APPLICATION OF ARTIFICIAL NEURAL NETWORKS IN RAINFALL - RUNOFF MODELLING

S.K. JAIN and DEEPA CHALISGAONKAR
National Institute of Hydrology, Roorkee

Abstract *The design, analysis and management of the water resources systems, involves modelling and prediction of the behavior of complex systems. The Artificial Neural Network (ANN) can be used in a large variety of problems, e.g. mapping, dynamic process modelling, optimisation, image processing, data analysis, forecasting, simulation, function approximation etc. Due to the distributed nature of ANNs, destruction of a few nodes or presence of some inconsistent data does not adversely affect the performance of ANNs. An ANN model was applied to rainfall-runoff simulation of an Indian catchment. Hourly rainfall, discharge and potential evaporation data were used. The results show acceptable match between the observed and computed discharges.*

INTRODUCTION

After the advent of digital computers, most of the information processing applications have been based on programmed computing approach. In hydrology, the systems may be nonlinear, multivariate and may have unknown interrelationships. Such problems can be tackled efficiently by the Artificial Neural Network (ANN). Because of their in-built mechanism of growing 'wiser' with 'experience', ANNs are capable of adapting their complexity and their accuracy goes on increasing as more and more of input data are made available to them. The processes involving several parameters are easily amenable to neurocomputing. In most of the existing models, the following areas are inadequately covered:

Unknown processes When the underlying physical laws are unknown or not precisely known, it is impossible to make a physically based model of the phenomenon using these techniques.
Complex process When all the existing complex relationships between the various aspects of the process under investigation may not be recognized, these cannot be properly represented in the model.
Incomplete data In all modelling exercises, preprocessing of data is absolutely necessary. A lack of data creates large uncertainties in the model results, as approximations and guesses have to be carried out on the data. Sometimes, the problem may be either poorly defined or misunderstood and observations of the process may be difficult or impossible to perform.
Simulating human decision paths Another difficult problem arises when a decision model has to be designed. Usually one predefines clear rules in a program (IF..THEN..ELSE rules) or tries to implement an expert system. These models need

some kind of predefinition and often become useless when exposed to unforeseen and undefined situations.

Data intensive tasks Detailed models often require many different parameters, especially while modelling dynamic systems. This requires data transfer and can lead to extensive processing time. For on-line operation, these slow but detailed models are unacceptable, since the state of the monitored system often changes faster than the simulation frequency of the model.

Changing environments To decrease processing time, the number of adjustable parameters is often reduced and to retain the authenticity of the model the results are calibrated. This calibration procedure can take a lot of time, fault tolerance decrease and the resulting model will become useless when the simulated process changes. Furthermore, in rapidly changing environments long calibration times are unacceptable.

Optimal solutions A standard optimisation approach or statistical model provides a solution only when allowed to run to completion; ANN always converges to optimal or near to optimal solutions and need not run to any prespecified solution condition.

For the points mentioned above, conventionally applied modelling techniques need to be refined and complemented to achieve performance by implementing new or different methods.

NEURAL NETWORKS

Attempts have been made to develop a technique that does not require algorithm or rule development and thus reduces the quality and complexity of the software. This technique is known as "Neurocomputing" and the networks laid out with many parallel processing elements to do this neurocomputing are called "Artificial Neural Networks" (ANN). ANNs represent highly idealized mathematical models of our understanding of complex systems. They include the ability to learn and generalize from example, to produce meaningful solutions to problems even when input data contains error or are incomplete, to adapt solutions overtime to compensate for changing circumstances, error tolerance, noise reduction, to process information rapidly and to transfer readily between computing systems. Since no algorithm development is involved, the technology can be easily understood and implemented.

