JOURNAL OF
ENVIRONMENTAL HYDROLOGY
The Electronic Journal of the International Association for Environmental Hydrology
On the World Wide Web at http://www.hydroweb.com
VOLUME 18 2010

EVALUATION OF DAILY RAINFALL-RUNOFF MODEL USING MULTILAYER PERCEPTRON AND PARTICLE SWARM OPTIMIZATION FEEDFORWARD NEURAL NETWORKS

Kuok King Kuok[1]
Sobri Harun[1]
Siti Mariyam Shamsuddin[2]
Po-Chan Chiu[3]

[1]Department of Hydraulics and Hydrology, University Technology Malaysia, Malaysia
[2]Department of Computer Graphics and Multimedia, University Technology Malaysia, Malaysia
[3]Department of Information Systems, University Malaysia Sarawak, Malaysia

In recent years, Artificial Neural Networks (ANNs) have been successfully used as a tool to model various nonlinear relations, and the method is appropriate for modeling the complex nature of hydrological systems. They are relatively fast and flexible, and are able to extract the relation between the inputs and outputs of a process without knowledge of the underlying physics. ANNs with sufficient hidden units are able to approximate any continuous function to any degree of accuracy by performing efficient training. In this study, two types of ANNs, namely, the multilayer perceptron neural network (MLP) and the newly developed particle swarm optimization feedforward neural network (PSONN) are applied to model the daily rainfall-runoff relationship for the Bedup Basin, Sarawak, Malaysia. Various models are investigated in searching for the optimal configuration of ANNs. Results are evaluated using the coefficient of correlation (R) and the nash-sutcliffe coefficient (E^2). With the input data of current rainfall, antecedent rainfall and antecedent runoff, MLP simulated the current runoff perfectly for training with R=1.000 and E^2=1.000, and R=0.911 and E^2=0.8155 for testing data set. Meanwhile, PSONN also simulated the current runoff accurately with R=0.872 and E^2=0.7754 for training data set, and R=0.900 and E^2=0.8067 for testing data set. Thus, it can be concluded that ANNs are able to model the rainfall-runoff relationship accurately. The performance of the newly developed PSONN is comparable with the well-known MLP network, which had been successfully used to model rainfall-runoff for the Bedup Basin.

INTRODUCTION

The rainfall-runoff relationship describes the time distribution of direct runoff as a function of excess rainfall. This relationship is known to be highly nonlinear and complex. The complexity and nonlinearity is mainly due to many highly complex hydrologic components, including interception, depression storage, infiltration, overland flow, interflow, percolation, evaporation and transpiration. Furthermore, runoff also depends on catchment topography, river network topology, river cross-sections, soil characteristics and antecedent moisture content. The relationship between rainfall and runoff for watersheds is the hardest problem faced by hydrologists and engineers (Irwan et al., 2007).

Various rainfall-runoff models such as HEC-HMS, HEC-RAS, MIKE-SHE, MIKE-11, SWMM, Infoworks, etc., have been successfully developed and applied. However, these models have their own weaknesses, especially in the calibration processes and the ability to adopt the nonlinearity of processes. Therefore, neural networks (NN) have been proposed to handle complex hydrologic processes with large volumes of data. In particular a backpropagation neural network (BPNN) is useful for handling real-time, non-stationary and non-linear natural phenomena (Nishimura and Kojiri, 1996). The natural behavior of rainfall-runoff systems is appropriate for the application of BPNN. The last decade has witnessed many applications of BPNN in water resources, such as modeling of the rainfall-runoff process (Elshorbagy et al., 2000; Bessaih et al., 2003); inflow estimation (Harun et al., 1996); runoff analysis in a humid forest catchment (Gautam et al., 2000); river flow prediction (Imrie et al., 2000; Dastorani and Wright, 2001); ungauged catchment flood prediction (Wright and Dastorani, 2001) and short term river flood forecasting (Garcia-Bartual, 2002).

However, the major disadvantages of BPNN are its relatively slow convergence rate (Zweiri et al., 2003) and solutions being trapped at local minima. Basically, BPNN learning is a hill climbing technique. Hence, it runs the risk of being trapped in local minima, where every small change in synaptic weight increases the cost function. Sometimes, the network is stuck where there exists another set of synaptic weights for which the cost function is smaller than the local minimum in the weight space. This makes termination of the learning process at local minima by BPNN undesirable.

Therefore, the present study was undertaken to develop rainfall runoff models using a novel method that is a hybrid of the particle swarm optimization (PSO) technique with artificial neural networks (ANNs) that can be used to provide reliable and accurate estimates of runoff. This hybrid method is called particle swarm optimization feedforward neural network (PSONN). This PSONN is proposed to improve the convergence rate of NN and avoid solutions being trapped at local minima. The performance of PSONN is then compared with the conventional multilayer perceptron neural network (MLP) with backpropagation algorithm.

This paper was divided into six sections. Section 2 reviews the basic concept of PSONN and MLP network architectures. Section 3 presents the selected study area, which is the Bedup Basin, located at Sarawak, Malaysia. Model development and the sensitivity of the PSONN and MLP performance to the length of the calibration data and related parameters using various data sets will be discussed in sections 4 and 5, respectively. Subsequently, the results obtained and discussion using various configurations of PSONN and MLP will be explained in section 6.

NEURAL NETWORK ARCHITECTURES

Particle Swarm Optimization Feedforward Neural Network (PSONN)

The PSO is made up of particles, where each particle has a position and a velocity. The idea of PSO in NN is to get the best set of weights (or particle positions) where several particles (problem solution) are trying to move to the best solution and this will avoid the solution trap at local minima (Eberhart and Shi, 2001).

The advantage of the PSO over many of the other optimization algorithms is its relative simplicity (Van den Bergh, 2001). According to Jones (2005), the only two equations used in PSO are the movement equation (Equation 1) and velocity update equation (Equation 2). The movement equation provides for the actual movement of the particles using their specific vector velocity while the velocity update equation provides for velocity vector adjustment given the two competing forces ("gbest" and "pbest"). The inertia weight (ω) was introduced by Shi and Eberhart (1998) to improve the convergence rate of the PSO algorithm.

$$presLocation = prevLocation + V_i \Delta t \tag{1}$$

$$V_i = \omega V_{i-1} + c_1 * rand() * (pbest - presLocation) + c_2 * rand() * (gbest - presLocation) \tag{2}$$

where V_i is the current velocity, Δt defines the discrete time interval over which the particle will move, ω is the inertia weight, V_{i-1} is the previous velocity, *presLocation* is the present location of the particle, *prevLocation* is the previous location of the particle and *rand()* is a random number between (0, 1), c_1 and c_2 are the acceleration constants for "gbest" and "pbest" respectively. Particle velocities are limited by a user-specified value, maximum velocity V_{max}, to prevent the particles from moving too far from a potential solution.

