ELSEVIER

Journal of Hydrology 199 (1997) 272–294

Journal of Hydrology

Application of a neural network technique to rainfall–runoff modelling

Asaad Y. Shamseldin

Department of Engineering Hydrology, University College Galway, Galway, Ireland

Received 1 February 1996; revised 1 November 1996; accepted 18 November 1996

Abstract

This paper deals with the application of a neural network technique in the context of rainfall–runoff modelling. The chosen form of neural network is tested using different types of input information, namely, rainfall, historical seasonal and nearest neighbour information. Using the data of six catchments, the technique is applied for four different input scenarios in each of which some or all of these input types are used. The performance of the technique is compared with those of models that utilize similar input information, namely, the simple linear model (SLM), the seasonally based linear perturbation model (LPM) and the nearest neighbour linear perturbation model (NNLPM). The results suggest that the neural network shows considerable promise in the context of rainfall–runoff modelling but, like all such models, has variable results. © 1997 Elsevier Science B.V.

Keywords: Neural network technique; Rainfall-runoff modelling; Simple linear model (SLM); Linear perturbation model (LPM)

1. Introduction

This paper is concerned with the use of the neural network technique for modelling the rainfall–runoff transformation, the objective being to evaluate the suitability of the technique by comparison with some traditional rainfall–runoff models.

One of the characteristics of the neural networks is that they provide a computational or mathematical technique which is powerful for modelling systems where the explicit form of the relationship between the variables involved is unknown (Fausett, 1994, p. 2). The technique has been used for solving various problems in different branches of science and engineering. In the hydrological context, it has been used for rainfall forecasting in space and time (French et al., 1992) and for river flow prediction (Karunanithi et al., 1994).

In the present work, neural networks are envisaged as non-linear input–output models

PII S0022-1694(96)03330-6

of the rainfall–runoff process. A neural network can have multiple inputs and likewise multiple outputs depending on the nature of the problem. In the present context, they are used as multiple-inputs single-output models. As with other empirical models, neural networks must be calibrated using actual observed input–output data and the optimum values of the parameters are found by search methods.

There are various forms of neural networks (Lippmann, 1987). However, for the present study, a multi-layer feedforward network is selected because of its versatility in function approximation (Nielsen, 1991, p. 131). This form of neural network is described in detail in Section 2 of this paper.

The application of neural networks, like any other rainfall–runoff model, requires input information. In this paper, different types of input information required for the operation of three selected rainfall–runoff models are used. The three selected models are the simple linear model (SLM) (Nash and Foley, 1982), the seasonally based linear perturbation model (LPM) (Nash and Barsi, 1983) and the nearest neighbour linear perturbation model (NNLPM) (Shamseldin and O'Connor, 1996).

In the SLM, it is assumed the rainfall and the discharge are related by a linear time-invariant system. Thus, the only input used in obtaining a forecast is the rainfall record over the recent past. This model was never intended as a serious operational rainfall–runoff model but rather to provide a base-line for the comparison of the performance of more substantive models.

The seasonally based LPM applies the same form of mathematical relationship, as used in the SLM, to relate the perturbations of the rainfall and the discharge from their respective seasonal expectations.

The NNLPM utilizes information of those historical rainfall–runoff events for which the rainfall time series segments are found to be close to the most recently observed rainfall time series segment. Such close rainfall time-series segments constitute the so-called nearest neighbours. The NNLPM model is based on the hypothesis that the departures of a rainfall time series segment from the mean rainfall time series segment of its nearest neighbours (i.e. close rainfall time series segments) are related by a linear time-invariant system to the corresponding departures of the observed discharge segment from the mean discharge hydrograph segment associated with these nearest neighbours.

Table 1 provides a summary of the various types of input information required for the operation of these three models. The models are also briefly reviewed in Section 5 of the paper.

The chosen form of neural network is applied in four different input scenarios. In each of the first three scenarios, the neural network utilizes input data corresponding in form to one of the three selected models, as shown in Table 1. However, in the fourth input scenario, which is the most complex, it uses the input data of both the LPM and the NNLPM, i.e. all three forms of input data listed in Table 1.

The inclusion of the fourth input scenario was indicated by the results of the study of Shamseldin and O'Connor (1996) which showed that, in the case of non-seasonal catchments, the NNLPM provided a more reliable indicator of the discharge than the seasonally based LPM. Thus, it seems reasonable to expect that if the seasonal information (as exploited by the LPM) and the nearest neighbours information, (as utilized by the

Table 1
Summary of the input data required for the operation of the SLM, the LPM and the NNLPM

Model	Input data type		
	Most recent rainfall	Seasonal information of rainfall and discharge	Nearest neighbours information
SLM	Yes	No	No
LPM	Yes	Yes	No
NNLPM	Yes	No	Yes

NNLPM) in addition to the most recent rainfall information, could be integrated into a single model structure, such a model might be expected to perform reasonably well for both seasonal and non-seasonal catchments.

The chosen form of neural network is tested on the synchronous rainfall and discharge data of six catchments from different geographical locations in the world. These data are those used by Shamseldin and O'Connor (1996) when testing the NNLPM. The data are the areal daily averaged values of rainfall and the daily averaged values of discharge, all expressed as equivalent depths of water over the catchment. Table 2 provides a summary description of these six catchments. The table shows the catchments areas, the names of the countries in which these catchments are located, the average rainfalls, the average discharges and the lengths of the calibration and the verification periods.

The performance of the neural networks in the four different input scenarios are compared with those of the three selected models (i.e. the SLM, the seasonally based LPM and the NNLPM). The comparison of model performances are assessed in terms of the R^2 model efficiency criterion (Nash and Sutcliffe, 1970). This criterion is briefly described in Appendix A.

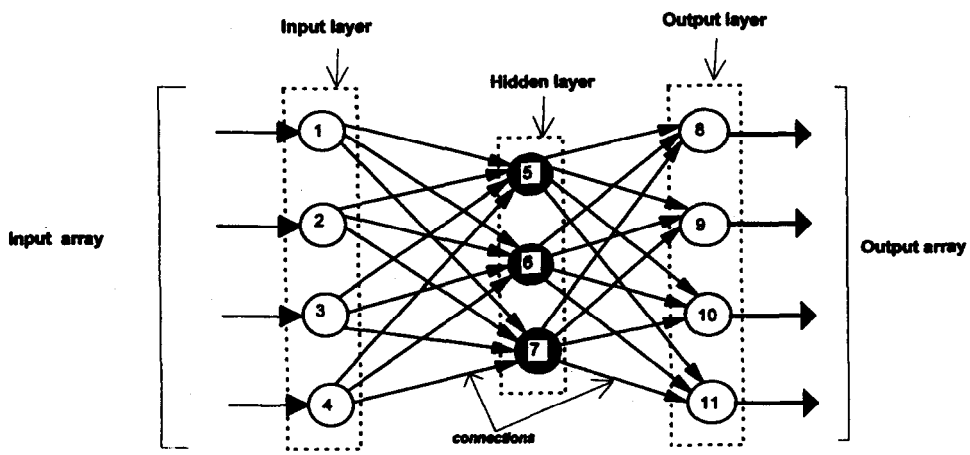

Fig. 1. A multi-layer feedforward neural network with one hidden layer.

