HYDROLOGICAL PROCESSES
Hydrol. Process. **23**, 2450–2459 (2009)
Published online 17 June 2009 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/hyp.7352

Auto-configuring radial basis function networks for chaotic time series and flood forecasting

Li-Chiu Chang,[1] Fi-John Chang[2]* and Yuan-Peng Wang[2]

[1] *Department of Water Resources and Environmental Engineering, Tamkang University, Taipei, Taiwan, ROC*
[2] *Department of Bioenvironmental Systems Engineering, National Taiwan University, Taipei, Taiwan, ROC*

Abstract:

The learning strategy of the radial basis function network (RBFN) commonly uses a hybrid learning process to identify the structure and then proceed to search the model parameters, which is a time-consuming procedure. We proposed an evolutionary way to automatically configure the structure of RBFN and search the optimal parameters of the network. The strategy can effectively identify an appropriate structure of the network by the orthogonal least squares algorithm and then systematically search the optimal locations of centres and the widths of their corresponding kernel function by the genetic algorithm. The proposed strategy of auto-configuring RBFN is first testified in predicting the future values of the chaotic Mackey-Glass time series. The results demonstrate the superiority, on both effectiveness and efficiency, of the proposed strategy in predicting the chaotic time series. We then further investigate the model's suitability and reliability in flood forecast. The Lan-Young River in north-east Taiwan is used as a case study, where the hourly river flow of 23 flood events caused by typhoons or storms is used to train and validate the neural networks. The back propagation neural network (BPNN) is also performed for the purpose of comparison. The results demonstrate that the proposed RBFN has much better performance than the BPNN. The RBFN not only provides an efficient way to model the rainfall-runoff process but also gives reliable and precise one-hour and two-hour ahead flood forecasts. Copyright © 2009 John Wiley & Sons, Ltd.

KEY WORDS radial basis function network; genetic algorithm; Mackey-Glass time series; flood forecast

Received 19 September 2008; Accepted 5 April 2009

INTRODUCTION

Accurate time series forecasting is very important in hydrology, especially for flood defense and water resources management. One of the most widely used methods in time series forecasting is the model described by Box and Jenkins (1976). This approach is both simple (parsimony) and yields rational results. It gives simplistic equations that only use several previous items of information to forecast the future. As computer power has increased, other approaches have attempted to find subtle trends and complex relationships in the data. The artificial neural networks (ANNs) have been accepted as a potentially useful tool for modelling complex non-linear systems and widely used for prediction. In the hydrological forecasting context, ANNs have also proven to be an efficient alternative to traditional methods for rainfall forecasting (Hsu *et al*., 1997; Chiang *et al*., 2007; Kumarasiri and Sonnadara, 2008), streamflow forecasting (Chang *et al*., 2004, 2007; Chiang *et al*., 2004; Jeong and Kim, 2005; Tayfur *et al*., 2007; Kisi, 2008; Lin and Chen, 2008), evapotranspiration simulation (Kisi, 2007; Jain *et al*., 2008), groundwater prediction (Krishna *et al*., 2008), reservoir operation (Aminian and Ameri, 2005; Chang *et al*., 2005; Kuo *et al*., 2006) and prediction of water quality (Maier *et al*., 2004; Sahoo *et al*., 2006; Chaves and Kojiri, 2007).

Of the many types of neural networks, one of the most widely used is the radial basis function neural (RBFN) network. One reason is that it can approximate any continuous function to any prescribed degree of accuracy by given sufficient size of the network (Poggio and Girosi, 1990; Ham and Kostanic, 2001). Moreover, training RBFN is usually faster than that of multilayer perceptron networks. The faster learning speed comes from the fact that RBFN has just two layers and each layer can be determined sequentially. The parameters of the hidden neurons are determined by clustering, which is relatively fast, while the output layer weights are determined by a linear least-square algorithm, which is simply solving a linear system. Even though RBFNs popularity is continuously increasing, it is not straightforward to construct an optimal architecture of RBFN to solve a given problem. To continuously increase their popularity and gain farther acceptance, the development of methodologies that can automatically configure RBF networks using a set of input-output patterns is a key element.

The standard RBF training procedures are usually divided into two stages: (i) The centres of hidden nodes are determined first in a self-organizing manner and (ii) the weights that connect the hidden layer with output layer are then searched. This is a time-consuming procedure, since it requires the examination of many different network structures by using a trial and error procedure.

* Correspondence to: Fi-John Chang, Department of Bioenvironmental Systems Engineering, National Taiwan University, Taipei, Taiwan, ROC. E-mail: changfj@ntu.edu.tw

Another drawback is that the centres of hidden nodes are determined by using a clustering approach that is separate from the searching objective. To overcome the time-consuming trial and error procedure, various methods have been proposed to determine the appropriate structure of the network, such as the orthogonal least squares (OLS) algorithm (Chen *et al*., 1991), constructive methods, where the structure of the network is incrementally built (Fritzke, 1994) and pruning methods that start with an initial selection of a large number of hidden units and then reduce the units as the algorithm proceeds (Musavi *et al*., 1992). These methods could substantially decrease the total required training time, but still suffer from the fact that the formulation of the hidden layer is based on local, rather than global, search methods. Therefore, they cannot guarantee the selection of the optimal number of hidden nodes.

An interesting alternative for solving this problem can be found by the recently developed evolutionary computation research area that involves a number of innovative optimization strategies such as genetic algorithms (GAs). Based on the principles of natural selection and evolution, GAs have been successfully utilized for the selection of the optimal structure of ANNs, such as feedforward neural networks (Whitely *et al*., 1990; Arifovic and Gencay, 2001; Boozarjomehry and Svrcek, 2001). Surprisingly, not many applications of GAs have been reported for the development of training algorithms for RBF networks (Billings and Zheng, 1995; Kuncheva, 1997). In this study, we propose a novel evolving strategy: Using the OLS algorithm to determine the appropriate structure of the network, then implementing the GA to directly search the optimizing locations of centres and the widths of their kernel function and using the least square errors (LSE) method to identify the weights of output layer.

