Hydrological Sciences Journal

ISSN: 0262-6667 (Print) 2150-3435 (Online) Journal homepage: http://www.tandfonline.com/loi/thsj20

Multivariate modelling of water resources time series using artificial neural networks

H. RAMAN & N. SUNILKUMAR

Multivariate modelling of water resources time series using artificial neural networks

H. RAMAN & N. SUNILKUMAR
Department of Civil Engineering, Indian Institute of Technology, Madras 600 036, India.

Abstract The artificial neural network (ANN) approach described in this paper for the synthesis of reservoir inflow series differs from the traditional approaches in synthetic hydrology in the sense that it belongs to a class of data-driven approaches as opposed to traditional model driven approaches. Most of the time series modelling procedures fall within the framework of multivariate autoregressive moving average (ARMA) models. Formal statistical modelling procedures suggest a four-stage iterative process, namely, model selection, model order identification, parameter estimation and diagnostic checks. Although a number of statistical tools are already available to follow such a modelling process, it is not an easy task, especially if higher order vector ARMA models are used. This paper investigates the use of artificial neural networks in the field of synthetic inflow generation. The various steps involved in the development of a neural network and a multivariate autoregressive model for synthesis are presented. The application of both types of model for synthesizing monthly inflow records for two reservoir sites is explained. The performance of the neural network is compared with the statistical method of synthetic inflow generation.

Modélisation multivariée de séries chronologiques hydrologiques grâce à l'utilisation de réseaux neuronaux artificiels

Résumé L'approche par les réseaux neuronaux artificiels de la génération de séries d'apports à des réservoirs décrite dans cet article diffère des approches traditionnelles de l'hydrologie synthétique dans la mesure où elle s'appuie sur les données plutôt que sur les modèles. La plupart des procédures de modélisation des séries chronologiques se situent dans le cadre des modèles autorégressifs de moyenne mobile (ARMA). Les procédures de modélisation statistique traditionnelles suggèrent un processus itératif en quatre étapes à savoir, choix du modèle, détermination de l'ordre du modèle, estimation des paramètres et vérification. Bien qu'un grand nombre d'outils statistiques soient disponibles pour appliquer une telle procédure, la tâche n'est pas aisée, particulièrement en ce qui concerne les modèles ARMA d'ordre élevé. Ce papier explore les possibilités d'utilisation des réseaux neuronaux artificiels dans le domaine de la génération d'apports. Les différentes étapes du développement d'un réseau de neurones ainsi qu'un modèle de synthèse autorégressif multivarié y sont présentés. Il explique également l'application des différents modèles en vue de la génération d'apports mensuels pour deux sites de réservoirs. Il compare enfin les performances du réseau de neurones à celles de la méthode statistique de génération d'apports.

Open for discussion until 1 October 1995

INTRODUCTION

Planning and management studies of water resources systems usually require the analysis of various hydrological variables such as precipitation, evaporation, inflow to and outflow from a reservoir, demands for irrigation, hydropower and domestic water use. For instance, operational studies of reservoir systems may require the extension of limited historic data of rainfall and runoff to evaluate the likely future performance.

A multivariate time series consists of sequences of values of several contemporaneous variables changing over time. Multivariate water resources time series may generally be represented by four components, namely:

(a) long term trends or other deterministic changes such as jumps exhibited beyond the annual cycle;
(b) seasonal or periodic changes of days, weeks or months within the annual cycle;
(c) almost periodic changes, such as tidal and solar effects; and
(d) stochastic or random variables.

The first three components are considered to be deterministic in the sense that their future values may be known in advance. On the other hand, the latter component is stochastic or random, in the sense that future outcomes may be known only on a probability basis.

Trends may be present in hydrological time series due to long term variations in astronomical phenomena, morphological changes in rivers and lakes, jumps created by manmade interventions such as construction of reservoirs and appurtenant water structures, or natural disruptions. The periodic characteristics of hydrological process are basically produced by the astronomical cycles. The almost periodic changes are attributed to the joint effect of astronomical cycles. The random characteristics of hydrological time series are caused by a combination of several factors, such as variable opacity of the atmosphere to solar radiation, fluctuating turbulence, large scale heat transfer in the atmosphere, and random thermodynamic processes.

Several stochastic models have been proposed for modelling hydrological time series and generating synthetic streamflows. These include autoregressive moving average (ARMA) models (Box & Jenkins, 1970), disaggregation models (Valencia & Schaake, 1973), models based on the concept of pattern recognition (Panu & Unny, 1980). Stedinger & Taylor (1982) studied the performance of five different models for streamflow simulation. All these models have their merits and have also been criticized.

Many of the available techniques for time series analysis assume linear relationships among variables. In the real world, however, temporal variations in data do not exhibit simple regularities and are difficult to analyse and predict accurately. Linear recurrence and their combinations for describing the behaviour of such data are often found to be inadequate. It seems necessary that nonlinear models such as neural networks, which are suited to complex nonlinear problems, be used for the analysis of real world temporal data.

The main purposes of this paper are to analyse and to discuss stochastic modelling of multivariate time series using artificial neural networks and traditional modelling techniques. The present study resorts to a neural network approach for nonlinear modelling of multivariate time series. Neural networks belong to a class of data-driven approaches as opposed to model-driven approaches. The analysis depends on the available data. A neural network is used to predict future values of possibly noisy multivariate time series based on past histories.

Chakraborty *et al.* (1992) successfully used neural networks for forecasting the flour prices in three cities. A forecasting framework of rainfall in space and time was investigated by French *et al.* (1992). They developed a three layer neural network to forecast rainfall intensity fields in space and time and results were compared with a space-time mathematical rainfall simulation model. They concluded that a neural network is capable of learning the complex and nonlinear relationships of rainfall events and performed well in multi-site rainfall forecasting.

This paper describes the various steps involved in the use of an artificial neural network model and a statistical model for synthetic monthly inflow generation. Both models are applied to synthesizing the monthly inflow data for two reservoir sites in the Bharathapuzha basin, Kerala state, South India, namely the Mangalam and Pothundy reservoirs. Finally, the results of both models are compared and conclusions are presented.

