 © 2007 ASCE

# An Intelligent Hybrid Genetic Annealing Neural Network Algorithms for Runoff Forecasting

**Huang Mutao**

（Ph.D, Center of Digital Engineering, Huazhong University of Science and Technology ,Luoyu Road 1037#, Wuhan City, Hubei Province, China, 430074 Fax: 86-27-87543992, Phone: 86-27-65011829, Email of corresponding author: rosemtcherish@gmail.com）

**Abstract**

This study tackles the problem of modeling of the complex, non-linear, and dynamic runoff process. To overcome local optima and network architecture design problems of ANN to make runoff forecasting of catchment more accurate and fast , an hybrid intelligent genetic annealing neural network (IHGANN ) algorithms is established by recombining and improving artificial neural network(ANN) and genetic algorithm (GA). The typical approach can be regarded as a hybrid evolution and learning system which can combine the strength of back propagation (BP) in weight learning and GA's capability of global searching the architecture space. However, the standard genetic algorithm(SGA) adopts constant crossover probability as well as invariable mutation probability. It has such disadvantages as premature convergence, low convergence speed and low robustness. Common adaptation of parameters and operators for SGA is hard to obtain high-quality solution, though it promotes the convergence speed. To address this problem, the IHGANN algorithm applies the simulated annealing algorithm to increase the fitness properly, the self adaptation technology to adjust the value of crossover probability and mutation probability. Meanwhile, a fitness normalization formula is introduced and it always gets a positive value. The new formula can guide the population to a proper direction and increase the press for selection of individuals. The similarity is defined to increase the varieties of individuals without increasing the size of population, thus solving the problem of local optimized solution. Moreover, IHGANN's real encoding scheme allows for a flexible and less restricted formulation of the fitness function and makes fitness computation fast and efficient. This makes it feasible to use larger population sizes and allows IHGANN to have a relatively wide search coverage of the architecture space.

In order to verify the feasibility and validity of the IHGANN, we give an example for some watershed located on the Jinsajiang river basin, Yunan province ,southwest China and carry out serial simulation experiments by using BP, the IHGANN separately. The simulations showed that problems faced by both back propagation algorithm and standard genetic algorithm were overcame by IHGANN.

Compared with BP, the IHGNN has faster convergence speed and higher robustness. Lastly, an dynamic intelligent interactive interface of the runoff forecasting system is developed by using the VC.net programming language.
**Key word:** Artificial Neural Networks; Genetic Algorithm; Simulated Annealing Algorithm; Runoff Forecast; Intelligent optimization

## 1 Introduction

The modelling of the process representing runoff occupies an very important place in the study of watershed hydrology. Thus, the development of a relationship which is capable of providing, as nearly as possible, a true representation of the runoff process has become increasingly indispensable in the decision making process of water resources planning and management. However, the runoff process involves many highly complex components, such as interception, depression storage, infiltration, overland flow, interflow, percolation, evaporation, and transpiration. The various physical mechanisms governing the river flow dynamics act on a wide range of temporal and spatial scales. Meanwhile, most of the hydrological processes in general and rainfall–runoff process in particular are non-linear and dynamic in nature and accordingly relationship developed considering the watershed system as non-linear and dynamic may provide a better representation. Development of mathematical relationships representing the runoff process, based on field studies or laboratory experiments, is time consuming, labor intensive and therefore expensive.

During the past few decades, a great deal of research has been devoted to the modeling and forecasting of river flow dynamics[1-5]. Such efforts have led to the formulation of a wide variety of approaches and the development of a large number of models. The existing models for runoff forecasting may broadly be grouped under three main categories:(1) physically based distributed models; and (2) empirical black-box models (3) conceptual models. Each of these types of models has its own advantages and limitations.

The physically-based models are specifically designed to mathematically simulate or approximate (in some physically realistic manner) the general internal sub-processes and physical mechanisms that govern the river flow process, whereas the black-box models are designed to identify the connection between the inputs and the outputs, without going into the analysis of the internal architecture of the physical process. While the physically-based models are very useful to our understanding of the physical mechanisms involved in the river flow (or any other hydrological) process, unfortunately, they also possess great application difficulties, essentially for the following reasons: (1) they require a large number of parameters pertaining to rainfall, physiography, soil type, cropping system and management practices for modeling the complexity of river flow dynamics ; and (2) extension of a particular model to even slightly different situations is very difficult[1].

The black-box models, on the other hand, though may not necessarily lead to a

better understanding of the river flow process (in a physically realistic manner), have an advantage in that they are easier to apply for even different conditions since the modeling and forecasting procedure is usually analogous. Furthermore, the analysis of the characteristic parameters of the black-box models can furnish useful information on the dynamics of the phenomenon. In the absence of accurate information about the physical mechanisms underlying or the 'exact' equations involved in the dynamics of river flow at a particular location, the use of black-box models seems to have an edge over the use of the physically- based model, since the former is capable of representing arbitrarily the complex non-linear river flow process, by relating the inputs and the outputs of the underlying system. The accuracy of developed black-box models depends to a large extent on the accuracy of its estimated parameters. To arrive at some logical estimation of parameters in general the developed models are calibrated by comparing the estimated and measured output values. Calibration which is basically an optimization process is labor intensive and based on the trial and error procedures. Conventionally, a model is calibrated by manipulating the parameters until the difference in model estimated values and actual output measurements, is minimal.

Artificial neural networks (ANNs) are frequently used for this purpose. ANN is a model inspired from the architecture of the brain, is well suited to such tasks as pattern recognition, combinatorial optimization, and discrimination. These tools contain no preconceived ideas about the manner in which a model ought to be structured or work. It also provides a flexible approach, with the power to provide different levels of generalization, and can produce a reasonable solution from small data sets. The modeller has control over the data inputs and irrelevant variables can be identified or removed during the model building process. There are numerous studies related to the application of ANNs to various problems frequently encountered in water resources[1-9]. The application domains of ANNs include the rainfall-runoff relationship, river runoff forecasting, various groundwater problems, unit hydrograph derivation, regional flood frequency analysis, estimation of sanitary flows and modeling hydraulic characteristics of severe contraction. In the majority of these studies feed forward neural network with back propagation algorithm , FFBP, is employed to train the neural networks. It was shown that multi layer perceptrons with FFBP method perform better than conventional statistical and stochastic methods in hydrological forecasting. However these algorithms have some drawbacks. They are very sensitive to the selected initial weight values and may provide performances differing from each other significantly. Another problem faced during the application of ANNs is the local minima issue. During the training stage the networks are sometimes trapped by the local error minima preventing them to reach the global minimum.

Recently, considerable attention has been paid on stochastic optimization techniques such as genetic algorithms (GAs) and simulated annealing(SA)[10]. The

main advantage of using the stochastic optimization algorithm is that it can solve the problem with arbitrary initial guesses and may give optimal results without any rules. Genetic algorithms (GAs) have been shown to have advantages over classical optimization methods (Holland, 1975; Goldberg, 1989) and have become one of the most widely used techniques for solving a number of hydrology and water resources problems[13,14]. Genetic algorithms (GAs) are search algorithms that are based on Darwin's natural selection theory of evolution where a population is progressively improved by selectively discarding the not-so-fit population and breeding new children from better population. GAs work by defining a goal in the form of a quality criterion (or objective function) and then use this goal to measure and compare solution candidates in a stepwise refinement of a set of data architectures and returns an optimal or nearly optimal solution after a number of iterations. GAs work with numerical values, and can also establish objective functions without difficulty. They are free from a particular model architecture and thereby only require an estimate of the objective function value for each decision set in order to proceed, regardless of whether such information comes from a simple equation or a very complex model. The advantages of GAs over conventional parameter optimization techniques are that they are appropriate for the ill-behaved problem, highly non-linear spaces for global optima and adaptive algorithm.

Another stochastic optimization method employed in this study is simulated annealing(SA). Metropolis et al. (1953)first applied SA in a two-dimensional rigid sphere system. Kirkpatrick et al.(1983) demonstrated the strengths of SA by solving large-scale combinatorial optimization problems. SA is a random search algorithm that allows, at least in theory or in probability, to obtain the global optimum of a function in any given domain[16,18]. One of the advantages of SA is its ability to use a descent strategy which allows random ascent moves to avoid possible traps in a local optimum. Ease of implementation is another advantage of SA. Many research results suggest that SA provide a rapid convergence to "good" solutions[19-28].