The rest of the article first briefly presents the RBF network architecture and training problem, the implemented GA algorithm and auto-configuring the network for a theoretical chaotic series and flood forecasting in the Lan-Yang River. The results of the proposed method are shown and compared with the back propagation neural network (BPNN) for the case of flood forecasting. The concluding remarks are then given.

RBFN THEORY AND CONFIGURATION

The RBF network was introduced into the neural network literature by Broomhead and Lowe (1988). The RBF network is motivated by the locally tuned response observed in biologic neurons. The architecture of the RBFN consists of three layers: (i) an input layer, (ii) a single hidden layer and (iii) an output layer. The input layer is only used to connect the network to its environment. The nodes in the hidden layer are associated with centres, which are vectors with dimension equal to the number of inputs. The output of the RBFN can be calculated according to following equation:

$$y_i = f(x) = \sum_{k=1}^{N} w_{ik}\phi_k(||x - c_k||_2), \quad i = 1, 2, \ldots, m \tag{1}$$

where $x \in \Re^{n\times 1}$ is an input vector, $\phi_k(\cdot)$ is a function, $||\cdot||_2$ denotes the Euclidean norm, w_{ik} are the weights in the output layer, N is the number of neurons in the hidden layer and $c_k \in \Re^{n\times 1}$ are the RBF centres in the input vector space. The centres of c_k are defined points that are assumed to perform an adequate sampling of the input vector space. The functional form of $\phi_k(\cdot)$ is chosen as the Gaussian function or Bell function. The generalized Gaussian functions and bell functions are defined in Equations (2) and (3).

$$\phi(||x - c||) = \exp\left(-\frac{||x - c||^2}{2\sigma^2}\right) \tag{2}$$

$$\phi(||x - c||) = \frac{1}{1 + \left(\frac{||x - c||}{a}\right)^{2b}} \tag{3}$$

where σ is the standard deviation and c is the centre of the Gaussian function. All parameters of bell functions (i.e. a, b and c) have their own physical meaning: c is the centre of the corresponding membership function, a is the distance between the centre point and the crossover point (where the value of function is 0·5) and $-\frac{b}{2a}$ is the slope at the crossover point.

Training a RBF network involves selecting the appropriate number of basis functions and their parameters and estimating the weights and biases. A two-stage procedure has been commonly used for training RBFNs. In the first stage, the number of hidden units (nodes) and parameters governing them are determined by clustering. The second stage involves estimating the weights in the output layer, locations of the RBFN centres and the width of their corresponding kernel functions. Most of the standard RBF training methods require a designer to fix the structure of the network and then proceed with the calculation of the model parameters. A major challenge in the design of the RBFN is the selection of the centres. Based on the Gram-Schmidt orthogonalization procedure, the OLS method offers a systematic way for centre selection and can significantly reduce the size of the RBFN. A detailed description of the algorithm can be found in the ANN text books, such as Ham and Kostanic (2001).

Here, we propose an automatically evolving strategy: (1) using the OLS algorithm to determine the appropriate structure of the network, (2) implementing the GA to search the optimizing locations of centres and the widths of their kernel function and then (3) using the LSE method to identify the weights of output layer. Figure 1 shows the flowchart of proposed methodology for auto-configuring RBFN. A detailed presentation of the OLS method can be found in the textbook by Ham and Kostanic (2001) and a brief description of the GA and its implementation is given in the following sections.