The hybrid of PSO and NN is applying PSO to train the feedforward NN for enhancing the convergence rate and learning process. According to Al-kazemi and Mohan (2002), the position of each particle in a PSONN represents a set of weights for the current iteration. The dimension of each particle is the number of weights associated with the network. The learning error of this network is computed using the Mean Squared Error (MSE) between the observed and simulated runoff. The learning process involves finding a set of weights that minimizes the learning error. Hence, the particle will move within the weight space attempting to minimize learning error.

The learning process of PSONN is further illustrated in Figure 1. The learning process of PSONN is initialized with a group of random particles (step 1), which are assigned random PSO positions (weight and bias). The PSONN is trained using the initial particles position (step 2). Then, the feedforward NN in PSONN will produce the learning error (particle fitness) based on an initial weight and bias (step 3). The learning error at the current epoch or iteration will be reduced by changing the particles position, which will update the weight and bias of the network.

The "pbest" value (each particle's lowest learning error so far) and "gbest" value (lowest learning error found in the entire learning process so far) are applied to the velocity update equation (Equation 2) to produce a value for position adjustment to the best solutions or targeted learning error (step 4). The new sets of positions (NN weight and bias) are produced by adding the calculated velocity value to the current position value using the movement equation (Equation 1). Then, the new sets of positions are used to produce new learning errors for the feedforward NN (step 5). This process is repeated until the stopping conditions, either minimum learning error or

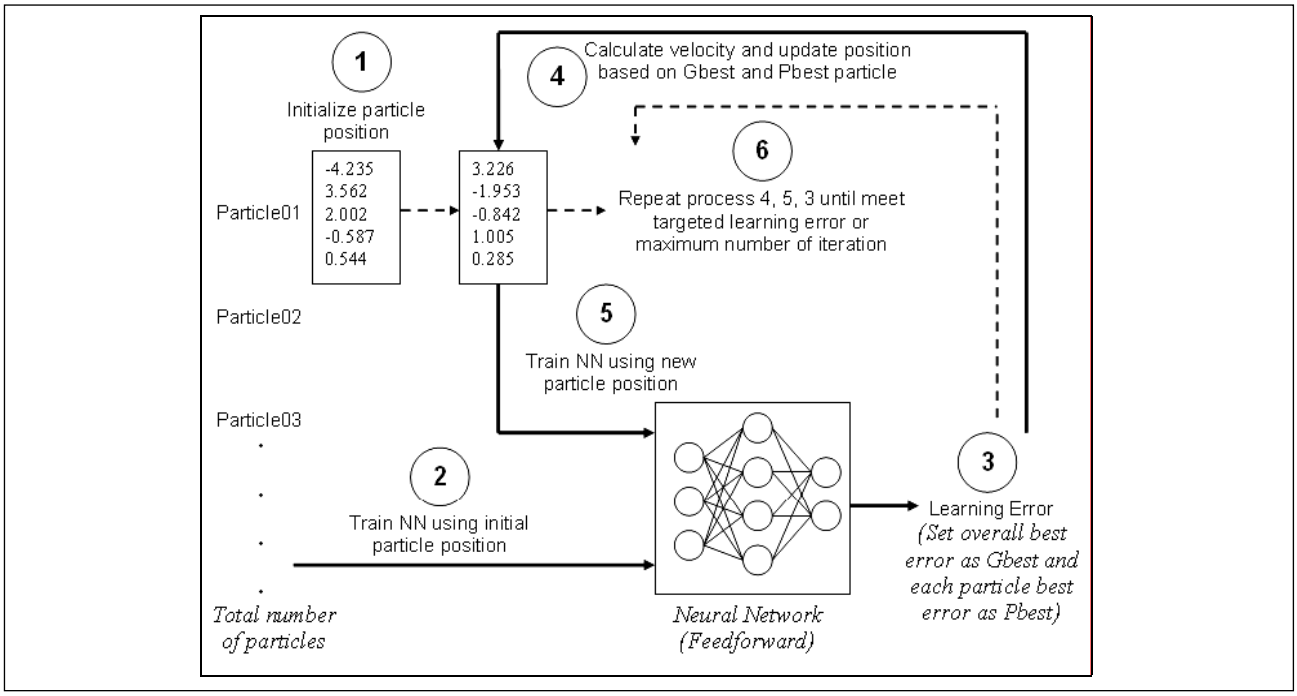


Figure 1. PSONN learning process (Van den Bergh, 2001).

maximum number of iterations, are met (step 6). The optimization output, which is the solution for the optimization problem, was based on the gbest position value.

Multilayer Perceptron Neural Network (MLP)

The MLP network used is a two-layer feedforward network trained with a backpropagation learning algorithm (Figure 2). The transfer function used in the hidden layer is tan-sigmoid (TANSIG) and linear transfer function (PURELIN) at the output layer (Palm, 2001). The variant of backpropagation algorithm used is Scaled Conjugate Gradient (TRAINSCG). TRAINSCG is selected as it was designed to avoid the time consuming line search, which produces generally faster convergence than the steepest descent directions used by the basic backpropagation (Demuth and Beale, 2001). The number of hidden neurons was determined through a trial and error method.

The neurons in each layer are connected to the neurons in the subsequent layer by a weight, which was adjusted during training. Figure 3 shows the input vector p_1 was propagated towards the hidden layer by multiplying with input weight matrices $IW_{1,1}$ to form $IW_{1,1}p_1$. Then the sum of bias b_1 and