These two optimization approaches could obtain the global optimal solutions. However, when the problem or the solution space is fairly complicated, both GA and SA approaches may have the problems of taking much computing time and effort to solve the optimization problem. Differing from the gradient type approach, the stochastic optimization methods should generate the trial solutions in the specified solution space. In addition, all the trial solutions require calculating the objective function values even though those solutions are incorrect. Besides, Youssef et al. (2001) pointed out that if excess population size and/or maximum evolutionary generation were specified, GA also took much time and effort to obtain global optimum solutions[19]. Similar to SA, the local optimum solution would be obtained if the initial temperature given was too low. On the other hand, if a higher initial temperature was given, more time would be consumed for using SA.

To overcome the problem of finding the gradient of the objective function, as

well as trapping of the convergence in local optima, an intelligent hybrid Genetic Annealing ANN algorithm IHGANN is proposed in this study. The hybrid algorithm are the neural network architectures into which the GAs and SA are incorporated. The main advantage of using the hybrid intelligent optimized algorithm is that it can solve the problem with arbitrary initial guesses and may give optimal results without any rules.

This hybrid predictive model differs from previously developed runoff predictive models in the following ways:

(1) input variables of the network (factors affecting runoff) are selected using construction experts' knowledge for each activity/task in question;

(2) the network identifies the sensitivity of input variables via the proposed network parameters;

(3) Runoff predictive models has a multi-layer perceptron network architecture but connection weights ,biases and network architecture parameters can be adjusted simultaneously by novel hybrid genetic annealing algorithm, which is based on a real-parameter genetic algorithm (RGA) with hybrid crossover operator and mutation operator composing of SA and GA, and adaptive mechanisms to determine the dynamic gene probability. Therefore, so the proposed approach has a more efficient learning mechanism.

(4) it does not assume a predefined functional form and also avoids time consuming experiments with alternative architectures, which is the case in standard multi-layer ANNs.

In order to verify the feasibility and validity of the hybrid intelligent optimized algorithm, the daily hydro series for jinsajiang river located in the southwest China is selected for the method application. The IHGANN forecasts compared well with BP algorithm in terms of the selected performance criteria. The simulations results show that the hybrid algorithm not only overcomes that problems faced by back propagation algorithm , such as the blindness of architecture and initial random weight choice, likely to be trapped by local minima ,rate tardiness of neural network training. and the GA's time-consuming defects, but also improves the network's performance and increase the speed of the network's convergence effectually.

## 2 Theory

### 2.1 ANN

ANNs are parallel architectures that comprise nonlinear processing nodes connected by fixed or variable weights. They can be designed to provide arbitrarily complex decision mappings and are often well suited for used in forecasting. The architecture of a multi-layer ANN is variable, in general, consists of several layers of neurons. The input layer plays no computational role but merely serves to pass the input vector to the network. The terms input and output vectors refer to the inputs

and outputs of the multi-layer ANN and can be represented as single vector. A multi-layer ANN may have one or more hidden layers and finally an output layer. By selecting a suitable set of weights and transfer functions, it is known that a multilayer ANN can approximate any smooth, measurable function between the input and vectors.

The ANN has the ability to learn through training, the training process requires a **set** of training sets, i.e., a series of input and associated output vectors. During training, the **ANN** is repeatedly presented with the training data set and the weights in the network are adjusted iteratively till the desired input-output mapping occurs. The error between the actual and the predicted function values is an indication of how successful the training is.

For a discrete time series with a sample set of $p$ units, consider a mapping function $F$ that maps an *m*-dimensional input or data space $R^m$ to a *n*-dimensional output or target space $R^n : F : R^m \to R^n$ as follows:

$$\left\{\left(x(t), y(t)\right) | \, x \in R^m, y \in R^n, t = 1,2,..., p\right\} \tag{1}$$

Where each of the $t$ known data points comprises an input vector $x(t)$ and a corresponding desired output $y(t)$ .For the construction of such time series mapping, multi-layer neural network is used to solve this problem, $m$ , $u$ and $n$ are the nodes number of input, hidden and output units, respectively. The details on the ANN architectures, connections and transfer functions are available in many of the references cited earlier and hence not repeated here [1-9]. The multi-layer neural network is based on the following equations:

$$\hat{y}_k(t) = f\left(\sum_{j=1}^{u} v_{jk} \cdot \varphi_{a,b}[\sum_{i=1}^{m} w_{ij} \cdot x_i(t) + \theta_j] + r_k\right) \tag{2}$$

Where, $f$ is sigmoid function, $k = 1,2,..., p, t = 1,2,..., p$ ; $x_i$ is thenetwork input; $\hat{y}_k$ is the network output; $w_{ij}$ is the weight from input-layer *i*th node to hidden layer *j*th node. $v_{jk}$ is the weight from hidden-layer; *m* is the numbers of nodes in input layer; *j* is numbers of nodes in hidden layer, *k* is numbers of nodes in output layer; $\theta_j$ is a bias input of *j*th node in hidden layer; $r_k$ is a bias input of *k*th node in output layer.

The key to solve the model is to determine network architecture and model

parameters(such as $w_{ij}$、$v_{jk}$、$\theta_j, r_k$). Since network input is determined by optimal objects, the decision of network architecture is to fix hidden-layer nodes number and transfer function style. Recently, when a neural network is designed, its architecture can be fixed in advance or progressive increase or decrease testing methods can be used. However, it has some defects, for instance, it is quite difficult to find the optimal solution when network architecture is very complex and nerve number is quite large. Therefore, the research here makes full use of the strong global searching ability of genetic algorithm to fix the hidden-layer nodes number, corresponding weights and biases of neural network.

In this study, the performance function of neural networks is defined as follows:

$$E = \frac{1}{2}\sum_{t=1}^{p}\sum_{k=1}^{n}[y_k(t) - \hat{y}_k(t)]^2 \le \varepsilon \qquad (3)$$

where $y_k(t)$is the expected output, $\hat{y}_k(t)$ is the predicted output, $n$is the number of output neurons, and $p$ is the number of training set samples. The stop criterion of network is that network total error is no more than $\varepsilon$,if $E$ is less than the given network training goal $\varepsilon$, network training is finished.

2.2 **Genetic Algorithm**(GA)

Genetic algorithm (GA) is an adaptive global search method that mimics the metaphor of natural biological evolution. Based on Darwin's theory of evolution, the better sub generation in GA will survive and generate the next generation. Naturally, the best generation will have better presentation to get with the conditions. The method can be applied to an extremely wide range of optimization problems. The genetic algorithm differs from other search methods in that this algorithm searches among a population of points, and works with a coding of the parameter set, rather than the parameter values themselves. It also uses objective function information without any gradient information. Owing to its ability to achieve the global or near global optimum, this algorithm has been applied to a large number of combinatorial optimization problems.

The standard genetic algorithm can be defined based on the following equations:

$$SGA = (C, E, P_0, M, \varphi, \Gamma, \phi, T) \qquad (4)$$

Where $C$ is the initial population coding scheme; $E$ is the fitness function; $P_0$ is the initial population; $M$ is the scale of population; $\varphi$ is the selection operator; $\Gamma$ is the crossover operator; $\phi$ is the mutation operator; $T$ is the stop criterion.

GA operates on a population of potential solutions by applying the principle of survival of the fittest to achieve an optimal solution. Every solution in the temporary population is ranked against other solutions based on a fitness criterion.

The selection process determines the number of parameter sets in the current generation that participates in generating new parameter sets for the next generation. Based on their fitness function values, individuals are appropriately selected for recombination. The first step is to assign fitness values to all individuals according to their values of objective function. Sometimes the fitness value needs to be scaled for further use. Scaling is important to avoid early convergence caused by dominant effect of a few strong candidates in the beginning, and to differentiate relative fitness of candidates when they have very close fitness values near the end of run [16]. There are several ways of implementing the selection mechanism. The main ones are: "roulette wheel" selection, tournament selection**,** expected value selection ,uniform raking ,crowing selection, stochastic remainder selection, Elitist selection, rank-based selection, and so on[17,18].