Table 2
Summary description of the six test catchments

Catchments	Country	Area	Average rainfall (mm day^{-1})		Average discharge (mm day^{-1})		Full years	Calib. period starting date	Full years	Verif. period starting date
			Calib.	Verif.	Calib.	Verif.				
Sunkosi-1	Nepal	18000	4.65	4.26	3.63	4.41	6	1 January 1975	2	1 January 1981
Yanbian	China	2350	3.28	3.36	2.55	2.64	6	1 January 1978	2	1 January 1984
Shiquan-3	China	3092	2.30	2.47	0.98	0.81	6	1 January 1973	2	1 January 1979
Brosna	Ireland	1207	2.20	2.52	0.98	1.22	8	1 January 1969	2	1 January 1977
Bird Creek	USA	2344	2.67	2.36	0.61	0.31	6	1 October 1955	2	1 October 1961
Wolombi Brook	Australia	1580	2.06	2.03	0.30	0.18	4	1 January 1963	1	1 January 1967

2. The multi-layer feedforward neural network

A neural network is made up of a number of computational elements (represented by circles in Fig. 1). These elements are usually known as neurons, each of which is connected to other neurons. Each neuron can receive an array of inputs and produces a single output. The output of a neuron can either be a final network output or otherwise be transmitted through the neuron output connection paths to contribute to the input array of other neurons. For the example shown in Fig. 1, the neuron labelled 6 receives inputs from those labelled 1, 2, 3 and 4 while its output becomes a component of the input array to each of those labelled 8, 9, 10 and 11.

The transformation of the inputs to output, in the case of each neuron, is defined by a mathematical function known as the neuron transfer function. The transformation within a single neuron is relatively simple; the complexity of the neural network system is generally achieved by the interaction of several neurons.

In the case of the multi-layer feedforward neural network, the neurons are arranged in a series of layers. A layer is usually a group of neurons each of which have the same pattern of connections to the neurons in the other layers (or layer).

The layers forming the multi-layer feedforward neural network are the input (external) layer, the output (external) layer and usually also a series of intermediate layers between the input and the output layers which are generally known as the hidden layers. Each type of layer has a different role in the overall operation of the network.

Fig. 1 shows an example of a three-layer feedforward neural network, containing only one hidden layer, which is the primary focus of the present study. The figure indicates that the neurons in the input layer (i.e. those left unshaded in Fig. 1) are connected only to those (shown dark-shaded in the figure) in the single hidden layer. The neurons in the hidden layer are connected to those of both the input layer and the output layer, while those of the output layer have no connections other than to those of the hidden layer. There are no connections between the neurons within any one of the three layers.

In general, the input layer of neurons receives an external input array. Each input neuron receives a single component element of this array. In contrast with that of the neurons of all other layers, the transfer function for all of the neurons in the input layer is the identity function, i.e. the output of an input layer neuron is equal to its external input (Fausett, 1994, p. 12). The output of each input layer neuron then becomes a component of the input array of each of the neurons in the next (i.e. following) layer to which it is connected. Thus, the function of the input layer is essentially to relay all of the external inputs to each of the neurons of the next layer.

A hidden layer of neurons may be defined as one without having direct connection paths to external inputs or outputs. Hidden layers add a degree of flexibility to the performance of the neural network and to the internal representation of the problem under consideration that, in most cases, considerably enhances the capability of the network to deal robustly and efficiently with inherently complex non-linear relations (Medsker, 1994, p. 12). A neuron of a hidden layer receives inputs from the neurons of the previous layer (input or hidden), while its output contributes to the input array of each of the neurons in the following layer, which in general may be either another hidden layer or the output layer of the network.

Each of the neurons of the output layer maps the inputs received from the neurons of the preceding layer to an output. The outputs of the neurons of this layer constitute the final network output array that represents the response of the network to the external input array. The neurons in the hidden and output layers transform their respective inputs to outputs through two separate stages.

Firstly, for each neuron, each of its inputs (i.e. each of the outputs of the neurons of the preceding layer connected to it) is multiplied by its corresponding weight, i.e. the weight assigned to each input connection path to that neuron, and the total sum of these products plus a constant term yields the neuron net input Y_{net}, that is

$$Y_{net} = \sum_{i=1}^{N} Y_i w_i + w_0 \tag{1}$$

where N is the total number of neurons in the preceding layer, Y_i is the neuron input received from the ith neuron in the preceding layer, w_i is the connection weight assigned to the path linking the neuron to that ith neuron and w_0 is the neuron threshold value. The neuron threshold value provides the means of adding a constant value to the sum term ($\sum_{i=1}^{N} Y_i w_i$) which can be used to scale this term into a useful range (Maren, 1990) that would generally enhance the flexibility of network. The inclusion of the term w_0 in Eq. (1) is analogous to that of considering an intercept in the context of linear regression.

Secondly, the neuron input Y_{net} is transformed to the neuron output Y_{out} by the application of the selected neuron transfer function, which is usually a non-linear function, that is

$$Y_{out} = f(Y_{net}) = f\left(\sum_{i=1}^{N} Y_i w_i + w_0\right) \tag{2}$$

where $f(\ )$ denotes the selected neuron transfer function.

The connection weights w_i reflect the relative importance of each input to the neuron. Moreover, they may act as normalizing or scaling factors in the case where the external inputs of the network have different measurement units and physical nature.

Generally, the same form of neuron transfer function is used for all of the neurons of the hidden and of the output layers. Various types of transfer functions have been applied (see Masters, 1993, pp. 81–82; Fausett, 1994, pp. 17–19). However, one of the most popular transfer functions used in neural network studies (Blum, 1992, p. 39), which is that also used in the present study for the neurons in the hidden and output layers, is the logistic function of the form

$$Y_{out} = f(Y_{net}) = \frac{1}{1 + e^{-Y_{net}}} \tag{3}$$

The logistic function has an S shape and its range varies between 0 and 1 (see Fig. 2). The use of such logistic functions introduces non-linearity in the operation of the neural network thereby underpinning and enhancing their ability to model non-linear processes.

Thus, in the case of multi-layer feedforward neural networks the information flows, as indicated in Fig. 1, only in one direction, i.e. successively from the input stage through the intermediate stages to the output stage, because there are only feedforward connections involved. On the other hand, a network is defined as a feedback network if the output of any neuron is fed back into the network as an input to other neurons in a preceding layer,

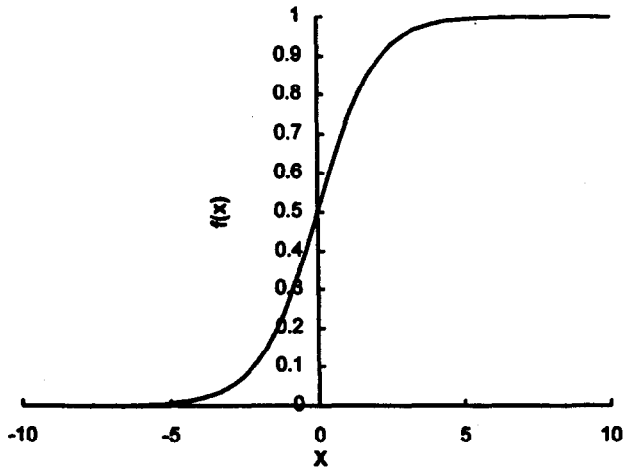

Fig. 2. Schematic diagram of the logistic function.

i.e. there are feedback connections in the network structure (cf. Tagliarini et al., 1991). Only the multi-layer feedforward network with a single hidden layer is considered for the present paper and such a network constitutes the model to be calibrated and verified for rainfall–runoff modelling using the data of the six selected catchments.

