African Journal of Science and Technology (AJST)
Science and Engineering Series Vol. 5, No. 2, pp 41-49

SEASONAL TIME SERIES FORECASTING: A COMPARATIVE STUDY OF ARIMA AND ANN MODELS

Kihoro, J.M.[1], Otieno, R.O.[1], Wafula, C.[2]
[1]Department of Mathematics & Statistics, Jomo Kenyatta University of
Agriculture & Technology,P.O. Box 62000, Nairobi – KENYA
[2]Department of Mathematics, Kenyatta University, P.O. Box 43844, Nairobi - Kenya

ABSTRACT:- *This paper addresses the concerns of Faraway and Chatfield (1998) who questioned the forecasting ability of Artificial Neural Networks (ANN). In particular the paper compares the performance of Artificial Neural Networks (ANN) and ARIMA models in forecasting of seasonal (monthly) Time series. Using the Airline data which Faraway and Chatfield (1998) used and two other data sets and taking into consideration their suggestions, we show that ANN are not as bad as Faraway and Chatfield put it. A rule of selecting input lags into the input set based on their relevance/ contribution to the model is also proposed.*

KEYWORDS: *Time Series; Seasonal Autoregressive Integrated Moving Average (SARIMA); Artificial Neural Network (ANN); Multilayered Perceptrons (MLP); Time lagged Neural Networks (TLNN); Automatic Relevance Determination (ARD)*

INTRODUCTION

Time series forecasting is a common problem. Many approaches to this problem have been used with Box and Jenkins (1976) developing the integrated autoregressive moving average (ARIMA) methodology for fitting a class of linear time series models. Statisticians in a number of ways have addressed the restriction of linearity in the Box-Jenkins approach and robust versions of various ARIMA models have been developed in addition to nonlinear time series models. More recently, Artificial Neural Networks (ANN) have been studied as an alternative to these nonlinear model-driven approaches. Because of their characteristics, ANN belong to the data-driven approach, where the analysis depends on the available data. In the recent past many statisticians have investigated the properties of neural networks and have found considerable overlap between statistical and neural network modelling, see for example Bishop (1995).

ANN have been used for a wide variety of applications, where statistical methods are traditionally employed. In time series applications they have been used in forecasting future values. Several authors have done comparison studies between statistical methods and ANN (see e.g. Titterington, 1999). In time series context Hill et al. (1996), Kuan and White (1994), among others have investigated the forecasting ability of ANN but Faraway and Chatfield (1998) reviewed their work and questioned their findings. This paper seeks to address the issues raised by those who do not embrace ANN models as an alternative for statistical modeling. In particular we focus on Faraway and Chatfield (1998) work and try to empirically dispel their fears on forecasting ability of ANN.

The statistical approach to forecasting involves the construction of stochastic models to predict the value of an observation x_{t+d} using previous observations. This is often accomplished using linear stochastic difference equation models, with random input. The most important class of such models is the linear autoregressive integrated moving average (ARIMA) model. Before we present our results, we give a brief review of ARIMA and ANN techniques respectively and optimal prediction from the models.

ARIMA MODELLING

Let $x_t : t = 0, 1, 2, ...N$ be an observed monthly time series. If this time series contains seasonality, a seasonal periodic component repeats itself after every $s = 12$

observations and thus we expect x_t to depend on terms such as x_{t-12} and perhaps x_{t-24} as well as terms such as x_{t-1} , x_{t-2} , ... Box and Jenkins (1976) have generalized the ARIMA model to deal with seasonality, thus coming up with a model known as SARIMA (seasonal autoregressive integrated moving average) written as ARIMA (p, d, q)(P, D,Q)s and given by

$$\phi_p(B)\Phi_p(w_t = \theta q(B)\Theta_Q(B^s)e_t \qquad (1)$$

where $\phi_p, \Phi_p, \theta q, \Theta_Q$ are polynomials of order p, P, q, Q respectively, $\nabla^d \nabla_s^d x_t.\nabla^d$ is the simple differencing operator of order d, ∇^D is the seasonal differencing operator of order $D\ B^i x_t = x_{t-i}$ is the backward shift operator and s.is the seasonal period.

The determination of the ARIMA model orders involves matching the patterns in the sample autocorrelation functions (ACF) and sample partial autocorrelation functions (PACF) with the theoretical patterns of the known models to identify the orders. Akaike Information Criterion (AIC), [Akaike (1974)] has also been widely used which can be used for statistical model identification in a wide range of situations and is not restricted to time series. The optimal order of the model is chosen by the number of model parameters, N_p , a function of p and q, which minimizes AIC(N_p). The performance of AIC has been criticized especially when N_p is large and a recent modification of AIC by Hurvich and Tsai (1989) adds a bias adjustment which leads to a criterion denoted by AICc and given by;