The crossover operator is mainly responsible for the global search property of the GA. The operator is used to create new parameter sets (i.e. offspring), by randomly selecting the location of the two parent parameter sets that were selected to participate in the next generation through selection and exchanges parts of chromosome. Crossover is not effective in environments where the fitness of an individual of the population is not correlated to the expected ability of its representational. The most commonly used crossover methods are single point, two point and uniform crossover[18,19]. For real-value encoding, these crossover methods does not change the value of each variable; so it cannot perform the search with respect to each variable. Therefore, it is not suitable in this study and consequently.

The mutation operator is used to add variability to the randomly selected parameter sets, obtained from the above crossover process. Mutation simply changes the value for a particular gene with a certain probability. It helps to maintain the vast diversity of the population and also prevents the population from stagnating. However, at later stages, it increases the probability that good solutions will be destroyed. Normally, the probability that mutation will occur is set to a low value (e.g., 0.01) so that accumulated good candidates will not be destroyed. For real value coding systems, the values in the randomly selected parameter set are being altered within the feasible parameter range. Several mutation methods are used in real value representation, uniformly distributed mutation, Gaussian mutation, range mutation and nonuniform mutation. These methods differ from each other by the frequency of mutating the parameters within the generation.

GA iterates over a large number of generations until the termination criteria have been fulfilled. The successful application of GA depends on the population size or the diversity of individual solutions in the search space. If GA cannot hold its

diversity well before the global optimum is reached, it may prematurely converge to a local optimum. Although maintaining diversity is the predominant concern of GA, it also reduces the performance of GA. Thus, various techniques have been attempted to find a trade-off between the population diversity and the performance of GA.

**2.3 Simulated Annealing(SA)**

SA is a general-purpose stochastic optimization method that has proven to be quite effective in finding the global optima for many different combinatorial problems. The concept of SA is based on an analogy with the physical annealing process. In the beginning of the process, the temperature is increased to enhance the molecular mobility. Next, the temperature is slowly decreased to allow the molecules to form crystalline architectures. When the temperature is high, the molecules have a high level of activity and the crystalline configurations assume a variety of forms. If the temperature is lowered properly, the crystalline configuration is in the most stable state; Thus, the minimum energy level may be naturally reached[19]. At a given temperature, the probability distribution of the system energy is determined by the Boltzman probability

$$P(E) \infty \exp(-E/kT) \tag{5}$$

where $E$=system energy; $k$=Boltzmann's constant; $T$=temperature; and $P(E)$ = occurrence probability. There exists a small probability that the system may have high energy even at low temperature. Therefore, the statistical distribution of energies allows the system to escape from a local minimum energy. This is the major reason why the solutions obtained from SA may not become trapped as a local optimum or result in a poor solution. The Boltzmann probability is applied in Metropolis' criterionto establish the probability distribution function for the trial solution. The Metropolis' criterion takes the place $\Delta E$ the difference between the current optimal and trial solution original solution and the new solution of $E$, and $k$ being equal to one. The modified Boltzmann probability which represents the probability that the trial solution will be accepted is given as

$$\begin{aligned} T_{(l)} &= T_0 / \log l \\ P_{(l)} &= \exp(-\Delta E/T) = \exp(-(E_r - E_q)/T_{(l)}) \end{aligned} \tag{6}$$

where $l$ denotes an integer time step, $T_0$ is an initial constant temperature, $T_{(l)}$ is a temperature sequence. The objective function values of the current solution and trial solution are represented as $E_q$ and $E_r$, respectively.

The new mutation operator operates as follows. Generate randomly a trial solution from the neighborhood of the current solution, which is obtained after the mutation process of GA. If the value of the new objective function is less than that of

the original objective function, that is $E_r - E_q < 0$, the new solution is better than the old one and it is accepted. On the other hand,if $E_r - E_q \geq 0$, the new one is accepted only when its acceptance probability $P_{(l)}$ given in Eq. (6) is larger than a random value between 0 and 1. There has been much work done about the choosing of the constant $T_0$ in Eq. (6). However, it is still difficult to determine $T_0$ because it is dependent on the strategies used for different problems. In general, $T_0$ is a function of $f_{\max}$ and $f_{\min}$, which represents the maximum and minimum objective function values of the initial population, respectively. In this paper, $T_0$ can be chosen as

$$T_0 = |f_{\min}| \tag{7}$$

where the influence of $f_{\max}$ is excluded. In addition, since a better initial population will lead to a faster convergence to the desired solution, the tournament criterion is applied to obtain a better initial population. Two populations of solutions are randomly generated. Then, one solution is randomly taken from each population of solutions, and the solution that has a higher fitness value can become the solution of the initial population.

## 3 IHGANN for Runoff Forecast

Determining an appropriate architecture of ANN for a particular problem is an important issue since the network topology directly affects its computational complexity and its generalization capability. Different theoretical and experimental studies have shown that larger-than-necessary networks tend to overfit the training samples and thus have poor generalization performance, while too-small networks (that is, with very few hidden neurons) will have difficulty learning the training data. This system first uses three layer (input, hidden and output layers) feedback neural network models, whose inputs and outputs can be any real numbers. Then, the novel IHGANN method is used to optimize the ANN. The main architecture and program process of IHGANN is shown in Fig.1. The details of the IHGANN are described as follows.

### 3.1 ANN Design

In the multivariate ANN forecasting context, the selection of appropriate input variables is very important since it provides the basic information about the system

considered. Thus, a sensitivity analysis is performed to determine the relative importance of each of the input variables. In addition to the daily runoff, several exogenous input variables (such as,previous and current precipitation, temperature, snowmelt, runoff, evaporation) are found relevant to the daily runoff forecasting in this context.

The watershed runoff was modeled as follows using ANN:

$$f(t+d)=\{\ f(t-1),...,f(t-n_f),r(t),r(t-1),...r(t-n_r),T(t),T(t-1),..T(t-n_T),\\ S(t),S(t-1),...S(t-n_S),E(t),E(t-1),...,E(t-n_E)\ \} \quad (8)$$

Where $f(t+d)$ is the ANN output representing daily runoff within certain forecast period, $d$ is the lead time, $n_f, n_r, n_T, n_S, n_E$ are the tapped delay line memory length of runoff, rainfall, temperature, snowmelt, evaporation respectively which are predefined using construction experts' knowledge. The number of ANN input variables $R = n_f + n_r + n_T + n_S + n_E$, the number of ANN output variables is 1.

**3.2 Training and testing details**

Data separation procedure divides the sample sets into three parts: training sets $\phi_1$, used to determine the network weights; validation sets $\phi_2$, used to estimate the network performance and decide when we stop training; prediction or test sets $\phi_3$, used to verify the effectiveness of the stopping criterion and to estimate the expected performance in the future.

The whole samples can be:

$$\phi_1 = \{(x(t), y(t)) \mid x \in R^m, y \in R^n, t = 1,2,..., p_1, p_1 \le p\} \quad (9)$$

$$\phi_2 = \{(x(t), y(t)) \mid x \in R^m, y \in R^n, t = p_1 + 1, p_1 + 2,..., p_2, p_2 \le p\} \quad (10)$$

$$\phi_3 = \{(x(t), y(t)) \mid x \in R^m, y \in R^n, t = p_2 + 1, p_2 + 2,..., p\ \} \quad (11)$$

Where $p_1$ is the sample sets length of training sets $\phi_1$, $p_2 - p_1$ is the sample sets length of validation sets $\phi_2$, and $p - p_2$ is the sample sets length of test sets $\phi_3$.

The following points are noted in order to make the sample sets selection: (1) it is necessary to have a training set that could represent the overall architecture of the runoff series to capture the input–output relationship, which means that it is important to include the extreme events; (2) since the objective is forecasting, it is important to capture the changes in the system with respect to time, which means that

events from the first few years, for instance, should form the basis for the events that follow; and (3) it may be necessary to have a reasonably long data set for training in order to sufficiently capture the dominant characteristics of the system under investigation[7].