DOI: 10.1002/hyp

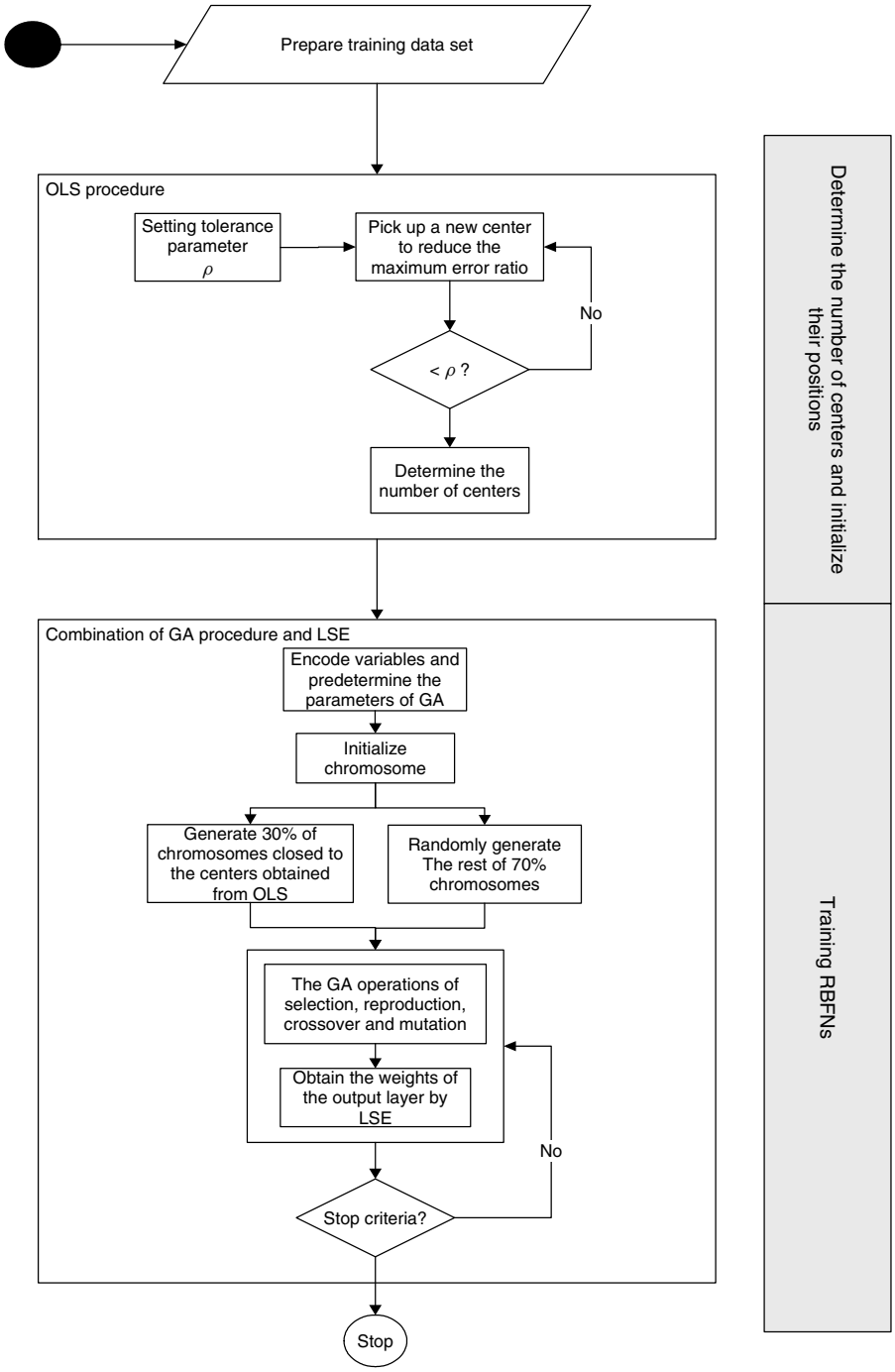

Figure 1. The flowchart of self-configuring radial basis function networks

GA

The GA is similar to Darwinian natural selection, and its primary monograph is Holland's adaptation in natural and artificial systems in 1975 (Holland, 1975). To evaluate the suitability of the derived solution, an objective function is required. The evolution starts from a set of coded solutions and proceeds from generation to generation through genetic operations: reproduction, crossover and mutation. The reproduction process copies parent chromosomes into a tentative new population. The probability of selected chromosomes for the next generation is directly proportional to its fitness value. Roulette Wheel selection is the most common method and is used in our study. The crossover recombines two parent chromosomes to produce offspring new chromosomes for the next generation. To sustain genetic diversity in the population, mutation is also made occasionally with small probability. Through the GAs, an optimal solution can be found and represented by the final winner of the processes.

Implemented GA to RBF network

In the present study, GAs are developed to optimize the centre locations and width of the hidden nodes (kernel functions). If the number of hidden nodes becomes large,

DOI: 10.1002/hyp

a huge number of parameters must be estimated. For example, if the optimum number of hidden node is 22 (in the case of chaotic series), 132 variables (22 nodes × 4 input dimensions in each node + 22 nodes' × 2 (coefficients of kernel function a, b)) must be identified. To make a global search in feasible solution domain more effectively and efficiently, we recommend the following:

(1) Choose 30% of initial solutions closed to the results obtained from OLS, and randomly generate the rest of the 70% initial solutions.
(2) Due to the long chromosomes (132 variables), multi-crossover points should be adopted (5 to 12 crossover points are recommended; we used 11 crossover points.)
(3) The GA parameters used in this study are population (1500), crossover rate (0·8), mutation rate (0·1) and generation (100).

TRAINING AND TESTING THE RBFN BY A CHAOTIC TIME SERIES

The prediction of future values of the chaotic Mackey-Glass time series generated by Equation (4) is recognized as a benchmark problem that has been used and reported by a number of researchers for comparing the learning and generalization ability of different neural networks (Moody and Darken, 1989; Jang, 1993). The Mackey-Glass chaotic time series is described by the following:

$$\frac{dx(t)}{dt} = \frac{0{\cdot}2x(t-\tau)}{1+x^{10}(t-\tau)} - 0{\cdot}1x(t) \qquad (4)$$

When $\tau > 17$, the equation shows chaotic behaviour. The goal of this task is to use known values of the time series up to the point $x = t$ to predict the value at some point in the future, $x = t + 6$ in our case. To demonstrate the goodness of fit of the proposed methods and make the comparison with earlier work fair (Jang, 1993; Gholipour *et al*., 2006), we predicted the $x(t + 6)$ using the input variables $x(t - 18)$, $x(t - 12)$, $x(t - 6)$ and $x(t)$, respectively. To obtain the Mackey-Glass time series $x(t)$, we also used the fourth-order Runge-Kutta method to find the numerical solutions of Equation (4). We extracted 1000 ($t = 118$ to 1117) input-output data pairs. The first 500 pairs were used to train the networks, and the rest of the 500 pairs were then used to test the trained network. The root mean square error (RMSE) is used as the criterion for determining the structure of the built network and for selecting the units in the hidden layer.

Results

The size of network can effectively and efficiently be identified by the OLS algorithm based on a designed error tolerance parameter ρ, which is important for balancing the accuracy and complexity of the network. To demonstrate the influence of the parameter, three different small values of tolerance parameter (i.e. 10^{-5}, 10^{-6}, 10^{-7}) were given. We found (1) the hidden node is 22, if $\rho = 10^{-5}$, (2) the hidden node is 31, if $\rho = 10^{-6}$ and (3) the hidden node is 63, if $\rho = 10^{-7}$. It appears that the number of hidden nodes would increase as the error tolerance parameter decreased. The results of implementing the constructed RBFN by fusing OLS with GA(s) are shown in Table I. It appears that (1) the values of RMSE would decrease as the number of hidden nodes increases; (2) the constructed network could closely fit the actual values and give very small values of RMSE in all sets, i.e. training and testing sets; and (3) implementing the GA into the OLS (i.e. OLS + GA) can significantly improve the accuracy of the RBFN. Using the RBFN with 63 hidden nodes, the time series of real and estimated values and the residual time series in two different stages, i.e. training and testing, are presented in Figure 2. The desired and predicted values for training and testing data sets are essentially the same, where their differences (errors) can only be seen on a finer scale (Figure 2). As the number of hidden nodes is 63, the values of RMSE in both training and testing sets are smaller than those of the adaptive-network-based fuzzy inference systems model (Jang, 1993). The results demonstrate that the RBFN can be effectively constructed and the network can provide reliable and accurate prediction. Considering that the values of RMSE in two stages are all very small, we conclude that the constructed RBFN has captured the essential components of the underlying dynamics and can be implemented to configure the non-linear dynamic system such as flood forecast.