In practical applications, the design of the multi-layer neural network structure requires the specification of many factors, such as the number of neurons in the different layers, the number of hidden layers, and the type of neuron transfer function to be used for all neurons (except of course for those of the input layer which all use the identity transformation). The number of neurons in the input and output layers is determined by the number of elements in the external input array and output array of the network, respectively (each element in an array being assigned to one neuron). The number of hidden layers as well as the number of hidden layer neurons are generally initially unspecified. The choice of the appropriate number of hidden neurons is crucial for the successful application of the neural networks. The issue of selection of these numbers is discussed in Section 4 of the present paper.

3. Calibration (training) and verification of the multi-layer neural network

After selecting a particular network structure, the next step is to estimate the connection weights (w_i) and the different neuron threshold values w_0.

The connection weights and the neuron threshold values are normally estimated by a procedure which is usually referred to as training (Hammerstrom, 1993). This is simply a procedure by which known inputs are applied and connection weights and neuron threshold values are adjusted according to a predetermined method so that it simulates the corresponding desired known outputs as closely as possible (i.e. the optimum performance of the network is achieved). Henceforth, in the present paper, the training of the

neural network is referred to as its calibration, as this is the standard term used for this procedure in hydrological modelling.

The estimated weights and the different neuron threshold values, as obtained by calibration, can be perceived as the parameter values of the neural network model. These parameters preserve what has been extracted from the data set applied in the calibration of the model and they represent the information which may be used with new input information in solving a particular problem.

A widely used algorithm for the calibration of the neural networks, in the case where samples of the network inputs and outputs are available, is the backpropagation algorithm (Werbos, 1990). This algorithm is simply a gradient descent method for function optimization. In the case of the present study, the function to be optimized is the sum of squares of errors of the estimated outputs of the network. The gradient decent method utilizes the first order partial derivatives of the function in searching for the optimum parameter set.

However, in the present study, the more complex conjugate gradient algorithm (see Press et al., 1989) is chosen instead for parameter estimation as it is generally faster and more efficient than the backpropagation algorithm (Johansson et al., 1990; Masters, 1993, p. 100). The practice adopted for calibration and verification is to divide the available data into two independent sets, i.e. split sampling (cf. Masters, 1993, p. 10). The first data set is used for calibration and the second data set, which is normally about one third of the total available data, is used for validation or verification (Hammerstrom, 1993).

4. The selection of the number of hidden neurons and the number of the hidden layers

The use of one hidden layer is generally recommended at least in preliminary studies. Naturally, the use of more than one substantially increases the number of parameters to be estimated. Such an increase in the number of the parameters may slow the calibration process (Masters, 1993, pp. 174–180) without substantially improving the efficiency of the network. A single hidden layer was adopted in the present study.

The determination of the appropriate number of neurons in the hidden layer is important for the success of the neural network, since it greatly enhances the performance of the neural network, i.e. the network efficiency is sensitive to this number. If the hidden layer has too few neurons i.e. the network is too parsimonious in its use of parameters, then the performance of the neural network may deteriorate below that of the appropriate number. On the other hand, if the hidden layer has too many neurons, then there are too many parameters and there is a danger of over-fitting the calibration data set, i.e. of fitting the noise in the calibration data set (Masters, 1993, pp. 174–180) with no significant improvement in calibration or even a drop in efficiency in the verification period.

The best strategy for selecting the appropriate number of neurons in the hidden layer is experiment, i.e. a trial and error procedure. An appropriate number of neurons can be found by calibrating the network and evaluating its performance over a range of different increasing values of number of hidden layer neurons (Hammerstrom, 1993) in order to obtain near-maximum efficiency with as few neurons as necessary, i.e. to be sensibly parsimonious in the number of parameters required.

4.1. The rescaling of the output

Excluding the neurons in the input layer of the network, all of which have the identity transfer function, most of the neuron transfer functions, which are used to convert the net input Y_{net} to its output Y_{out} in both the hidden layer(s) and in the output layer, have a bounded output range. For example, the logistic function, defined by Eq. (3), has an output which is bounded in the range [0, 1] (cf. Fig. 2).

As the actual external outputs of the network are generally outside the bounded range of the neuron transfer function, it becomes necessary to rescale or transform the actual (i.e. observed) external outputs in such a way as to be within the bounded output range so that a proper direct comparison can then be made between the network estimated outputs and the external rescaled actual outputs. In practice, the effective rescaling range is generally less than the full bounded range of the transfer function of the neurons of the output layer. This effective rescaling range is normally adopted in order to facilitate the calibration process, in particular when derivative-based optimization techniques are used for calibration of the network.

In the present work, since the logistic function is selected as the transfer function of the neurons in the hidden layer and of the single neuron in the output layer, the observed discharge series q_i is rescaled according to the following linear transformation;

$$qs_i = 0.1 + 0.75\left(\frac{q_i}{q_{max}}\right) \tag{4}$$

where qs_i is the rescaled observed discharge series, q_{max} being the maximum observed discharge in the calibration period. Adopting such a linear transformation, the rescaled discharge series qs_i in the calibration period is bounded in the range [0.1, 0.85]. This bounded range is adopted mainly because the derivatives of the logistic function outside this range are generally small giving rise to problems with derivative based algorithms such as the conjugate algorithm which is adopted in the present study.

5. A brief review of the three selected rainfall–runoff models

5.1. The simple linear model (SLM)

The SLM postulates a linear time invariant relationship between the rainfall X_i and the discharge q_i. It was introduced by Nash and Foley (1982) as a naive model against which the performance of more sophisticated rainfall–runoff models could be compared.

In discrete form, embodying a model fit error term e_i, it may be expressed as (Kachroo and Liang, 1992)

$$q_i = G \sum_{j=i-m+1}^{i} X_j h_{i-j+1} + e_i \tag{5}$$

where G is the gain factor (i.e. amplification) of the system, m is the memory length of the system and h_i is a set of discrete pulse response ordinates such that

$$\sum_{i=1}^{m} h_i = 1 \tag{6}$$

5.2. The linear perturbation model (LPM)

The LPM was originally proposed by Nash and Barsi (1983). This models exploits the seasonal information of the observed rainfall and the discharge time series.