Network Topology

A Neural Network is a massive system of parallel, distributed information processing system that relates an input vector to an output vector. An ANN consists of a large number of information processing elements called neurons/nodes interconnected via unidirectional, weighted signal channels called connections. The neurons are grouped in layers. The neurons in a layer share the same input and output connections, but do not interconnect with themselves. Each layer performs

specific functions. The input layer processing elements get the input vector and transmit the values to the second (hidden) layer of processing elements across connections. Weighted values converging at a node in the hidden layer are summed along with a weighted bias associated with the node. The result is then put through a simple function to generate a level of activity for the node.

The activation levels of the hidden nodes are transmitted across links/connections with the nodes in the output layer. Again these values are weighted during transmission, then summed at the output node and are put through an activation function. The level of activity generated at the output node(s) is the network's solution to the problem presented at the input nodes. All the nodes within a layer act synchronously, meaning at any point of time, they will be at the same stage of processing. The equations governing the mode of operation of such a network, in the generalized form are as below:

$$h_i = f\,[(w_{j,i} \,.\, x_j) + b_i] \qquad (1)$$

$$y_i = f\,[(v_{j,i} \,.\, h_i) + b_i] \qquad (2)$$

where h_i is the activity level generated at the i^{th} hidden node; y_i is the activity level generated at the i^{th} output node; x_j is j^{th} component of the input layer; $w_{j,i}$ and $v_{j,i}$ are weights on the connections to the hidden and output layers of nodes, respectively; b_i are weights biases at i^{th} node (hidden node in case of Eq.(1) and output node in case of Eq. (2)); and f( ) is activation function.

The function f( ) is chosen such that its value lies in the ranges [-1,+1] or [0,1]. This function can be a sigmoidal function, a sine function, a Gaussain function etc.

The ANNs operate on the principle of learning from a training set that involves adjusting the weights of interconnections. The data passing through the connections from one neuron to another, can be manipulated by weights. These weights indicate the strength of a passing signal. Consequently, when these weights are modified, the data transfers through the ANN will change and the overall network performance will alter.

The manipulating parameters can all be adjusted and optimised to get a specific response from an ANN. This process of adjustment and optimisation is called learning/training and is defined by the learning algorithm of an ANN. The learning algorithm is a set of optimisation function that adjusts the weights in a manner, through which an input signal is correctly associated with a desired output signal. Several learning examples can be presented to the network, each attributing to the optimisation of the weight distribution. Accuracy of ANNs goes on increasing as more and more of previous data are made available to it because it has an in-built mechanism of growing 'wiser' with 'experience'. Finally, when an ANN has learned enough examples it is considered trained.

Training the ANNs

In general, it is assumed that the network does not have any prior knowledge about the problem before it is trained. ANNs are trained with asset of typical

input/output pairs called the training set. So at the beginning of training the network weights are initialised with a set of random values say between –1.0 and 1.0. During training, the weights are adjusted to reduce the residual error of the training set. A parameter 'Errorindex-Threshold' represents the root-mean square of sum squared residual error of the training set normalized by the standard deviation of the training outputs. Thus by controlling this parameter one can constrain the number of hidden units added to the network. After examining a few values, a particular value for the ErrorIndex-Threshold can be settled for which the resulting networks provide a better prediction results than other values. The final weight vector of a successfully trained neural network represents its knowledge about the problem.

Backpropagation (BP) Algorithm for Training of ANN

The data enters the network through the input layer. The nodes in the input layer are not computational nodes and each simply broadcasts a single data value over weighted connections to the hidden nodes. Each BP node also has an extra input called the threshold input, which acts as a reference level or bias for node. All hidden nodes thus receive all input data, but because each has a different set of weight, the sets of values differ.

Each hidden node processes its inputs and broadcasts its result to the output layer. The output nodes also have distinct sets of weights and process input values to produce a result. For BP, the network's result is asset of continuously variable values, one per output node. Hidden nodes have no direct connection to input or output. Introducing intermediate layers enhances the network's ability to model complex functions.