NORMALIZATION OF THE SERIES

The skewness coefficients of the historic data from both stations were 1.40 and 1.56 for Mangalam and Pothundy reservoirs, respectively. Therefore, a transformation to reduce this skewness closer to zero was warranted. In this regard, different transformations were made to the original flows including log normal, Pearson type III, Wilson-Hilferty and square root transformations. The Wilson-Hilferty transformation fits well to the given data on the basis of chi-square statistics for different distributions. The procedure for a Wilson-Hilferty transformation (Salas *et al.*, 1985) is as follows:

Firstly, normalization by log-transformation is applied to the original flows, $x_{v,\tau}$ via:

$$w_{v,\tau} = \log(x_{v,\tau} + cx_{\tau}) \qquad (1)$$

where c is a small increment for positive incrementation of x to cater for zero flows and thereby prevent numerical overflows when logarithms are taken, x_{τ} is the mean for month τ, and v denotes the year.

Standardization is then obtained by:

$$y_{v,\tau} = \frac{w_{v,\tau} - w_{\tau}}{\sigma_{\tau}} \qquad (2)$$

where the resulting $y_{v,\tau}$ will have zero mean and unit variance, and w_τ and σ_τ are the mean and standard deviation of $w_{v,\tau}$.

Finally, the modified Wilson-Hilferty transformation (Kirby, 1972), which is valid for any value of skewness coefficient, is given by:

$$z_{v,\tau} = \frac{6}{\gamma_\tau}\left\{\left[\frac{\gamma_\tau y'_{v,\tau}}{2}+1\right]^{1/3} - 1\right\} + \frac{\gamma_\tau}{6} \tag{3}$$

where $y'_{v,\tau}$ satisfies the expression:

$$y'_{v,\tau} = \begin{cases} \max[y_{v,\tau}, -2/\gamma_\tau] & \text{if } \gamma_\tau > 0 \\ \min[y_{v,\tau}, -2/\gamma_\tau] & \text{if } \gamma_\tau < 0 \end{cases} \tag{4}$$

where $y_{v,\tau}$ is the log-transformed standardized variable and γ_τ is the coefficient of skewness of the original series.

NEURAL NETWORKS

An artificial neural network (ANN) is a computing system made up of a highly interconnected set of simple information processing elements, analogous to a neuron, called units. The neuron collects inputs from both a single and multiple sources and produces output in accordance with a predetermined nonlinear function. A neural network model is created by interconnecting many of these neurons in a known configuration. The primary elements characterizing the neural network are the distributed representation of information, local operations and nonlinear processing.

The area addressed by ANN techniques includes pattern matching, optimization, data compression and function optimization. French *et al.* (1992) discuss the advantageous characteristics of the neural network approach to problem solving. These include:

(1) application of a neural network does not require *a priori* knowledge of the underlying process;

(2) one may not recognize all of the existing complex relationships between various aspects of the process under investigation;

(3) a standard optimization approach or statistical model provides a solution only when allowed to run to completion whereas a neural network always converges to an optimal (or suboptimal) solution and need not run to any prespecified solution condition; and

(4) neither constraints nor an *a priori* solution structure is necessarily assumed or strictly enforced in the ANN development.

The learning process or training forms the interconnection between neurons and is accomplished by using known inputs and outputs, and presenting these to the ANN in some ordered manner. The strength of these interconnections is adjusted using an error convergence technique so that a desired output will be produced for a known input pattern.

BACK PROPAGATION ALGORITHM

The training process by which a neural network determines the appropriate connection strengths is described in the following section. Many training procedures are discussed in the literature (Lippman, 1987). Error back propagation (Lippman, 1987; Pao, 1990) was used in this research. The processing units in back propagation neural networks are arranged in layers. The method is generally an iterative nonlinear optimization approach using a gradient descent search method. Error back propagation provides a feed forward neural network, giving the capacity to capture and represent relationships between patterns in a given data sample.

A neural network has an input layer, a hidden layer and an output layer. Each layer is made up of several nodes, and layers are interconnected by sets of correlation weights. The pattern of connectivity and the number of processing units in each layer may vary within some constraints. No communication is permitted between the processing units within a layer, but the processing units in each layer may send their output to the processing units in the succeeding layers. The nodes receive input either from the initial inputs or from the interconnections.

Error back propagation involves two phases: a feed forward phase in which the external input information at the input nodes is propagated forward to compute the output information signal at the output unit, and a backward phase in which modifications to the connection strengths are made based on the differences between the computed and observed information signals at the output units.

At the beginning of a training process, the connection strengths are assigned random values. The learning algorithm modifies the strength in each iteration until the successful completion of the training. When the iterative process has converged, the collection of connection strengths captures and stores the knowledge and the information present in the examples used in the training process. Such a trained neural network is then ready to be used. When presented with a new input pattern, a feed forward network computation results in an output pattern which is the result of the generalization and synthesis of what the ANN has learned and stored in its connection strengths.

The neural network employed in the present study possessed a three-layer learning network consisting of an input layer, a hidden layer and an output layer, as shown in Fig. 1. A portrayal of a unit and its function is given in Fig. 2 with N data input patterns, each having a set of input values, x_i, $i = 1, \ldots, k$ at the input nodes with output values, T_n, $n = 1, \ldots, m$ at the output nodes. The input values, x_i are multiplied by the first interconnection weights, w_{ij}, $j = 1, \ldots, h$ at the hidden nodes, and the products are summed over the index, i, and become the inputs to the hidden layers i.e.:

$$H_j = \sum_{i=1}^{k} w_{ij} x_i \quad j = 1,\ldots,h \tag{5}$$

where H_j is the input to the jth hidden node, w_{ij} is the connection weight from the ith processing element (PE) to the jth PE. Each hidden node is transformed through a sigmoid function to produce a hidden node output, HO_j and is defined as:

$$HO_j = f(H_j) = \frac{1}{1+\exp[-(H_j+\theta_j]} \tag{6}$$

where H_j is the input to the node, $f(H_j)$ is the node output, and θ_j is a threshold or bias. The threshold, θ_j will be learned in the same manner as the weights. The output, HO_j serves as the input to succeeding layer and this procedure is continued until the output layer is reached. This is referred to as forward activation flow. The input to the m output nodes, IO_n, is expressed as:

$$IO_n = \sum_{j=1}^{h} w_{jn} HO_j \quad n = 1,\ldots,m \tag{7}$$

These input values are processed through the sigmoidal function defined earlier to give the neural network output values, O_n. The subsequent weight adoption or learning process is accomplished by the back propagation learning algorithm.