STUDY WATERSHED AND IMPLEMENTING THE PROPOSED RBFN FOR FLOOD FORECAST

To illustrate the practical application of the proposed model, the Lan-Yang River, northeastern Taiwan, is used

Table I. The root mean square error values of radial basis function networks (RBFN; orthogonal least squares − OLS + genetic algorithms − GA) for forecasting a chaotic time series

Method # data	RBFN (22)		RBFN (31)		RBFN (63)		ANFIS[a]
	OLS	OLS + GA	OLS	OLS + GA	OLS	OLS + GA	
Training (500)	0·00923	0·00473	0·00604	0·00390	0·00275	0·00143	0·0016
Testing (500)	0·00933	0·00475	0·00635	0·00387	0·00249	0·00140	0·0015

[a] ANFIS, Adaptive-network-based fuzzy inference systems. These results are obtained from Jang (1993) for the purpose of comparison.

DOI: 10.1002/hyp

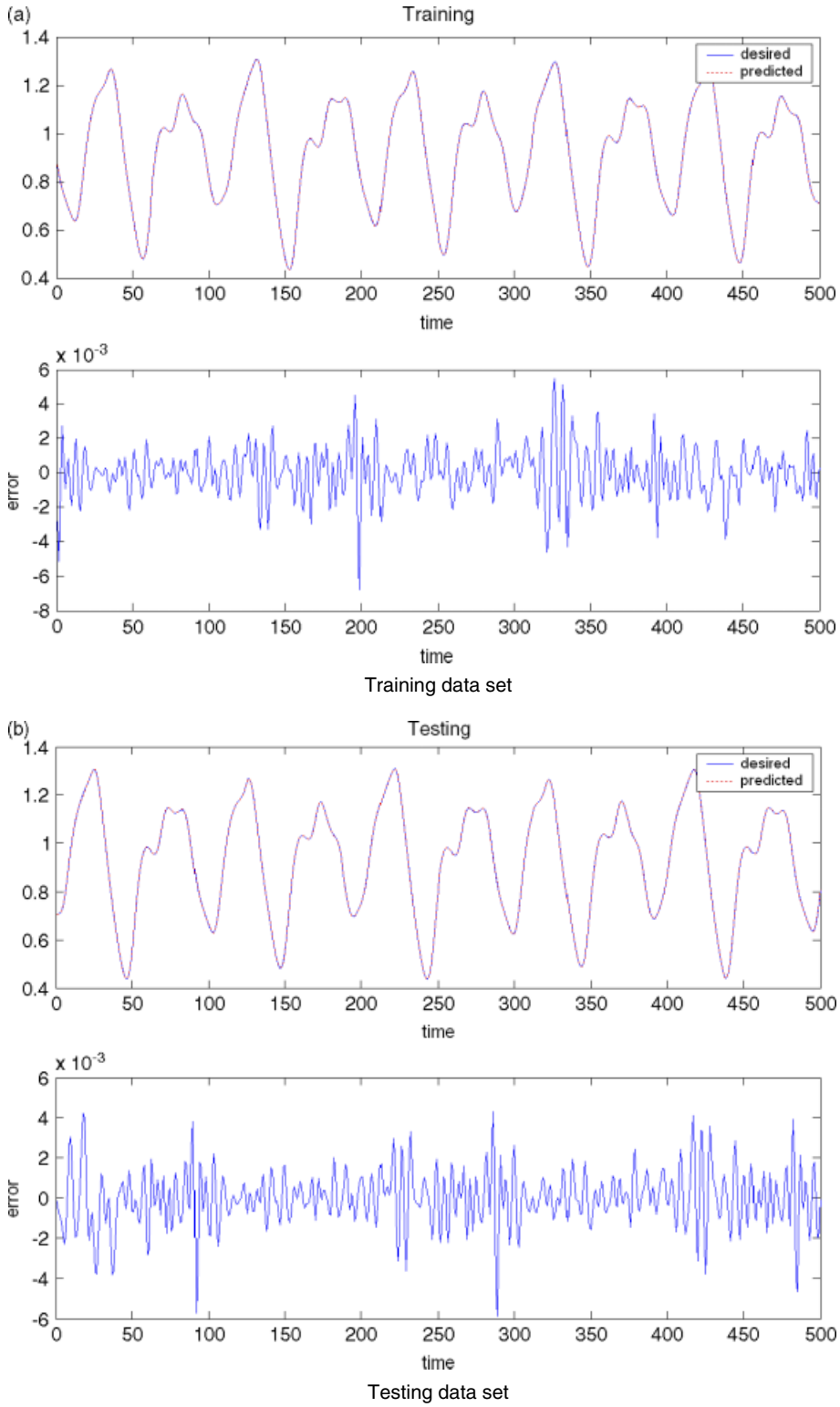

Figure 2. Desired and predicted chaotic time series by radial basis function networks

as a case study. The river originates on Nanhu Mountain at 3535 m above sea level and gradually descends from the west to the east in 73-km long running through to the Pacific Ocean. The steep Lan-Yang River sweeps masses of sandstone downstream, forming the alluvial plain with a drainage area of 980 km^2. The average annual temperature is 21·8 °C, has mean annual rainfall of 3170 mm that results in luxuriant vegetation and beautiful scenery. About 40% of the runoff is derived from summer rain related to typhoons. There is no dam on the entire river. The river provides the Lan-Yang Plain with water for irrigation and household use, and its estuary is one of most important coastal wetlands in Taiwan. The estuary is one of the nation's three major fishing grounds and attracts more than 230 kinds of birds.

DOI: 10.1002/hyp

The available data

There are four rainfall stations upstream of the watershed and a water-level gauge station at the Lan-Yang bridge, which are all equipped with automatic rainfall amount (or water level) recorders and transmitted by wireless (Figure 3). The hourly rainfall and water level of the gauge stations are used. These are 23 typhoon (or heavy rainfall) events from the past 21 years (1981 to 2001), published by the Water Resources Agency, Taiwan. A total of 1632 data sets are obtained. The 23 events data are divided into three independent subsets: (i) the training, (ii) validation and (iii) the testing subsets. The training subset includes 15 events (1104 data sets), the validation subset has 5 events (288 data sets), while the testing subset has the remaining 3 events (240 data sets). The ratio of training sets to verification sets to testing sets is approximately 4 : 1 : 1. First, the training subsets are repeatedly used to build networks and to adjust the connected weights of the constructed networks. Afterwards, the validation subset is used to simulate the performance of the built models for checking its suitability of generalization, and the best network is selected for later use. The testing data set is then used for final evaluation of the selected network performance.