The hidden and output nodes process their inputs in two steps. Each multiplies every input by its weight, adds the product to a running total, and then passes the sum through a function to produce its result. This transfer function is usually a steadily increasing S-shaped curve, called a sigmoid function. The attenuation at the upper and lower limbs of the "S" constrains the raw sums smoothly within fixed limits. The transfer function also introduces a non-linearity that further enhances the network's ability to model complex functions.

The key to the back propagation learning algorithm is its ability to change the values of its weights in response to errors. For calculating the errors, the training data must contain a series of input patterns labeled with their target output patterns. (Labeled training data is a common, but not universal, requirement for neural networks). During BP training, a network passes each input pattern through the hidden layer to generate a result at each output node. It then subtracts the actual result from the target result to find the output-layer errors. Next, the network passes the derivatives of the output errors back to the hidden layer, using the original weighted connections. This backward propagation of errors gives the algorithm its name. Each hidden node then calculates the weighted sum of the back-propagated errors to find its indirect contribution to the known output errors.

After each output and hidden node finds its error value, the node adjusts its weights to reduce its error. The equation that changes the weights is designed to minimize the sum of the network's squared errors. This minimization has an

intuitive geometric meaning. To see it, all possible sets of weights must be plotted against the corresponding sum-of-squares of errors. The result is an error surface shaped like a bowl, whose bottom marks the set of weights with the smallest sum-of-squares error. Finding the bottom of the bowl-that is, the best set of weights-is the goal during training.

Generating Results from Trained ANN

After the learning cycles, the learning algorithm of trained ANN is often deactivated and the weights are frozen. Then, a test data set, which it has never encountered before, is presented to the ANN enabling a validation of its performance. This is called testing of ANN. Depending on the outcome, either ANN has to relearn the examples with some modifications or it can be implemented for its designed use.

RAINFALL-RUNOFF MODELLING

Problem of estimating the runoff in a river has received considerable attention of hydrologists as runoff prediction is vital for planning and design of reservoir for hydropower, water supply scheme, irrigation projects, designing of hydraulic structures, flood prediction etc. The success with which ANNs have been used to model dynamic systems in other fields of science and engineering suggests that the ANN approach may prove to be an effective and efficient way to model the rainfall-runoff (R-R) process in simulations where explicit knowledge of the internal hydrologic subprocess is not required. ANNs for daily and hourly streamflow forecasting have been successfully developed and used. French et al. (1992) demonstrated that an ANN is capable of forecasting the complex temporal and spatial distribution of rainfall generated by a rainfall simulation model. Chang and Tsang (1992) used an ANN to model snow water equivalent from multichannel brightness temperatures and obtained better results than from a multiple-regression mode; ANNs have also been applied to groundwater reclamation problems and to prediction of average air temperatures.

In R-R modelling, the input pattern consists of rainfall depths and the output the discharges at the catchment outlet. The contributions from different parts of the catchment arrive at the outlet at different times and the variations in the discharge output are determined by the rainfall depths at both the concurrent and previous time intervals. Hall and Minns (1993) have indicated that the number of antecedent rainfall ordinates required is broadly related to the lag time of the drainage area. Since the ANN relate the pattern of inputs to the pattern of outputs, volume continuity is not a constraint. However, care must be taken to avoid the presentation of contradictory information to the ANN. More specifically, the input pattern may contain many zeros both at the start of the rising limb of the output hydrograph and during the recession when rainfall has ended and flows are decreasing. These two situations could be distinguished by providing an extra input consisting of a binary variable (say, zero for pre-storm and unity for post-storm conditions), but Hall and

Minns (1993) have indicated that antecedent flow ordinates both perform the same function and provide additional information about the input pattern, i.e. the longer the input rainfalls remain zero, the more the output decreases. Hsu et al. (1995) have shown that ANN model approach provides a better representation of the R-R relationship of a medium sized basin that the linear ARMAX approach of the Sacramento soil moisture accounting model. Minns and Hall (1996) have reported a series of numerical experiments in connection with the application of ANN to R-R modelling and concluded that the ANNs are capable of identifying usable relationships between discharges and antecedent rainfalls.