In the LPM, it is presumed that if, during a particular year, the actual observed rainfall is identical to its seasonal expectation then the corresponding observed actual discharge hydrograph would also be identical to its seasonal expectation. However, when this is not the case, the departures of the actual observed rainfall and of the discharge time series from their respective seasonal expectations are assumed to be related by a linear time invariant subsystem. The preceding assumptions can be mathematically written as

$$q_i' = G \sum_{j=i-m+1}^{i} X_j' h_{i-j+1} + e_i \tag{7}$$

where X'_i and q'_i are the rainfall departures and the corresponding discharge departures from their seasonal expectations, respectively.

In the LPM, the estimation of the seasonal expectations is indispensable. When both the rainfall and the discharge time series are measured at daily intervals, their seasonal expectations are found by estimating the average daily rainfall and discharge, for each day of the year, over the entire calibration period. The resulting series is then smoothed using discrete Fourier series (Kachroo et al., 1992) to yield the data series having a period of 1 year from which the perturbations are calculated.

5.3. The nearest neighbour linear perturbation model (NNLPM)

This model was developed by Shamseldin and O'Connor (1996) and integrates both the concept of the nearest neighbours and the concept of perturbations from a mean value analogous to that used in a seasonal context in the LPM of Nash and Barsi (1983).

In the NNLPM, it is assumed that if the most recent rainfall time series segment agrees exactly with the mean of its historical nearest neighbours, i.e. with the mean of similar rainfall time series segments which occurred in the past, then it would produce an output segment which agrees with the mean of the corresponding output segments of these nearest rainfall segments (i.e. nearest neighbours). Where the most recent rainfall time series segment departs from the mean of its nearest neighbours, which is generally the case, it is further postulated that these departures are related by a linear time-invariant system to the corresponding departures of the observed discharge segment from the mean discharge hydrograph segment associated with these nearest neighbours. The most recent rainfall time series segment is expressed as the rainfall vector $\boldsymbol{x}_i$, consisting of m discrete rainfall observations, which may be written as;

$$\boldsymbol{x}_i = (X_i, X_{i-1}, X_{i-2}, \ldots, X_{i-m+1})^T \tag{8}$$

where T denotes the transpose of the vector.

The overall operation of the NNLPM, incorporating a model fit error term e_i, may be expressed as;

$$q_i - {}_n\bar{q}_i = G \sum_{j=i-m+1}^{i} (X_j - {}_n\overline{X}_j) h_{i-j+1} + e_i \tag{9}$$

where ${}_n\bar{X}_i$ and ${}_n\bar{q}_i$ denote the time series of the mean rainfall time series segment and that of the corresponding mean discharge hydrograph segment of the nearest neighbours, respectively.

The main prerequisite for the operation of the NNLPM is prior knowledge of the nearest neighbours (giving ${}_n\bar{X}_i$ and ${}_n\bar{q}_i$) which is obtained from searching the available historical records. The nearness of the rainfall time series segments is assessed in terms of a rainfall index series RI_i defined by

$$RI_i = \sum_{j=i-m+1}^{i} X_j h_{i-j+1} \tag{10}$$

Thus, each rainfall time series segment, i.e. vector $\boldsymbol{x}_i$, has its own corresponding rainfall index RI_i. In order to curtail the laborious effort required in searching for the nearest neighbours at each time step, the available historical rainfall time series segments and their corresponding discharges are first ranked and divided into bands according to their RI_i values. Each of these RI bands has, approximately, an equal number of RI_i values, except for the one containing all the zero RI_i values as additional values. Thus, the nearest neighbours of the most recent rainfall time series segment are those of the band to which its corresponding rainfall index value belongs (cf. Shamseldin and O'Connor, 1996).

6. Application of the multi-layer neural network, the SLM, the LPM, the NNLPM and a comparison of the results

6.1. The specification of the inputs to the neural network, for the four input scenarios

As indicated earlier, the neural network is tested under four different external input scenarios, in each of which it utilizes certain types (i.e. different combinations) of input information. In all of these scenarios, the input information includes the most recent rainfall values combined with (or without) either the seasonal rainfall and discharge time series information or the rainfall and the discharge information of the nearest neighbours or all three of these inputs.

In the first scenario, the input information corresponds to the m most recent rainfall measurements, i.e. to the rainfall vector $\boldsymbol{x}_i$ defined by Eq. (8). This input information is the same as that utilized by the SLM.

If each of the elements of the vector $\boldsymbol{x}_i$ was considered as an external input element to the network, then each element would be assigned to one neuron in the input layer. However, if the memory length (m) is large then this would require a large input layer to accommodate these input elements and, as a consequence, it requires a large number of parameters (i.e. connection weights w_i and neuron threshold values w_0) to be estimated, which is by no means a simple task. To circumvent this problem, the rainfall index RI_i is considered as the only external input factor to the network, as RI_i characterizes the rainfall vector. This rainfall index is a weighted sum of the m most recent rainfall values defined by Eq. (10).

Thus, while the neural network itself does not incorporate storage elements, storage is implicitly accounted for by the use of the rainfall index RI_i as an input to the network.

In the second scenario, the input information includes the seasonal information of the discharge and the rainfall time series in addition to the most recent observed rainfall values. The inputs are the seasonal expectation of the discharge ${}_sq_i$ and the corresponding seasonal rainfall index ${}_sRI_i$ as well as the current rainfall index RI_i. These seasonal expectations are estimated by the same procedure as that used in the case of the seasonally based LPM (see Kachroo et al., 1992). Thus, in this scenario, the neural network utilizes similar input information to that of the LPM although the structures of the neural network and the LPM are quite different.

In the third scenario, the input information for the network is the same as that used by the NNLPM. This input information corresponds to the most recent rainfall and the nearest neighbours information, i.e. the rainfall and the discharge information of rainfall time series segments similar to the most recent segment. The nearest neighbours are obtained by the same procedure used in the case of the NNLPM. For this scenario, the three inputs to the network are the rainfall index RI_i, the mean nearest neighbour discharge ${}_n\bar{q}_i$ and the corresponding rainfall index ${}_n\overline{RI}_i$ of the nearest neighbours which can be obtained from the mean rainfall index and discharge of the RI band to which the current value of RI_i belongs. In the present study, the number of the RI bands used is 250.

Finally, in the fourth scenario, there are five input types, namely, the rainfall index RI_i, its seasonal expectation ${}_sRI_i$, the mean nearest neighbour rainfall index ${}_n\overline{RI}_i$, the discharge ${}_n\bar{q}_i$ of the nearest neighbours and the seasonal discharge expectation ${}_sq_i$. All of these inputs are obtained in the manner adopted in the case of the three previous scenarios.

The four different external input scenarios outlined above are schematically shown in Fig. 3.

In the present work, the ordinates of the pulse response in Eqs. (5)–(10) are obtained in parametric form, using the well known gamma function model which was introduced by Nash (1957) as a general equation of the instantaneous unit hydrograph. The impulse response of the gamma function model $H(t)$ is given by

$$H(t)=\frac{1}{k\Gamma(n)}\left(\frac{t}{k}\right)^{n-1}e^{-t/k} \tag{11}$$

where $\Gamma(n)$ is the gamma function of the variable n given by the following improper integral

$$\Gamma(n)=\int_0^\infty e^{-y}y^{n-1}dy \tag{12}$$

n and k being the parameters of the gamma function model of the h_i series. Moreover, in this model form, n and the lag product nk are usually considered as its parameters rather than n and k (cf. Kachroo and Liang, 1992).

