An application of artificial intelligence for rainfall–runoff modeling

Ali Aytek[1,*], M Asce[1] and Murat Alp[2]

[1] *Gaziantep University, Civil Engineering Department, Hydraulics Division, 27310 Gaziantep, Turkey.*
[2] *State Hydraulics Works, 14 Regional Directorate, 34696 Küçükçamlıca, Istanbul, Turkey.*
[*] *e-mail: aytek@gantep.edu.tr*

This study proposes an application of two techniques of artificial intelligence (AI) for rainfall–runoff modeling: the artificial neural networks (ANN) and the evolutionary computation (EC). Two different ANN techniques, the feed forward back propagation (FFBP) and generalized regression neural network (GRNN) methods are compared with one EC method, Gene Expression Programming (GEP) which is a new evolutionary algorithm that evolves computer programs. The daily hydrometeorological data of three rainfall stations and one streamflow station for Juniata River Basin in Pennsylvania state of USA are taken into consideration in the model development. Statistical parameters such as average, standard deviation, coefficient of variation, skewness, minimum and maximum values, as well as criteria such as mean square error (MSE) and determination coefficient (R^2) are used to measure the performance of the models. The results indicate that the proposed genetic programming (GP) formulation performs quite well compared to results obtained by ANNs and is quite practical for use. It is concluded from the results that GEP can be proposed as an alternative to ANN models.

1. Introduction

The advances in the field of artificial intelligence influence many science topics as well as water resources engineering applications. New algorithms and models, especially those based on soft computing, enable researchers to solve the most complex systems in different ways. The use of forecast methods not based on physics equations, such as ANN and EC methods are becoming widespread in various engineering fields. The relationship between rainfall and runoff is an important issue in surface hydrology. The accurate amount of streamflow from rainfall occupies an important place in the hydrological cycle. The need to predict the quantitative amount of rainfall and runoff is necessary to avoid risks of rain on the catchments and to warn the floods.

In the last decade, ANNs have been successfully employed in modeling a wide range of hydrologic processes, including rainfall–runoff processes; Smith and Eli (1995); Hsu *et al* (1995); Minns and Hall (1996); Shamseldin (1997); Dawson and Wilby (1998); Cigizoglu and Alp (2004) studied on neural-network models of rainfall–runoff process, Mason *et al* (1996) used radial basis function (RBF) ANN for rainfall–runoff modeling, Shamseldin *et al* (1997) have presented methods for combining the outputs of the different rainfall–runoff models, Loke *et al* (1997) studied the application of an ANN for prediction of runoff coefficient by the use of simple catchment data. Fernando and Jayawardena (1998) studied on runoff forecasting using RBF networks with OLS algorithm, Tokar and Johnson (1999) developed an ANN model to predict daily runoff as a function of daily precipitation, temperature, and snowmelt for a watershed in Maryland, USA. Sajikumara and Thandaveswara (1999) presented a non-linear rainfall–runoff model using

Keywords. Artificial intelligence; artificial neural networks; evolutionary computation; genetic programming; gene expression programming; rainfall; runoff.

an ANN, Tokar and Markus (2000) applied an ANN to predict monthly streamflow for the Fraser River Watershed in Colorado, Tayfur and Singh (2006) used ANN and fuzzy logic models for simulating event-based rainfall–runoff, Garbrecht (2006) compared three alternative ANN designs for monthly rainfall–runoff simulation, Antar *et al* (2006) applied rainfall–runoff modeling using ANN technique using the Nile catchment as case study, Chiang *et al* (2004) compared the static-feed forward and dynamic-feedback neural networks for rainfall–runoff modeling, Srinivasulu and Jain (2006) presented a comparative analysis of training methods for ANN rainfall–runoff models, Riad and Mania (2004) studied on rainfall–runoff models using an ANN approach, Rajurkar *et al* (2004) modeled the daily rainfall–runoff relationship with ANN, Agarwal and Singh (2004) modeled the runoff through back propagation ANN with variable rainfall–runoff data, Tayfur *et al* (2007) presented a model to predict and forecast flow discharge at sites receiving significant lateral inflow.