**3.3 Preprocess sample sets**

There are several methods to preprocess the sample sets , including using wavelet analysis method to filter and eliminate the yawp of data (the detailed algorithm is omitted here) and considering whether to classify the sample series or not, to abstract the data with uniform features from the mass data group to construct local neural network model.

For the data set considered in the present study, the input variables as well as the target variables are first normalized linearly in the range of 0.1~0.9. This range is selected because of the use of the logistic function (which is bounded between 0.0 and 1.0) as the activation function for the output layer. The normalization is done using the following equation:

$$X_{norm} = 0.1 + 0.8(X - X_{\min})/\left(X_{\max} - X_{\min}\right) \tag{12}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values in the data set, respectively.

**3.4 ANN parameter set**

The synaptic weights of the ANNs are initialized with normally distributed random numbers in the range of -1 to 1. The same initial weights are adopted for all the simulations in one set of simulations in order to make a direct comparison. The training is carried out in a pattern mode and the order of presenting the training samples to the network is also randomized from iteration to iteration.

The accuracy of forecasts is evaluated using a variety of (absolute and relative) error indicators, as follows: mean absolute error (MAE), mean square error (MSE), root mean square error (RMSE),maximum absolute error (MAXAE), minimum absolute error (MINAE), correlation coefficient (r ),coefficient of determination ( $R^2$ ), coefficient of efficiency (E ), and modified coefficient of efficiency(E). Among these, the MAE, the MSE, the RMSE, and $R^2$ are considered the most important.

Two stopping criteria, the error function and the maximum number of iterations, are adopted. The error curves for the training and the testing sets are used to evaluate the convergence speed of the networks. The related training parameter of ANNs, including learning rate $\eta(0 < \eta < 1)$、inertia factor $\alpha$、network training goal , net training time , net training epochs and so on is set properly.

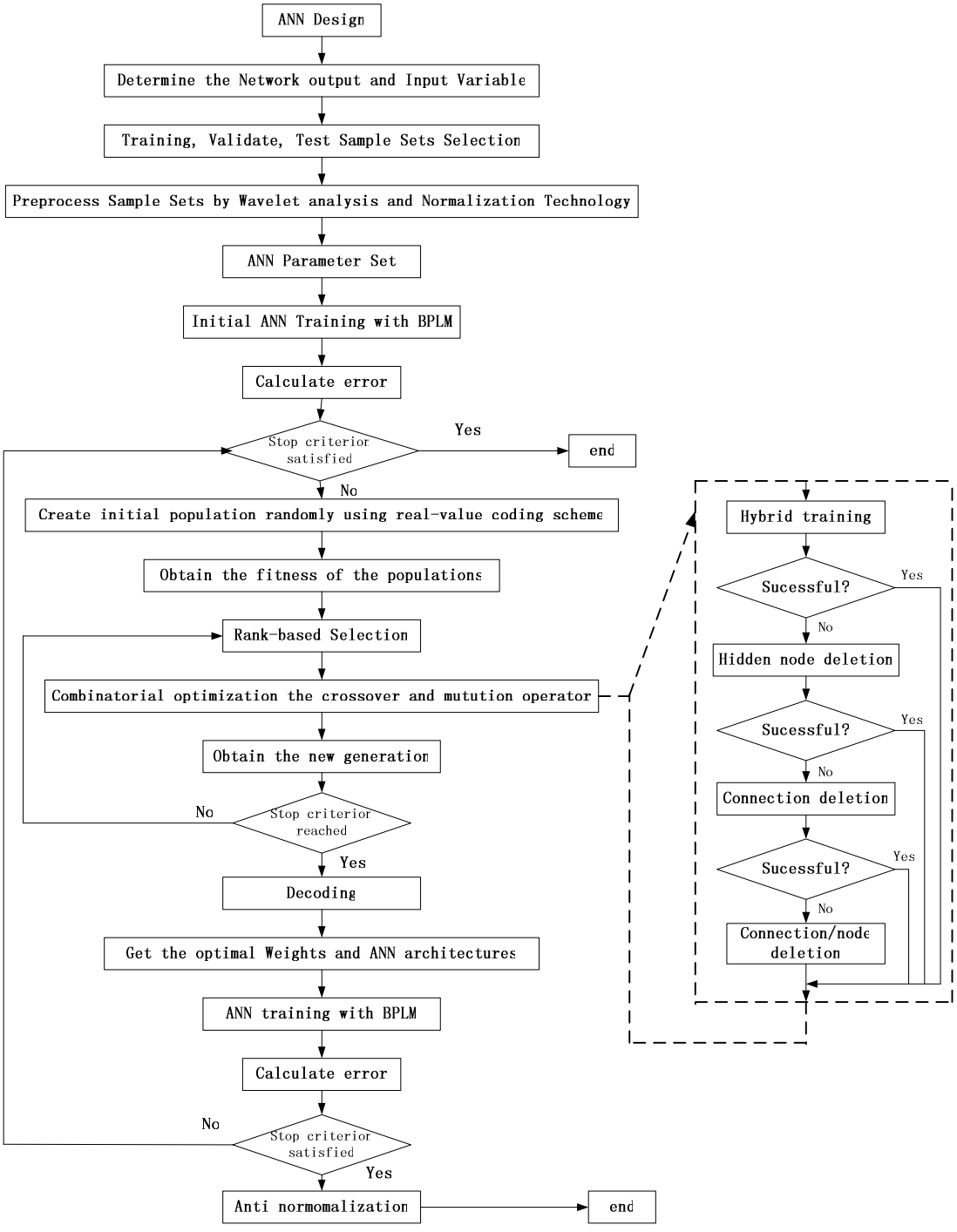



Fig.1 Main architecture and program process of IHGANN

### 3.5 Simultaneous evolution of the ANN architectures and weights

A difficult task with ANNs involves choosing network architecture and model parameters ,such as the number of hidden nodes, and the initial weights.It is known

that a network smaller than needed would be unable to learn, and a network larger than needed would probably end in over-training. For a typical BP evolution where only the weights are adapted and the architecture remains fixed, it is a common knowledge that it is prone to underfitting or overfitting the training data if the size of the network chosen is smaller or bigger than necessary. However, finding the ideal network complexity remains a major problem.

Once we have stated our problem, there are many algorithms that could be applied to its solution. Among the most widely used algorithms for combinatorial optimization are simulated annealing, genetic algorithms, particle swarm optimization and ant colonies[10]. However, we have a prerequisite to be met by any algorithm to be useful in optimization: it must not be computationally expensive. On the other hand, it must also be effective in finding good optima[24]. Mixing these two conditions, in this study, the combinatorial optimization algorithm IHGANN with an adaptive learning mechanism is used. The most important steps of the simultaneous evolution of both architectures and weights can be summarized as follows:

1) Generating initial population. Randomly generating certain number of neural networks as initial population, whose hidden neuron number and linking weights are generated in their initial scope. The number of nodes of hidden layer *h*, is obtained from a uniform distribution: $0 < h \le h_{\max}$ ;each node is created with a number of connections *c*, taken from a uniform distribution: $0 < c \le c_{\max}$ The initial value of the weights is uniformly distributed in the interval $\left[w_{\min}, w_{\max}\right]$.

2)Fitness computation. Evaluate each individual based on its error and/or other performance criteria. According to the giving sample set, training network by "sample counterchanging method", and transforming the computation error of each network as the fitness of training network individual.

3)Select individuals for reproduction and genetic operation by use of the rank-based Selection approach.

4)Apply the self adaptation technology to adjust genetic operators, such as crossover and mutation, to simultaneous evolution the ANN's architectures and weights. It is carried out through the combined use of GA and SA. Then a population of the next generation is created.

5)The fitness of the individuals of the new generation is calculated according to termination criterion, and the process is repeated until the stop criterion (the total evolution generation *K* )is reached.

