Feed-Forward Artificial Neural Network Model for Forecasting Rainfall Run-off.

Roger D. Braddock
Environmental Sciences, Griffith University, Nathan 4111, Australia

Michael L. Kremmer
Commerce and Administration, Griffith University, Nathan 4111, Australia

Louis Sanzogni
Australian Catholic University, PO Box 247, Everton Park 4053, Australia

Abstract This paper presents the results of a blind test of the ability of a Feed-Forward Artificial Neural Network to provide out-of-sample forecasting of Rainfall Run-off using real data. The results, while comparable with those obtained using best methods currently available, are surprising in that they cast some doubt upon the widely held assumption that this hydrological system is by definition non-linear. The focus of the paper has been an easily repeatable experiment applied to rainfall and runoff data for a catchment area; which particular catchment was not revealed to the experimenters i.e., a blind experiment. To this end, a simple model has been specified, and the architecture of the neural network and the data preparation procedures adopted are discussed in detail. The results are presented and discussed in detail and the extent to which the system was found to be non-linear is quantified.

1. INTRODUCTION

The implementation of a feed-forward artificial neural network model (*ffann*) for forecasting rainfall runoff for a catchment normally requires a knowledge of the relationship between rainfall to the catchment (input), and the runoff or stream flows within the catchment (output). Frequently models of various forms are used in a predictive capacity to accept rainfall inputs, and to predict the stream flows [Wheater *et al.* 1993]. These models consist of empirical or statistically based time series models such as those based on the unit hydrograph, conceptual catchment-scale models such as the Stanford Watershed model, IHACRES [Jakeman *et al.* 1990] and deterministic, spatially distributed models such as SHE, TOPOG and IHDM. The IHACRES model in particular has undergone considerable development, and has been applied widely to about 100 catchments in Australia, as well as a range of hydrological methodologies worldwide.

Recently Minns and Hall [1996] have applied neural networks in a hydrological setting and have obtained promising results. However their experiments were performed using a data set generated by a mathematical model, rather than using the observed behaviour of a physical catchment.

The aim of this study is to provide an easily repeatable experiment requiring no experience in the application of *ffann* or extensive knowledge of hydrology. The aim is to develop a *ffann* to model a catchment, using the observed rainfall to predict the physical runoff. In the first section of the paper, the architecture of the *ffann* and the training method utilised are described in detail. The preprocessing of the data has been kept to a minimum, and a simple model has been specified on the basis of the data alone. The data set was obtained by arrangement from the ANU's Center for Resource and Environmental Studies without information which would identify the catchment or a description of its characteristics[1]. It consisted of daily observations of the average temperature in degrees, precipitation in millimetres and stream flow in cubic metres per second from 12/2/1972 to 27/4/1990, some 6641 observations in total. The steps in preparing the data and selecting the *training*, *test* and *acceptance* sets are also described and the model is trained on the *training* set data. The model is then used in its predictive mode to obtain 1096 one step ahead, unconditional out-of-sample forecasts. The predictive performance of the *ffann* is then analysed and compared with the performance of other models, particularly the IHACRES model which is known to compare very favourably with other models [Wei *et al.* 1996]. Finally, the extent to which the *ffann* is exploiting non-linearities in the data is assessed.

[1]We gratefully acknowledge the assistance of Dr S Schreider in making this data set available.

2. NEURAL NETWORK ARCHITECTURE

Figure 1 is a stylised representation of the typical architecture of a multi-layered (*ffann*) of the kind implemented in this study [Masters 1993:78]. The network consists of sets of neurons organised in hierarchical layers from the input layer at the bottom to the output layer at the top. The input and output layers require at least one neuron each. The input layer performs no processing; it simply stores the input values, x_j, $j=1,\ldots,J$, to be processed. Between the input and output layers is at least one "hidden layer" of neurons. The network depicted in Figure 1 is referred to as a three layered feed forward artificial neural network, as it has one hidden layer and its processes run exclusively from the output of one layer to the successive layer. The values fed to the neurons in the hidden layer net_h, $h=1,\ldots,H$, are the weighted sum of the x_i inputs to which is added the bias neuron b_h, $h=1,\ldots,H$, in the form:

$$net_h = \sum_{j=1}^{J} w_{hj} x_j + b_h \text{ ,for } h = 1,\ldots,H, \qquad (1)$$

where $W=\{ w_{hj} \}$, is a matrix of the weights assigned to the links between the input neurons and the hidden layer neurons.