The use of an output variable in the input is referred to as recurrent back-propagation. The inclusion of the flow at time (t-1) as an input to determine the flow at time t may appear to introduce an element of flood routing into the model, but that is not the purpose of the ANN. Unlike the conventional R-R models, the network seeks to learn patterns and not to replicate in detail the physical processes involved in transforming input into output. The learning process does not depend upon any number of (active) parameters or their possible physical interaction. Some people have a feeling that the ANN could perhaps be regarded as the ultimate black-box model.

The Study Area and Data Availability

The Kolar subbasin of the Narmada basin, located in central India, was chosen as the application for this study. This Kolar basin is located in the latitude range of 22°40' to 23°08' N and longitude 77°01' to 77°29' E (Fig. 1).

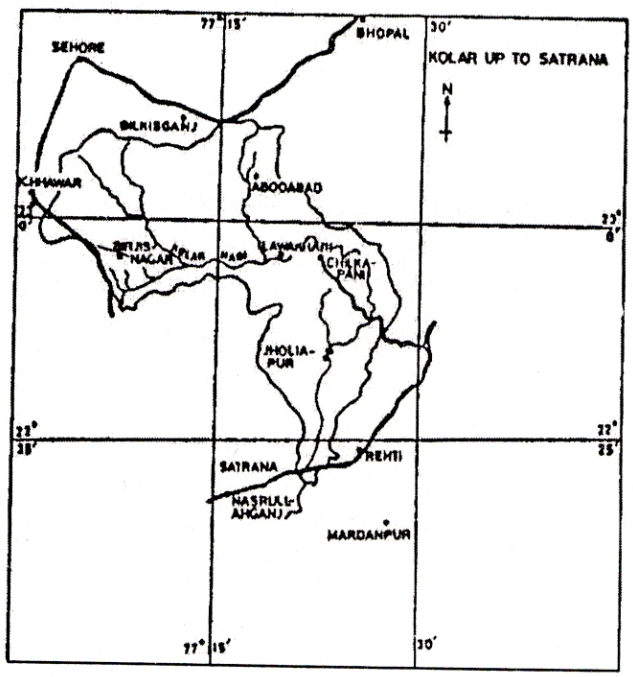

Fig. 1 Study area

The basin elevation varies from 300 m to 600 m. The catchment area of 820 km^2 up to a gauge and discharge site measurement at Satrana has been modelled. The hourly rainfall data at four stations was available for the period 1983-1988 and was used to get weighted average hourly rainfall for the basin. The hourly gauge discharge data for the monsoon season only was available at Satrana. Rating curves were developed for this site to convert hourly stages and to hourly discharge. The pan evaporation data for a station located near the basin in an agricultural area was used. A time step of one hour was used in the simulations. It may be mentioned that the availability of the input data for the catchment is far from ideal. The coverage of the basin through the rainfall stations is also not uniform. However, the data from this basin provides an opportunity for rainfall-runoff modelling in a noisy environment.

Desingn of the ANN

Let R_t, E_t and Q_t represent the rainfall (mm/hr), potential evaporation (mm/hr) and runoff (mm) at time t. In the design of the ANN six variables R_t, E_t, R_{t-1}, Q_{t-1}, R_{t-2} were used as input and the output variable was Q_t. These data were normalised so that the data were available in 0 to 1 range. This normalized data was used for training the ANN. The number of hidden layers in an ANN and the number of nodes in each of these hidden layers are two important parameters in the design of an ANN. In general, the performance of the ANN improves as the number of hidden layers increases though the network becomes more complex to solve.