Some details of the IHGANN algorithm are given as:

1) real-value coding scheme

One major drawback of the standard genetic algorithm (SGA) is that it encodes

parameters as finite-length strings such that much computation time is wasted in the encoding and decoding processes. Hence, a real-parameter genetic algorithm (RGA) is proposed to overcome this problem. Instead of the coding processes, RGA directly operates on the parameters and much computation time is saved. IHGANN uses a real-value coding scheme to represent the chromosome, and each chromosome vector is coded as a vector of real-value point numbers of the same length as the solution vector. Let $x = (x_1, x_2, ..., x_n)$ be the encoding of a solution, here $x_i \in R$ represents the value of the *i*th gene in the chromosome $x$. Initially, $i$ is selected within the desired domain, and reproduction operators of GA are carefully designed to preserve this constraint[21]. As for the genetic operators, RGA is the same as SGA in the reproduction process, but they are different in the crossover and mutation processes in this study.

2) Individual representation

To optimize ANN, it needs to be expressed in proper form. There are some methods to encode an ANN like binary representation, tree, linked list, and matrix [14]. We have used a matrix to encode an ANN since it is straight forward to implement and easy to apply genetic operators. According to requirement of IHGANN model, the individual should include number of hidden neuron and linking weights and thresholds of whole network. The maximum number of hidden nodes $H$ must be predefined in this representation. The number of input nodes and output nodes is dependent on the problem as described before. Though the maximum number of hidden nodes $H$ is pre-defined, it is not necessary that all hidden nodes are used. Some hidden nodes that have no useful path to output nodes will be ignored.

When $N$ is the total number of nodes in an ANN including input, hidden, and output nodes, the matrix is $N \times N$, and its entries consist of connection links and corresponding weights. In the matrix (see Fig. 2), upper right triangle has connection link information that is 1 when there is a connection link and 0 when there is no connection link. Lower left triangle describes the weight values corresponding to the connection link information[15,16,22]. There will be no connections among input nodes. Architectural crossover and mutation can be implemented easily under such a representation scheme. Node deletion and addition involve flipping a bit in the matrix. Fig. 2 shows an example of encoding of an ANN that has two input node, three hidden nodes, and one output node. At the initialization stage, connectivity information of the matrix is randomly determined and if the connection value is 1, the corresponding weight is represented with a random real value. This representation don't allows direct links between input nodes and output nodes.

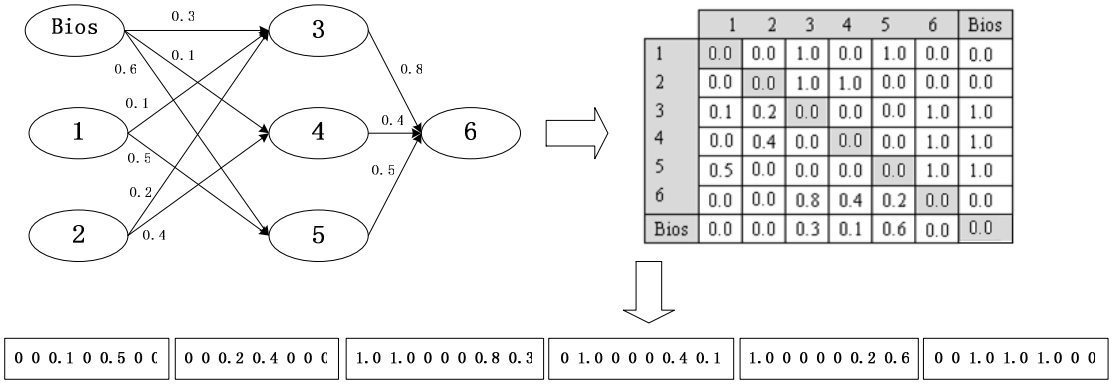



Fig.2 ANN individual representation by means of the linearization of the connectivity matrix for a real-coded genetic algorithm.

3) Expression of fitness function

The individual fitness of neural network is expressed by the following transformation of error function of neural network.

$$f = 1/1 + E \tag{13}$$

Where $E$ has been defined in Eq.(2).

Genetic algorithm determines its searching direction only by the fitness transformed from objective functional value. Eq.(13) is the most common used fitness normalization formula. it always gets a positive value. At the genetic initial stage, some super-normal individuals with high fitness control the selection process, which influences the global optimization performance of the algorithm. In this study, a new fitness normalization formula named fitness stretch method is used to guide the population to a proper direction and increase the press for selection of individuals. The similarity is defined to increase the varieties of individuals without increasing the size of population, thus solving the problem of local optimized solution. The improved fitness normalization formula is expressed as follow:

$$F_i = \exp(f_i / t) / \sum_{i=1}^{m} \exp(f_i / t) \tag{14}$$

$$f_i = 1/\left( \sum_{k=1}^{n} \left( y_k(t) - \hat{y}_k(t) \right)^2 \right), t = t_0\left((0.99)^{K-1}\right) \tag{15}$$

Where $f_i$ *is i*th individual fitness before the improvement; $n$ is input layer neurons number; $y_k(t)$ is the network expected output; $\hat{y}_k(t)$ is the network actual output; $K$ is the current genetic evolutionary generation; $t_0$ is the initial temperature; $t$ is the

current temperature;

4) Selection

In this paper, the rank based fitness selection is used. The selection probability $s_i$ of *i*th individual after ranking operation is determined by the following formula:

$$r = \frac{q}{1-(1-q)^M} \tag{16}$$

$$s_i = r(1-q)^{(b-1)} \tag{17}$$

Where: $q$ is the selection probability of optimal individual; $M$ is the population scale; $r$ *is* the value of normalized q; $b$ is the ranking location of *i*th individual.

5) Combinatorial optimization approach to genetic operator

Some basic steps are needed in the application of hybrid Genetic annealing schedule to the optimization problem. The first step in hybrid algorithm is to initialize a solution and set the initial solution to equal the current optimal solution. The second step is to update the current optimal solution, if the trial solution generated from the initial solution within the boundary is better than the current optimal solution; otherwise, continue generating trial solutions until the algorithm satisfies the temperature decrease criterion. The algorithm will be terminated when hybrid algorithm obtains the optimal solution or the obtained solution satisfies the stopping criteria. In general, the stopping criteria are defined to check whether the temperature or the iteration number reaches the specified value or not.

There are five classes of elemental operators:

(1) Addition of a node. A new node is added, which gets two inputs and one output connection obeying the layer restrictions (i.e., a maximum number of layers).

(2) Deletion of a node. A node is selected randomly and deleted together with its connections. A hidden node, say $h_j$, and all connections to or from $h_j$ are deleted. If $h_j$ was the only input to a hidden node, this node is connected with one of the former inputs to $h_j$, which is chosen randomly. If after the deletion a hidden node has no output connection, this node is connected with one of the former outputs of $h_j$.

(3) Addition of a connection. A connection is added, with 0 weight, to a randomly selected node. A forward connection is added obeying the layer restrictions.

(4) Deletion of a connection. A connection that is not necessary for the network

to be valid is deleted. A randomly selected connection is removed.

(5)Adjustment of Weights, which is an operator that adjusts the weights of the evolved networks. For each weight, a random value is drawn from a Gaussian distribution with zero mean and variance $\sigma^2 = 0.1$ and added to the weight.

In every generation, each parent produces one offspring. Elemental operators are chosen randomly and are applied to the offspring. The crossover operator exchanges the architecture of two ANNs in the population to search ANNs with various architectures. In the population of ANNs, crossover operator selects two distinct ANNs randomly and chooses one hidden node from each ANN selected. These two nodes should be in the same entry of each ANN matrix encoding the ANN to exchange the architectures. Once the nodes are selected, the two ANNs exchange the connection links and corresponding weights information of the nodes and the hidden nodes after that. The mutation operator is used to add variability to the randomly selected parameter sets, obtained from the above crossover process. Mutation simply changes the value for a particular gene with a certain probability.

For real value coding systems, the values in the randomly selected parameter set are being altered within the feasible parameter range. In IHGANN, both the adaptive crossover mechanism and the adaptive mutation mechanism are included. Each step of the algorithm consists of adding a small random value to every weight of the each ANN.

As SA algorithm implies a high computational cost, two modifications to the SA approach are introduced in this study to ensure that the solutions obtained from SA are the optimum solutions[23-26].