$$AICc(N_p) = AIC + \frac{2N_p(N_p+1)}{N-N_p-1} \qquad (2)$$

Burnham and Anderson (1998) insist on the use of AICc irrespective of the size of the data while Faraway and Chatfield (1998) recommends the use of Bayesian Information Criterion (BIC) because it penalises extra parameters just like AICc.The BIC is defined by;

$$BIC(N_p) = N\ell n\hat{\sigma}_e^2 + N_p + N_p \log(N) \qquad (3)$$

Once the form of the model has been specified its parameters are then estimated using maximum likelihood estimation method or any other convenient one. The basic assumption in ARIMA models is that the errors are uncorrelated random variables with mean zero and constant variance. It is therefore expected that the residuals have the characteristics of the white noise. Chatfield (1975), Abraham and Ledolter (1983) have suggested that we ''Just'' look at the first few values of autocorrelations, r_k (for residuals) particularly at lags 1, 2 and the first seasonal lag (if any) and see if they are significantly different from zero (i.e whether they fall outside the limits ±. If only one (or two) values of estimated r_k are significant at lags which have no obvious physical meaning there would be no enough evidence to reject the model. A more comprehensive treatment of linear ARIMA-models may be found in Box and Jenkins (1976). Box-Pierce (1970) came up with a test for serially correlated residuals in ARMA (p,q). The test has been extended to cover seasonal models, Chatfield (1975). It has been shown that if the appropriate model for the differenced series is ARIMA (p, d, q)(P, D,Q)s process then the statistic

$$Q = n(n+2)\sum_{k=1}^{m}(n-k)^{-1}\hat{r}_k^{\,2} \sim \chi^2_{(m-p-q-P-Q)}$$

If the calculated Q statistic exceeds the tabulated $\chi^2_{\alpha,(m-p-q-P-Q)}$, then the adequacy of the fitted model would be questioned. It is however advisable to combine several methods in diagnostic checking. Finally and once the model has been found adequate, prediction / forecasting of future values is done.

ARTIFICIAL NEURAL NETWORKS

An artificial neural network (ANN) is, basically, a non-parametric attempt to model the human brain. ANN acts 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, ANN has found applications in many different fields, especially for forecasting and classification purposes. The topology of ANN has been sketched in Fig. 1.

In figure 1, various inputs to the network are represented by the mathematical symbol, x_i ; each of these n inputs is multiplied by a connection weight w_{ih} depending on the hidden neuron it is connected to. In the simplest case, these products are simply summed, fed through a transfer function to generate a result and then output y.

Even though all artificial neural networks are constructed from this basic building block the fundamentals may vary in these building blocks and there are differences. The processing elements are able to ''learn'' by receiving weighted inputs that, with adjustment, time, and repetition, can be made to produce appropriate outputs. A multi layer feedforward network with at least one hidden layer and a sufficient number of hidden units/neurons is

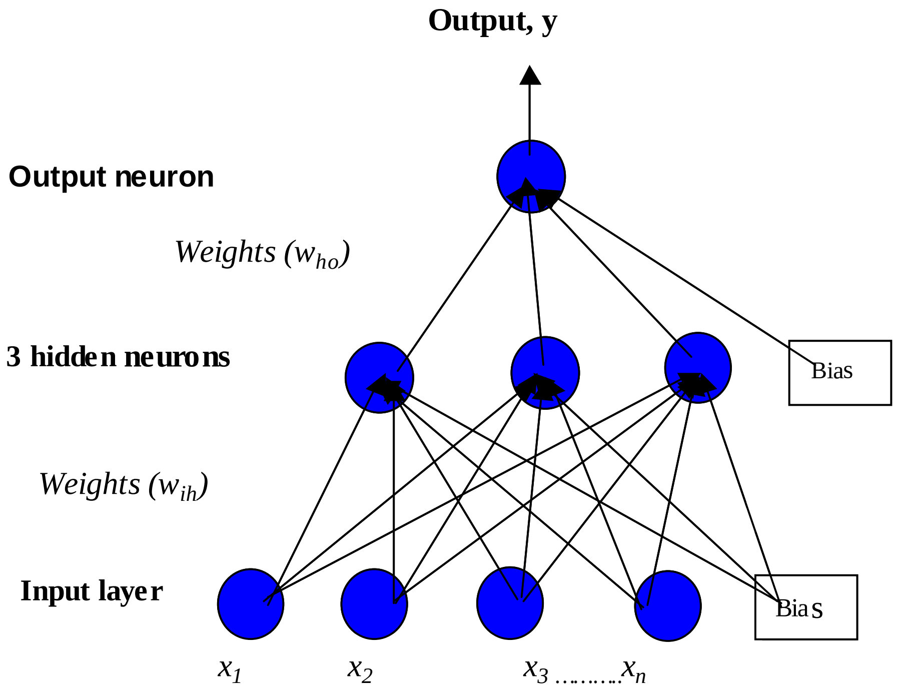

Figure 1: Topology of Artificial Neural Network

capable of approximating any Borel-measurable function, Hornik *et al* (1989), and therefore ANN is powerful enough to represent any form of time series.