GP has been applied to a wide range of problems in artificial intelligence, engineering and science applications, industrial, and mechanical models. GP can be successfully applied to areas where:

- the interrelationships among the relevant variables are poorly understood (or where it is suspected that the current understanding may well be wrong),
- finding the size and shape of the ultimate solution is hard and is a major part of the problem,
- conventional mathematical analysis does not, or cannot, provide analytical solutions,
- an approximate solution is acceptable (or is the only result that is ever likely to be obtained),
- small improvements in performance are routinely measured (or easily measurable) and highly prized,
- there is a large amount of data, in computer readable form, that require examination, classification, and integration (Banzhaf *et al* 1998).

It was observed that only a few studies existed in the literature related to the use of GP in the field of water resources engineering. Cousin and Savic (1997); Savic *et al* (1999); Drecourt (1999); Whigham and Crapper (1999, 2001); Babovic and Keijzer (2002) applied GP to rainfall–runoff modeling. Dorado *et al* (2003) studied on prediction and modeling of the rainfall–runoff transformation of a typical urban basin using ANNs and GP. Rabunal *et al* (2007) determined the unit hydrograph of a typical urban basin using GP and ANNs, Harris *et al* (2003) studied on velocity predictions in compound channels with vegetated floodplains using GP, Giustolisi (2004) determined Chezy resistance coefficient in corrugated channels by using GP, Guven *et al* (2008) applied GP for estimation of reference evapotranspiration and recently Aytek and Kişi (2008) modeled suspended sediment by using GP. Researchers continue to develop new algorithms and models for rainfall–runoff modeling due to the importance of the subject. Therefore, the purpose of this study is to develop a mathematical model for rainfall–runoff prediction based on GEP and to compare it with ANN techniques. Towards this aim, three rainfall meteorological stations for Juniata catchment (Lewistown, station no: 364992, Mapleton Depot, station no: 365381, Newport River, station no: 366297) from National Climatic Data Center, NCDC and one streamflow station (Juniata river, station no: 01567000) from USGS, are used as case studies

2. Artificial intelligence

Generally, AI methods can be divided into two main categories:

(1) symbolic AI, which deals with the development of knowledge-based systems, and
(2) computational intelligence, which includes neural networks (NN), fuzzy systems (FS), and evolutionary computing.

The most important of AI application areas are:

- system identification and function approximation, which is concerned with building empirical dynamic models of systems from measured data, or mapping system inputs to outputs,
- nonlinear prediction focuses on the prediction of the behavior of systems where the relationship between input and output is not linear,
- control focuses on controlling a system so as to achieve a desired output,
- pattern recognition or classification describing a broad range of problems where the goal is to classify an object or put it in its right class or category,
- clustering which refers to the problem of grouping cases with similar characteristics together, and identifying the number of groups or classes,
- planning which refers to the act of formulating a program for a definite course of action intended to achieve a desired goal.

Evolutionary computation is a computational technology made up of a collection of randomized global search paradigms for finding the optimal solutions to a given problem. The term evolutionary is borrowed from the terminology introduced by Charles Darwin (1864), describing the process of adaptation of survival capabilities through natural selection, fitness improvement of individual species, etc. To achieve this, evolutionary computation tries to model the natural evolution process for a successful survival battle, where reproduction and fitness play predominant roles. Being an evolutionary process, it is essentially based on the genetic material of offspring inherited from the parents. Therefore, if this material is of bad quality then the offspring cannot win the battle of survival. The evolutionary process considers the population of individuals represented by chromosomes, each chromosome bearing its characteristics called genes. The genes are assigned their individual values. Through the process of crossover the offspring are generated by combining the gene values of their parents. During the combination, the genes can undergo a (low probability) mutation process consisting of random changes of gene value in a chromosome, in order to insert fresh genetic material into the chromosomes. Finally, the winner will be the offspring with the highest value of fitness, i.e., with the best characteristics inherited from the parents.