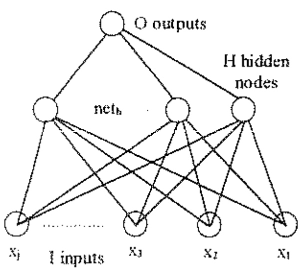


Figure 1: Architecture of a feed-forward neural network

The net_h vector values are inputs to a nonlinear activation function g( ); the outputs of this function are commonly referred to as hidden neuron activations or more formally as translates of the activation function g( ). Several forms of activation function g( ) have been discussed in the literature [Kalman and Kwasny 1992]; here, the hyperbolic tangent activation function from will be used. The weighted sum, with weights $\overline{W}_{oh}$, $h=1,\ldots,H$, of these activations are then offset by the addition of another bias neuron $\overline{b}_o$ to provide the network output o_o:

$$o_o = \sum_{h=1}^{H} \overline{w}_{oh}\, g(net_h) + \overline{b}_o . \qquad (2)$$

where $\overline{W} = \{ \overline{W}_{oh} \}$ is a matrix of the weights between the hidden layer neurons and the output neuron.

The output, $o_0 = o_0(x_1, x_2, \ldots, x_J)$, of the *ffann* can be described as a surface above the $(x_1, x_2, \ldots, x_J)$-plane of inputs [Weigend 1991:16]. In time series forecasting, this surface describes a function which could have given rise to the time series used to train the network.

2.1 Training

The application of a *ffann* depends upon the successful implementation of a training procedure which will establish the number of hidden neurons necessary to balance the often conflicting requirements of accuracy in training and out-of-sample generalisation. The training of a *ffann* is in essence a method of performing inductive inference.

The training procedure consists of finding the optimal number, H, of hidden neurons, and associated weights W and $\overline{W}$ and biases b_h and $\overline{b}_o$ which result in a closed form equation from which forecasts can be generated. The most common method of finding the number of hidden neurons is the cross validation procedure [Vemuri and Rogers 1994]. This procedure requires three data sets which are called the *training* set, the *test* set and the *acceptance* set. The *training* set is computationally involved in the training of the *ffann*. The performance of the training network with respect to this data set determines how the weights and biases of the network are adjusted. This is an iterative process referred to as supervised learning.

This process seeks to minimise the mean square global error E=E(H) where:

$$E(H) = \frac{1}{N}\sum_{n=1}^{N}\left(O^n(H) - T^n\right)^2 , \qquad (3)$$

where T^n, $n=1\ldots,N$, are the expected or observed outputs from the physical data, and $O^n(H)$ are the predictions by the *ffann*. Note that N is the number of training samples in the *training* set, and the minimisation is over the weights and biases.

The *test* set is used to monitor the ability of the training network to generalise (ie. extrapolate) to inputs it has not seen. The *ffann's* performance on this data set is periodically monitored during the training process and helps to determine whether to increase or decrease the number of hidden neurons. The *acceptance* set is used to evaluate the performance of the trained network by means of out-of-sample forecasts. It is assumed that if the *ffann* provides adequate performance *ex-post* there is an *a priori* case that it will continue to do so *ex-ante*.

Training begins with a network which has relatively few hidden neurons and the weights and biases are initialised with random numbers. This network is then trained, using an iterative algorithm which continuously adjusts the weights and biases, until the mean prediction error obtained over the *training* set becomes asymptotic as shown in Figure 2; a stylised representation of this process. This procedure is repeated a moderate number of times (five has proven adequate) to ensure that the *ffann* has not become trapped in a local minimum of the mean predictive error function.