TIME LAGGED NEURAL NETWORKS

Time lagged Neural Networks (TLNN) is network in which temporal dependence in time series data is captured by supplying the network with present value of the input, x_t in addition to p past values of the input $x_{t-1}, x_{t-2}, ...x_{t-p}$. The relationship between output y_t and the input is assumed to be of the form

$$yt = f(xt\ ,\ xt-1,\ xt-2,...xt-p)+et \qquad (5)$$

where e_t is a zero mean Gaussian variable with variance $\sigma_e{}^2$ and $f(.)$ is a non-linear function in its arguments. We may view this as a non-linear autoregressive process. In neural network context we write ARNN to mean Autoregressive Neural Network.

Let s be the seasonal period and assume that the series one season in the past contains enough information about the patterns in the time series data. Since data is available up to time t the input window is shifted to start at s steps below this time. The network is then trained with inputs $x_{t-s}, x_{t-(1+s)}, x_{t-(2+s)}, ..., x_{t-(p+s)}$, s > 0 which we denote by the vector $\mathbf{x}_t$ and desired or target variable x_t in an autoregressive model. Upon convergence, the network is fed with $x_t, x_{t-1}, x_{t-2}, ...x_{t-p}$, so as to output x_{t+s} an s steps ahead forecast of x_t. Assuming that the output neuron is linear, the NN output of the k^{th} hidden neuron is given by

$$\hat{x}_k(t) = \phi\left(\sum_{l=s}^{p} w_{lk} x_{t-l} + b_k\right) \qquad (6)$$

where ϕ (.) is the activation function of the neuron k and w_{lk} are its connection weights, [Cichocki and Unbehauen (1993), Faraway and Chatfield (1998)]. Some of the commonly used activation functions are the linear function, where k is a real-valued constant, the logistic function $\phi(z) = \frac{e^z}{1+e^z}$ and the hyperbolic tangent function . For s steps ahead forecasting, the training phase model is given by

$$\hat{x}_t = \sum_{h=1}^{N_h} w_{ho}\phi\left(\sum_{l=s}^{p} w_{lh} x_{t-l} + b_h\right) + b_o \qquad (7)$$

where N_h is the number of neurons in the hidden layer with connection weights w_{ho} to the output neuron, which has a bias b_o . The forecasting model is given by

$$\hat{x}_{t+s} = \sum_{h=1}^{N_h} w_{ho} \phi \left(\sum_{l=0}^{p-s} w_{lh} x_{t-l} + b_h \right) + b_o \qquad (8)$$

The parameter space includes the vector of biases, the matrix $\mathbf{w_1} = ((w_{lh}))$ of weights connecting the inputs with the hidden neurons and the matrix $\mathbf{w_2} = ((w_{ho}))$ of weights linking the hidden neurons to the output neuron. To estimate $\psi = (b, w_1, w_2)$, nonlinear least squares procedures are used to minimize

$$E(\psi) = \sum_{t=1}^{N_h} e_t^2 = \sum_{t=1}^{N_h} (x_t - \hat{x}_t)^2 \qquad (9)$$

The optimization techniques for minimizing the error function 9 are referred to as learning rules. The best-known learning rule is the error backpropagation, (Rumelhart *et al*., 1986) or back error propagation, which is also called the generalized delta rule. The rule is based on the idea of continuously modifying the strengths of the input connections to reduce the difference (the delta) between the desired (target) output, and the actual output. Other variations of this rule include gradient descent with momentum, gradient descent with adaptive learning rate, quasi-Newton, conjugate gradient, Scaled conjugate gradient and Levenberg-Marquardt. Standard batch backpropagation is the most popular training method of all, but it is slow, unreliable, and requires the tuning of the learning rate, which can be a tedious process. Levenberg-Marquardt is very fast and reliable for small least-squares networks, Quasi-Newton techniques are good for medium-sized networks while conjugate gradient techniques are good for large networks, see Bishop (1995) for discussion of the learning rules.

Selection of Network Architecture

The two main problems in network specification are; selection/determination of input variables/lags and determination number of units/neurons in the hidden layer. Problems, which can occur due to poor selection of the parameters, include: increased input dimensionality, increased computational complexity and memory requirements, increased learning difficult, mis-convergence and poor model accuracy. There is also the problem of understanding results from complex models.

Determination of Input Lags

To determine which lags to include in an input set X of variables, autocorrelations and partial autocorrelations analysis together with AIC and its variation have been used, but they have not been very helpful. Network pruning, which involves removing small magnitude weights, has been developed, (Hassibi & Stork, 1993). A large network is sequentially reduced by removing some network connections (number of hidden units) on the bases of elements of the inverse of ‘Hessian’ matrix, Bishop (1995). The problem is the difficulty involved in computing the elements of a Hessian matrix, which are the second derivatives of the error function with respect to the training weights.