To assess the models performances, RMSE, mean absolute error (MAE) and G_{bench} are used. G_{bench} is to compute the goodness of fit measures and shown as below.

$$G_{\text{bench}} = 1 - \frac{\sum_{i=1}^{n}(\hat{Q}_i - Q_i)^2}{\sum_{i=1}^{n}(\hat{Q}_i - Q_{i,\text{bench}})^2}. \quad (5)$$

Here, $\hat{Q}_i$ and Q_i represent the forecasted and observed water level, respectively. The variable n means the number of data points. $Q_{i,\text{bench}}$ is the previous observed value, e.g. for n-step ahead prediction $Q_{i,\text{bench}} = Q_{i-n}$. If G_{bench} is negative, the forecasting performance is poorer than the benchmark; if G_{bench} is equal to zero, the performance is as good as the benchmark; If G_{bench} is equal to one, it means perfect fit.

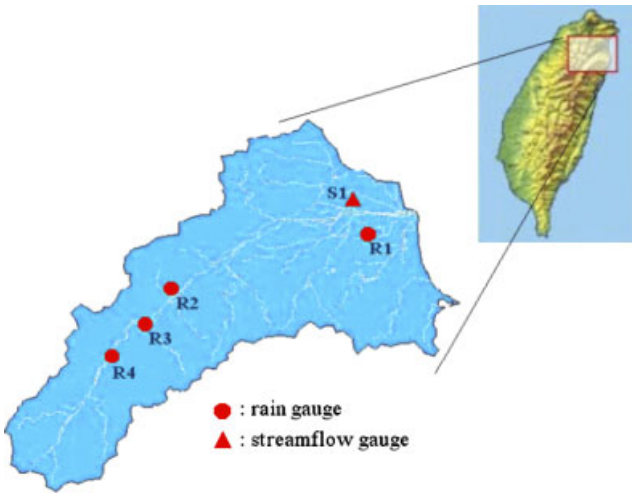

Figure 3. Location of study area and gauge stations

In the present study, the input dimensions are determined by the input variables and the lag time. To determine an appropriate feed-forward neural network structure for forecasting the streamflow at time $t+1$ and $t+2$ in this selected basin, we develop two different models for the BPNN and RBFN, namely,

Model 1: $Q(t+1) = f(Q(t), Q(t-1), Q(t-2), R_1(t), R_1(t-1), R_1(t-2), R_2(t), R_2(t-1), R_2(t-2), R_3(t), R_3(t-1), R_3(t-2), R_4(t), R_4(t-1), R_4(t-2))$;

Model 2: $Q(t+2) = f(Q(t), Q(t-1), R_1(t), R_1(t-1), R_2(t), R_2(t-1), R_3(t), R_3(t-1), R_4(t), R_4(t-1))$,

where $Q(t-i)$ represents the value of the Lan-Yang Bridge streamflow gauge station at time $t-i$ and $R_1(t-i)$, $R_2(t-i)$, $R_3(t-i)$ and $R_4(t-i)$ represents the precipitation of the four rainfall gauge stations at time $t-i$. Because the drainage area of the study watershed is less than 1000 km^2 and the basin slope is quite steep, the time of concentration is short. Consequently, the input pattern is focused on the previous two-hours information only.

BPNN

The BP network is trained by using supervised learning to obtain the optimal values of the connected weights such that the energy function is a minimum. The conjugate gradient method is a well-known numerical technique used for solving various optimization problems. The conjugate gradient algorithm searches a system of conjugate directions on the error surface and updates the weights along these directions. In practice, the process makes good uniform progress towards the solution at every time step and has been found to be effective in finding better optimization than the standard BP algorithm (Ham and Kostanic, 2001). We use this algorithm to update the connecting weights during the training process.

Finding the optimal size of the networks

Determining the appropriate size of neural networks is very important for validity and efficiency. Since there is no systematic or standard method for finding the optimal number of hidden nodes, the optimal network size is based on trial and error in the circumstance. To determine the optimal BPNN for the given input-output patterns, we investigated several network architectures. Because the input vector is set the same as RBFN, i.e. 15 for one-step ahead and 10 for two-step ahead, respectively, and the output vector is 1 (one-step ahead or two-step ahead), only the number of nodes in the hidden layer needs to be determined. The networks with various numbers of hidden nodes, from 1 to 12, are performed; furthermore, each network is executed with 50 initial sets of connected weights and the best result is selected. The search results

DOI: 10.1002/hyp

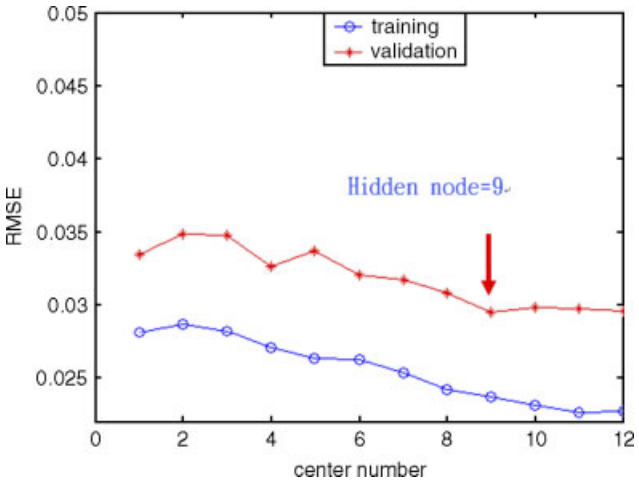

Figure 4. The root mean square error versus the number of node (centrr) in the constructed back propagation neural network

of one-step ahead are shown in Figure 4. As shown, when the hidden layer has 9 nodes for one-step ahead, the constructed BPNN has suitable performance in the set and best performance in the validation set. For two-step ahead forecast, as the hidden layer has 13 nodes, the BPNN has the best performance. These results are used as a base to evaluate the adequacy of our proposed RBF neural networks. To determine the optimal RBFN for the given input-output patterns, the OLS method, which offers a systematic way for centre selection and can significantly reduce the size of the RBFN, is used. The Gaussian function is implemented as the Kernel function of the RBFN. The optimal number of nodes in the hidden layer obtained by the OLS method is 11 and 12 for one-step ahead and two-step ahead, respectively.