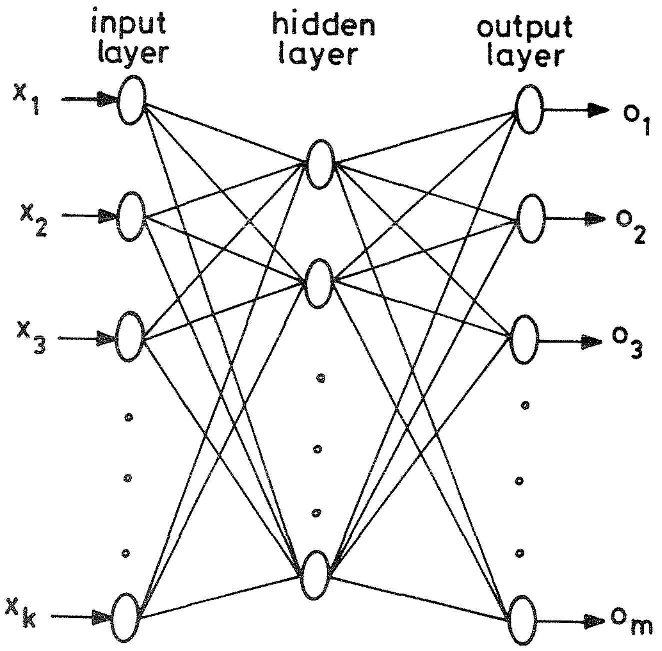
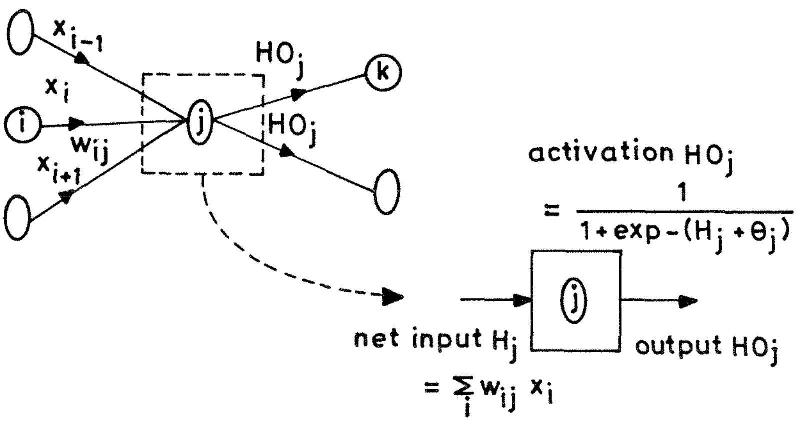


Fig. 1 Structure of a three layer neural network model.



x_{i-1}
x_i
w_{ij}
x_{i+1}
i
j
k
HO_j
HO_j
activation $HO_j = \frac{1}{1+\exp-(H_j+\theta_j)}$
net input $H_j = \sum_i w_{ij}\, x_i$
output HO_j

Fig. 2 Portrayal of a unit and its function.

The O_n at the output layer will not be the same as the target value, T_n. For each input pattern, the sum of the squares of error, e_p for the pth input pattern is:

$$e_p = \frac{1}{2}\sum_{n=1}^{m}(T_n - O_n)^2 \tag{8}$$

and the average system error or mean square error (MSE), E, for all input patterns is:

$$E = \frac{1}{2N}\sum_{p=1}^{N}\sum_{n=1}^{m}(T_{pn} - O_{pn})^2 \tag{9}$$

where T_{pn} is the target value, T_n, for the pth pattern and O_{pn} is the neural network output value, O_n for the pth pattern.

The aim of the back propagation learning algorithm is to minimize iteratively the average squared error between values of the output, O_{pn}, at the output layer and the correct pattern, T_{pn}, provided by a teaching input using the gradient descent approach. This is accomplished by first computing the gradient (δn) for each processing element on the output layer:

$$\delta_n = O_n(1 - O_n)(T_n - O_n) \tag{10}$$

where T_n is the correct teaching value for the output unit, n, and O_n is the neural network output. The error gradient (δ_j) is then recursively determined for the hidden layers by computing the weighted sum of the errors at the previous layer:

$$\delta_j = HO_j(1 - HO_j)\sum_{n=1}^{m}\delta_n w_{jn} \tag{11}$$

The errors are propagated backwards one layer, and the same procedure is recursively applied until the input layer is reached. The error gradients are then used to update the network weights:

$$\Delta w_{ji}(r) = \eta \delta_j x_i \tag{12}$$

$$\Delta w_{ji}(r+1) = w_{ji}(r) + \Delta w_{ji}(r) \tag{13}$$

where r is the iteration number and η is the learning rate which provides the step size during the gradient descent. The learning rate governs the rate at which the weights are allowed to change at any given presentation. Higher learning rates speed the convergence process, but can result in non-convergence. Slower learning rates produce more reliable results at the expense of increased training time. Generally, to assure rapid convergence, large step sizes which do not lead to oscillations are used. The weight change after the nth data presentation is:

$$\Delta w_{ji}(r) = \eta \delta_j x_i + \alpha \Delta w_{ji}(r-1) \tag{14}$$

where α, a momentum rate term, is used to improve the convergence, which determines the effect of previous weight changes on the present change in the weight space.

Procedure for training the network

Training of the neural network is accomplished by providing inputs to the model, computing the output, and adjusting the interconnection weights until the desired output is reached. The error back propagation algorithm is used to train the network, using the mean square error (MSE) over the training samples as the objective function. For training, the data are divided into three parts (Rumelhart *et al.*, 1994). One part is used for training, the second is used for cross-validation and the third part is used for testing.