In the preset case, a number of trials were made by varying the number of hidden layers from 1 to 3. Moreover, number of nodes in each hidden layer was varied from 1 to 15 in each case. For each of these combinations, the output mean squared error was observed. It was found that with a five-layer ANN consisting of input layer, three hidden layers and output layer having 6 nodes in input layer, 5, 10 and 5 nodes in the first, second and third hidden layer respectively and one node in the output layer, the normalized output mean squared error was the minimum. The weights for this configuration are given in Table 1. It may be mentioned that for most of the real-life applications, three to five layers give acceptable results. Further, it was observed that there was not much improvement in the results if the number of nodes was increased beyond limit. In a few cases, as the number of nodes in the hidden layer was increased, the error was found to increase rather than decrease.

Validation of the ANN Model

In the validation phase, the weights calculated during the design phase were freezed. The rainfall and potential evaporation data along with the weights were used to compute the discharge. The observed and computed values of the discharge for the year 1983, 1984 and 1985 have been plotted in Figs. 2, 3 and 4, respectively. It is seen from these figures that there is a very good match between the observed and computed discharges.

Table 1 Weights of various layers in the designed ANN

Layer Node	Weights received at node									
	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10
1/1	-1.242061	-0.514925	-0.553550	-2.154959	0.324895					
1/2	0.503652	-0.613605	0.165679	0.284863	-1.650433					
1/3	0.838053	0.001248	0.435091	0.165630	-0.213159					
1/4	2.711155	3.620046	-8.455657	-0.989244	-0.903086					
1/5	-1.171448	0.443034	-0.342390	-0.293410	-0.459904					
1/6	-0.167194	-2.649626	-7314860	-1.254268	-0.497127					
H1/1	0.162116	-0.053748	1.412494	-2.42631	-0.133750	2.682387	-0.136062	1.378423	0.79143	-0.91723
H1/2	2.069242	-1.769924	2.241963	-2.608817	-1.291923	2.837254	2.141025	1.127490	0.51471	-0.23997
H1/3	-2.100625	1.973427	-4.688900	4.525123	0.382227	5.362124	-1.635203	-2.893311	-1.06479	-0.94399
H1/4	-2.208805	2.528255	0.977842	-0.446711	2.418357	0.061290	-2.028490	-0.030812	-0.34263	0.76737
H1/5	-0.006321	2.769861	-0.120657	1.385977	1.215614	-0.309028	-1.599527	-1.046756	1.10556	1.07391
H2/1	-1.355428	-1.012967	-0.585804	-2.624857	2.505579					
H2/2	5.708912	-0.079519	-0.143218	3.995101	-1.288877					
H2/3	0.670749	-2.119931	-0.126507	-5.127364	0.202332					
H2/4	3.439794	2.306121	-0.598031	6.493931	1.051443					
H2/5	3.108166	0.162573	-1.019526	0.687301	-1.413632					
H2/6	-1.091403	-2.233878	-0.983292	-6.647303	-0.526793					
H2/7	-2.937249	-0.443145	0.865016	-2.472623	2.435343					
H2/8	-0.732827	-1.445113	-0.056799	-3.572592	0.806846					
H2/9	1.065212	-1.130065	-0.164246	-1.032324	0.474226					
H2/10	2.047807	-0.504638	-0.981260	-0.615693	0.212573					
H3/1	-6.257359									
H3/2	-3.170694									
H3/3	0.439842									
H3/4	-8.830539									
H3/5	5.885727									

Note: I represents input layer and H1, H2, H3 represent hidden layer 1, 2, and 3, respectively.