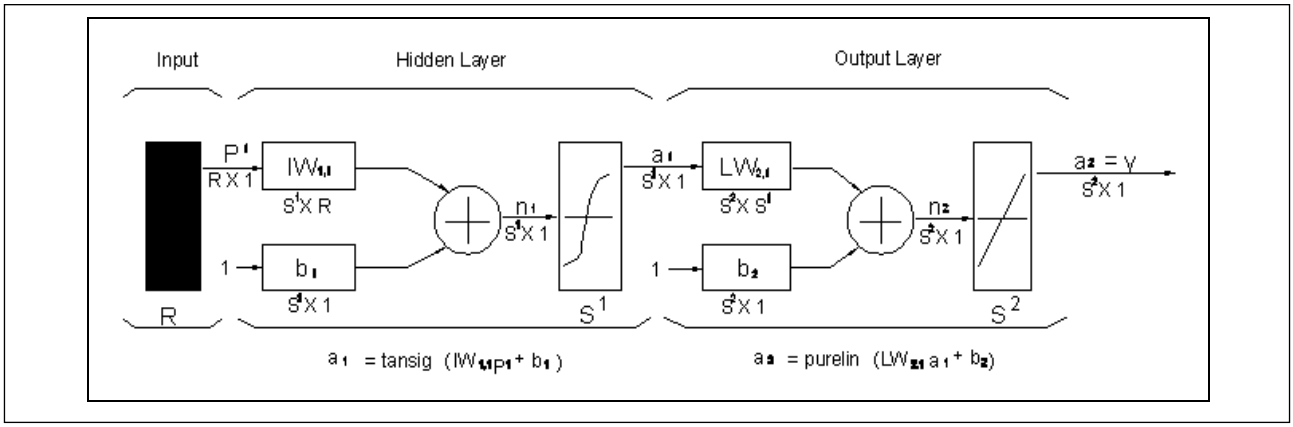


Figure 2. MLP network architecture (Demuth and Beale, 2001).

the product of $IW_{1,1}p_1$ propagated forward to *tansig* transfer function. This sum was passed through *tansig* transfer function to get the hidden neurons's output a_1, where $a_1 = tansig(IW_{1,1}\, p_1 + b_1)$.

Similarly, the output a_1 from hidden layer is propagated forward through the network towards the output layer. At the output layer, a_1 is then multiplied with weight matrices in layer outputs called layer weights, $LW_{2,1}$ to form $LW_{2,1}\, a_1$. Then, the sum of bias b_2 and product $LW_{2,1}\, a_1$ will transform through the *purelin* transfer function to get the neurons output a_2, where $a_2 = purelin(IW_{2,1}\ a_1 + b_2)$.

Thereafter, feedback iteration calculated error signals that propagated backwards through the network are used to adjust the weights. The weights of the output layer are adjusted first. Then, adjustments are made for interconnection weights between each layer, based on the computer error and learning rate parameter.

STUDY AREA

The selected study area is Sungai Bedup Basin, a sub-basin of Sadong Basin, Sarawak, Malaysia. This basin is located approximately 80km from Kuching City. Model calibration used data series of daily rainfall and observed runoff from 1997 to 1999.

Figure 3a shows the location of Sadong Basin in Sarawak, Malaysia. The main boundary of the Sadong Basin, rainfall and river stage gauging stations within Sadong Basin, are shown in Figure 3b. The rainfall and water level stations available in Bedup basin, which is a non-tidal influence river basin, are presented in Figure 3c. The stations are 5 rainfall gauging stations, named Bukit Matuh (BM), Semuja Nonok (SN), Sungai Busit (SB), Sungai Merang (SM), and Sungai Teb (ST), and one river stage gauging station, located at the outlet of the basin.

The basin area is approximately 47.5km^2 and the elevation varies from 8m to 686m above mean sea level (JUPEM, 1975). The Sungai Bedup basin has a dendritic type channel system. The maximum stream length for the basin is approximately 10km, which is measured from the most remote point on the stream network to the basin outlet.

The input data used are daily rainfall data from the 5 rainfall stations. Observed daily mean runoff data are converted from water level data through a rating curve given by Equation (3) (DID, 2004).

$$Q = 9.19(H)^{1.9} \tag{3}$$

where Q is the discharge (m^3/s) and H is the stage discharge (m). These observed runoff data were used to compare the model runoff.

MODEL DEVELOPMENT

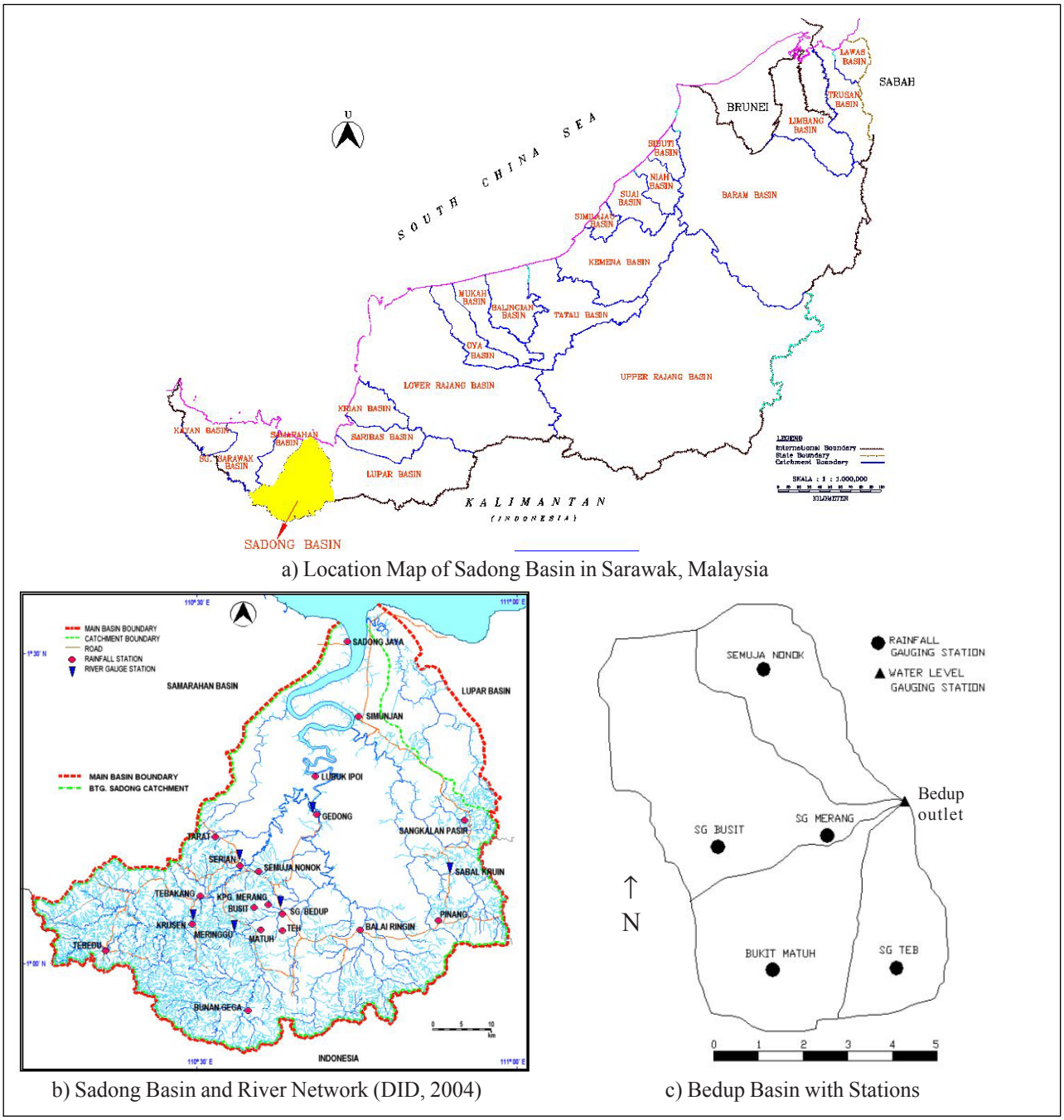
The training and testing processes of the daily rainfall-runoff model are investigated using the PSONN and MLP models. Various parameters that affect PSONN and MLP performance are investigated for searching the best model configuration of PSONN and MLP for simulating daily runoff at Sungai Bedup.