The architecture of the network in the present study consisted of an input layer, a hidden layer and an output layer. The number of intermediate units was obtained through a trial-and-error procedure. Initially two intermediate units were used and the training process carried out. The error between the value predicted by the network and the value actually observed was then measured and propagated backwards along the feed forward connection. The final error after a given number of training cycles was observed. The number of intermediate units which gave the minimum system error was accepted. During training, the performance of the network was also evaluated on the validation set. As long as the network gave an improvement on the validation, training was continued. If over-fitting occurred, the network at some point began to show poorer performance than that with the validation data. At that point, the training was stopped and the weights which gave the optimal performance on the validation set were selected and tested against the test data set.

Procedure for testing the networks

Some of the historical observations were set apart when building the training data set. These observations were not shown to the network during training. They were used after the training was finished in order to test the network and monitor its performance on test samples in terms of the mean squared error criterion.

STATISTICAL MODELLING PROCEDURES

The multivariate time series modelling comprised four basic stages, namely: selection of the model, identification of the order of the model, parameter estimation, and diagnostic checking for model adequacy. This strategy, formalized by Box & Jenkins (1970) and advocated by Salas *et al.* (1980), is an iterative procedure of model building to ensure satisfactory model fitting and utilization.

Model identification

Consider a multivariate time series of $X_t^{(i)}$, $i = 1, 2, \ldots, n$; $t = 1, \ldots, N$, where n is the number of time series and N is the sample size with normalization of the data done using the Wilson-Hilferty transformation. In the model identification stage, the relationship between two time series may be studied by examining the cross-correlation structure of the series (Salas *et al.*, 1980). The elements of the lag k cross-correlation coefficients, $r_{ij}(k)$, are estimated by:

$$r_{ij}(k) = \frac{\sum_{t=1}^{N-k} [X_t^{(i)} - x_t^{(i)}][X_{t+k}^{(j)} - x_{t+k}^{(j)}]}{\left[\sum_{t=1}^{N-k} (X_t^{(i)} - x_t^{(i)})^2 \sum_{t=1}^{N-k} (X_{t+k}^{(j)} - x_{t+k}^{(j)})^2\right]^{1/2}} \qquad (15)$$

where $x_t^{(i)}$ is the mean of the first $N - k$ values of the series i, and $x_{t+k}^{(j)}$ is the mean of the last $N - k$ values of the series j.

The cross-correlation function of an autoregressive AR(p) model is infinite in extent and the partial autocorrelation function is zero after lag p. The autocorrelation and partial autocorrelation analysis of individual series complements the cross-correlation analysis. The order of the model is determined from the significance test of cross-correlation and partial autocorrelation (Salas *et al.*, 1980).

Parameter estimation

The method of moments and maximum likelihood are among the most popular methods of parameter estimation. The series is normalized by applying an appropriate transformation and the periodic means and variances may be removed by seasonal standardization. A multivariate autoregressive model is selected for the synthesis based on the model identification procedures. The multivariate AR(1) model can be expressed as:

$$Z_t = A_1 Z_{t-1} + B\epsilon_t \tag{16}$$

where Z_t is a vector of $(n*1)$ elements, Z_{it} of a stationary multivariate series; A_1 and B are $(n*n)$ matrices of parameters a_{ij} and b_{ij}; and ϵ_t is an $(n*1)$ vector of independent normally distributed random variables, ϵ_{it} with zero mean and unit variance. The subscripts i and j are indices of sites. The ϵ values are assumed to be uncorrelated in time and space, namely $E(\epsilon_{it}\epsilon_{jt}) = 0$ for $i \neq j$. The moment estimates of the parameters A_1 and B may be obtained from (Salas *et al.*, 1980):

$$A_1 = M_1 M_0 \tag{17}$$

$$BB^T = M_0 - A_1 M_1^T \tag{18}$$

where M_0 and M_1 are the lag 0 and lag 1 cross-correlation matrices.

The extension of the AR(1) model to a multivariate lag two autoregressive model AR(2) is given by:

$$Z_t = A_1 Z_{t-1} + A_2 Z_{t-2} + B\epsilon_t \tag{19}$$

In this model, Z, the inflow data at time t at each site depend on Z at time $t - 1$ at all sites and Z at times $t - 2$ at all sites. A_1, A_2 and B are $(n*n)$ matrices of parameters.

Diagnostic test for model adequacy

Having estimated the model parameters, diagnostic checks are necessary in order to see whether the model has been mis-specified and to search for model improvements. Statistical tests for the independence and normality of residuals are employed. The residuals of the multivariate model are tested for normality considering each series individually using skewness and chi-square tests (Salas *et al.*, 1980). A portmanteau lack-of-fit test is used for checking dependence. An Akaike Information Criterion (AIC) is used to select the best among the competing models. The AIC for an AR(p) model takes the form:

$$\mathrm{AIC}(p) = N\ln(\sigma_\epsilon^2) + 2p \tag{20}$$

where σ_ϵ^2 is the maximum likelihood estimate of the variance and p is the order of the model.

CASE STUDY

The artificial neural network and statistical procedures presented above were applied to modelling the monthly inflows to two reservoir sites, *viz* the Mangalam and Pothundy reservoirs located in the Bharathapuzha basin, Kerala state, India. The monthly inflows to both reservoirs measured during the concurrent period from 1977 to 1990 were used in the present study. The live storage capacities of Mangalam and Pothundy reservoirs are 24.67 Mm^3 and 43.89 Mm^3, respectively.

Application of neural network model

The neural network approach was executed with two testing experiments using the historic inflow series of both the reservoirs. In the first experiment, the training set was prepared by taking the series continuously. In the second experiment, each historical series was divided into 12 monthly input data sets for training. The latter approach gave better results and was used in the present study.

Consider the periodic time series of inflows, $X_{v,\tau}$ and $Y_{v,\tau}$ for both reservoirs where v denotes the year and τ denotes the time interval within the year. From these two series, input data sets for training, cross-validation and testing were prepared. Ten years data were used for training, two years data were used for cross-validation and two years data were used for testing.