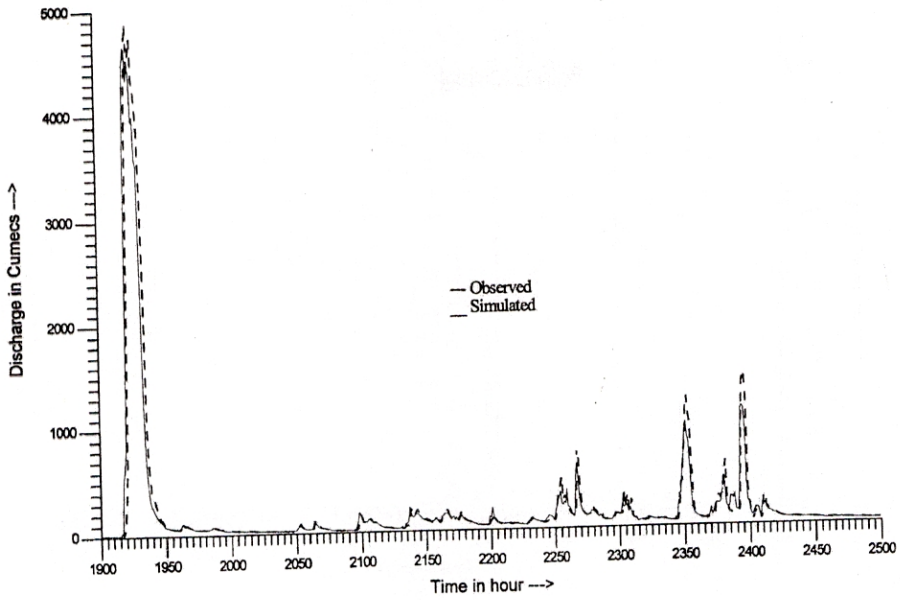

Fig. 2 Plot of observed and simulated discharges
(hour one on x-axis corresponds to July 1, 1983)

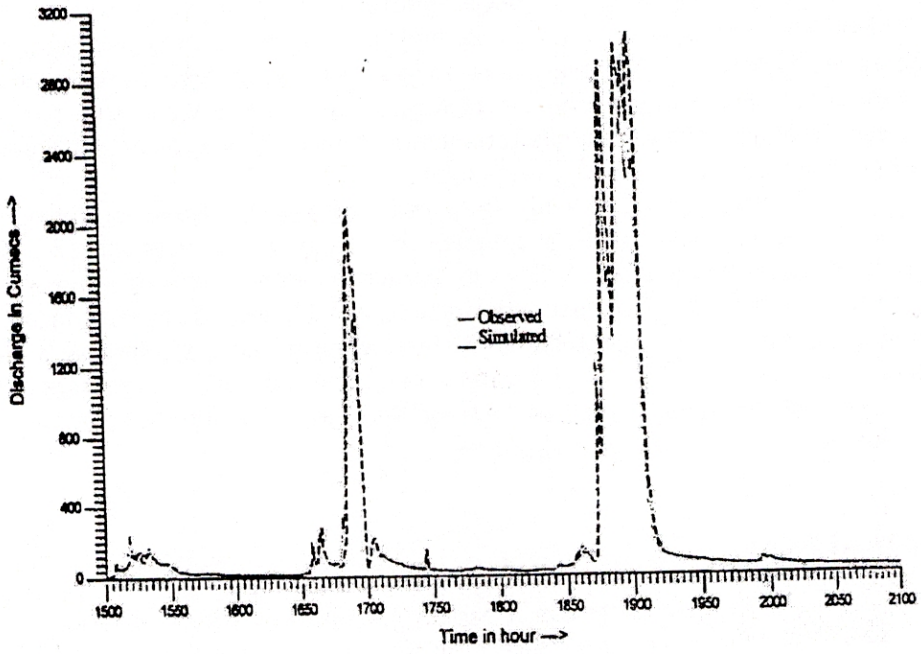

Fig. 3 Plot of observed and simulated discharges
(hour one on x-axis corresponds to July 1, 1984)