The performance of PSONN was investigated for nine parameters and these are:

a) Number of antecedent days.

b) Acceleration constants for "gbest" (c_1)

c) Acceleration constants for "pbest" (c_2)



a) Location Map of Sadong Basin in Sarawak, Malaysia

b) Sadong Basin and River Network (DID, 2004)

c) Bedup Basin with Stations

Figure 3. Location map of Bedup Basin, Sub-basin of Sadong Basin, Sarawak, Malaysia.

d) The time interval (Δt) constant

e) The number of particles.

f) Maximum iteration (stopping condition)

g) Number of hidden neurons in the hidden layer

h) Length of input data

i) Number of antecedent days.

In PSONN, the number of particle dimensions refers to the number of weights and the bias that depends on the training dataset and PSONN architecture. The particle dimension relies on the

number of input neurons, number of hidden neurons in the hidden layer, and number of output neurons. The dimension of PSONN is calculated using Equation (4).

$$Dimension = (input*hidden\ input) + (hidden*output\ hidden) + hidden_{bias} + output_{bias} \quad (4)$$

In contrast, the performance of MLP is investigated for four main parameters:

a) Number of antecedent days.

b) Length of training data

c) Number of hidden neurons

d) Learning rate values

Five models were developed for investigating the effect of the number of antecedent days on the performance of PSONN and MLP. The input data of the models consists of antecedent rainfall, *P(t-1), P(t-2),...,P(t-n),* antecedent runoff, *Q(t-1), Q(t-2),...,Q(t-n)* and current rainfall, *P(t)*. The output is the runoff for the current day, *Q(t)*. The configurations of five PSONN models with different number of antecedent days are listed below:

PSONND1 and MLPD1 Model

$$Q(t)=f[P(t),P(t-1),Q(t-1)] \quad (5)$$

PSONND2 and MLPD2 Model

$$Q(t)=f[P(t),P(t-1),P(t-2),Q(t-1),Q(t-2)] \quad (6)$$

PSONND3 and MLPD3 Model

$$Q(t)=f[P(t),P(t-1),P(t-2),P(t-3),Q(t-1),Q(t-2),Q(t-3)] \quad (7)$$

PSONND4 and MLPD4 Model

$$Q(t)=f[P(t),P(t-1),P(t-2),P(t-3),P(t-4),Q(t-1),Q(t-2),Q(t-3),Q(t-4)] \quad (8)$$

PSONND5 and MLPD5 Model

$$Q(t)=f[P(t),P(t-1),P(t-2),P(t-3),P(t-4),P(t-5),Q(t-1),Q(t-2),Q(t-3),Q(t-4),Q(t-5)] \quad (9)$$

where t = time (days), P = precipitation (mm), Q = discharge (m^3/s). Equations 5, 6, 7, 8 and 9 represent operations to forecast discharge at current day with 1, 2, 3, 4, and 5 days of antecedent data for PSONN and MLP, respectively. The inputs were arranged sequentially as time is one of the important factors in the model.

The objective function used is Mean Squared Error (MSE). This optimization objective will ensure that MSE or learning error is getting lesser with the increase of number of iterations as the simulated runoff is getting closer to observed runoff. The performance of the PSONN is measured by the 'coefficient of correlation' (R) and 'Nash-Sutcliffe coefficient' (E^2). These two criteria measure the overall differences between the simulated and observed runoff. R and E^2 values of 1.0 imply a perfect fit. The formulas of these two coefficients are given in Table 1.

LEARNING MECHANISM

The PSONN and MLP models are composed of three layers, namely, the input layer, the hidden layer and the output layer. The PSONN model is investigated with:

Table 1. Statistics for model comparison.

Coefficient	Symbol	Formula
Coefficient of Correlation	R	$\frac{\sum(obs-\bar{obs})(pred-\bar{pred})}{\sqrt{\sum(obs-\bar{obs})^2\sum(pred-\bar{pred})^2}}$
Nash-Sutcliffe Coefficient	E^2	$E^2 = 1-\frac{\sum_i^j(obs-pred)^2}{\sum_i^j(obs-\overline{obs})^2}$
where obs = observed value, pred = predicted value, $\bar{obs}$ = mean observed values, $\bar{pred}$ = mean predicted values and j = number of values.		

a) 1, 2, 3, 4 and 5 antecedent days.

b) Different *c1* and *c2* values ranging from 1.2 to 2.2.

c) Different time interval constants ranging from 0.0025 to 0.03.

d) 16, 18, 20 and 22 number of particles.

e) Different length of training data from 11 months to 23 months tested with 4 to 7 months of testing data.

f) Different max iterations ranging from 300 to 600.

g) Different number of hidden neurons ranging from 70 to 150.

In contrast, the learning mechanism for MLP is investigated with:

a) 1, 2, 3, 4 and 5 antecedent days.

b) 9, 11, 13, 15, 17, 19 and 21 months of training data.

c) 100, 125, 150, 175, 200, 225 and 250 neurons in the hidden layer

d) Learning rate values of 0.2, 0.4, 0.6 and 0.8.

The daily rainfall and runoff data were used for model calibration and validation. The model was initially trained with 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 and 23 months of data taken from the period October 1997 to December 1999. The calibrated model was then tested with new data taken from January 1996 to September 1997, which is about 1/3 of the length of training data.