MacKay,(1992) and Neal,(1996) developed input selection method referred to automatic relevance determination (ARD) model based on regression problem with many input variables. This method defines a prior over the regression parameters that embody the concept of uncertain relevance, so that the model is effectively able to infer which variables are relevant and then switch the others off thus preventing those inputs from causing significant overfitting. This is achieved by looking at the distribution of the synaptic weights, which connect one input unit to all of the units in the next layer. The variance of this distribution can give an idea about size of the weights controlled by each one of the input units, namely:

- A small variance suggests that the weights are quite close to 0 thus the input controlling those weights is not very relevant.
- Conversely a large variance is typical of distribution of weights, which are connected to a relevant input.

The variance of the distribution of weights is controlled through a hyper parameter α, where α is inversely proportional to the square root of the actual variance. A small value of α indicates that the variance of the weights is large and thus the associated variable/lag is relevant. Computation of α involves evaluating the eigenvalues of the ‘Hessian’ matrix, which is not easy. Bishop (1995) however notes that this may be avoided if all the weight parameters are ‘’well determined’’ which can only be the case if $N_t >> N_h$.

This method may be applied to any univariate time series problem if an autoregressive neural network (ARNN) is to be fitted to the data. In such a case, lagged inputs of order p are treated as variables among which there maybe some irrelevant variables (lags) to the prediction of the output variable.

Automatic Relevance Determation Method: Proposed Rule

Suppose the Neural Network is fed with the first *p* input lags, ignoring the biases which are generally constant terms, the l^{th} input lag influences the output through the weights w_{lh} , h = 1, 2, 3......N_h linking it with the hidden neurons and w_{ho} indirectly linking it with the output. The total influence on output due to this input which we denote by *inf(l)* is

$$inf\,(l) = \sum_{h=1}^{N_h} w_{lh} w_{ho} \tag{10}$$

Assuming that the network has converged to a local minima with the correct number of hidden units, the relevance of x_{t-l} in predicting the output x_{t+s} is given by $r(l) = |inf(l)|$. A small value of *r(l)* indicates that the input lag is not relevant to the model. The rule may be summarized as follows;

1. Determine the number of hidden neurons, N_h using the existing methods

2. Choose the first *p* lags as the input set where *p* is the largest lag that we suspect to be having an influence on the predicted value x_{t+s}.

3. Train the network in the usual way using all the selected lags until some stopping criterion is satisfied. (Convergence is necessary).

4. Compute the (1 x p) matrix of influence measures as **Inf = $\mathbf{w_2}$x $\mathbf{w_1}$.**

Choose *k*, a pre-set threshold for selection of input variables into the input set X then

$$P(x(t-l)\in X) = \begin{cases} 1 & if \quad |r(l)| > k \\ 0 & if \quad |r(l)| \le k \end{cases} \qquad l = 1,2,3............p \tag{11}$$

where *r(l*) is the absolute value of the l^{th} element of the influence matrix **Inf** or

5. Sort the relevance statistics in descending order and select the first $N_{l.}$
6. Drop all the delays, which do not meet the criteria and proceed with the selected lags.

This method maybe viewed as an improvement of Mackay's ARD method discussed in the previous section in which the distribution of weights associated with a certain lag is assumed to be Gausian with mean 0 and constant variance. While Mackay's method uses the whole set of synaptic weights, the proposed rule uses a sub set of the weights, Bishop (1995) with a clear cut method of selecting the set to ensure that the effects of a particular input lag are well represented. This method is computationally fast, as it requires no evaluation of eigen values of the Hessian matrix, which is not easy.

DETERMINATION OF THE NUMBER OF NEURONS IN THE HIDDEN LAYER

Selecting a small number of hidden units leads to poor approximation of the true data generating process. On the other hand a large number of hidden units may lead to over fitting (poor generalization). The number of neurons in the hidden layer is therefore a concern in the application of neural networks to time series forecasting. Hidden units selection can be based on the nonlinearity test (TLG) proposed by Tersvirta *et al*,(1993) or alternatively on the test proposed by White (1989). Both are Lagrange Multiplier (LM) type statistics). It is even possible to use BIC and AIC,(Stone,1977).

Baum and Haussler (1989) used what is referred to as *Vapnik-Chervonenkis* dimension to show that if N_t is the number of patterns in a network with binary inputs and we wish to correctly classify $(1-\varepsilon/2)\%(\varepsilon \le 0.125)$ of the patterns, then

$$N_t \le (N_w/\varepsilon)\log_2(N_h/\varepsilon) \tag{12}$$

where N_w is the total number of weights(including biases) in the network. To classify correctly a fraction $1-\varepsilon$ ε of new patterns drawn from the same distribution, they were able to show that $N_t \ge N_w/\varepsilon$ for a large two-layer network. Since for such a network $N_w = (N_l + 2)N_h + 1$ we can substitute N_w and solve for N_h to get an approximate rule of thumb given by

$$N_h \le \frac{\varepsilon(N_t - 1)}{N_l + 2} \tag{13}$$

Another suggested rule is to train a network successfully with one hidden neuron then two and so on as one monitors the error for the validation data set. This error decreases with every increment in N_h until overfitting begins. At this point training is stopped and this N_h is taken to be the best choice. This latter rule and

Baum-Haussler's rule (though not developed for continuous data) were used in this study.