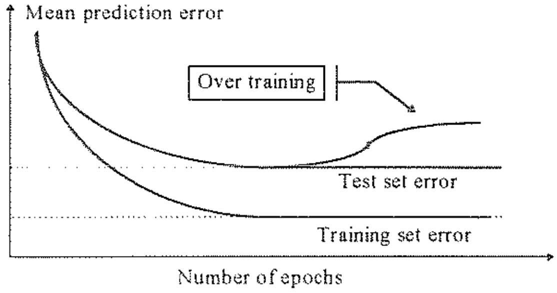


Figure 2: Training set error vs test set error

If the mean predictive error on the *test* set, which is periodically sampled, diverges from that on the *training* set, as shown in Figure 2, the network has been over trained and has begun to learn the idiosyncrasies of the *training* set. In this case, a hidden neuron is removed and the process is repeated. If the error on the *test* set does not diverge from that of the *training* set, an additional neuron is added, in the hope that performance will be improved, and the process is repeated. The objective of this process is to find a network architecture with just enough resources to map the underlying relationship between the network's inputs and outputs. If the *test* set error diverges from the *training* set error when using the minimum number of neurons which will provide adequate performance, then the *training* set and the *test* set are not representative subsets of the entire population and the size of the *training* set must be increased [Masters 1993:183]. The *acceptance* data set is independent of both the *training* set and *test* set, and its operation with the neural network model in also different. The *training* and *test* data set lead to changes in the architecture, and to changes in the values of the weights and biases. Neither the architecture nor the values of the weights and biases are altered during acceptance testing. The performance of the *ffann* is evaluated using the independent *acceptance* data set, where this data is not used to further develop the *ffann*. This is clearly a more stringent test of the performance of the *ffann*, than one which allows the network model to be retrained every time a new data observation (in the *acceptance* set) becomes available.

3. THE EXPERIMENT

The data supplied was analysed on a yearly basis, in terms of mean, standard deviation and range, and some of these results are shown in Table 1. The data for years 1974, 1984 and 1987 were set aside and together constitute the *acceptance* set upon which the performance of the network was tested. These three years were selected because in terms of their mean, standard deviation and range, they fall in the top (1974) the bottom (1987) and the middle (1984) of the data set available and should give some indication of the performance that could be expected from the network over an extended period. From the remaining data, two subsets, the *training* set and the *test* set, were extracted. These two data sets were selected so as to be statistically representative of the entire data set, as indicated by the various statistics in Table 1. The three data sets were then scaled into the effective range of the *ffann* activation function, g( ): in this case -0.9 to 0.9, the effective range of a hyperbolic tangent [Masters 1993].

Let t represent time in days, and assume that the stream flow, F_{t+1}, on day t+1, will be some unknown non-linear function of the stream flow, the precipitation and the temperature observed on previous days i.e. on days t, t-1, t-2,.... Consider the model.

$$F_{t+1} = f\begin{pmatrix} F_t, F_{t-1}, \ln(F_t / F_{t-1}), \ln(F_{t-1} / F_{t-2}), \\ P_t, P_{t-1}, T_t \end{pmatrix} \quad (4)$$

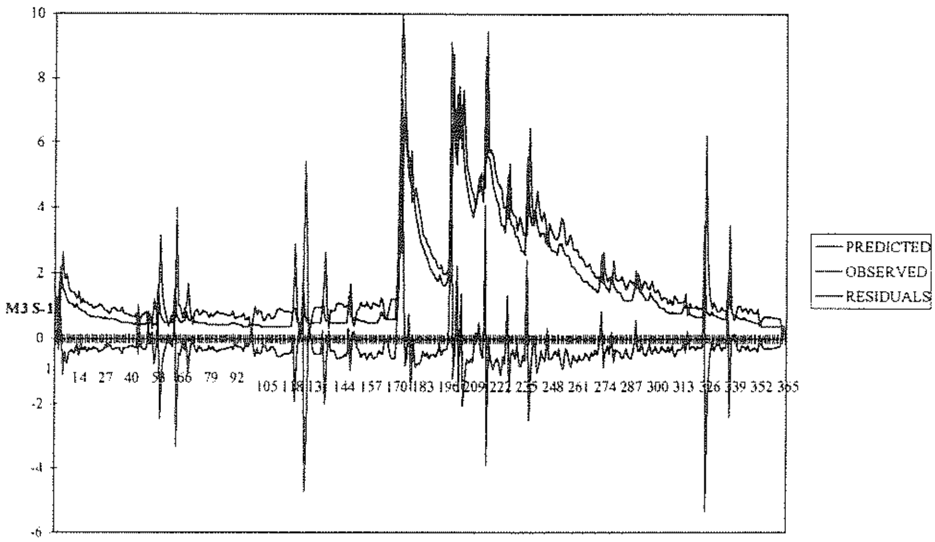
where: F_t = Stream Flow at time t
P_t = Precipitation at time t
T_t = Temperature at time t