6.2. *The procedure for calibration of the neural network*

In all of the four external input scenarios described in Section 6.1, once the structure of the multi-layer feedforward neural network had been chosen (i.e. the number of the layers

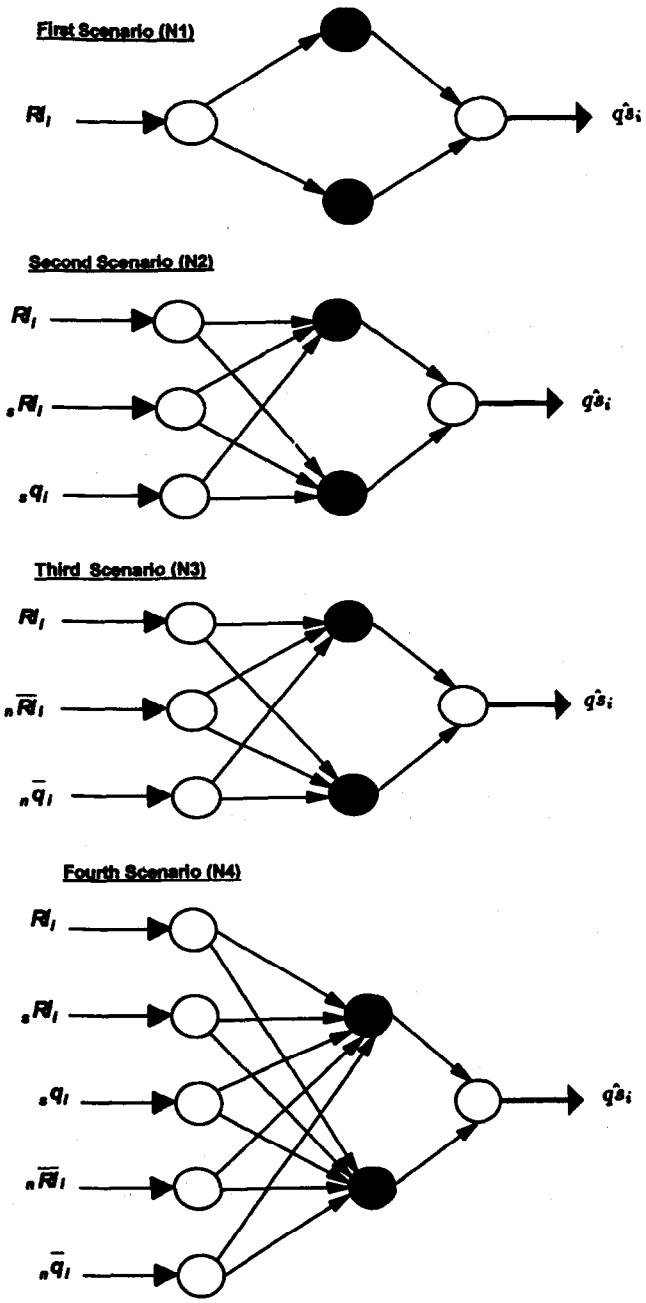

Fig. 3. Schematic diagram for the multi-layer feedforward neural networks for the four external input scenarios.

and the number of neurons in each layer), the application of the neural network requires the estimation of the parameter values of the gamma function model (n and nk) as well as the parameter values of the network. In the present paper, an iterative calibration procedure is adopted for the estimation of these parameters (gamma and network) which can be briefly summarized as follows.

1. With suitable initial values of the parameters of the gamma function, calculate the inputs to the input layer of the network.
2. Choose initial values of the parameters of the network (i.e. the connection weights and the neuron threshold values) and using the conjugate gradient algorithm (see Press et al., 1989), find estimates of the parameter values of the network in such a way as to minimize the objective function, i.e. the sum of the squares of the differences between the network outputs and the rescaled observed discharges.
3. Taking the parameters of the network, estimated in Step 2, as being fixed (i.e. constants), use the simplex method (Nelder and Mead, 1965) to find suitable values of the parameters n and nk of the gamma function by minimizing the same objective function as that used in Step 2.
4. With these values of n and nk, estimated in Step 3, as new initial values, return to Step 1.
5. Repeat Steps 1–4 until there is no improvement in the overall performance of the neural network with the performance viewed in terms of the objective function specified in Step 2.

6.3. Discussion of the results obtained for six catchments using the four input scenarios for the neural network and comparisons with the other models

In the case of the SLM, the LPM and the NNLPM, the ordinates of the pulse responses of these models were also obtained in a parametric form using the gamma function model, for the six chosen test catchments (Table 2). Thus, the application of these three models requires the estimation of the parameters of the gamma function (i.e. n and nk) in addition to the gain factor G.

The optimum parameter values (i.e. n, nk and G) for these three models are estimated by minimizing the sum of the squares of the errors between the observed discharges and the model estimated discharges and are shown in Table 3.

In the case of the SLM and the LPM, these values were obtained from a draft report compiled by Liang (1993), while in the case of the NNLPM, the results were obtained from Shamseldin and O'Connor (1996) as this latter model was also tested on these six catchments.

In Table 3, the optimum parameter values of the gamma function (i.e. n and nk) used in conjunction with the neural networks are presented, in which the four external input scenarios described in Section 6.1 and shown in Fig. 3 are referred to as N1, N2, N3 and N4, respectively. The values of these parameters were obtained by the calibration procedure outlined in Section 6.2 and using the data of the entire calibration period as shown in Table 2. In all of the external input scenarios, there is only one hidden layer containing only two neurons. This number of two hidden neurons is chosen because

Table 3
Optimum parameters of the SLM, the LPM, the NNLPM and the gamma function, used in conjunction with the N1, N2, N3 and N4 neural network forms