Results and Discussion

As presented above, the structures of BPNN for river flood forecasting have been identified based on a time-consuming trial and error processes, whereas the proposed methodology that can systematically configure RBF network and estimate its parameters using a set of input-output patterns. Based on these constructed networks, the results of BPNN and RBFN are summarized in Table II. The results show that (1) the RBFN is superior to the BPNN in forecasting ability, where the RBFN obtains much lower forecasting errors and higher G_{bench} values as compared with the BPNN and (2) the RBFN can adequately produce one-step ahead streamflow forecasting with small MAE and high G_{bench} value. To closely investigate the performance of the RBFN, Figure 5 shows the hydrographs and scatter plots of observed versus one-step ahead forecasting of the RBFN in all the three phases, i.e. training, validation and testing, respectively. One can easily find that the forecast values of RBFN are highly correlative to the observed values of hydrographs and close to the ideal line in the scatter plots in all three phases.

For further investigating the applicability of the RBFN and BPNN networks on two-hour ahead forecasting,

Table II. One-step ahead flood forecast in three different phases by back propagation neural network (BPNN) and radial basis function networks (RBFN)

		Training	Validation	Testing
BPNN	RMSE (cms)	98·18	115·58	134·58
	MAE (cms)	59·43	68·20	74·05
	G_{bench}	0·23	0·26	0·34
RBFN	RMSE (cms)	74·02	74·12	97·07
	MAE (cms)	37·87	38·82	51·74
	G_{bench}	0·56	0·70	0·38

RMSE, root mean square error; MAE, mean absolute error.

Table III. Two-step ahead flood forecast in three different phases by back propagation neural network (BPNN) and radial basis function networks (RBFN)

		Training	Validation	Testing
BPNN	RMSE (cms)	134·2	200·6	179·3
	MAE (cms)	77·3	99·8	101·0
	G_{bench}	0·54	0·35	0·31
RBFN	RMSE (cms)	110·5	110·4	150·9
	MAE (cms)	57·6	57·9	73·2
	G_{bench}	0·69	0·80	0·60

RMSE, root mean square error; MAE, mean absolute error.

the above procedures were re-executed and the results are shown in Table III. Again, we can easily find that both networks can reasonably make two-hour ahead forecast, while the RBFN is superior to the BPNN, in terms of smaller RMSE and MAE values and higher value of G_{bench} in all three phases. Figure 6 represents the two-hour ahead forecast results, where the RBFN reasonably fits the ideal line and slightly wider spread from the ideal line than the one-hour ahead forecasts. It appears that the constructed RBFN has better forecasting performances, on both one-step and two-step ahead, than the constructed BPN in this study case. We demonstrate that the auto-configuring RBFN can effectively capture the essential components of the non-linear dynamic system and accurately forecast the short-term flood.

CONCLUSION

RBF networks have been widely used for modelling complex non-linear systems and for time series prediction. The configuration of an RBFN was usually divided into two stages: Using the unsupervised cluster method to select the number of hidden neurons for determining the structure of RBFN, and then search the optimal position of the RBF centres and the width of the RBFs. Even though previous works have demonstrated, or claimed, that it is usually faster to construct and train this type of network than that of multilayer perceptron networks, it could only be classified as a local search method and suffers from the time-consuming trial and error procedures. To continuously increase their popularity, we propose a systematic and efficient way to automatically construct