and the terms $\ln(F_t / F_{t-1})$ are the relative rates of

Table 1: Data Set Statistics

Data Sets	Mean			Standard Deviation			Range			
	TEMP	PRECIP	FLOW	TEMP	PRECIP	FLOW	TEMP	PRECIP	FLOW	N
Test Set	14.540	2.093	2.709	6.155	5.629	5.200	30.350	62.300	79.398	1109
Training Set	14.336	1.935	2.851	6.249	5.534	5.806	30.200	68.900	102.199	1109
Available Data	14.365	2.269	2.825	6.167	6.248	5.557	31.300	75.900	102.199	5542
1974 Set	14.138	3.589	11.960	5.815	8.941	12.659	23.400	70.200	78.020	365
1984 Set	13.253	2.474	2.876	5.282	7.204	4.209	24.750	61.200	24.965	366
1987 Set	13.818	1.862	1.510	5.641	5.258	1.648	25.250	31.600	9.607	365
Acceptance Set	13.736	2.642	5.446	5.590	7.320	9.033	25.650	70.200	78.935	1096

1987 Out-Of-Sample Rainfall-Runoff Forecasts



Days

Figure 3: 1987 Out-of-Sample Rainfall-Runnoff Forecasts

The extent to which the observed errors are a consequence of the model being unconditional, can be quantified, using ordinary least squares (OLS) regression, by regressing the residuals on the observed precipitation at time t+1 as shown in Table 3.

This regression indicates that 33% of the error of the model can be explained by this *ex ante* precipitation. This problem could perhaps be addressed by generating conditional forecasts, the accuracy of which would depend upon how reliably the precipitation at time t+1 could be anticipated. The greatest weakness of the model is that the absolute magnitude of the errors is clearly a function of the average stream flow in the year being forecast. Ideally the errors would be of uniform magnitude regardless of the year being forecast. This problem could perhaps be overcome by biasing the *training* set in favour of years presenting extreme values, or including a year to-date moving average as an explanatory variable.

Finally, the extent to which the *ffann* is exploiting the hypothesised non-linearities in the model specified, needs to be quantified. To this end, an estimate of an OLS regression equation can be obtained using the training data set of the *ffann*. The results of this experiment are provided in Table 4 where not only has the performance of the OLS model estimated upon the *training* set been maintained out-of-sample, the results mirror those obtained using the *ffann* in almost every respect.

This would indicate that the *ffann* has not found the anticipated non-linearities in the data set. This can be further investigated by using a variation on the R_W statistic [Weigend *et al.* 1990:20] given by:

$$R_w = \frac{\text{Residual Variance(nonlinear)}}{\text{Residual Variance(linear)}}, \qquad (6)$$

where the denominator is the residual variance of a linear neural network. Now replace the linear network with OLS estimates of the model fitted to the *training*, *test* and *acceptance* data sets used in this experiment. The ratio of the remaining variances between the fourteen hidden neuron *ffann* and the OLS estimation are:

$R_w(train)=0.99$
$R_w(test)=1.16$
$R_w(acceptance)=1.10$

While these results do indicate that the *ffann* has not been over trained, they clearly demonstrate that the

Table 3 Ordinary Least Squares Estimation of Precipitation t+1 on Residuals.

	Coefficients	*Standard Error*	*t Stat*	*P-value*
Intercept	0.963625	0.114894	8.387065	1.52216E-16
R_{t+1}	-0.34847	0.014907	-23.3763	2.47428E-98
R Square	0.333314	Adjusted R Square 0.332704		

change in the stream flow from day to day. This model necessitates a *ffann* with seven input neurons, for the seven independent variables in the function *f*, and one output neuron. The *training* and *test* data sets from Table 1, were used in training the network, and resulted in an optimal architecture of H=14 hidden neurons. The values of the weights and biases are not given here but are available from the corresponding author.