Catchment	Model	n	nk	G
Sunkosi-1	SLM	1.466	12.619	0.820
	LPM	0.481	9.812	0.572
	NNLPM	0.861	25.013	1.000
	N1	1.005	16.207	–
	N2	0.502	9.029	–
	N3	1.171	27.676	–
	N4	0.566	7.5095	–
Yanbian	SLM	0.956	8.645	0.755
	LPM	1.044	6.264	0.748
	NNLPM	0.899	17.06	0.827
	N1	0.909	10.685	–
	N2	0.939	8.576	–
	N3	0.916	18.308	–
	N4	1.007	8.026	–
Shiquan-3	SLM	2.271	1.488	0.524
	LPM	1.353	1.950	0.646
	NNLPM	1.707	1.628	0.638
	N1	1.225	1.938	–
	N2	1.533	1.675	–
	N3	1.016	1.903	–
	N4	1.002	2.024	–
Brosna	SLM	1.074	9.953	0.457
	LPM	1.029	9.483	0.357
	NNLPM	0.984	15.608	0.469
	N1	1.045	11.872	–
	N2	1.000	10.000	–
	N3	1.036	10.092	–
	N4	1.050	10.164	–
Bird Creek	SLM	4.128	1.810	0.371
	LPM	2.323	2.102	0.459
	NNLPM	3.469	2.014	0.977
	N1	2.110	2.261	–
	N2	2.133	2.421	–
	N3	2.287	2.419	–
	N4	1.222	3.338	–
Wolombi Brook	SLM	1.027	1.355	0.246
	LPM	0.542	2.923	0.331
	NNLPM	0.556	5.640	1.000
	N1	0.560	6.929	–
	N2	0.690	2.016	–
	N3	0.585	6.505	–
	N4	0.619	5.004	–

preliminary tests carried out by the present author indicated that there is no substantial improvement in the overall performance by increasing the number beyond two in most of the cases considered. Table 4 shows the number of the network parameters for the four scenarios. Fig. 4(a) and (b) show comparisons of the observed and estimated discharge

hydrographs by the three models and the four neural network forms for the Yanbian catchment for 1 complete year.

Table 5 provides the initial mean square of errors (MSE_0), the mean square of errors (MSE), the R^2 model efficiency values and also the R^2 rank for the SLM, the seasonally based LPM, the NNLPM, as well as for the N1, the N2, the N3 and the N4, i.e. for the four different scenarios of application of the neural network.

Examination of Table 5 indicates that, in the calibration period, the SLM has, as would be expected, the worst performance, i.e. the lowest R^2 values, in all of the six catchments. However, in the verification period, it is the worst only in the case of the Bird Creek catchment, where it has a negative R^2 value of –45.75%, which indicates that its verification performance on Bird Creek is even worse than that of the naive 'no model' case for which the forecast is always taken as the mean discharge of the calibration period. These results generally confirm that the rather naive postulate of the SLM is unsatisfactory in the case of these six test catchments.

More interestingly, Table 5 reveals that, in calibration, one or other form of neural network, i.e. either the N1, the N2, the N3 or the N4, has the highest R^2 value, except in the case of the Bird Creek catchment where its R^2 value of 82.36% is not notably different from that of 83.58% for the NNLPM. However, in the verification period, one or other form of the neural network has the highest R^2 in four out of the six test catchments, specifically, the Yanbian, the Brosna, the Bird Creek and the Wolombi Brook catchments. These results reveal that the neural network can be an efficient and, indeed, an alternative tool for river flow forecasting in simulation (design) mode, i.e. without any updating, as considered in this paper.

Comparison of the results of the N1 form of the neural network with the SLM, both of which utilize only the current and the most recent rainfall measurements, shows that the N1 has a better performance than the SLM in all of the test catchments, for both the calibration and the verification periods, except in the case of the verification period of the Brosna catchment, where the R^2 value of 43.81% of the N1 neural network form is just marginally lower than that of 44.59% of the SLM.

Likewise, when the results of the N2 neural network form are compared with the model which uses the same type of input information, i.e. the seasonally based LPM, it is conspicuous that, in the calibration period, the N2 has a better performance than the LPM in all of the six catchments. Similarly, in the verification period, the N2 form the neural network is better than the LPM except for two catchments, namely, the Sunkosi-1 and the Wolombi Brook catchments, where the differences in the R^2 values are not significant. These results indicate that the N2 neural network form is capable of extracting more information from the same input data than the LPM.

Table 4
The number of network parameters of the neural network forms the N1, the N2, the N3 and the N4

	Neural network form			
	N1	N2	N3	N4
No. of network parameters	7	11	11	15

Table 5
Summary of results of the SLM, the LPM, the NNLPM and the four neural network forms, the N1, the N2, the N3 and the N4

Catchments	Models	Calibration				Verification			
		MSE_0	MSE	R^2 (%)	R^2 rank	MSE_0	MSE	R^2 (%)	R^2 rank
Sunkosi-1	SLM	17.914	3.157	82.37	7	23.350	4.248	81.81	6
	LPM		1.439	91.97	3		2.135	90.86	1
	NNLPM		2.027	88.68	5		3.881	83.38	5
	N1		2.540	85.82	6		3.849	83.52	4
	N2		1.340	92.52	2		2.155	90.77	2
	N3		2.019	88.73	4		4.308	81.55	7
	N4		1.073	94.01	1		2.273	90.27	3
Yanbian	SLM	12.161	3.571	70.64	7	12.230	3.445	71.83	5
	LPM		2.277	81.82	3		2.870	76.53	3
	NNLPM		2.589	78.70	5		3.778	69.10	7
	N1		3.140	74.18	6		3.327	72.79	4
	N2		2.056	83.10	2		2.415	80.25	1
	N3		2.546	79.07	4		3.613	70.45	6
	N4		1.623	86.65	1		2.656	78.28	2
Shiquan-3	SLM	8.697	2.521	71.01	7	8.245	4.036	51.05	5
	LPM		2.206	74.64	6		4.195	49.13	7
	NNLPM		1.289	85.18	3		2.390	71.01	1
	N1		1.619	81.38	5		2.717	67.04	2
	N2		1.385	84.07	4		2.872	65.17	3
	N3		1.286	85.21	2		3.520	57.32	4
	N4		1.113	87.20	1		4.080	50.51	6
Brosna	SLM	0.583	0.375	35.68	7	1.148	0.636	44.59	4
	LPM		0.181	68.93	3		0.277	75.88	3
	NNLPM		0.313	46.31	4		0.669	41.72	6
	N1		0.361	38.15	6		0.645	43.81	5
	N2		0.121	79.19	2		0.167	85.46	1
	N3		0.325	44.28	5		0.692	39.64	7
	N4		0.096	83.50	1		0.172	85.03	2
Bird Creek	SLM	7.851	3.226	58.90	7	1.010	1.472	−45.75	7
	LPM		3.012	61.63	6		1.369	−35.50	6
	NNLPM		1.290	83.58	1		0.771	23.60	3
	N1		1.621	79.36	5		0.747	25.98	1
	N2		1.544	80.33	4		1.050	−4.06	5
	N3		1.385	82.36	2		0.886	12.20	4
	N4		1.479	81.17	3		0.749	25.72	2
Wolombi	SLM	3.318	1.830	44.85	7	0.996	1.081	−8.62	5
Brook	LPM		1.709	48.51	6		1.146	−15.05	7
	NNLPM		0.697	78.99	4		0.282	71.65	2
	N1		0.845	74.55	5		0.231	76.80	1
	N2		0.505	84.78	2		1.127	−13.20	6
	N3		0.593	82.13	3		0.479	51.84	3
	N4		0.173	94.80	1		0.678	31.84	4

Similarly, the comparison of the results of the N3 neural network form and the NNLPM, both of which use the nearest neighbours information, signifies that, in calibration, the N3 is better than the NNLPM in four out of the six catchments, namely, the Sunkosi-1, the Yanbian, the Shiquan-3 and the Wolombi Brook catchments. However, in verification, the NNLPM has higher R^2 values than the N3 in five out of the six catchments, i.e. except for the Yanbian catchment.