For training the network, twelve input layer series were prepared from the historic series for both reservoirs and yielded twelve ANN models. The input layer consisted of four units, i.e. two consecutive inflow values of normalized data from each reservoir. The output layer consisted of two units, i.e. the third consecutive value from each series, prepared as follows. $X_{v,\tau-1}$, $X_{v,\tau-2}$ for the first reservoir and $Y_{v,\tau-1}$, $Y_{v,\tau-2}$ for the second reservoir, $1 \leq v \leq$ N, formed one input layer pattern, and $X_{v,\tau}$ and $Y_{v,\tau}$, the corresponding output layer pattern. If $(\tau - 1) < 1$ then $X_{v,\tau-1}$ was replaced by $X_{v-1,12+\tau-1}$ and if $(\tau - 2) < 1$ then $X_{v,\tau-2}$ was replaced by $X_{v-1,12+\tau-2}$. Similarly twelve input layer patterns and output layer patterns were prepared and normalized separately between 0 and 1 combining each input layer pattern and its output layer pattern. The time series were not transformed to normal and standardized prior to input layer preparation.

The input layer of four units was selected based on the minimum mean square error criterion after trying various number of input units. This indicated that every piece of data was dependent on the previous two data values. The number of intermediate units was obtained through a trial-and-error procedure. Initially a single layer of two intermediate units was used and the training process was carried out. The error between the value computed by the network and the value actually observed was then measured and propagated backwards along the feed forward connection. Neural networks were developed using various hidden layers and hidden nodes. The final error after a given number of training cycles was observed. The number of intermediate units which gave the minimum system error was accepted. The variation of system error for different hidden layers is shown in Fig. 3. The learning rate was set at 0.5 and the momentum rate at 0.9. The number of intermediate nodes and iterations were different for each input layer pattern. The number of intermediate nodes varied from two to seven and the number of iterations varied from 3000 to 6000 for converging to a desired mean squared error and cross-evaluation on the validation set.

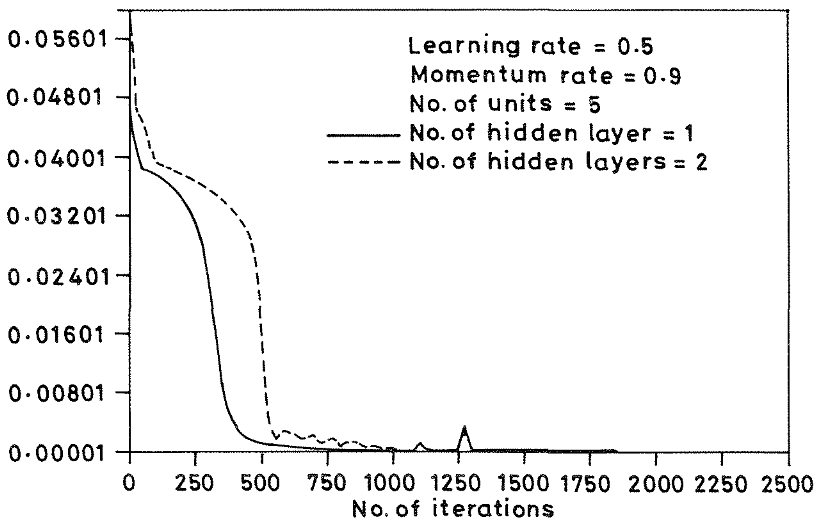


Fig. 3 Variation of system error with number of hidden layers for February.

When training was finished the weights were collected from each training sample to test the network and monitor its performance on test samples in terms of the mean squared error criterion. The mean squared error for the entire training set taken continuously was 1.884 and 0.627 for Mangalam and Pothundy, respectively, and that of the test set was 1.149 and 0.750 for Mangalam and Pothundy, respectively. When the training process was completed, the weights for each interconnection were available, to produce outputs by feeding in the last measured outputs as inputs along with the appropriate antecedent inputs and repeating as desired. Ten series each of fourteen years' inflows were generated using the model.

The time series plot of historical monthly inflows superimposed with simulated inflows using the ANN model are shown in Figs 4(a) and 4(b) for Mangalam and Pothundy, respectively. Comparisons of the autocorrelation function of the historical inflows with that of the simulated series using the ANN model are shown in Figs 5(a) and 5(b) for Mangalam and Pothundy, respectively.