Once the network has been trained, it is in fact a closed form equation expressed in terms of its activation function. Thus the neural network model, trained on a data set which entirely excludes data from the years 1974, 1984, and 1987, remains unchanged throughout the forecasting test; the values of the weights and biases are no longer adjusted. The generation of the out-of-sample forecasts as the output of the network involves no more than the presentation of the independent variables as input to the network. The forecasts so generated in this study are unconditional and strictly out-of-sample.

4. RESULTS

The results of this experiment are presented in Table 2, and Figure 3 shows the results for the year 1987. Table 2 contains the mean of the residuals (BIAS) the mean absolute error (MAE), the mean square error (MSE), the root mean square error (RMSE), the coefficient of efficiency E, and the coefficient of determination R^2. Schreider *et al.* [1996:870] note that the E statistic quantifies the "proportion of observed stream discharge variance explained by the model", and is defined as

$$E = 1 - \frac{\sum\left(F - \hat{F}\right)^2}{\sum\left(F - \bar{F}\right)^2}, \tag{5}$$

where: F = Observed Flow,
$\hat{F}$ = Predicted Flow, and
$\bar{F}$ = Mean Observed Flow.

In the results for the *acceptance* set in Table 2, the negative sign of the BIAS indicates that the model tends to overestimate the stream flow, while an R^2 of 0.87 indicates that the predictions cluster around a regression line indicative of perfect forecasts. Note that the *acceptance* set, which was designed to include the extreme values of years 1974 and 1987 out performs the *test* set which is statistically similar to the *training* set, in terms of its Bias, R^2, and E statistics. In the results for the individual years which make up the *acceptance* set, note that the magnitude of the forecasting error is a function of the average stream flow in the individual years. This can be seen by comparing the RMSE for these three years in Table 2 with their average flow rates in Table 1. Figure 3 indicates that a similar pattern is apparent, a substantial number of large errors are associated with sudden increases in the stream flow. While this pattern is consistent across all three years, it is interesting to compare the performance of the years 1974 and 1987 which represent the extremes of the data set, with the performance of the model for 1984 which represents an average year. In both 1974 and 1987 there is a significant decline in performance of the model compared to the test set in terms of the explanatory value of its predicted flow rates, R^2, and in the ability of the model to explain the variance in the stream flow as indicated by the E statistic. In contrast, the results for 1984 in terms of the R^2 and E statistics, does outperform the *test* set results.

Turning to the errors, with the exception of the MAE which is marginally poorer, the absolute magnitude of the errors, in terms of the MSE and RMSE, are significantly smaller than those of the *test* set. This would indicate that the frequency and magnitude of the comparatively larger error in 1984 was some what less than that indicated by the representative sample, i.e., the *test* set. Over all, the performance of the model, is, in terms of the BIAS and the E statistic, comparable with the IHACRES model of Schreider *et al.* [1996:869] and the results obtained by Minns and Hall [1996:411], using artificially generated data, when that model is tested out of range.

Table 2: Data Set Statistics

Data Sets	Mean			Standard Deviation			Range			
	TEMP	PRECIP	FLOW	TEMP	PRECIP	FLOW	TEMP	PRECIP	FLOW	N
Test Set	14.540	2.093	2.709	6.155	5.629	5.200	30.350	62.300	79.398	1109
Training Set	14.336	1.935	2.851	6.249	5.534	5.806	30.200	68.900	102.199	1109
Available Data	14.365	2.269	2.825	6.167	6.248	5.557	31.300	75.900	102.199	5542
1974 Set	14.138	3.589	11.960	5.815	8.941	12.659	23.400	70.200	78.020	365
1984 Set	13.253	2.474	2.876	5.282	7.204	4.209	24.750	61.200	24.965	366
1987 Set	13.818	1.862	1.510	5.641	5.258	1.648	25.250	31.600	9.607	365
Acceptance Set	13.736	2.642	5.446	5.590	7.320	9.033	25.650	70.200	78.935	1096