In the meantime, various evolutionary algorithms and their modifications are found. But still, the following variants are only considered as basic evolutionary algorithms: genetic algorithms, which model genetic evolutionary processes in a generation of individuals genetic programming, which is an extension of genetic algorithms to the population in which the individuals are themselves computer program evolutionary strategies, which deal with 'evolution of evolution' by modeling the strategic parameters that control variations in evolutionary process evolutionary programming, which models adaptive evolutionary phenomena (Palit and Popovic 2005).

2.1 *Overview of genetic programming*

There are five major preliminary steps for solving a problem by using GP:

- set of terminals,
- set of functions,
- fitness measure,
- values of the numerical parameters and qualitative variables for controlling the run, and
- criterion for designating a result and terminating a run (Koza 1992).

The first major step in preparing to employ the GP paradigm is to identify the set of terminals to be used in the individual computer programs in the population. The major types of terminal sets contain the independent variables of the problem, the state variables of the system and the functions with no arguments. These types of terminal sets are given in a table by Koza (1992). The second major step is the set of functions; arithmetic operations, testing functions (such as IF and CASE statements) and boolean functions. The third major step is fitness measure which identifies the way of evaluating how good a given program solves a particular problem. The terminals and the functions are components of the programs which form the junctions in the tree. The choice of components of terminals and functions (the program) and the fitness function establishes the space that GP searches for. The fourth major step is the selection of certain parameters to control the runs. The control parameters contain the size of the population, the rate of crossover, etc. The fifth and last step is the criteria to terminate the run. For most of the problems, if the sum of the differences becomes zero (or reasonably close to zero), then, the solution is considered acceptable. The termination criterion is basically a rule for stopping. Characteristically the rule is to stop either on finding a program which solves the problem or after a certain number of generations.

2.2 *Overview of gene expression programming*

Gene expression programming (GEP) is an extension of GP that evolves computer programs of different sizes and shapes encoded in linear chromosomes of fixed length. The chromosomes are composed of multiple genes, each gene encoding a smaller subprogram. Furthermore, the structural and functional organization of the linear chromosomes allows the unconstrained operation of important genetic operators such as mutation, transposition, and recombination. One strength of the GEP approach is that the creation of genetic diversity is extremely simplified as genetic operators work at the chromosome level. Another strength of GEP consists of its unique, multigenic nature which allows the evolution of more complex programs composed of several subprograms. As a result GEP surpasses the old GP system in 100–10,000 times (Ferreira 2001a and 2001b). GP starts with an initial population of randomly generated computer programs composed of functions and terminals appropriate to the problem domain. The functions may be standard arithmetic operations, standard programming operations, standard mathematical functions, logical functions, or domain-specific functions. Depending on the particular problem, the computer program may be Boolean-valued, integer-valued, real-valued, complex-valued, vector-valued, symbolic-valued, or multiple-valued. GEP is, like GAs and GP, a genetic algorithm as it uses populations of individuals, selects them according to

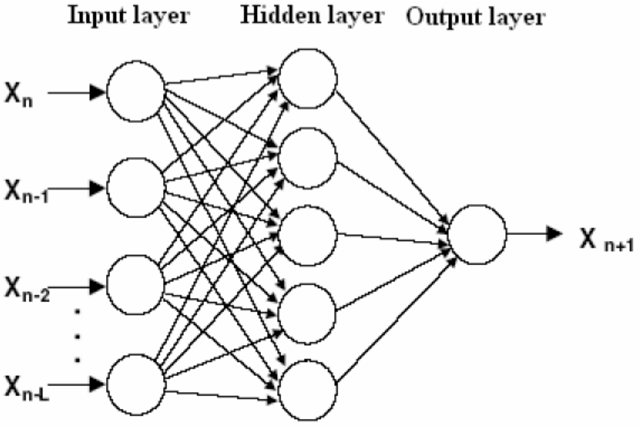

Figure 1. The structure of the FFBF.

fitness, and introduces genetic variation using one or more genetic operators (Ferreira 2006). The fundamental difference between Genetic Algorithm (GA), GP and GEP is due to the nature of the individuals: in GAs the individuals are linear strings of fixed length (chromosomes); in GP the individuals are nonlinear entities of different sizes and shapes (parse trees); and in GEP the individuals are encoded as linear strings of fixed length (the genome or chromosomes) which are afterwards expressed as nonlinear entities of different sizes and shapes (i.e., simple diagram representations or expression trees). Thus the two main parameters of GEP are the chromosomes and expression trees (ETs). The process of information decoding (from the chromosomes to the ETs) is called translation which is based on a set of rules. The genetic code is very simple where there exist one-to-one relationships between the symbols of the chromosome and the functions or terminals they represent (Ferreira 2006).