RESULTS AND DISCUSSION

Results and Discussion for PSONN

Many computations were conducted to find the optimal configuration of PSONN. The effects of each parameter on PSONN are presented below.

Number of Antecedent Days

The aim of investigating the effect of the number of antecedent days is to determine the best time series to produce the best network. As shown in Table 2, the performance of PSONND3 consistently yields the highest correlation and efficiency coefficients compared to other PSONN models. This indicates that PSONND3 is the best model for simulating daily runoff in Bedup Basin.

Acceleration Constants c_1 and c_2

The performance of PSONN improved as c_1 and c_2 values increased from 1.2 to 2.0. This is because the particles are attracted towards "pbest" and "gbest" with higher acceleration and abrupt

Table 2. The performance of PSONN with different number of antecedent days.

Different Antecedent Days	Training		Testing	
	R	E^2	R	E^2
PSONND1	0.680	0.6439	0.797	0.6516
PSONND2	0.826	0.7358	0.819	0.7590
PSONND3	0.872	0.7754	0.900	0.8067
PSONND4	0.773	0.6838	0.795	0.6656
PSONND5	0.687	0.7448	0.681	0.6588

Note: PSONN calibrated with c_1 and c_2 values of 2.0, 20 number of particles, Δt of 0.01, 21 months of training data, 7 months of testing data, 500 numbers of max iteration and 100 numbers of hidden neurons.

movements when c_1 and c_2 values increased from 1.2 to 2.0 since c_1 and c_2 parameters are determining the particles acceleration. At *c1* and *c2* values of 2.2, PSONN was unable to converge. The best c_1 and c_2 values for PSONN are 2.0 where the values of R and E^2 were 0.872 and 0.7754 for training, 0.900 and 0.8067 for testing, respectively. The results for PSONN trained with different c_1 and c_2 values are shown in Table 3.

Table 3. Performance of PSONND3 according to different *c1* and *c2* values.

c_1 and c_2 values	Training		Testing	
	R	E^2	R	E^2
1.2	0.700	0.5871	0.771	0.6638
1.4	0.665	0.7100	0.718	0.6618
1.6	0.626	0.7070	0.827	0.6212
1.8	0.778	0.7088	0.704	0.6454
2.0	0.872	0.7754	0.900	0.8067
2.2	0.646	0.6244	0.756	0.7161

Note: PSONND3 trained with 21 months of training data, 7 months of testing data, Δt of 0.01, 20 numbers of particles, 500 max iteration and 100 hidden neurons.

Time Interval (Δt) Constant

The performance of PSONN is increased as Δt increases from 0.0025 to 0.01. PSONN is unable to converge at Δt of 0.0025 due to the high granularity movement within the solution space. Then, the PSONN performance is decreased as Δt decreases from 0.01 to 0.03 due to the low granularity movement within solution space. Results show that the optimal Δt for PSONN in this study is consistently given as 0.01. Table 4 represents the performance of PSONN when trained and tested with different Δt values.

Table 4. Performance of PSONND3 according to different time interval (Δt) constant.

Δt constant values	Training		Testing	
	R	E^2	R	E^2
0.0025	0.775	0.6340	0.841	0.5894
0.0050	0.814	0.8164	0.781	0.6861
0.0100	0.872	0.7754	0.900	0.8067
0.0150	0.744	0.6780	0.816	0.6891
0.0200	0.639	0.7713	0.859	0.6644
0.0250	0.749	0.6983	0.646	0.6544
0.0300	0.774	0.6284	0.729	0.7357

Note: PSONND3 trained with c_1 and c_2 values of 2.0, 21 months of training data, 7 months of testing data, 20 numbers of particles, 500 max iteration and 100 hidden neurons.

Number of Particles

The performance of PSONN is increased with the increase of number of particles from 16 to 20. PSONN is unable to simulate well with 16 particles because the space covered in the problem is not sufficient. Then, at 22 particles, the PSONN performance starts decreasing (Table 5) due to the space covered in solving the problem being too wide. The best number of particles was found

to be 20. It was observed that the optimization period was getting longer with the increase of number of particles. This is because when more particles are presented, the amount of space that is covered in the problem is greater and therefore, more optimization time is required.

Table 5. Performance of PSONND3 according to numbers of particles.

Number of Particles	Training		Testing	
	R	E^2	R	E^2
16	0.711	0.6501	0.757	0.7105
18	0.848	0.7325	0.843	0.7229
20	0.872	0.7754	0.900	0.8067
22	0.690	0.6529	0.802	0.6613

Note: PSONND3 trained with c_1 and c_2 values of 2.0, 21 months of training data, 7 months of testing data, Δt of 0.01, 500 max iteration and 100 hidden neurons.

Length of Training and Testing Data

The performance of PSONN is increased as the length of training is increased. It was found that a minimum of 21 months of training data was required to achieve best accuracy, and yielded R = 0.872 and E^2 = 0.7754 values for the training data set and R = 0.900 and E^2 = 0.8067 for validation data set. Table 5 shows that when more data are used, PSONN may perform better (produce higher coefficient results), because a more accurate determination of the synaptic weights is made by the PSONN. The results of PSONN calibration with different lengths of training and testing data are tabulated in Table 6.

Table 6. Performance of PSONND3 according to length of training and testing data investigated.

Length of Training Data	Training		Length of Testing Data	Testing	
	R	E^2		R	E^2
11 months	0.815	0.6971	3 months	0.820	0.8156
12 months	0.640	0.6711	4 months	0.650	0.6741
13months	0.687	0.7734	4 months	0.838	0.8010
14 months	0.839	0.7150	4 months	0.851	0.7076
15 months	0.802	0.6078	5 months	0.722	0.6291
16months	0.779	0.7734	5 months	0.814	0.7928
17 months	0.762	0.6963	5 months	0.772	0.7213
18 months	0.859	0.7575	6 months	0.814	0.7865
19months	0.811	0.7655	6 months	0.878	0.8639
20 months	0.808	0.7221	6 months	0.831	0.7245
21months	0.872	0.7754	7 months	0.900	0.8067
23months	0.800	0.7654	7months	0.855	0.7652

Note: PSONND3 calibrated with c_1 and c_2 of 2.0, 20 number of particles, Δt of 0.01, 500 max iteration and 100 hidden neurons.