Table 4: *ffann*-Out-Of-Sample Forecasts

ffann	BIAS	MAE	MSE	RMSE	E	R^2
Training Set	0.000415	0.675938	3.78345	1.945109	0.885898	0.941225
Test Set	0.178978	0.820467	9.029704	3.004947	0.743961	0.865782
1974 Set	0.509082	2.969363	52.859356	7.270444	0.669673	0.819712
1984 Set	-0.339408	0.934415	3.653841	1.911502	0.793245	0.896662
1987 Set	-0.329609	0.592354	0.956673	0.978097	0.647099	0.840680
Acceptance Set	-0.053572	1.498196	19.142479	4.375212	0.765237	0.874799

BIAS= Mean of the residuals. MAE= Mean absolute error.
MSE= Mean square error. RMSE=Root mean square error.
E=Coefficient of efficiency R^2=Coefficient of determination.

neural network has simply performed OLS by iteration; i.e., it is making no use of non-linearities in the data. Given that this *ffann* had 181 potentially effective parameters with which to exploit non-linearities in the data this experiment is indicative of the possibility that the system modelled is in fact essentially linear one step ahead. While this does not prove that the system is linear one step ahead, or that the system is itself linear, it does indicate that forecasting errors can not be simply attributed to presumed non-linearities in the system rather than to, for example, measurement errors in the data collection process, or inadequate sampling rates.

5. CONCLUSION

This study provides an introduction to the application of *ffann* to time series forecasting in hydrology. The results suggest that *ffanns* can provide performance comparable with current methods. For the data set considered, considerable doubt is cast upon the assumption that this system is by definition non-linear.

As the experiment was designed to provide results which could be considered representative rather than optimal, several possibilities for improvement suggest themselves. The differences in the flow rate could be forecast rather than the flows themselves, the inputs could be averaged to smooth out the noise, calculated variables for effective rainfall could be included as inputs, the network model could be applied to the residuals of existing models; the possibilities are endless.

REFERENCES

Jakeman, A.J., I.G Littlewood, and P.G. Whitehead, Computation of the instantaneous hydrograph and identifiable component flows with application to two small upland catchments. *Journal of Hydrology 117*, 275-300, 1990.

Kalman, B.L., and S.C. Kwasny, Why Tanh: Choosing a sigmoidal function, *Proceedings of the international Joint Conference on Neural Networks, 4*, 578-81, 1992.

Masters, T. *Practical Neural Network Recipes in C++*, Academic Press, Inc, Boston, 1993.

Meese, R.A., and K. Rogoff, Empirical exchange rate models of the seventies, *Journal of International Economics 14*, 3-24, 1983.

Minns, A.W., and M.J. Hall, Artificial neural networks as rainfall-runoff models, *Hydrological Sciences - Journal des Sciences Hydrologiques 41*, 399-417, 1996.

Schreider, S.Y., A.J. Jakeman, and A.B. Pittock, Modelling rainfall-runoff from large catchment to basin scale: the Goulburn Valley, Victoria, *Hydrological Processes, 10*, 863-876, 1996.

Vemuri, V., and R. Rogers, *Artificial Neural Networks: Forecasting Time Series*, IEEE Computer Society Press, Los Alamitos, 1994.

Wei, Y., B.C. Bates, N. Viney, M. Sivapalan, and A. Jakeman, Performance of conceptual rainfall-runoff models in low yielding catchments, *Water Resources Research*, 1996.

Weigend, A., D.E. Rumelhart, and B.A. Huberman, Predicting the future: a connectionist approach, *Systems, 1*(193), 1-26, 1990.

Weigend, A., Connectionist architectures for time series prediction of dynamical systems, *UMI Dissertation Services*, Michigan, 1991.

Wheater, H. S., A.J. Jakeman, and K.T. Beven, Progress and Directions in Rainfall-runoff Modelling, in *Modelling Change in Environmental Systems*, edited by A.J. Jakeman, M.B. Beck, and M.J. McAleer, John Wiley & Sons, Southampton, UK, 1993.