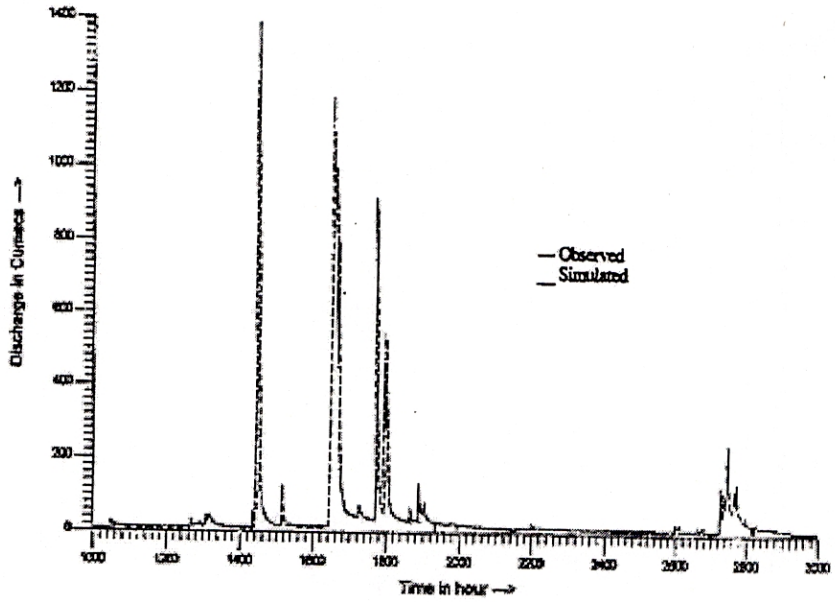

Fig. 4 Plot of observed and simulated discharges
(hour one on x-axis corresponds to July 1, 1985)

CONCLUSIONS

The ANN approach has been applied to the rainfall runoff modelling problem. The results obtained for the example catchment are quite promising. So far the applications of ANNs in the area of hydrology have been to several diverse nature of problems and the results in each case have been very encouraging. Due to the ease of application and simple formulation, this technique has already become a prospective research area with great potential

The ANN approach should be viewed as an alternative to conventional computing techniques not as a complement. Many researchers are considering hybrid systems integrating ANNs, in particular with knowledge-based expert systems, to exploit the advantages specific to each technique. They have many short comings of their own, most notably, the production of inexact solutions, a lack of theory to guide selection of the most appropriate size and configuration for a network, and slow progress during training. Such issues need to be resolved before the potential of ANNs can be realized.

REFERENCES

Chang, A.T.C. and Tsang, I. (1982) A Neural Network Approach to Inversion of Snow Water Equivalent from Passive Microwave Measurements. Nordic Hydrol.,23, 173-182.

French, M.N., Krajewski, W.F. and Cuykendall, R.R. (1992) Rainfall Forecasting in Space and Time using a neural network. Journal of Hydrology, 137, 1-31.

Hall, M.J. and Minns, A.W. (1993) Rainfall-runoff modelling as a Problem in Artificial Intelligence: Experience with Neural Network. In: Proc. 4th Nat. Hydrology Symposium, British Hydrological Society, London.

Hsu, K., Gupta, H.V. and Sorooshian, S. (1995) Artificial Neural Networks Modelling of the Rainfall-Runoff Process. Wat. Resour. Res., 31(10), 2517-2530.

Flood, Ian and Kartam, N. (1994). Neural Networks in Civil Engineering I: Principles and Understanding. ASCE J. Comput. in Civil Engg, 8(2).

Flood, Ian and Kartam, N. (1994) Neural Networks in Civil Engineering II: System and Applications. ASCE J. of Comput. in Civil Engg, 8(2).

Minns, A.W. and Hall, M.J. (1996) Artificial Networks as Rainfall-Runoff models. Hydrol. Sc. J., 41(3), 399-417.

Muller, B. and Reinhardt, J. (1990) Neural Networks. Springer Verlag.

Rao Valluru and Rao, H. (1996) Neural Networks and Fuzzy Logic. BPB Publicaitons.

Nielsen, R. H. (1990) Neuro Computing. Addison Wesley Publishing Company.

Wasserman, P. (1990) Neuro Computing: Theory and Practice. Van Nostrand Reinhold.