2.3 *Feed forward back propagation ANN*

The neural network structure in this study possess a three-layer learning network consisting of an input layer, a hidden layer and an output layer as shown in figure 1. Given a training set of input-output data, the most common learning rule for multi-layer perceptrons is the back-propagation algorithm (BPA). Back propagation involves two phases: a feed forward phase in which the external input information at the input nodes is propagated forward to compute the output information signal at the output unit, and a backward phase in which modifications to the connection strengths are made based on the differences between the computed and observed information signals at the output units (Eberhart and Dobbins 1990).

2.4 *Generalized regression neural network*

Generalized Regression Neural Network (GRNN) proposed by Specht (1991) does not require an iterative training procedure as in the back propagation method. It approximates any arbitrary function between input and output vectors, drawing the function estimate directly from the training data. Furthermore, it is consistent; that is, as the training set size becomes large, the estimation error approaches zero, with only mild restrictions on the function. The GRNN is used for estimation of continuous variables, as in standard regression techniques. It is related to the radial basis function network and is based on a standard statistical technique called kernel regression. By definition, the regression of a dependent variable y on an independent x estimates the most probable value for y, given x and a training set. The regression method will produce the estimated value of y which minimizes the mean-squared error. GRNN is a method for estimating the joint probability density function (pdf) of x and y, given only a training set. Because the pdf is derived from the data with no preconceptions about its form, the system is perfectly general.

If $f(x, y)$ represents the known joint continuous probability density function of a vector random variable, x, and a scalar random variable, y, the conditional mean of y given X (also called the regression of y on X) is given by

$$E[y|X] = \frac{\int_{-\infty}^{\infty} y f(X, y) dy}{\int_{-\infty}^{\infty} f(X, y) dy}. \tag{1}$$

When the density $f(x, y)$ is not known, it must usually be estimated from a sample of observations of x and y. The probability estimator $\hat{f}(X, Y)$ is based upon sample values X^i and Y^i of the random variables x and y, where n is the number of sample observations and p is the dimension of the vector variable x:

$$\hat{f}(X, Y) = \frac{1}{(2\pi)^{(p+1)/2}\sigma^{(p+1)}} \frac{1}{n} \times \sum_{i=1}^{n} \exp\left[-\frac{(X - X^i)^T (X - X^i)}{2\sigma^2}\right] \times \exp\left[-\frac{(Y - Y^i)^2}{2\sigma^2}\right]. \tag{2}$$

A physical interpretation of the probability estimate $\hat{f}(X, Y)$ is that it assigns sample probability of width σ for each sample X^i and Y^i, and

