10**, 5–15.

Hill, T., O'Connor, M. and Remus, W. (1996) Neural network models for time series forecasts. *Man. Science*, (to appear).

Hoptroff, R. (1993) The principles and practice of time series forecasting and business modelling using neural nets. *Neural Computing and Applications*, **1**, 59–66.

Priestley, M. B. (1981) *Spectral Analysis and Time Series.* London: Academic Press.

Ripley, B. (1993) Statistical aspects of neural networks. In *Chaos and Networks - Statistical and Probabilistic Aspects* (eds O. Barndorff-Nielsen, J. Jensen, and W. Kendall), pp. 40–123. London: Chapman and Hall.

Ripley, B. D. (1995) Statistical ideas for selecting network architectures. In *Neural Networks: Artificial Intelligence and Industrial Applications* (eds B. Kappen and S. Gielen), pp. 183–190. Berlin: Springer.

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press.

Tang, Z., de Almeida, C. and Fishwick, P. A. (1991) Time series forecasting using neural networks versus Box-Jenkins methodology. *Simulation*, **57**, 303–310.

Venables, W. N. and Ripley, B. D. (1994) *Modern Applied Statistics with* S-PLUS. New York: Springer-Verlag.

Appendix

Software for fitting neural nets to time series data may be found on WWW at *http://www.stat.lsa.umich.edu/~faraway/* together with full details on installation and use. We used various S-PLUS functions from Venables and Ripley (1994) (including *nnet* and *nnet.Hess*) in conjunction with some functions written by the first author. Some knowledge of S-PLUS is required for the software to be useful.

Here is an example of its use, where we read in the airline data and rescale it by dividing by 100 in the first command line. A NN(1,12;2) is fit to the first 132 observations in the second line. We restart the algorithm 50 times from different random starting weights and take the best of the models found. A summary of the fit is requested in the third line.

```
> air <- scan("air.data")/100
> g <- nnts(air[1:132],c(1,12),2,retry=50)
> summary(g)
a 2-2-1 network with 9 weights

Unit 0 is constant one input
Input units:  Lag 1=1, Lag 12=2,
Hidden units are  3 4
Output unit is 5

   0->3   1->3   2->3   0->4   1->4   2->4   0->5   3->5   4->5
  -0.10   1.26  -1.31  -0.10   0.66  -0.55  -7.43 -15.60  31.27
Sum of squares is  2.310301
AIC : -456.0137 , BIC : -421.9262 , residual se : 0.1442689
```

We can now predict the next 12 observations.

```
> predict(g,12)
 3.997191 3.803972 4.383948 4.370344 4.652647 5.185713 5.889187 6.055508
 4.930448 4.410727 3.994916 4.466562
```