APPLICATION TO TIME SERIES DATA

Three data sets were used for the empirical study namely:

- The 'Airline' data (N=144): Monthly totals in thousands of international airline passengers from Jan. 1949 to Dec. 1960 from Box & Jenkins (1976).
- The 'Tourist' data (N=156): Monthly totals in thousands of world tourists visiting Kenya from Jan. 1971 to Dec.1983 from Kenya Bureau of statistics: statistical abstracts.
- The 'Nottem' data (N=240): Mean monthly air temperature at Nottingham Castle from Jan 1920 to Dec. 1939 available at http://www.personal.buseco.monash.edu.au/~hyndman/TSDL/.

Time plots in figure 2 clearly show that the three data sets have different properties with Airline data showing nonlinearity (multiplicative seasonality) and trend, Nottem

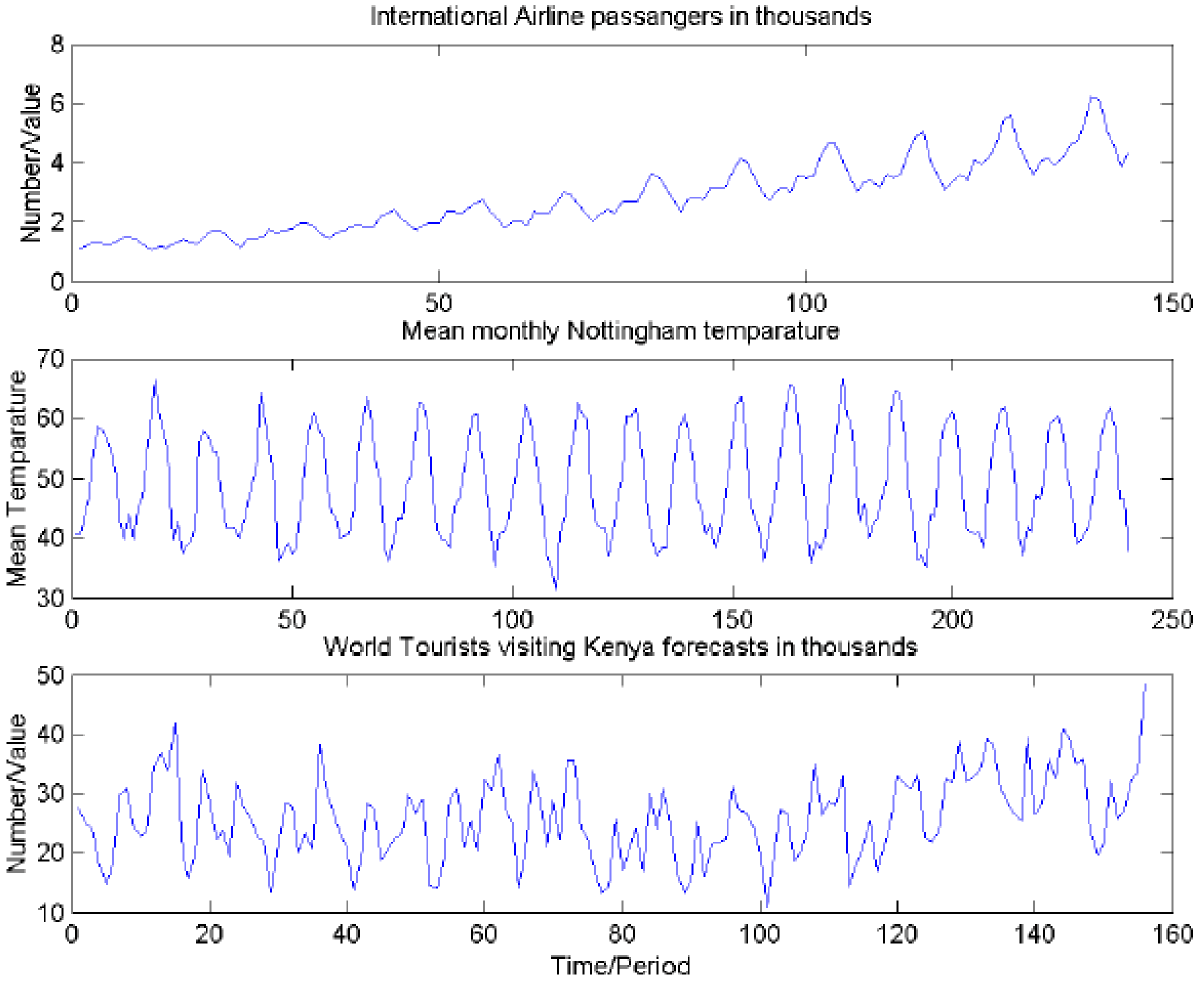

Fig. 2: Time plots for the raw data sets

data is dominated by additive seasonal patterns while Tourists data appears to have seasonal patterns and somehow quadratic trend.