The incorporation of the seasonal information of the rainfall and the discharge time series in the neural network has generally improved its performance as the N2 form has better R^2 results than those of the N1 form for all of the test catchments in the calibration period. However, in verification the N2 has better performance than the N1 form in three catchments.

However, in the case on the nearest neighbours information, i.e. comparing the results of the N1 and the N3 neural network, the N3 performs better than the N1 form in calibration in all of the six test catchments. However, in verification, this situation is reversed. Surprisingly, also the N1 form performs better than the N3 form, in the case of all catchments.

When the results of the N4 are contrasted with those of the N1, N2 and N3 which use subsets of input information utilized by the N4, it can be inferred, based on the calibration period, that the N4 has the highest R^2 efficiency values for all of the six catchments, except the Bird Creek catchment. In verification, the N4 has R^2 model efficiency values which are in the vicinity of the maximum R^2 among the other three forms in four catchments, namely, Yanbian, Sunkosi-1, Brosna and Bird Creek.

The comparison of the results, for the calibration period, of the N4 form with those of the other three models, i.e. the SLM, the LPM and the NNLPM, shows that the N4 has the highest R^2 model efficiency values in all of the six test catchments, excluding the Bird Creek catchment. Nonetheless, in verification, the N4 has the highest R^2 values in three catchments, namely, the Brosna, the Yanbian and the Bird Creek catchments.

These results show that it is not always to be expected that the N4 would yield the best performance, when compared with that of the SLM, LPM and NNLPM. This may be, perhaps, a reflection of the fact that, in some cases, the information carrying capacity of the data, used in the present study, does not support a more sophisticated model or method.

7. Conclusions

In this study, the multi-layer feedforward neural network is applied in the context of rainfall–runoff modelling on the data of six catchments. It has been tested under four different external input scenarios using the input information required for the operation of three selected rainfall–runoff models, namely, the SLM, the LPM and the NNLPM. In each of the first three scenarios, the input information corresponds to that used by one of these models, while in the fourth scenario it corresponds to those of both the LPM and NNLPM.

The comparison of the results of the neural network forms corresponding to each of the four external input scenarios with those of the SLM, the LPM, and NNLPM shows that, in calibration, one or other form of neural network has substantially higher R^2 model

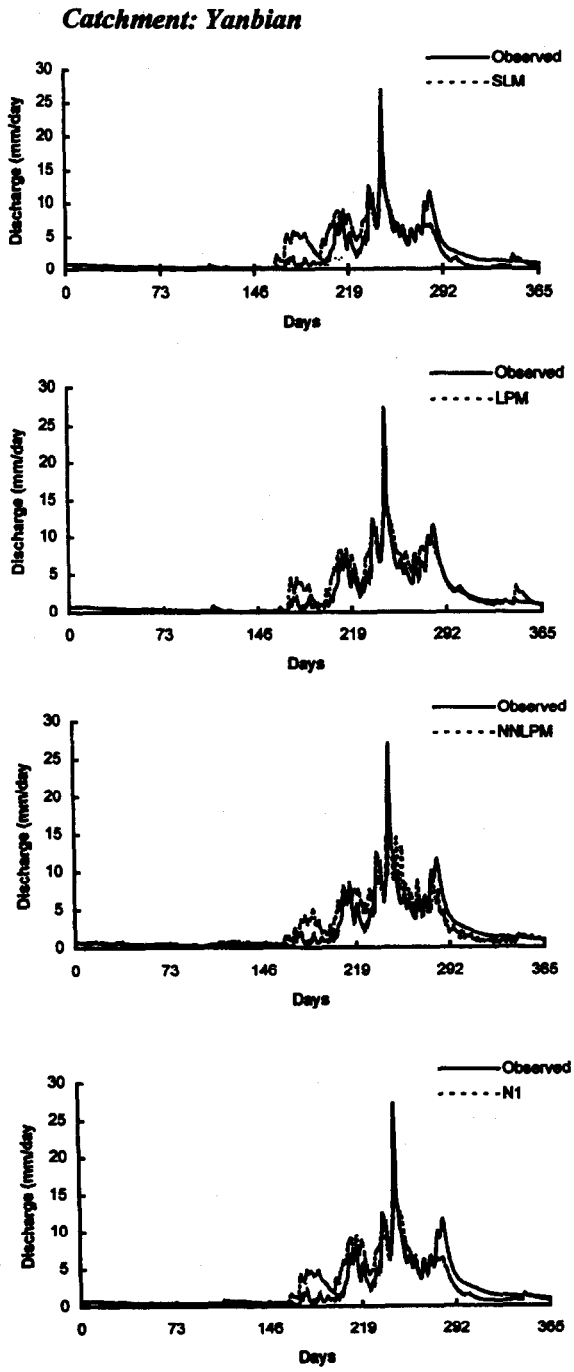

Fig. 4. Comparisons of the observed and the estimated discharge hydrographs of the SLM, LPM, NNPLM, N1, N2, N3 and N4 for the year 1979.

Catchment: Yanbian

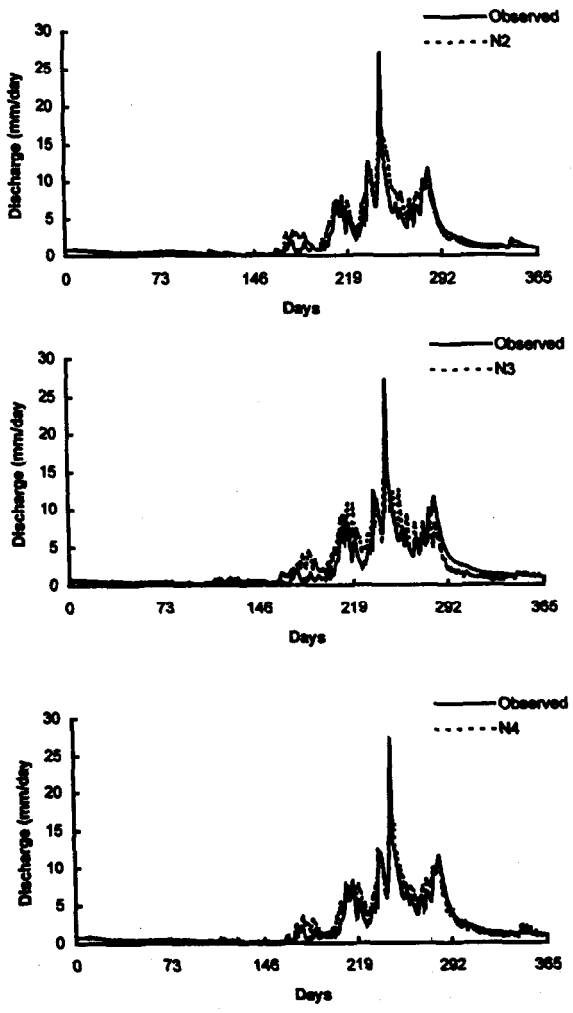

Fig. 4 (*continued*)

efficiency values than these three models in the case of all of the six test catchments. However, in verification, one or other form of neural network is better than these three models in the case of four out of the six catchments (Table 5).