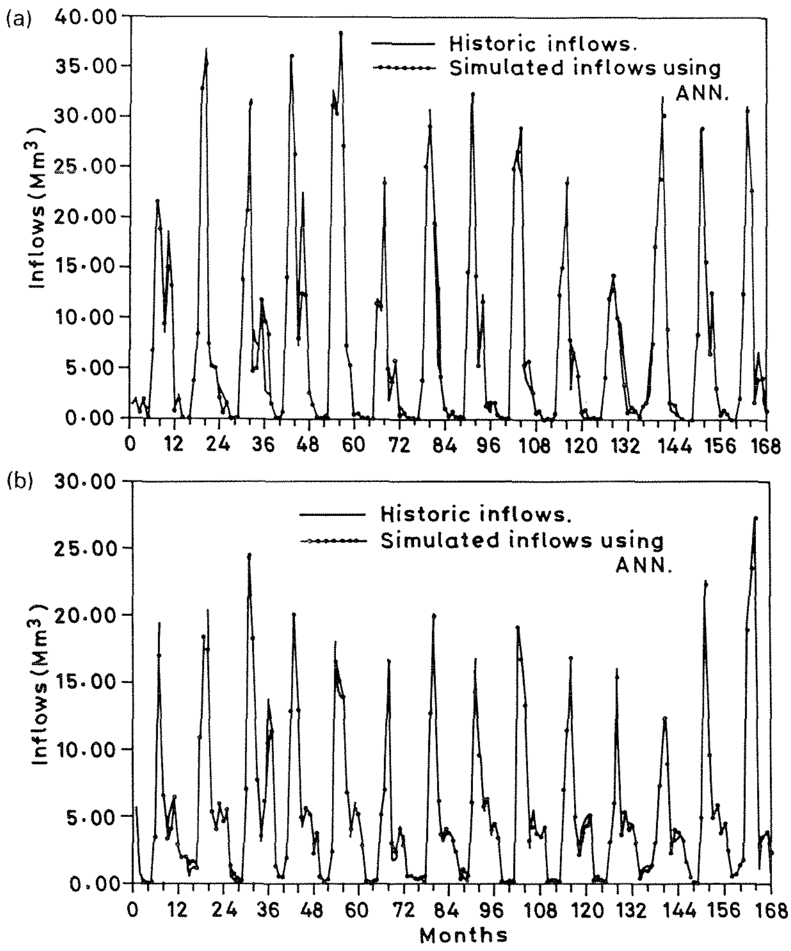


Fig. 4 Time series plot of historic monthly inflows (a) into Mangalam; and (b) into Pothundy, superimposed with simulated series using the ANN model.

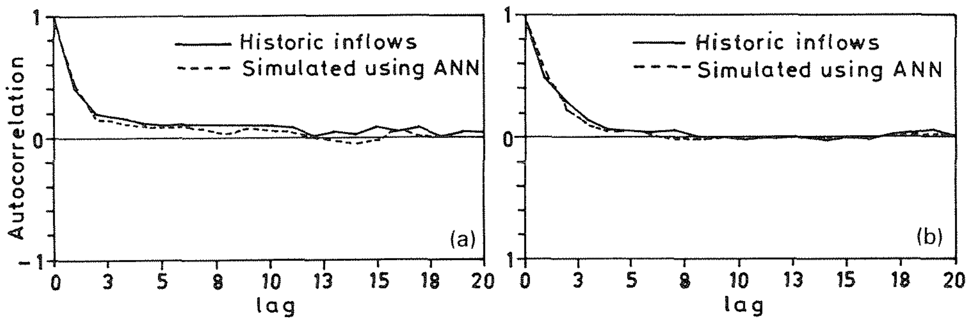


Fig. 5 Comparison of autocorrelation of historic inflows (a) into Mangalam; and (b) into Pothundy, with that of simulated series using the ANN model.

Application of statistical model

The synthesis of inflows was carried out using statistical procedures for the purpose of comparing the neural network results. The autocorrelation function of inflows to Mangalam and Pothundy are shown in Figs 6(a) and 6(b). The partial autocorrelation coefficients of inflows into both reservoirs are shown in Figs 7(a) and 7(b). The cross-correlation matrices of the two series for three lags are given in Table 1. The autocorrelation, partial autocorrelation and cross-correlation analysis suggests the modelling of inflows using multivariate autoregressive models. AR(1) and AR(2) models were applied to generate the series and an inverse path of model fitting used to obtain the original variables. The residual series were tested for independence and normality. The test for normality indicated that the residual series followed a normal distribution. The portmanteau lack-of-fit test for checking independence indicated that the residual series was serially uncorrelated and not cross-correlated. Based on the Akaike information criteria, the AR(2) model was found to be the appropriate model for the synthesis of inflow series.

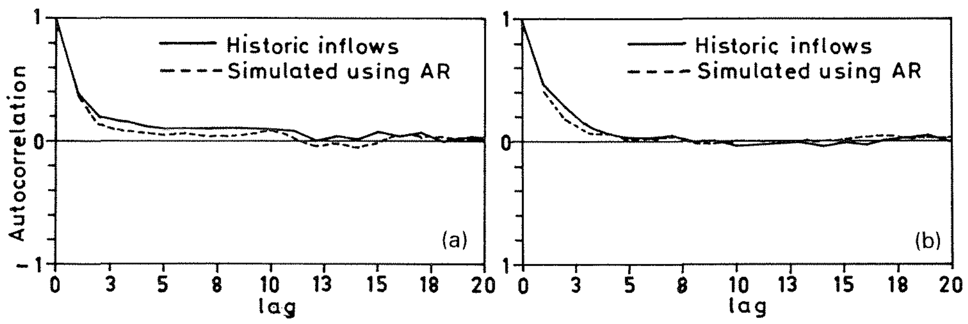


Fig. 6 Comparison of autocorrelation of historic inflows (a) into Mangalam; and (b) into Pothundy, with that of simulated series using the AR model.

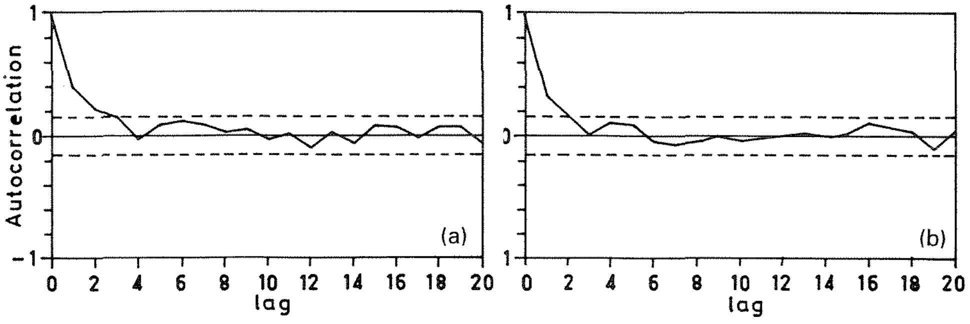


Fig. 7 Partial autocorrelation (PACF) of inflows (a) into Mangalam; and (b) into Pothundy.

Table 1 Cross-correlation matrix

Cross-correlation for lags 0-3		
$\begin{bmatrix} 1.000 & 0.463 \\ 0.463 & 1.000 \end{bmatrix}$	$\begin{bmatrix} 0.399 & 0.261 \\ 0.214 & 0.333 \end{bmatrix}$	$\begin{bmatrix} 0.168 & 0.127 \\ 0.047 & 0.107 \end{bmatrix}$

The time series plot of historic monthly inflows superimposed with simulated inflows using the AR(2) model is shown in Figs 8(a) and 8(b) for Mangalam and Pothundy, respectively. Ten series each of fourteen years inflows were generated using the model.