ARIMA Results: For each data set a Seasonal ARIMA model was fitted to the first *N-12* values after appropriate transformation of the raw data. The best model was found by inspecting the AIC, the AICc and the BIC for the minimum and the results are given in table 1 of appendix. Residuals analysis for normality and test for significantly correlated lags using Box-Ljung statistics (Qc) was done and the results are shown in in table 2 of appendix 1.

Neural Networks Results: Each of the three data sets is first scaled by dividing each value by the maximum value within it to ensure that the inputs lie in the interval [0, 1]. The Time lagged multilayered feedforward Neural Network (TLNN) with one hidden layer and bias in both layers was used. The last 12 values were dropped (Forecasting targets) and the rest *N -12* were used for training *(N_t* values) and validation (N_v values) such that $N_t + N_v = N$ *- 12* with N_t taking at least two thirds of *N - 12*.

Backpropagation with Levenberg-Marquardt optimization, hyperbolic tangent activation function neurons at the first layer and a linear transfer function on the output neuron were used. The training was done until mean squared error (MSE) reached constant values as a sign of convergence. The validation error was also monitored for the selected lags to ensure that the AIC AICc and BIC were minimum, see table 3 for Network model selection statistics. The proposed ARD rule was empirically verified and the results compared with those obtained using Mackay's rule. Both rules picked the similar lags for each of the three data sets, see table 4 (appendix) for Airline data's ARD statistics, ARD statistics for the other data sets can easily be verified. In the table 3, the notation ARNN (2, 13; 1) means that the best lags for 12 steps ahead forecasting (equation/model 8) were 2 and 13 in network with 1 output neuron.

Table 3: Neural Network model selection statistics

DATA	AIC	AICc	BIC	Model
Airline	-788.22 (-684.57)	-788.63 (-684.04)	-770.81 (-665.63)	ARNN(0,12;1)
Tourist	-538.31 (-696.84)	-537.31 (-695.84)	-511.8 (-669.66)	ARNN(0,1,8,12;1)
Nottem	-1303.3 (-1329.7)	-1303 (-1329.4)	-1281.8 (-1307.8)	ARNN(1,12;1)

KEY: Training statistics (validation statistics)

Table 4: ARD statistics for Airline data

When *s = 1* in model 7 and 8 we got one-step ahead

Lag	α	r(l)	Lag	α	r(l)
1	2.681	0.8266	12	1.7	0.6683
2	262.898	0.0722	13	94	0.0728
3	344.274	0.0612	14	130.2	0.0664
4	144.226	0.1051	15	2740.1	0.005
5	76.77	0.1489	16	2171.5	0.0064
6	64.744	0.1627	17	119.9	0.0668
7	82.167	0.1423	18	24.1	0.1726
8	71.221	0.1548	19	74.2	0.0834
9	220.799	0.0796	20	66.3	0.0949
10	396.664	0.0529	21	795.9	0.0112
11	102.983	0.1247	22	63.4	0.0974
12	1.956	0.9594	23	39.6	0.1223
13	2.925	0.7866	24	2.8	0.5164

Table 1: SARIMA model selection statistics

DATA	**Transformation**	**AIC**	**AICc**	**BIC**	**Model**
Airline	Natural log.	-443.25	-443.16	-435.49	ARIMA (0, 1, 1)(0, 1, 1)$_{12}$
Tourist	Divide by range	-194.2	-194.11	-186.26	ARIMA (0, 1, 1)(0, 1, 1)$_{12}$
Nottem	None	870.7	870.9	888.4	ARIMA (1, 0, 0)(2, 1,)$_{12}$

Data set	**df**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Airline	p-value	0.0558	0.0951	0.159	0.224	0.335	0.435	0.409	0.510
	Qc	3.657	4.705	5.182	5.684	5.719	5.899	7.194	7.250
Tourist	p-value	0.0515	0.015	0.031	0.0448	0.0825	0.126	0.155	0.148
	Qc	3.793	8.404	8.872	9.752	9.754	9.977	10.636	12.071
Nottem	p-value	0.195	0.38	0.582	0.642	0.764	0.589	0.628	0.667
	Qc	1.683	1.938	1.954	2.515	2.584	4.656	5.264	5.821

prediction model with which Faraway and Chatfield produced forecasts. The two ARD rules picked x_{t-1} , x_{t-12} and x_{t-13} as the best input variables for such a model and with s = 12 the rule picks x_{t-12} , x_{t-24} as best predictors of x_t which is equivalent to x_t , x_{t-12} as predictors of x_{t+12} , that leads to ARNN(0, 12; 1). The optimal models were then used to forecast the last 12 values in the original data sets using model 8, see results in table 5 and 6.