DOI: 10.1002/hyp

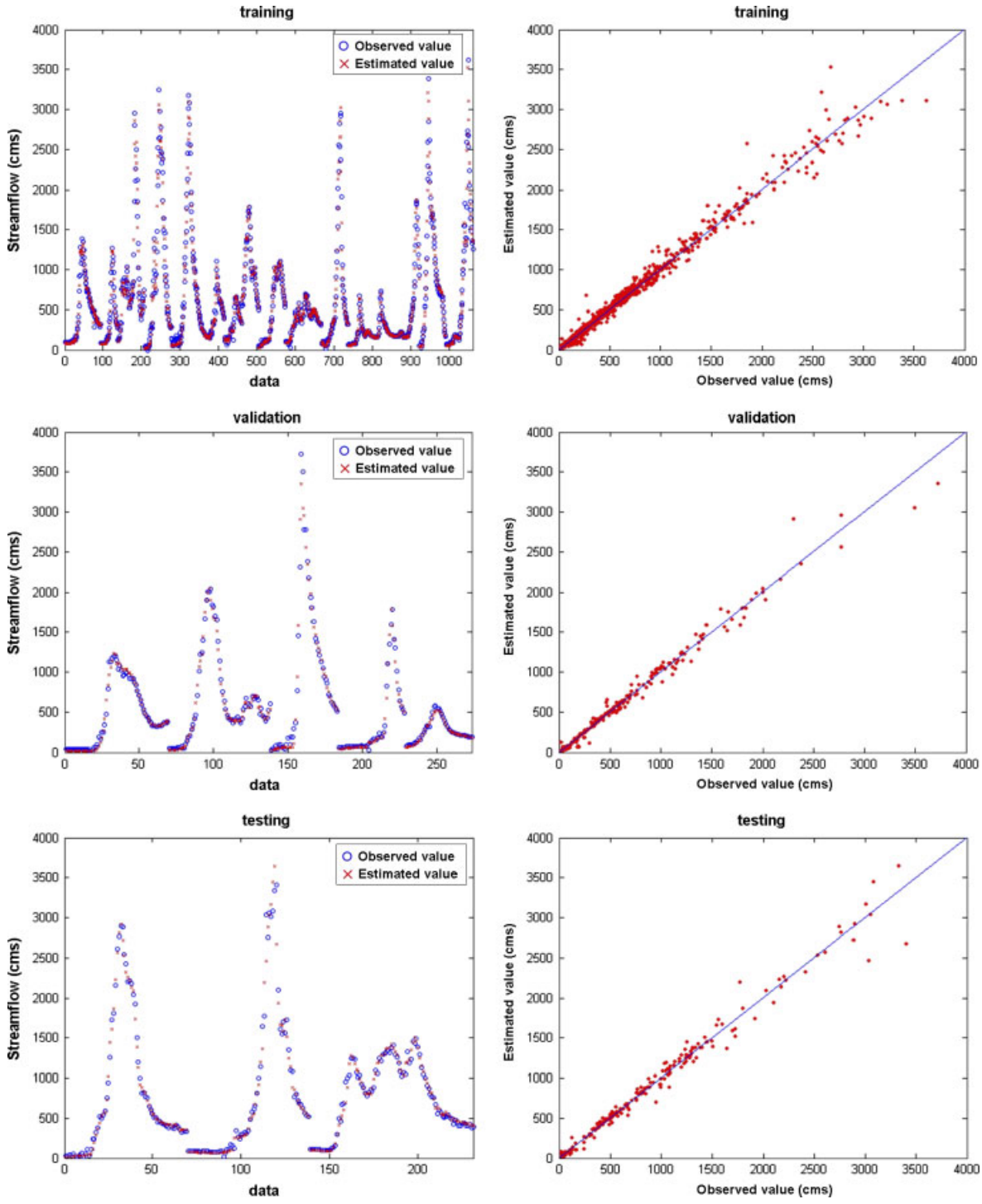

Figure 5. The hydrographs and scatter plots of observed versus one-step ahead forecasting of the radial basis function networks in three phases

the RBFN. Our strategy entails fusing the OLS algorithm, the GA and the LSE method to (1) directly identify the structure of RBFN, (2) search the position of the centres and the width of RBFs and (3) identify the linear weights of output layer. The time-consuming search problem is effectively solved and the resulting networks can provide better performance.

The superiority of the proposed algorithm over the conventional method in constructing and training the RBFN is first demonstrated through a benchmark problem—the chaotic Mackey-Glass time series. Our experimental results show the efficiency and effectiveness of the proposed approaches in predicting the chaotic time series. The auto-configuring strategy increases the confidence and promises of the RBFN to be widely and easily used. The proposed method is then applied to flood forecasting in the Lan-Yang River, Taiwan. For the purpose of comparison, the commonly used BPNN is also performed. The results demonstrate that the RBFN is superior to the BPNN in making one-hour ahead and two-hour ahead flood forecasts. The proposed strategy can systematically construct the structure (the number of hidden nodes)

DOI: 10.1002/hyp

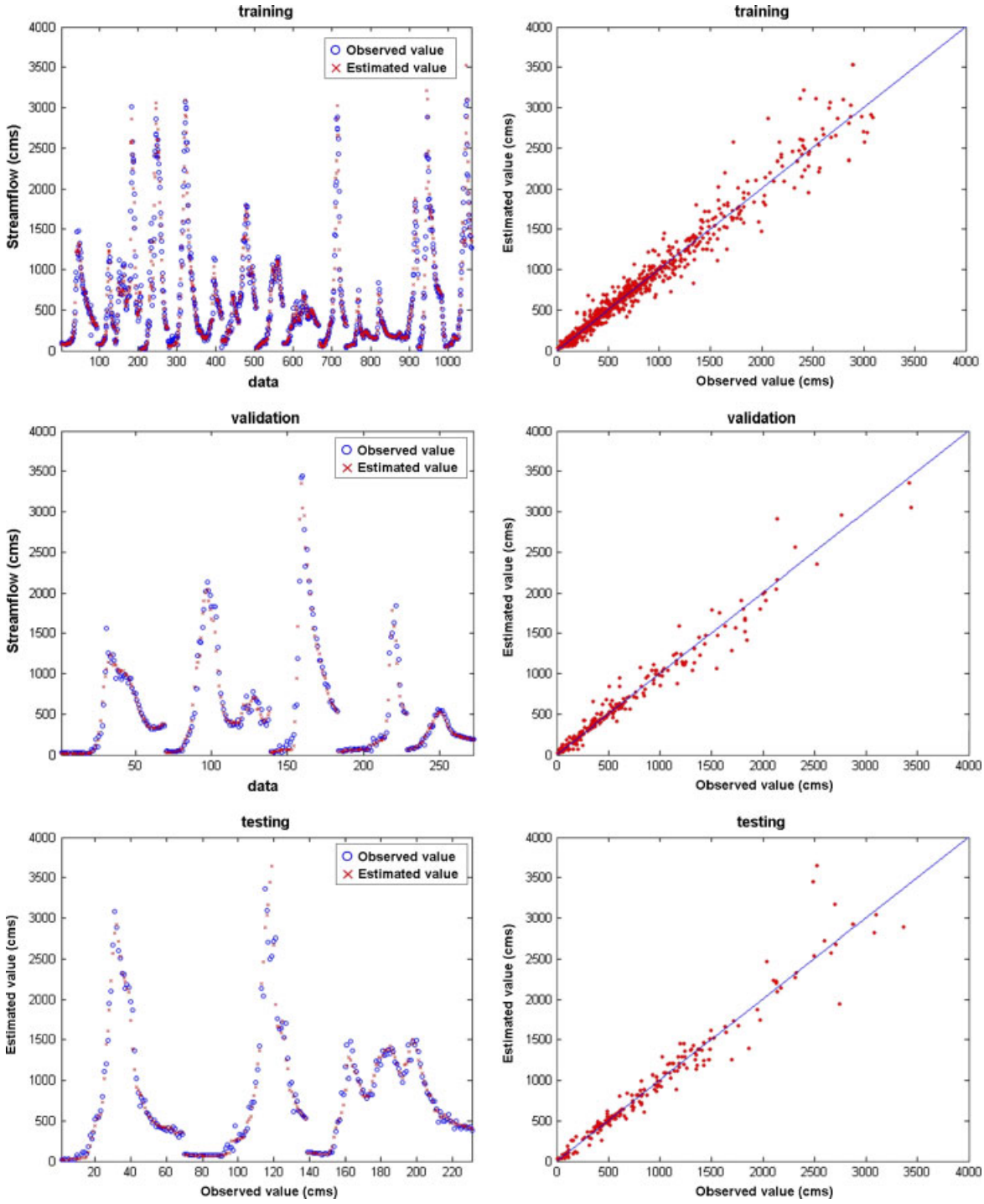

Figure 6. The hydrographs and scatter plots of observed versus two-step ahead forecasting of the radial basis function networks in three phases

of RBFN and properly identify the network's relative parameters (centres and their widths of kernel functions and linear weights of output layer) to make accurate flood forecasts.