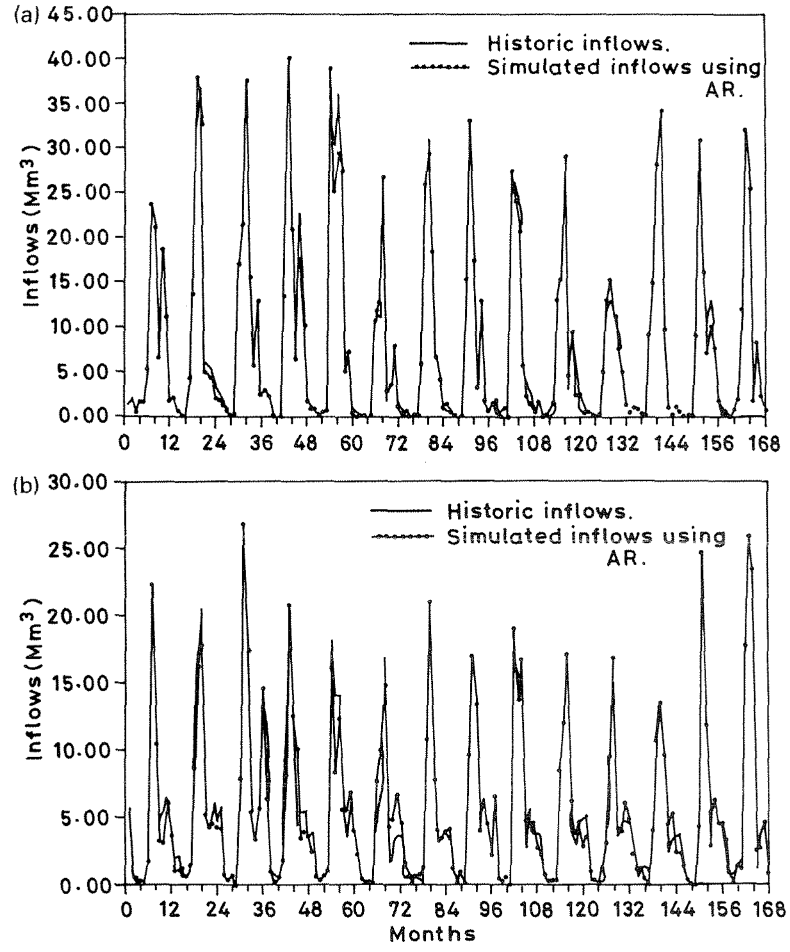


Fig. 8 Time series plot of historic monthly inflows (a) into Mangalam; and (b) into Pothundy, superimposed with simulated series using AR model.

The Storage-Yield-Reliability relationship for different percentages of average inflows was computed for checking the model adequacy. Since the reservoirs are already built, only the reliability which the existing reservoirs can yield was determined. Yield-Reliability relationships were computed using both the historic series and the generated series.

COMPARATIVE ANALYSIS OF ANN MODEL AND STATISTICAL MODEL

The results obtained using the neural networks compared well with those obtained using an alternative statistical model indicating that the network exhibits a potential for a competitive alternative tool for the analysis of multivariate time series. Tables 2 and 4 show the comparison of monthly means and variances of the series generated using both the neural network and the statistical model for Mangalam and Pothundy reservoirs, respectively. Tables 3 and 5 give the comparison of the monthly skewness of the generated series using the neural network and the statistical model for Mangalam and Pothundy, respectively. The higher order moments of variance and skewness showed rather high variability. The neural network model preserved the mean of the generated series well in comparison to that of the statistical model. The Yield-Reliability relationships computed using the historic and generated flows using both the models are shown in Figs 9(a) and 9(b) for Mangalam and Pothundy, respectively.

Table 2 Comparison of mean and variance of historic and generated inflow series for Mangalam reservoir

	Mean			Standard deviation		
Month	Historic data	ANN model	AR model	Historic data	ANN model	AR model
January	1.254	1.397	1.412	0.793	0.697	0.793
February	0.641	0.643	0.962	0.659	0.653	0.972
March	0.200	0.210	0.178	0.201	0.141	0.145
April	0.413	0.392	0.641	0.560	0.599	0.545
May	0.898	0.744	1.199	1.157	1.254	1.059
June	12.528	12.507	13.190	8.051	7.671	9.523
July	24.275	24.416	23.131	8.220	8.125	7.654
August	24.452	24.091	24.432	7.287	7.415	6.488
September	10.630	10.324	11.323	9.540	8.821	9.005
October	9.644	9.993	7.998	5.501	5.112	4.515
November	5.633	5.787	4.929	4.010	3.894	3.236
December	1.477	1.778	1.880	1.011	1.368	0.864

Table 3 Comparison of skewness of historical and generated inflow series for Mangalam reservoir

	Skewness		
Month	Historic data	ANN model	AR model
January	0.734	0.898	0.809
February	1.280	1.233	1.170
March	1.917	1.649	1.572
April	2.336	2.323	2.088
May	2.120	2.048	2.194
June	1.516	1.415	1.553
July	0.342	0.345	0.269
August	0.338	0.238	0.223
September	1.626	1.608	1.694
October	1.392	1.102	1.485
November	1.180	1.160	0.921
December	0.938	0.902	0.867

Table 4 Comparison of mean and variance of historic and generated series for Pothundy reservoir

	Mean			Standard deviation		
Month	Historic data	ANN model	AR model	Historic data	ANN model	AR model
January	4.570	4.123	5.464	2.644	2.550	4.265
February	0.910	0.808	1.088	0.741	0.681	1.189
March	0.503	0.443	0.578	0.365	0.279	0.483
April	0.531	0.489	0.901	0.455	0.407	1.329
May	0.836	0.848	0.752	0.930	0.984	0.592
June	8.519	8.594	9.668	6.331	6.294	5.098
July	15.543	15.616	15.691	5.997	5.967	8.445
August	15.615	15.148	14.457	4.363	4.596	4.042
September	5.140	5.134	5.124	2.782	3.496	2.357
October	4.180	4.029	4.072	1.413	1.471	2.074
November	4.683	4.268	4.627	1.061	0.967	1.546
December	4.479	4.633	4.486	2.941	2.847	2.059