Number of Maximum Iteration

The performance of PSONN was also investigated using different numbers of maximum iterations ranging from 300 to 600. When R is used as the criterion of performance, the best maximum number of iterations obtained was 500 and yielded R=0.872 and R=0.900 for training and testing respectively. When E^2 is used as the criterion, the best maximum iteration is 550 with and E^2=0.8433 and E^2=0.8222 for training and testing respectively. The best maximum number of iterations adopted was 500 to avoid over training and over fitting of model. Once over trained, PSONN will only simulate accurately for trained data, but is unable to simulate for different sets of input data accurately. If the maximum number of iterations is not sufficient (from 300 to 450), the network is under trained and unable to approximate any continuous function to any degree of accuracy. In contrast, when too many iterations are used (at maximum iteration of 550 and 600), PSONN may be over trained and sometimes might over fit the data. Table 7 compares the effects of various maximum iterations to PSONND3.

Table 7. The performance of PSONND3 with different maximum iteration.

Maximum Iteration	Training		Testing	
	R	E^2	R	E^2
300	0.811	0.6366	0.785	0.7606
350	0.770	0.6777	0.654	0.6176
400	0.872	0.7375	0.858	0.7702
450	0.845	0.7713	0.893	0.8028
500	0.872	0.7754	0.900	0.8067
550	0.822	0.8433	0.840	0.8222
600	0.809	0.7855	0.801	0.7988

Note: PSONND3 calibrated with c_1 and c_2 values of 2.0, 20 numbers of particles, Δt of 0.01, 21 months of training data, 7 months of testing data and 100 hidden neurons.

Number of Hidden Neurons

The investigation initially was started from 10 hidden neurons. However, it was observed that 10 to 60 hidden neurons are unable to produce accurate results. Thus, the simulations reported here have been carried out with 70 to 150 hidden neurons in the hidden layer. Increasing the number of hidden neurons from 70 to 100 increases the accuracy of simulation results as shown in Table 8. However, the performance of PSONND3 decreases from 125 hidden neurons to 150 hidden neurons. This is because when the number of hidden neurons is small, the network may not have sufficient degrees of freedom to learn the process correctly. In contrast, if the number is too high, the network will take a long time to get trained and may sometimes over fit the data. In this study, the optimal PSONN model uses 100 hidden neurons.

Table 8. Performance of PSONND3 according to different number of hidden neurons investigated.

Different Hidden Neurons	Training		Testing	
	R	E^2	R	E^2
70	0.776	0.6111	0.826	0.7359
80	0.673	0.6482	0.826	0.6571
90	0.779	0.6940	0.831	0.7960
100	0.872	0.7754	0.900	0.8067
110	0.790	0.7315	0.784	0.7910
120	0.832	0.7309	0.827	0.7281
130	0.800	0.7177	0.797	0.6909
140	0.770	0.7194	0.709	0.6776
150	0.721	0.6653	0.687	0.6653

Note: PSONND3 calibrated with *c1* and *c2* values of 2.0, 20 number of particles, Δt of 0.01, 21 months of training data, 7 months of testing data and 500 numbers of max iteration.

Results and Discussion for MLP

Different Antecedent Data

The number of antecedent data will determine the best time series to produce the best network. Besides, the number of antecedent data will also determine the number of input nodes to the input layer. Results revealed that the performance of the MLP network increased rapidly with the increase of antecedent data starting from MLPD1 to MLPD5. In this study, the best number of antecedent days is found to be 5 where R and E^2 obtained are 0.911 and 0.8155 respectively. This indicated that 5 antecedent days offers sufficient degrees of freedom for MLP network to learn the process correctly. Table 9 presents the effect of different antecedent days on the MLP network.

Table 9. Results for MLP network at different number of antecedent days.

Different Antecedent Days	Training		Testing	
	R	E^2	R	E^2
MLPD1	1.000	1.000	0.686	0.6651
MLPD2	1.000	1.000	0.833	0.6867
MLPD3	1.000	1.000	0.851	0.7649
MLPD4	1.000	1.000	0.903	0.8091
MLPD5	1.000	1.000	0.911	0.8155

Note: LR=0.8, 1000 epochs, 175 hidden nodes, TRAINSCCG, 15 months of training data, 6 months testing data.

Different Length of Training Data

The performance of MLP network is clearly lower than expectations when trained with short length of training data. Hence, the minimum length of training data presented here is 9 months. With 9 months training data using TRAINSCG, 150 hidden neurons, learning rate of 0.8 and 5 antecedent data, the R and E^2 values given by MLPD5 for testing is only 0.866 and 0.7036 respectively. As the length of training data increase from 9 months to 15 months, the R and E^2 values increase to 0.911 and 0.8155 using 15 months of training data (refer to Table 10). This indicates that 15 months of training data is sufficient for MLPD5 to determine the synaptic weights of MLP network. However, MLPD5 performance is decreased at 17 and 19 months of training data. The MLPD5 slightly recovers to R=0.910 and E^2=0.8020 for 21 months of training data. Therefore, the optimum length of training data required is 15 months.

Table 10. Results for MLPD5 at different length of training data.

Length of Training Data	Training		Length of Testing Data	Testing	
	R	E^2		R	E^2
9 months	1.000	1.000	3 months	0.866	0.7036
11 months	1.000	1.000	3 months	0.875	0.7493
13 months	1.000	1.000	4 months	0.891	0.7922
15 months	1.000	1.000	5 months	0.911	0.8155
17 months	1.000	1.000	5 months	0.899	0.8002
19 months	1.000	1.000	6 months	0.878	0.7911
21 months	1.000	1.000	7 months	0.910	0.8020

Note: LR=0.8, 5 antecedent data, 1000 epochs, 150 hidden nodes, TRAINSCG.

Different Number of Hidden Neurons

The performance of MLP network is increased with the increase of number of hidden nodes (refer to Table 11) from 100 hidden neurons to 150 hidden neurons. Meanwhile, it was also found that the training period was getting longer with the increase of number of hidden nodes. The performance of MLPD5 decreased as the hidden neurons increased from 150 to 250. During this investigation, the best number of hidden neurons was found to be 150 with R and E^2 of 0.911 and 0.8155 respectively. This is because MLPD5 has sufficient degrees of freedom to learn the process correctly at 150 hidden neurons.

Different Learning Rate Value

The effect of different learning rate starting from 0.2 to 0.8 is shown in Table 12. It was found there is no clear relationship between learning rate and the performance of MLP network. However, all the learning rates investigated are able to converge the data. Meanwhile, the training period was reduced with the increase of learning rate values. Therefore, the recommended learning rate value for this particular MLP network is 0.8.