In the first modification, an initial point $x$ is required to evaluate the objective function value $f(x)$. Let $x'$ assume the position as the neighbor of $x$ and its objective function value is $f(x')$. In the minimization problem, if $f(x')$ is smaller than $f(x)$, then the trial solution $x'$ takes the place of the current optimal solution $x$. if $f(x')$ is not smaller than $f(x)$, then one has to test Metropolis's criteria and generate a new random number $D$ between zero and one. For solving minimization problems, the Metropolis's criterion is given as :

$$P_{SA}\{accept \quad j\} = \begin{cases} 1 \quad if \quad f(j) \le f(i) \\ \exp(\dfrac{f(i) - f(j)}{T}) \quad if \quad f(j) > f(i) \end{cases} \tag{18}$$

where $f(i)$ and $f(j)$ are, respectively, the function value when

$x = x_i$ and $x = x_j$ . $x_j$ and $x_i$ are, respectively, the current optimal solution and neighborhood trial solution of *x*. Here *T*, a control parameter, is usually the current temperature.

This modified criterion is different from the general Metropolis criterion as mentioned previously. In Eq.(18), the increment between the current best solution and the neighborhood trial solution is divided not only by parameter *T*, but also by the neighborhood trial solution. After the temperature decreases several times, any acceptance probability obtained from the modified Metropolis criterion will be smaller than that obtained from the general Metropolis criterion. The best solution obtained from the modified Metropolis criterion will converge much faster than that using the general Metropolis criterion because unfavorable solutions will not be accepted in the algorithm.

The second modification is to adjust the searching number with a factor $a$ for decreasing temperature. In general, *a* is given as 1.1. Due to an increasing of the searching number, more trial solutions will be created and a much higher possibility will be achieved to obtain the optimal solution.

In searching the optimum solutions, when the best solution keeps the same for some successive generations, the executed algorithm may be stuck at a local minimum, and some changes should be done on the searching strategy of the algorithm. Therefore, adaptive mechanisms are proposed to do the change. In these mechanisms, if the best solution is the same for the lasting $K$ generations and $K > K_{frozen}$ , the crossover probability and mutation probability are changed according to the following two equations.

$$P_c = P_{c0} + \frac{K - K_{frozen}}{K}(\alpha - P_{c0}) \tag{19}$$

$$P_m = P_{m0} + \frac{K - K_{frozen}}{K}(\beta - P_{m0}) \tag{20}$$

where $K_{frozen}$ is a positive integer constant, $P_{c0}$ and $P_{m0}$ are the initial crossover probability and initial mutation probability, respectively. Besides, $\alpha$ and $\beta$ are constant real numbers. If $K \le K_{frozen}$ or the best solution changes such that *K* is reset to zero, the crossover probability and mutation probability remain equal to their initial values

$$P_c = P_{c0} \tag{21}$$

$$P_m = P_{m0} \tag{22}$$

Eqs. (19) and (21) are called adaptive crossover mechanism, and Eqs. (20) and (22) are called adaptive mutation mechanism. If there is no adaptive mechanism included in the algorithm, the crossover probability and mutation probability will remain the same as their initial values as Eqs. (21) and (22). In addition, the elitist strategy, by which the best solution of each generation is copied to the next generation, is adopted here to insure the solution quality.

The five elemental operators are attempted sequentially. If one operator leads to a better off-springs , it regarded successful, no future operator will be applied(see Fig.1). The order of deletion first and addition later encourages the evolution of compact ANNs. It deletes and adds connections probabilistically according to their importance in the ANN. Nodes deletion is done at random ,but node addition is achieve through splitting an existing nodes.

**3.6 Decoding and ANN training**

Get the optimal network weights and biases by decoding the $K$ th generation individual with the highest fitness firstly. Then using BPLM algorithm to train network. With the optimized weights and biases, network will be trained to calculate the error between actual output and expected output. If the stop criterion is satisfied, network training stop, or else, the program goes to step 3.6 to optimize the architecture and weights of ANN again until reach the performance goal.

## 4 Case Study and Results

In order to evaluate how well a model can be applied to approximate the relationship between a set of inputs and a set of outputs, it is necessary to compare the predictive capabilities of a model with existing approaches. The comparison of models is usually accomplished by testing all the models of interest on a data set from the same watershed. Therefore, we give an example for some watershed located on the Jinsajiang river basin, Yunan province ,southwest China and carry out serial simulation experiment by using BP, the IHGANN separately.

The watershed contains 7 rain gauge station and 1 control hydrology station station. The original data consist of 20 years (1981–2000) of daily natural inflows, precipitation (rain and snow), evaporation. In view of the sample sets selection principle mentioned above, it is decided to use 6350 daily data points for analysis in this study. Out of these 6350 points, the first 5400 points, which represent about 85% of the series, are selected as training set, whereas the remaining 950 points, accounting for about 15% of the series, are used for testing the forecasting performance of IHANNS approaches. Also, in the case of IHANNS, the training set of 5400 points is further divided into two parts; training set and validation set. The first 4500 points are selected as the training set and the next 900 points are taken as the validation set. The choice of the length of the validation set is based on the

recommendation to use about 15–20% of the training set. Having said that, the consideration of (only) the first 4500 values of the river flow series for the purpose of training may raise serious questions for (at least) two reasons: (1) the 4500 values used for training happen to contain the highest recorded flow event and also exhibit significant variations; and (2)the testing set used (i.e. the latter part of the series) does not exhibit significant variations. The concern implied in these reasons is that the testing set is less variable and more predictable than the training set and, therefore, there may be a bias in the analysis.

In this study, a three-layer BP neural network with Levenberg–Marquardt learning algorithm is used for daily forecast. Network output is the daily runoff of the control hydrology station. Table 1 summarizes the architecture and input variables for the ANN, the tapped delay line memory length of daily natural inflows, precipitation (rain and snow), evaporation are set respectively.

Table 1 ANN input variables

| Model | Time periods for input variables | | | number of ANN input variables |
|---|---|---|---|---|
| | Precipitation | daily runoff | evaporation | |
| IHGANN(2-1-3) | $t \to t-1$ | $t-1$ | $t \to t-2$ | 6 |
| IHGANN(3-2-7) | $t \to t-2$ | $t-1, t-2$ | $t \to t-6$ | 12 |
| IHGANN(5-3-7) | $t \to t-4$ | $t-1 \to t-3$ | $t \to t-6$ | 15 |
| BP(3-2-7) | $t \to t-2$ | $t-1, t-2$ | $t \to t-6$ | 12 |

The related training parameter of ANNs, include learning rate $\eta = 0.01$, inertia constant $\alpha = 0.15$、network training goal 0.01 , net training epochs 2000, the tansig sigmoid function is taken as the transfer function between input layer and hidden layer, and the logsig sigmoid function as the transfer function between hidden layer and output layer. The performance of the IHANNS approaches for forecasting the runoff series is tested by making forecasts for different lead times, from 1 day to 7 days.

In order to overcome local optima and network architecture design problems of ANN to make runoff forecasting of catchment more accurate and fast,we use IHGANN algorithm to optimize the ANN architectures and weights simultaneously. The population has 50 networks with a maximum of hidden nodes of 50. The probability of mutation is 0.02. The simulated annealing algorithm was run 1000 steps. Fig.3,4 shows the comparison runoff forecast result of 3 IHGANN models and 1 BP model by use of the regression analysis and the output fitting technology. The simulations showed that problems faced by both back propagation algorithm and standard genetic algorithm were overcame by IHGANN. Compared with BP, the HIGNN has faster convergence speed and higher robustness, significantly improves the overall prediction accuracy.