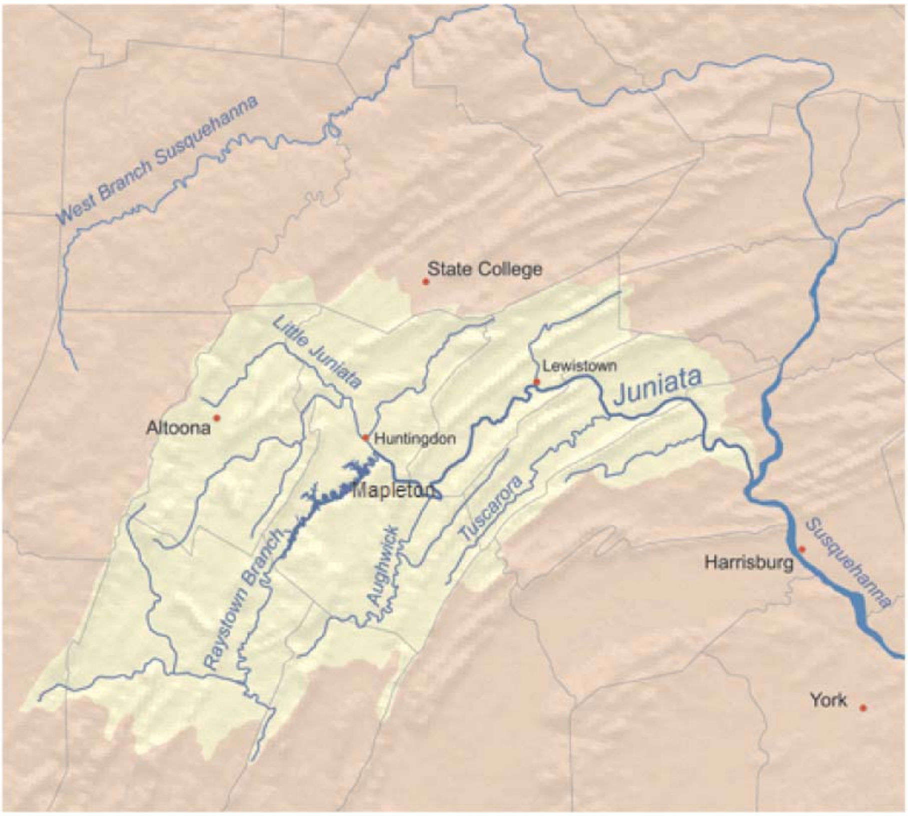

Figure 2. Map of the study area.

the probability estimate is the sum of those sample probabilities (Specht 1991). Defining the scalar function D_i^2

$$D_i^2 = (X - X^i)^T (X - X^i) \qquad (3)$$

and performing the indicated integrations yields the following:

$$\hat{Y}(X) = \frac{\sum_{i=1}^{n} Y^i \exp\left(-\frac{D_i^2}{2\sigma^2}\right)}{\sum_{i=1}^{n} \exp\left(-\frac{D_i^2}{2\sigma^2}\right)}. \qquad (4)$$

The resulting regression (equation 4) is directly applicable to problems involving numerical data. When the smoothing parameter σ is made large, the estimated density is forced to be smooth and in the limit becomes a multivariate Gaussian with covariance $\sigma^2 I$. On the other hand, a smaller value of σ allows the estimated density to assume non-Gaussian shapes, but with the hazard that wild points may have too great an effect on the estimate (Specht 1991). Differing from the FFBP, the GRNN consists of four layers: input layer, pattern layer, summation layer and output layer.

3. A brief description of the study area and data

As shown in figure 2, the study area is located in Pennsylvania state, USA. The dataset used in this study was obtained from USGS (Juniata river, station no: 01567000) and NCDC (Juniata catchment (Lewistown, station no: 364992, Mapleton Depot, station no: 365381, Newport River, station no: 366297). The averages of daily total precipitation values were computed using the Thiessen Method. The Thiessen weights for rainfall stations are computed as 0.41 for Mapleton Depot, 0.39 for Lewistown, and 0.20 for Newport River. Information for these stations can be acquired from the USGS web server (http://webserver.cr.usgs.gov) and NCDC web server http://www.ncdc.noaa.gov/oa/ncdc.html respectively. The data of January 01, 1983–June 22, 1988 were chosen for calibration and data of June 23, 1988–September 23, 1989 were chosen for validation. The training set comprises the first 2000 values of daily data and the testing set covers the last 458 values.