Table 5: Forecasts for Airline data

period	ARIMA	ANN	ANN (F & C)	Actual air
1	432.6	401.4	399.7	417
2	411	381	380.4	391
3	487.9	443.2	438.4	419
4	475.9	431.3	437.4	461
5	504.7	454.2	465.3	472
6	567.2	515.4	518.5	535
7	658.6	590	588.9	622
8	671.8	602.6	605.6	606
9	556.4	499.3	493.3	508
10	489.1	443	441.1	461
11	435	393.9	399.5	390
12	486.7	434.9	446.7	432

Key: F & C used to stand for Faraway and Chatfield

Table 6: Forecasts for Tourist and Nottem data

Period	Arima.T	ANN.T	Actual.T	Arima. Nott	ANN. Nott	Actual. Nott
1	46.2	33	39.3	40.31	42.1	39.4
2	45.1	32.79	34.9	40.89	41.2	40.9
3	37.9	30.14	35.7	39.31	42.4	42.4
4	35.4	26.54	24.1	46.83	46.6	47.8
5	33.6	24.08	19.6	53.52	52.4	52.4
6	32.4	24.97	21.4	58.26	59	58
7	46.3	31.56	32.2	60.76	59.6	60.7
8	33.7	28.36	25.9	61.03	60.4	61.8
9	35.3	28.44	27.5	57.37	57	58.2
10	42.7	30	32.4	51.84	50.7	46.7
11	39.6	30.62	33.8	42.05	47.8	46.6
12	47.9	33.06	48.6	39.14	39.2	37.8

Comparison Statistics: After retransformation of the variables to the original scale, the forecast “ $\hat{x}_{t+i}$ of x_{t+i} made at time *t* was used to compute the error E_i due to this forecast given by $E_1 = x_{t+i} - \hat{x}_{t+i}$. To compare the forecasting ability of different method/models one could use Mean Square Error $MSE = s^{-1}\sum_i E_i^2$. Mean Absolute Percentage Error, $s^{-1}\sum_i \left|E_i / x_{t+i}\right| x\,100\%$% as suggested by Wei (1990). The Mean Euclidean Distance $MED = s^{-1}\sqrt{\sum_i E_i^2}$ as proximity dissimilarity measures and Product Moment Correlation Coefficient (PMC) as a proximity similarity measure may also be used. We chose to use MED, MAD and PMC and the results are shown in table 7. These statistics and the forecasts (plotted in figure 3) show that ANN out-performed ARIMA model in two cases (Airline and Tourist data) while in the case of Nottem data the statistics are not significantly different. From the time plots, we noted that Airline and Tourist data sets are not apparently linear while Nottem data is clearly linear with dominant regular seasonal patterns. We may therefore infer that NN are better forecasters than ARIMA models especially with economic data, which naturally exhibits non-linear properties. As noted by Faraway and Chatfield, care must be taken in model specification. Although the approach used is different, the results are almost similar in the case of Airline data.

Table 7: Models comparison statistics

Data set	Model	MED	MAD	PMC
Nott	ARNN	0.6608	1.5876	0.963
	ARIMA	0.4847	1.2917	0.984
Airline	ARNN	5.2481	15.484	0.981
	F & C(NN)	5.06	15.583	0.97
Tourist	ARIMA	12.253	38.667	0.977
	ARNN	1.621	4.134	0.906
	ARIMA	2.716	8.508	0.854