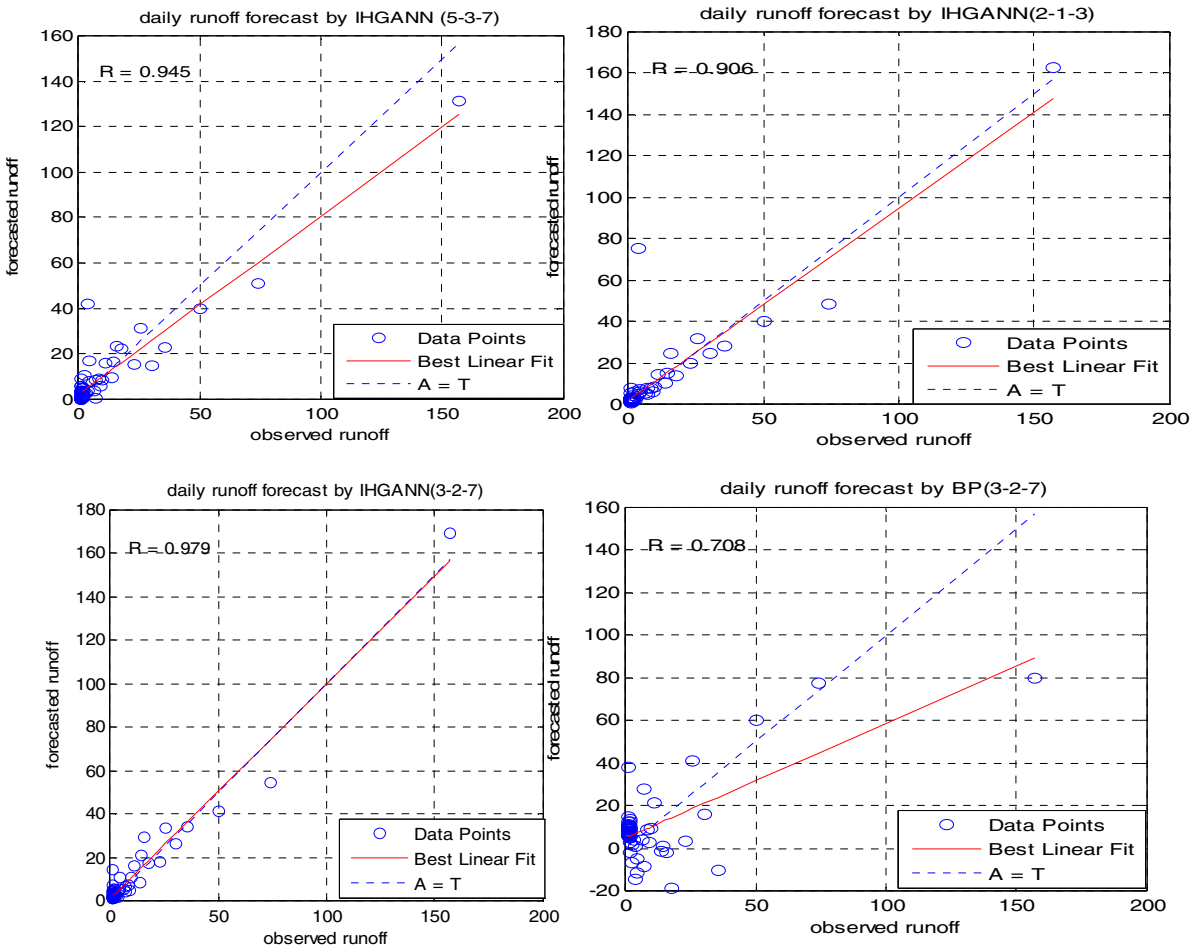
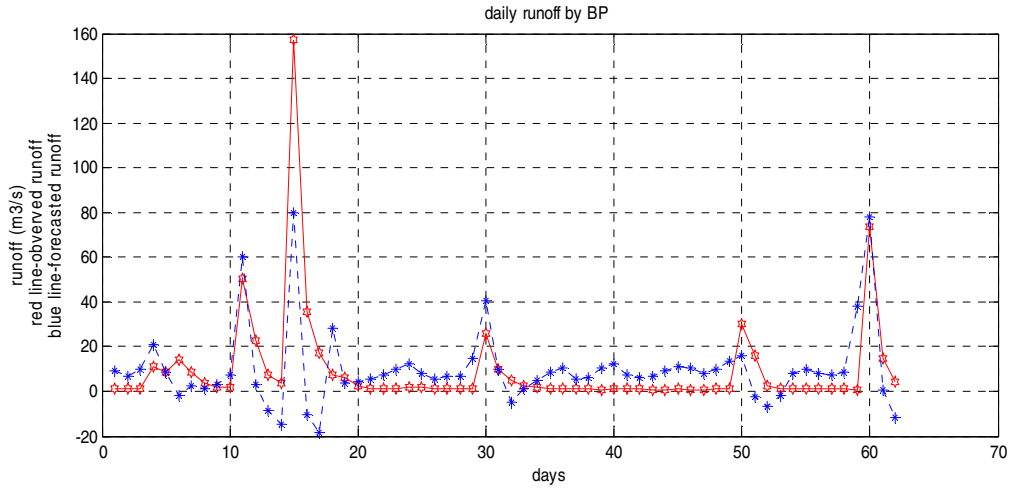



Fig.3 Comparison daily runoff forecast result of 3 IHGANN models and 1 BP model by use of the regression analysis



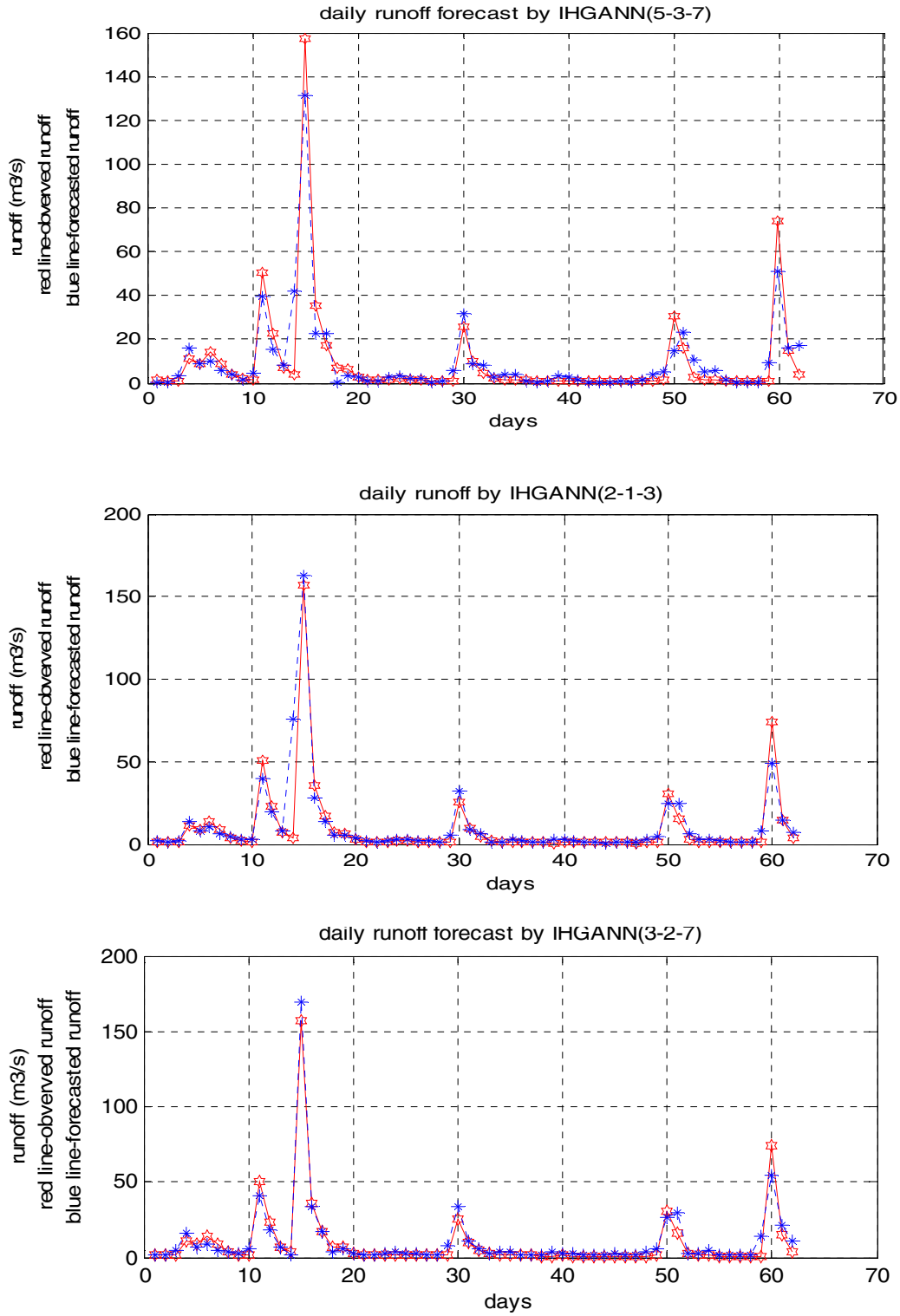



Fig. 4 Comparison daily runoff forecast result of 3 IHGANN models and 1 BP model by use of output fitting technology

Table2 shows the comparative performance of BP Model and IHGANN for daily runoff forecast in terms of relative MSE error index. Tables 2 suggest there is no systematic deterioration in the forecast skill with the growth in the forecast lead time. This may indicate the dynamic forecast skill and robustness of the IHGANN.