Table 5 Comparison of skewness of historic and generated inflow series for Pothundy reservoir

	Skewness		
Month	Historic data	ANN model	AR model
January	2.160	2.398	2.287
February	1.253	1.609	1.378
March	1.640	1.621	1.431
April	1.454	0.352	2.923
May	1.467	1.383	1.676
June	0.827	0.705	0.476
July	0.274	0.114	0.222
August	0.598	0.674	0.663
September	0.928	1.021	0.491
October	0.142	0.216	0.262
November	0.353	0.287	0.425
December	2.777	2.957	2.425

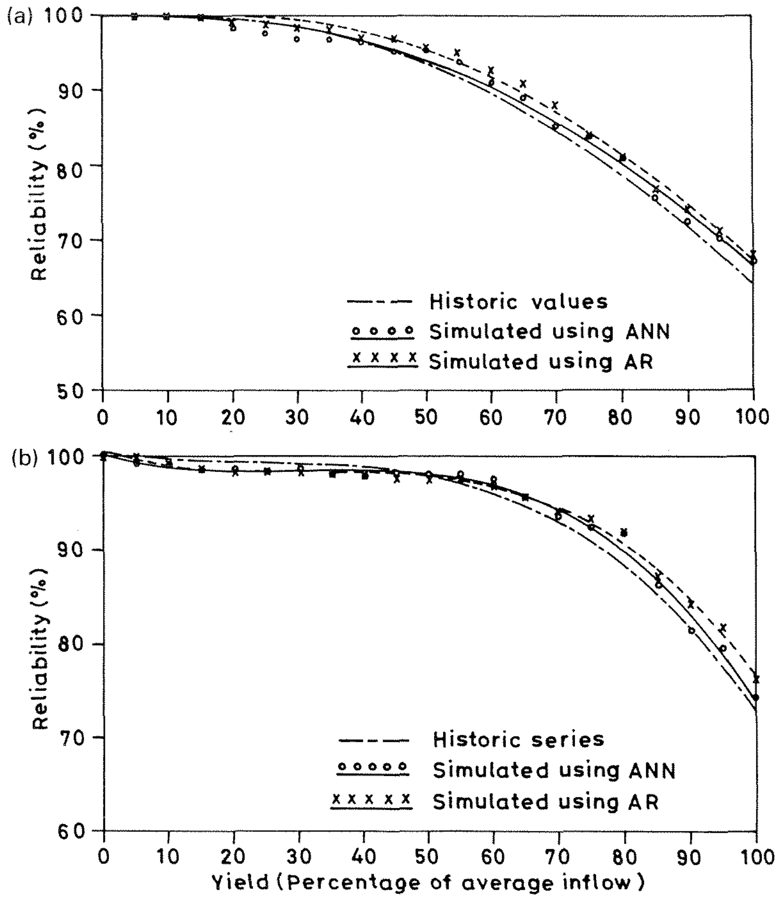


Fig. 9 Yield-Reliability relationships for inflows (a) into Mangalam; and (b) into Pothundy.

CONCLUSIONS

In this paper, the use of an artificial neural network for the synthesis of inflows was investigated. An appropriate multivariate AR(2) model was identified and used to generate inflow values for the purpose of comparing the results obtained using an artificial neural network model. Real world observations were used to train and test the predictive power of feed forward neural networks. A remarkable success was achieved in training the networks to learn the patterns in the time series without normalizing and standardizing prior to input. The neural network model provided a very good fit with the data, as the mean square errors were very low for the training samples. The results obtained using a neural network model compared well in the mean with those obtained using an autoregressive model. This indicates that a neural network offers a viable alternative for multivariate modelling of water resources time series.

Acknowledgements The authors gratefully appreciate the anonymous reviewers for their valuable comments and suggestions resulting into the modification of the paper in the present form.

REFERENCES

Box, G. E. P. & Jenkins, G. M. (1970) *Time Series Analysis Forecasting and Control.* Holden-Day Press, San Francisco, California, USA.

Chakraborty, K., Mehrotra, K., Mohan, C.K. & Ranka, S. (1992) Forecasting the behaviour of multivariate time series using neural networks. *Neural Networks* **5**, 961-970.

French, M. N., Krajewski, W. F. & Cuykendall, R. R. (1992) Rainfall forecasting in space and time using a neural network. *J. Hydrol.* **137**, 1-31.

Kirby, W. (1972) Computer-oriented Wilson-Hilferty transformation that preserves the first three moments and the lower bound of the Pearson type 3 distribution. *Wat. Resour. Res.* **8**(5), 1251-1254.

Lippman, R. P. (1987) An introduction to computing with neural networks. *IEEE ASSP Magazine*, 4-22.

Panu, U. S. & Unny, T. E. (1980) Extension and application of feature prediction model for synthesis of hydrologic records. *Wat. Resour. Res.* **16**(1), 77-96.

Pao, Yoh-Han (1990) *Adaptive Pattern Recognition and Neural Networks.* Addison-Wesley, New York, USA.

Rumelhart, D. E., Widrow, B. & Lehr, M. A. (1994) The basic ideas in neural networks. *Commun. ACM* **37**(3), 87-92.

Salas, J. D., Deulleur, J. W., Yevjevich, V. & Lane, W. L. (1980) *Applied Modelling of Hydrologic Time Series.* Water Resources Publ., Littleton, Colorado, USA.

Salas, J. D., Tabios, Q. B. & Bartolini, P. (1985) Approaches to multivariate modelling of water resources time series. *Wat. Resour. Bull.* **21**(4), 683-708.

Stedinger, J. R. & Taylor, M. R. (1982) Synthetic streamflow generation - model verification and validation. *Wat. Resour. Res.* **18**(4), 909-918.

Valencia, R. D. & Schaake, J. C. (1973) Disaggregation processes in stochastic hydrology. *Wat. Resour. Res.* **9** (3), 580-585.

Received 24 March 1994; accepted 10 September 1994