The daily statistical parameters of the rainfall and run-off data for the stations are given in table 1. In the table, the x_{mean}, S_x, C_v, C_{sx}, $x_{\max}$ and $x_{\min}$ denote the mean, standard deviation,

Table 1. *Statistical parameters of the rainfall and runoff data.*

	Rainfall (mm)			Runoff (m^3s^{-1})		
	Training	Testing	Whole data	Training	Testing	Whole data
x_{mean}	2.78	2.88	2.79	120.81	120.56	120.69
S_x	5.29	5.06	5.24	154.75	145.77	153.09
C_{sx}	3.80	2.59	3.60	4.58	2.81	4.30
$x_{\min}$	0.00	0.00	0.00	13.70	19.70	13.70
$x_{\max}$	72.33	35.64	72.33	2470.00	1050.00	2470.00
C_v	1.90	1.76	1.88	1.28	1.21	1.27

coefficient of variation, skewness, maximum and minimum, respectively. In the calibration flow data, $x_{\min}$ and $x_{\max}$ values for run-off fall in the ranges 13.70–2470 m^3s^{-1} for the Juniata station. However, the testing flow dataset extremes are $x_{\min} = 19.70\,m^3s^{-1}$, $x_{\max} = 1050\,m^3s^{-1}$. The range between two series of flow data is higher and this may cause extrapolation difficulties in the estimation of peak and low flow values.

4. Model development

The rainfall–runoff process was assumed to be a Markovian process for developing the proposed models, which means that the run-off value at a given location in space and time is a function of a finite set of previous realizations. With this assumption, a model structure can be mathematically expressed as;

$$Q(t) = f(R(t), R(t-1), R(t-2), \ldots, R(t-k+1), \ldots, Q(t-1), Q(t-2), \ldots, Q(t-k+1)) + \varepsilon(t) \qquad (5)$$

where $R(t)$ and $Q(t)$ represent rainfall and runoff respectively, $\varepsilon(t)$ is an error function (to be minimized), and k is the number of past rainfall realizations contributing to rainfall at the next time-step; usually, k refers to the lag of the network; if $k = 1$, the rainfall at the next time-step is related only to the present rainfall, thus giving a lag-l network.

4.1 *Rainfall–runoff modeling using GEP*

Generally, selection of input parameters does not completely define the environment from which the system will learn. The researcher must also choose specific past examples from the learning domain. Each example should contain data that represent one instance of the relationship between the chosen inputs and the outputs. These examples are often referred to as 'training cases' or 'training instances' while they are called 'fitness cases' in the case of GP. Collectively, all of the training instances are referred to as the 'training set'. Once the training set is selected, one could say that the learning environment of the system is defined (Banzhaf *et al* 1998).

There are five major steps for constructing a model by using gene expression programming. The first is the fitness function. For this problem, the fitness, f_i of an individual program, i is measured by

$$f_i = \sum_{j=1}^{C_t} (M - |C_{(i,j)} - T_j|), \qquad (6)$$

where M is the range of selection, $C_{(i,j)}$ is the value returned by the individual chromosome i for fitness case j (out of C_t fitness cases) and T_j is the target value for fitness case j. If $|C_{(i,j)} - T_j|$ (the precision) is less than or equal to 0.01, then the precision is equal to zero, and $f_i = f_{\max} = C_t M$. In this case, $M = 100$ was used, therefore, $f_{\max} = 1000$. The advantage of this kind of fitness function is that the system can find the optimal solution by itself (Ferreira 2002).

The second major step consists of choosing the set of terminals T and the set of functions F to create the chromosomes. In this problem, the terminal set consists obviously of the independent variables, i.e., $Q_t = \{R_t, R_{t-1}, R_{t-2} \ldots Q_{t-1}, Q_{t-2} \ldots\}$ where R_t and Q_t denote the rainfall and runoff at time t. The choice of the appropriate function set is not so obvious; however, a good guess can always be helpful in order to include all the necessary functions. In this study, four basic arithmetic operators $(+, -, *, /)$ and some basic mathematical functions $(\sqrt{}, \ln(x), \log(x), e^x, 10^x$, power) were utilized.