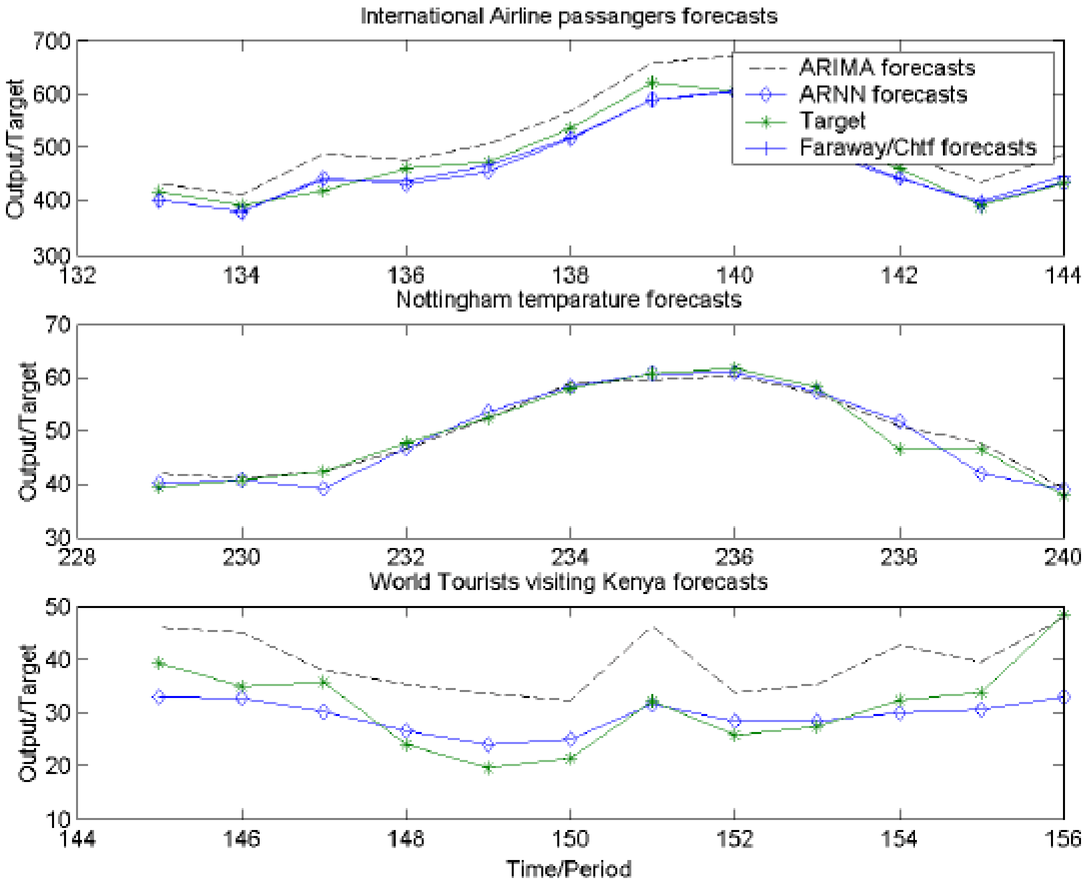

Fig. 3: Twelve (12) steps ahead forecasts

CONCLUSION

In this paper we have tried to offer an empirical comparative evaluation of the performance of ANN to the problem of univariate time series forecasting. Most of the recent literature has focused on comparing ANN and ARIMA models and the results have been conflicting. Our results show that the ANN are relatively better than ARIMA models in forecasting ability but the nature of the data may influence the results. More research need to be done on the same perhaps with linear or non-linear series and/or shorter or longer series in order to generalize the results.

We have shown that instead of modeling using recursive estimation, which introduces recursion errors to successive forecasts, the relationship between x_t and x_{t+s} may be modeled and the *s* steps ahead forecast generated all at once.

The main problems with ANN seem to be their lack of explanation capabilities and of a proper building methodology to define the network architecture. Most of the ANN modeling process is basically empirical and we have proposed an easier ARD rule, which seems to be working well empirically. This rule may be investigated further and perhaps a theory developed to be included in Time Series modeling methodology for Artificial Neural Networks.

REFERENCES

Abraham, B and Ledolter, J. (1983). *Statistical methods for forecasting.* New York: John Wiley & Sons.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control* 19, 716-723.

Bishop, C.M. (1995). *Neural networks for pattern recognition*. Oxford: Oxford University press.

Baum, E.B. and Haussler D. (1989). What size net gives valid generalization? *Neural Computation* **1(1),**151-160.

Box, G. E. P and Jenkins, G. M. (1976). *Time analysis, Forecasting and Control*. San Francisco, Holden – Day.

Box,G.E.P and Pierce,D.A (1970). Distribution of the residual autocorrelations in ARIMA time series models, *Biometrika*,**52**,181-192.

Burnham, K. P., and Anderson, D.R. (1998). Model selection and inference. Springer-Verlag, New York.

Chatfield, C. (1975). *The analysis of Time Series*. London, Chapman and Hall.

Cichocki, A. and Unbehauen, R. (1993). Neural Networks for Optimization and signal processing John Wiley and Sons, New York.

Faraway J. and Chatfield C.(1998). Time series forecasting with neural networks: A comparative study using the airline data. *Journal of applied statistics* ***47,***231-250.

Hassibi, B. and Stork D. G. (1993). Second order derivatives for network pruning:Optimal brain surgeon. *Advances in Neural information processing systems* **5**, 164-171. San Mateo.

Hill, T., O'connor, M. and Remus, W. (1996) . Neural network models for time series forecasts, *Management Science*, **42,** 1082-1092.

Hornik, K., Stinchcombe, M. and White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, **2**, 359-366.

Hurvich, C. M., and Tsai, C.L. (1989). Regression and Time series model selection in small samples. *Biometrika* **76**, 297-307.

Kuan, C. M. and White H. (1994). Artificial neural networks: an econometric perspective (with discussion). *Econometrics. Rev.*, 13, 1-143.J.

MacKay, D. J. C. (1992). Bayesian interpolation, *Neural Computatio*n, vol. **4**, no. 3, pp. 415-447.

Marquardt, D. (1963). An algorithm for least squares estimation of nonlinear parameters. *J. Soc. Ind. Appl. Math*. **11**, 431.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. New York: Springer.

Rumelhart, D.E., Hilton, G.E. and Williams, R.J. (1986). Learning international representations by error propagation. *Parallel Distributed Processing*. Eds. D.E. Rumelhart and J.L. McClelland. MIT Press, Cambridge, MA.