ACKNOWLEDGEMENT

This article is based on partial work supported by the National Science Council, ROC under NSC-95-2313-B-002-031. In addition, the authors are indebted to the reviewers for their valuable comments and suggestions.

REFERENCES

Aminian K, Ameri S. 2005. Application of artificial neural networks for reservoir characterization with limited data. *Journal of Petroleum Science and Engineering* **49**(3–4): 212–222.

Arifovic J, Gencay R. 2001. Using genetic algorithms to select architecture of a feedforward artificial neural network. *Physica A* **289**: 574–594.

Billings SA, Zheng GL. 1995. Radial basis function network configuration using genetic algorithms. *Neural Networks* **8**(6): 877– 890.

Boozarjomehry RB, Svrcek WY. 2001. Automatic design of neural network structures. *Computers & Chemical Engineering* **25**(7–8): 1075–1088.

Box GEP, Jenkins GM. 1976. *Time Series Analysis Forecasting and Control*, 2nd edn. Holden-Day: San Francisco, CA.

Broomhead DS, Lowe D. 1988. Multivariable function interpolation and adaptive networks. *Complex System* **2**(3): 321–355.
Chang LC, Chang FJ, Chiang YM. 2004. A two-step-ahead recurrent neural network for stream-flow forecasting. *Hydrological Processes* **18**(1): 81–92.
Chang YT, Chang LC, Chang FJ. 2005. Intelligent control for modeling of real-time reservoir operation, part II: artificial neural network with operating rule curves. *Hydrological Processes* **19**(7): 1431–1444.
Chang FJ, Chang LC, Wang YS. 2007. Enforced self-organizing map neural networks for river flood forecasting. *Hydrological Processes* **21**(6): 741–749.
Chaves P, Kojiri T. 2007. Conceptual fuzzy neural network model for water quality simulation. *Hydrological Processes* **21**(5): 634–646.
Chen S, Cowan CFN, Grant PM. 1991. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks* **2**(2): 302–309.
Chiang YM, Chang LC, Chang FJ. 2004. Comparison of static-feedforward and dynamic-feedback neural networks for rainfall-runoff modeling. *Journal of Hydrology* **290**: 297–311.
Chiang YM, Chang FJ, Jou JD, Lin PF. 2007. Dynamic ANN for precipitation estimation and forecasting from radar observations. *Journal of Hydrology* **334**: 250–261.
Fritzke B. 1994. Fast learning with incremental RBF networks. *Neural Processing Letters* **1**(1): 2–5.
Gholipour A, Araabi BN, Lucas C. 2006. Predicting chaotic time series using neural and neurofuzzy models: a comparative study. *Neural Processing Letters* **24**(3): 217–239.
Ham FM, Kostanic I. 2001. *Principles of Neurocomputing for Science & Engineering*. McGraw-Hill: New York.
Holland JH. 1975. *Adaptation in Natural and Artificial System*. University of Michigan Press: Ann Arbar, MI.
Hsu KL, Gao X, Sorooshian S, Gupta HV. 1997. Precipitation estimation from remotely sensed information using artificial neural networks. *Journal of Applied Meteorology* **36**: 1176–1190.
Jain SK, Nayak PC, Sudheer KP. 2008. Models for estimating evapotranspiration using artificial neural networks, and their physical interpretation. *Hydrological Processes* **22**(13): 2225–2234.
Jang RJS. 1993. ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Transactions on Systems, Man, and Cybernetics* **23**(3): 665–685.
Jeong D, Kim Y-O. 2005. Rainfall-runoff models using artificial neural networks for ensemble streamflow prediction. *Hydrological Processes* **19**(19): 3819–3835.
Kisi O. 2007. Evapotranspiration modelling from climatic data using a neural computing technique. *Hydrological Processes* **21**(14): 1925–1934.
Kisi O. 2008. River flow forecasting and estimation using different artificial neural network techniques. *Hydrology Research* **39**(1): 27–40.
Krishna B, Satyaji Rao YR, Vijaya T. 2008. Modelling groundwater levels in an urban coastal aquifer using artificial neural networks. *Hydrological Processes* **22**(8): 1180–1188.
Kumarasiri AD, Sonnadara UJ. 2008. Performance of an artificial neural network on forecasting the daily occurrence and annual depth of rainfall at a tropical site. *Hydrological Processes* **22**(17): 3535–3542.
Kuncheva LI. 1997. Fitness functions in editing k-NN reference set by genetic algorithms. *Pattern Recognition* **30**: 1041–1049.
Kuo JT, Wang YY, Lung WS. 2006. A hybrid neural–genetic algorithm for reservoir water quality management. *Water Research* **40**(7): 1367–1376.
Lin GF, Chen GR. 2008. A systematic approach to the input determination for neural network rainfall-runoff models. *Hydrological Processes* **22**(14): 2524–2530.
Maier HR, Morgan N, Chow CWK. 2004. Use of artificial neural networks for predicting optimal alum doses and treated water quality parameters. *Environmental Modelling & Software* **19**(5): 485–494.
Moody J, Darken D. 1989. Fast learning in networks of locally-tuned processing units. *Neural Computation* **1**: 281–294.
Musavi MT, Ahmed W, Chan KH, Faris KB, Hummels DM. 1992. On the training of radial basis function classifiers. *Neural Networks* **5**: 595–603.
Poggio R, Girosi F. 1990. Networks for approximation and learning. *Proceedings of the IEEE* **78**(9): 1481–1497.
Sahoo GB, Ray C, De Carlo EH. 2006. Use of neural network to predict flash flood and attendant water qualities of a mountainous stream on Oahu, Hawaii. *Journal of Hydrology* **327**: 525–538.
Tayfur G, Moramarco T, Singh VP. 2007. Predicting and forecasting flow discharge at sites receiving significant lateral inflow. *Hydrological Processes* **21**(14): 1848–1859.
Whitely D, Starkweather T, Bogartm C. 1990. Genetic algorithms and neural networks—optimizing connections and connectivity. *Parallel Computing* **14**: 347–361.

DOI: 10.1002/hyp