The third major step is to choose the chromosomal architecture, i.e., the length of the head and the number of genes. Length of the head, $h = 8$, and three genes per chromosome were employed. The fourth major step is to choose the linking

Table 2. *Parameters of the training of the GEP model.*

P_1	Function set	$+, -, *, /, \surd, \ln(x), \log(x), \mathrm{e}^x$
P_2	Chromosomes	50
P_3	Head size	8
P_4	Number of genes	3
P_5	Linking function	Addition
P_6	Fitness function error type	MSE (mean square error),
P_7	Mutation rate	0.044
P_8	Inversion rate	0.1
P_9	One-point recombination rate	0.3
P_{10}	Two-point recombination rate	0.3
P_{11}	Gene recombination rate	0.1
P_{12}	Gene transposition rate	0.1

Table 3. *The MSE and determination coefficients of AI models for rainfall–runoff relation in test period.*

	GEP		FFBP			GRNN		
Input combinations	MSE $\mathrm{m^6s^{-2}}$	R^2	Nodes in layer	MSE $\mathrm{m^6s^{-2}}$	R^2	s	MSE $\mathrm{m^6s^{-2}}$	R^2
(i) R_t	22082	0.057	1	20622	0.028	0.02	19923	0.080
(ii) R_t, R_{t-1}	17958	0.234	2	16238	0.252	0.03	16081	0.247
(iii) R_t, R_{t-1}, R_{t-2}	15341	0.346	2	14009	0.343	0.02	15217	0.285
(iv) $R_t, R_{t-1}, R_{t-2}, R_{t-3}$	14615	0.376	3	18510	0.271	0.04	15122	0.302
(v) $R_t, R_{t-1}, R_{t-2}, Q_{t-1}$	2249	0.905	3	2800	0.900	0.03	4742	0.784
(vi) $R_t, R_{t-1}, R_{t-2}, Q_{t-1}, Q_{t-2}$	2285	0.903	3	3834	0.869	0.03	4549	0.802
(vii) $R_t, R_{t-1}, Q_{t-1}, Q_{t-2}$	2301	0.897	3	3255	0.876	0.03	4857	0.763

function. In this study, the subexpression trees (ETs) were linked by addition. Finally, the fifth major step is to choose the set of genetic operators that cause variation and their rates.

5. Application and results

A combination of all genetic operators (mutation, transposition and recombination) is used for this purpose (table 2). Several input combinations (table 3) are tried using GEP, FFBP and GRNN to estimate rainfall–runoff relation for the used station.

5.1 *Modeling the daily flow from the rainfall*

The MSE and determination coefficient of GEP models in test period are given in table 3 for Juniata station. As seen from the table, the GEP model whose inputs are current rainfall, one previous rainfall, two previous rainfalls and one previous runoff (input combination (v)) has the lowest MSE (2249) and the highest R^2 (0.905). Similarly the best combination is alternative (v) for FFBP in which MSE = 2800 and determination coefficient (R^2) = 0.900. However, two previous runoffs are needed for GRNN application which still has low statistics performance with respect to FFBP and GEP; (MSE = 4549 and $R^2 = 0.802$). The results indicate that using only rainfall data $(R_t, R_{t-1}, R_{t-2}, R_{t-3}, \ldots)$ is not adequate for rainfall–runoff modeling.

The best of generation individual, by setting 30 chromosomes and 4 gene, has fitness 763. The best model was found after 80000 generations. The best model that describes the dynamics of the Juniata river flow using GEP is presented as:

$$Q_t = \frac{R_{t-1} - 9.80}{3.65} \times (2R_t + 9.8) + 0.39(R_{t-1} - Q_{t-1} - R_t - 22.3) + (R_{t-1} + R_{t-2}) + (R_t + 14.83) \times Q_{t-1}^{1/3} + \frac{Q_{t-1} - 9.45R_{t-1}^2 + R_t \times R_{t-1}}{Q_{t-1}}. \quad (7)$$

The runoff estimations from rainfall by the GEP method is given in figure 3 in the form of