Table 2 Comparative Performance of BP Model and IHGANN for Daily Runoff Forecast in terms of Relative MSE error Index

| Lead time (days) | observed flow ($m^3/s$) | BP | | IHGANN | |
|---|---|---|---|---|---|
| | | Forecasted runoff($m^3/s$) | Relative error (%) | Forecasted Runoff ($m^3/s$) | Relative error (%) |
| 1d | 0.37 | 0.364 | 1.6 | 0.371 | 0.27 |
| 2d | 0.37 | 0.361 | 2.4 | 0.366 | 1.1 |
| 3d | 0.4 | 0.379 | 5.25 | 0.394 | 1.5 |
| 4d | 21.4 | 19.98 | 6.64 | 21.02 | 1.78 |
| 5d | 2.41 | 2.2 | 9.72 | 2.39 | 0.83 |
| 6d | 0.75 | 0.67 | 8.71 | 0.73 | 2.67 |
| 7d | 0.65 | 0.71 | 6.15 | 0.64 | 1.54 |

All the algorithms are programmed in MATLAB programming language, and integrated into the runoff forecast simulation system by use of COM technology, the dynamic interactive interface of the runoff forecasting system is developed by using the Visual Studio.C# programming language.

**Reference**

[1] Rao S. Govindaraju, Assoc. Artificial Neural Networks in Hydrology. Journal of Hydrologic Engineering, Vol. 5, No. 2, April,2000,124-137.

[2] P. Coulibaly, F. Anctil, B.Bobee. Daily Reservoir Inflow Forecasting using Artificial Neural Networks. Journal of Hydrology 230 (2000) ,244–257.

[3] Paulin Coulibaly, Francois Anctil, Bernard Bobee. Multivariate Reservoir Inflow Forecasting Using Temporal. Journal of Hydrologic Engineering, Vol. 6, No.5, September/October, 2001,367-376.

[4] Sezin Tokar1, Momcilo Markus. Precipitation-Runoff Modeling using Artificial Neural Networks and Conceptual Models. Journal of Hydrologic Engineering, Vol. 5, No. 2, April, 2000,156-161.

[5] A.Sezin Tokar1 and Peggy A. Johnson. Rainfall-Runoff Modeling using Artificial Neural Networks. Journal of Hydrologic Engineering, Vol. 4, No. 3, July, 1999, 232-239.

[6] R.Baratti,B.Cannas,A.Fanni. River Flow Forecast for Reservoir Management through Neural Networks. Neurocomputing,55(2003),412-437.

[7] Sivakumar, A.W. Jayawardena, T.M.K.G. Fernando . River Flow Forecasting:use of Phase-space Reconstruction and Artificial Neural Networks

Approaches. Journal of Hydrology, 265 (2002) ,225–245.

[8] Cameron M. Zealand, Donald H. Burn, Slobodan P. Simonovic. Short Term Stream Flow Forecasting using Artificial Nneural Networks. Journal of Hydrology 214 (1999), 32–48.

[9] Tsung-yi Pan, Ru-yih Wang. State Space Neural Networks for Short Term Rainfall- Runoff Forecasting. Journal of Hydrology 297 (2004), 34–50.

[10] Nicolas Garcıa-Pedrajas, Domingo Ortiz-Boyer, Cesar Hervas-Martınez. An Alternative Approach for Neural Network Evolution with a Genetic Algorithm Crossover by Combinatorial Optimization. Neural Networks 19 (2006) ,514–528.

[11] N.Garcıa-Pedrajas, C.Hervas-Martınez, J.Munoz-Perez. Multi-objective Cooperative Coevolution of Artificial Neural Networks. Neural Networks 15 (2002) 1259–1278.

[12] WE1 GAO. New evolutionary neural networks. 2005 First International Conference on Neural Interface and Control Proceedings; 26-28 May 2005; Wuhan, China.

[13] Christian Igel,Martin Kreutz. Operator adaptation in evolutionary computation and its application to architecture optimization of neural networks. Neurocomputing, 55 (2003), 347-361.

[14] Kyung-Joong,Sung-Bae Cho. Prediction of colon cancer using an evolutionary neural network. Neurocomputing 61(2004),361-379.

[15] David Aubert, Cecile Loumagne, Ludovic Oudin. Sequential assimilation of soil moisture and streamflow data in a conceptual rainfall–runoff model. Journal of Hydrology 280 (2003), 145–161.

[16] Wei Gao. Study on New Evolutionary Neural Network. Proceedings of the Second International Conference on Machine Learning and Cybernetics, Wan, 2-5 November 2003.

[17] Walter Boughton. The Australian Water Balance Model. Environmental Modelling &Software,19(2004),943-956.

[18] Hongmei Yu, Haipeng Fang, Pingjing Yao. A Combined Genetic Algorithm Simulated Annealing Algorithm for Large Scale System Energy Integration. Computers and Chemical Engineering 24 (2000) 2023–2035.

[19] T.W.Leung,C.H.Yung,Marvin D.Ttoutt. Applications of Genetic Search and Simulated Annealing to the Two-dimensional Non-guillotine Ccutting Stock Problem. Computers & Industrial Engineering 40(2001),201-214.

[20] Z.G.Wang, Y.S.Wong, M.Rahman. Development of a Parallel Optimization Method based on Genetic Simulated Annealing Algorithm. Parallel Computing 31 (2005), 839- 857.

[21] Habib Youssef, Sadiq M.Sait, Hakim Asiche. Evolutionary Algorithms, Simulated Annealing and Tabu Search. Engineering Applications of Artificial Intelligence,14 (2001),167-181.

[22] Yu-Chung Lin,Hund-Der Yeh. Trihalomethane Species Forecast Using Optimization Methods: Genetic Algorithms and Simulated Annealing. Journal of Computing in Civil Engineering,Vol. 19, No. 3, July 1, 2005.

[23] Shun-Fa Hwang, Rong-Song He. Improving Real-parameter Genetic Algorithm with Simulated Annealing for Engineering Problems. Advances in Engineering Software 37 (2006) 406–418

[24] Jian Fang,Yugeng Xi. Neural Network Design based on Evolutionary Programming. Artificial Intelligence in Engineering, 11(1997),155-161.

[25] Qiang Luo, Wenqiang Yang, Puyin Liu. Promoter Recognition based on the Interpolated Markov Chains Optimized via Simulated Annealing and Genetic Algorithm. Pattern Recognition Letters 27 (2006),1031-1036.

[26] P.P. Palmes, S.Usui. Robustness, Evolvability, and Optimality of Evolutionary Neural Networks. BioSystems 82 (2005) 168–188.

[27] Fang Zhao, P.E.,M.ASCE. Simulated Annealing–Genetic Algorithm for Transit Network Optimization. Journal of Computing in Civil Engineering, Vol. 20, No. 1,January 1, 2006.57-68.

[28] Wei Fan, Randy B. Machemehl. Using a Simulated Annealing Algorithm to Solve the Transit Route Network Design Problem. Journal of Transportation Engineering, Vol. 132, No. 2,February 1, 2006.121-132.