Table 11. Results of MLPD5 at different number of hidden nodes.

Different Hidden Neurons	Training		Testing	
	R	E^2	R	E^2
100	1.000	1.000	0.883	0.7656
125	1.000	1.000	0.885	0.7718
150	1.000	1.000	0.911	0.8155
175	1.000	1.000	0.910	0.8081
200	1.000	1.000	0.907	0.8079
225	1.000	1.000	0.904	0.8043
250	1.000	1.000	0.901	0.8016

Note: Trained with 15 months data, Tested with 6 months data, LR=0.8, 1000 epochs, 5 antecedent data, TRAINSCG.

Table 12. Results of MLPD4 at different learning rate value.

Different Learning Rate	Training		Testing	
	R	E^2	R	E^2
0.2	1.000	1.000	0.897	0.7844
0.4	1.000	1.000	0.910	0.8273
0.6	1.000	1.000	0.906	0.8059
0.8	1.000	1.000	0.911	0.8155

Note: Trained with 15 months data, Tested with 6 months data, 150 hidden nodes, 1000 epochs, 5 antecedent data, TRAINSCG.

BEST CONFIGURATION

The PSONN has shown encouraging and promising results in terms of simulating daily runoff. The optimal configuration of PSONN for modeling daily rainfall-runoff relationship was found to be:

a) 3 antecedent days

b) c_1 and c_2 values of 2.0

c) time interval of 0.0100

d) 20 particles

e) 21 months of training data and 7 months of testing data

f) 500 max iteration

g) 100 hidden neurons

Meanwhile, the optimal configuration for MLP network was found to be as follow:

a) Number of antecedent days = 5

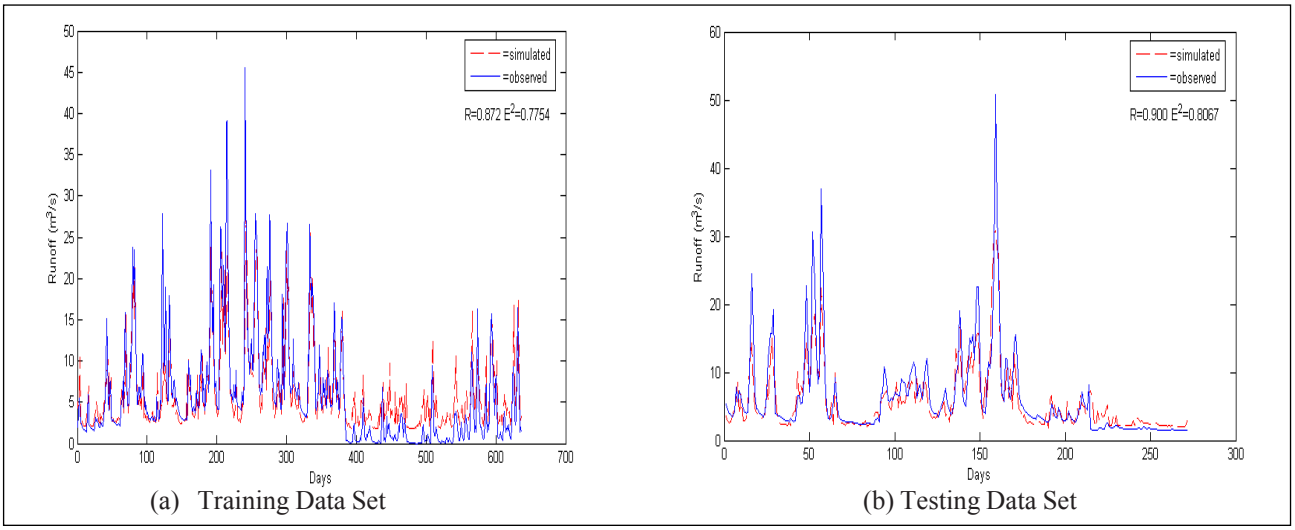
b) Length of training data = 15 months

c) Number of hidden nodes = 150

d) Learning rate value = 0.8

The comparison between simulated and measured runoff for the optimum PSONN is presented in Figure 4. The results show the simulated runoff generated by optimum PSONN is successful in approaching the highest peak of measured runoff for both training and testing data. However, the simulated runoff generated by PSONN is less successful in calculating low runoff especially where the runoff that is approaching zero.

The comparison between observed and simulated runoff for testing obtained using MLP optimal configuration is presented in Figure 5.



(a) Training Data Set (b) Testing Data Set

Figure 4. Comparison between simulated and measured runoff for the optimum PSONN investigation.

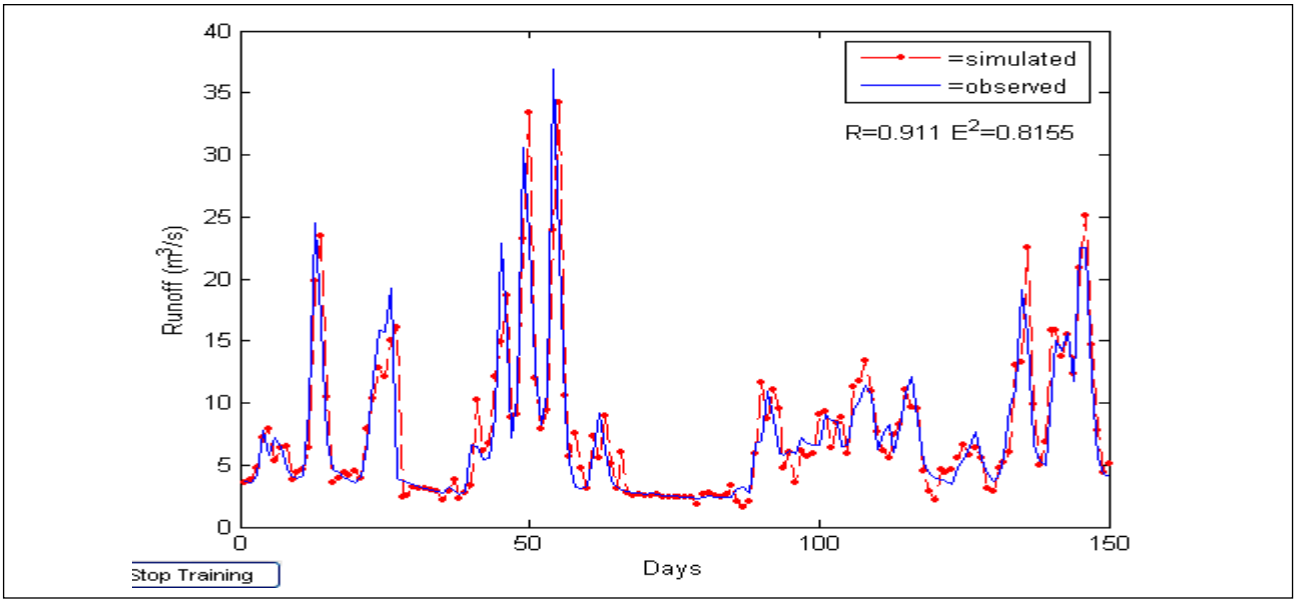


Figure 5. Comparison between observed and simulated runoff using optimum MLP configuration.