The results obtained using the neural network in the four different external input scenarios, as adopted in the present study as an alternative approach for river flow forecasting generally, suggest that the neural network has considerable potential in this context but, like all such models, has variable results (i.e. good on some catchments but on others

poor). The results of the present study are generally in agreement with similar studies (e.g. Bonafe et al., 1994; Karunanithi et al., 1994) in the sense that the neural networks can provide more accurate discharge forecasts than some of the traditional models.

In future applications of neural networks in the context of hydrological forecasting, consideration should be given to testing alternative network structures, different forms of neuron transfer functions (i.e. other than the logistic function used in the present study) and also alternative rescaling functions of the output (other than that used in Eq. (4)), possibly also with optimization of the parameters of such rescaling functions.

Acknowledgements

The author would like to express his appreciation to the reviewers for their helpful comments.

Appendix A. The model efficiency criterion

In the present study, the criterion used for assessing the performance of the different models is the established R^2 criterion of Nash and Sutcliffe (1970). This criterion may be expressed as

$$R^2 = \frac{F_o - F}{F_o} \tag{A1}$$

where F_o is the initial variance for the discharges about their mean given by

$$F_o = \sum (q_i - \bar{q})^2 \tag{A2}$$

where $\bar{q}$ is the mean discharge and F is the residual model variance, i.e. the sum of the squares of the differences between the observed discharges q_i and the model estimated discharges $\hat{q}_i$, which is

$$F = \sum (q_i - \hat{q}_i)^2 \tag{A3}$$

In Eq. (A1), the initial variance F_o can be visualized as the variance of a primitive (naive) model, having at any time, a constant discharge forecast which is equal to the mean of the observed discharges. Thus, the R^2 criterion is a measure the performance of the substantive model relative to that of the primitive model.

In the application of the R^2 criterion to the calibration period, all of the quantities in Eqs. (A1) to (A3) are estimated within that period. In the case of the verification period, the initial variance F_o is calculated using the mean discharge of the calibration period. The rationale behind this is that the primitive forecast in the verification period is the mean discharge of the calibration period.

Eq. (A2) can alternatively be expressed as

$$R^2 = 1 - \frac{F}{F_o} = 1 - \frac{\text{MSE}}{\text{MSE}_0} \tag{A4}$$

where MSE_0 and MSE are the initial and residual means of the sum of squares of errors, given respectively by

$$MSE_0 = \frac{F_0}{M} \tag{A5}$$

and

$$MSE = \frac{F}{M} \tag{A6}$$

where M is the number of the time periods.

A value of R^2 of 90% indicates a very satisfactory model performance while a value in the range 80–90% indicates a fairly good model. Values of R^2 in the range 60–80% would indicate unsatisfactory model fit (Kachroo, 1986).

References

Blum, A., 1992. Neural Networks in C++; An Object-Oriented Framework for Building Connectionist Systems. John Wiley and Sons., Inc., USA.

Bonafe, A., Galeati, G., Sforna, M., 1994. Neural networks for daily mean flow forecasting. In: Blain, W.R., Katsifarakis, K.L. (Eds.), Hydraulic Engineering Software V Vol. 1, Water Resources and Distribution. Computational Mechanics Publications, Southampton, UK, pp. 131–138.

Fausett, L., 1994. Fundamentals of Neural Networks: Architectures, Algorithms, and Applications. Prentice-Hall International., Inc., USA.

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.

Hammerstrom, D., 1993. Working with neural networks. IEEE Spectrum 46–53.

Johansson, T.M., Dowla, F.U., Goodman, D.M., 1990. Back-propagation learning through time for multi-layer feed-forward neural networks using the conjugate gradient method. Report UCRL-JC-104850, Lawrence Livermore National Laboratory, Livermore, CA.

Kachroo, R.K., 1986. HOMS workshop on river flow forecasting, Nanjing, China. Unpublished internal report, Dept. Eng. Hydrol., University College Galway, Ireland.

Kachroo, R.K., Liang, G.C., 1992. River flow forecasting. Part 2. Algebraic development of linear modelling techniques. J. Hydrol. 133, 17–40.

Kachroo, R.K., Sea, C.H., Warsi, M.S., Jemenez, H., Saxena, R.P., 1992. River flow forecasting. Part 3. Applications of linear techniques in modelling rainfall–runoff transformations. J. Hydrol. 133, 41–97.

Karunanithi, N., Grenney, W.J., Whitley, D., Bovee, K., 1994. Neural networks for river flow prediction. J. Comput. Civ. Eng. 8 (2), 201–220.

Liang, G.C., 1993. Results obtained for the intercomparison study of mathematical models for river flow forecasting. Unpublished internal workshop report, Dept. Eng. Hydrol., University College Galway (UCG), Ireland.

Lippmann, R.P., 1987. An introduction to computing with neural nets. IEEE ASSP Mag. 4, 4–22.

Maren, A.J., 1990. Neural network structures: from follows functions. In: Maren, A.J., Harston, C.T., Pap, R.M. (Eds.), Handbook of Neural Computing and Applications. Academic Press, Inc., p. 51.

Masters, T., 1993. Practical Neural Networks Recipes in C++. Academic Press, Inc., USA.

Medsker, L.R., 1994. Hybrid Neural Network and Expert systems. Kluwer Academic Publishers, USA.

Nash, J.E., 1957. The form of the instantaneous unit hydrograph. Int. Assoc. Sci. Hydrol. Publ. 45 (3), 114–118.

Nash, J.E., Barsi, B.I., 1983. A hybrid model for flow forecasting on large catchments. J. Hydrol. 65, 125–137.

Nash, J.E., Foley, J.J., 1982. Linear models of rainfall–runoff systems. In: Singh, V.P. (Ed.), Rainfall–Runoff Relationship, Proceedings of the International Symposium on Rainfall–Runoff Modelling, May 1981, Mississippi State University, USA. Water Resources Publications, pp. 51–66.

Nash, J.E., Sutcliffe, J.V., 1970. River flow forecasting through conceptual models. Part 1. A discussion of principles. J. Hydrol. 10, 282–290.

Nelder, J.A., Mead, R., 1965. A simplex method for function optimization. Comput. J. 7, 308.

Nielsen, R.H., 1991. Neurocomputing. Addison-Wesley Publishing Company, USA.

Press, W.H., Flannaery, B.P., Teukolsky, S.A., Vetterling, W.T., 1989. Numerical Recipes. Cambridge University Press, New York, pp. 301–306.

Shamseldin, A.Y., O'Connor, K.M., 1996. A nearest neighbours linear perturbation model for river flow forecasting. J. Hydrol. 179, 353–375.

Tagliarini, G.A., Christ, J.F., Page, W.E., 1991. Optimization using neural networks. IEEE Trans. Comput. 40 (12), 1347–1358.

Werbos, P.J., 1990. Backpropogation through time: what it does and how to do it. Proc. IEEE 78 (10), 1550–1560.