CONCLUSIONS

A new type of neural network algorithm (PSONN) is successfully applied to solve the optimization problem, particularly in calibrating the rainfall-runoff model relationship for the Bedup Basin, Malaysia. This is shown by the results where PSONN displays a promising ability to simulate daily runoff accurately. The best model is found to be PSONND3 with a configuration of c_1 =2.0, c_2 = 2.0, time interval of 0.0100, 20 particles, 21 months of training data and 7 months of testing data, 500 max iterations, and 100 hidden neurons. The optimal results obtained are R = 0.872 and E^2 = 0.7754 for training and R = 0.900 and E^2 = 0.8067 for testing data.

The results also revealed that the best performance MLP uses 5 antecedent days, 15 months of training data, learning rate of 0.8 and 150 hidden nodes in hidden layer, where the R and E^2 are 0.911 and 0.8155 respectively for testing. Generally, the performance of MLP is slightly better than PSONN currently.

Results revealed that the performance of newly developed PSONN is comparable with the well-known more conventional MLP. However, the research for PSONN in hydrology is still in the nascent stage, as shown by using the basic PSO algorithm as a hybrid with NN in this study. The

future research directions of PSONN would be adapting newly developed PSO techniques to hybridize with NN, such as the modified dynamic neighborhood PSO algorithm, guaranteed convergence PSO algorithm, and multi-objective PSO optimization algorithm, to improve the convergence speed and obtain better accuracy of simulation results.

ACKNOWLEDGMENTS

The main author would like to thank Associate Professor Dr. Sobri Harun from Department of Hydraulics and Hydrology, Faculty of Civil Engineering, University Technology Malaysia, Johor, Malaysia and Professor Dr. Siti Mariyam Shamsuddin from Department of Computer Graphics and Multimedia, Faculty of Computer Science and Information System, University Technology Malaysia, Johor, Malaysia for their invaluable excellent guidance, technical support, encouragement, concern, critique, and advice throughout the research work, including reviewing and correcting this paper.

REFERENCES

Al-kazemi, B., and C.K. Mohan. 2002. Training feedforward neural network using multi-phase particle swarm optimization. Proceeding of the 9th International Conference on Neural Information Processing, New York.

Bessaih, N., Y.S. Mah, S.M. Muhammad, K.K. Kuok, and A.B. Rosmina. 2003. Artificial neural networks for daily runoff simulation. Proceeding of Engineering and Technology Conference 2003, Kuching, Sarawak, Malaysia.

Dastorani, M.T., and N.G. Wright. 2001. Artificial neural network based real-time river flow prediction. School of Civil Engineering, University of Nottingham, Nottingham NG7 2RD, UK.

Demuth, H., and M. Beale. 2001. Neural network toolbox - for use with MATLAB. The Math Works, Inc.

DID. 2004. Hydrological year book year 2004. Department of Drainage and Irrigation Sarawak, Malaysia.

Eberhart, R., and Y. Shi. 2001. Particle swarm optimization: developments, application and resources. IEEE, Vol. 57, pp. 81-86.

Elshorbagy, A., S.P. Simonovic, and U.S. Panu. 2000. Performance evaluation of artificial neural networks for runoff prediction. Journal of Hydrologic Engineering, Vol. 5(4), pp. 424-427.

Garcia-Bartual, R. 2002. Short term river flood forecasting with neural networks. Universidad Politecnica de Valencia, Spain, pp. 160-165.

Gautam, M.R., K. Watanabe, and H. Saegusa. 2000. Runoff analysis in humid forest catchment with artificial neural networks. Journal of Hydrology, Vol. 235, pp. 117-136.

Harun, S., A.H. Kassim, and T.V.N. Van. 1996. Inflow estimation with neural networks. Proceeding of 10th Congress of the Asia and Pacific Division of the International Association for Hydraulic Research, pp. 150-155.

Imrie, C.E., S. Durucan, and A. Korre. 2000. River flow prediction using artificial neural networks: generalization beyond the calibration range. Journal of Hydrology, Vol. 233, pp. 138-153.

Irwan, A.N., S. Harun, and M.K. Hashim. 2007. Radial basis function modeling of hourly streamflow hydrograph. Journal of Hydrologic Engineering, ASCE/January/February, pp. 113-123.

Jones, M.T. 2005. AI application programming. 2nd Ed. Hingham, Massachusetts.

JUPEM. 1975. Jabatan ukur dan pemetaan Malaysia. Scale 1:50,000.

Nishimura, S., and T. Kojiri. 1996. Real-time rainfall prediction using neural network and genetic algorithm with weather radar data. Proceeding of 10th Congress of the Asia and Pacific Division of the International Association for Hydraulic Research, pp. 204-211.

Palm, W.J. 2001. Introduction to MATLAB 6 for engineers. Mc Draw-Hill Companies, Inc. United States of America.

Shi, Y., and R.C. Eberhart. 1998. A modified particle swarm optimizer. Proceeding of the 105 IEEE Congress on Evolutionary Computation, pp. 69–73, May 1998.

Van den Bergh, F. 2001. An analysis of particle swarm optimizers. Ph.D dissertation, University of Pretoria, South Africa.

Wright, N.G., and M.T. Dastorani. 2001. Effects of river basin classification on artificial neural networks based ungauged catchment flood prediction. Proceeding of the 2001 International Symposium on Environmental Hydraulics.

Zweiri, Y.H., J.F. Whidborne, and L.D. Sceviratne. 2003. A three-term backpropagation algorithm. Neurocomputing, Vol. 50, pp. 305-318.

ADDRESS FOR CORRESPONDENCE
Kuok King Kuok
Department of Hydraulics and Hydrology
Faculty of Civil Engineering
University Technology Malaysia
81310 UTM, Johor
Malaysia

Email: kelvinkuok100@gmail.com