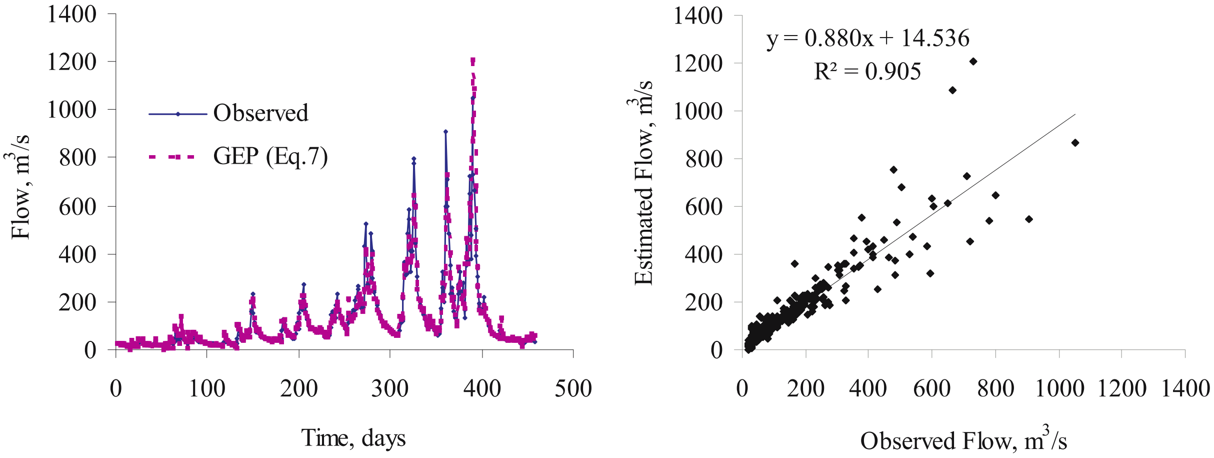

Figure 3. The observed and estimated flow in the validation period for the GEP model (equation 7).

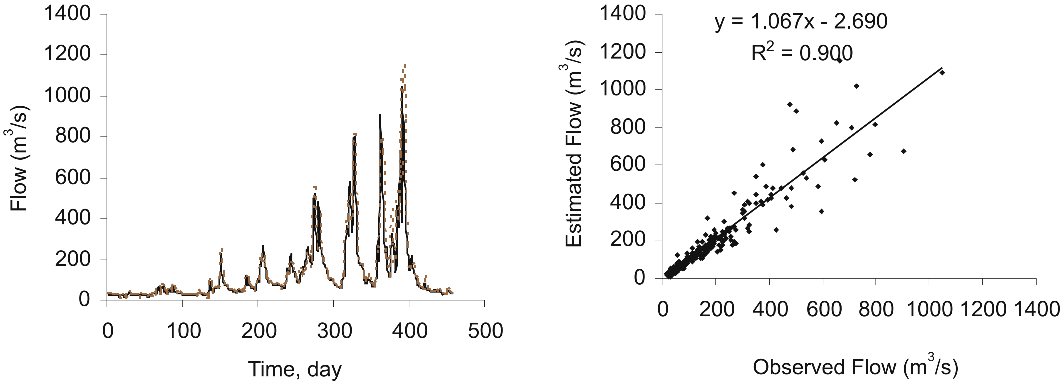

Figure 4. The observed and estimated flow in the validation period for the FFBP model.

hydrograph and scatterplot for the best combination given in table 3 (combination *v*). As seen from the figure, the GEP model approximates the corresponding observed runoff values and are as good as those obtained by ANN techniques. The hydrograph and scatter plots for FFBP are given in figure 4 for test periods.

5.2 *Modeling the daily flow*

Most of the ANNs employed only daily or monthly hydrometeorological data in the input vector for predicting long term estimations. Rajurkar *et al* (2002) applied an ANN for predicting daily flows during monsoon flood events for a large size catchment in India by using only daily rainfall data. In this study, three artificial intelligence techniques; two ANNs (FFBP and GRNN) and one CE (GEP) is applied to data obtained by the USGS database for modeling daily runoff. Input combinations for the three models are given in table 4. The best combination based determination coefficient and MSE is as follows; input combination (iii) for GEP (MSE $= 3389$ and $R^2 = 0.858$), (v) for FFBP (MSE $= 2779$ and $R^2 = 0.872$) and (i) for GRNN (MSE $= 3901$ and $R^2 = 0.818$). The observed and estimated flows in the validation period for GEP model (equation 8) and for FFBP were given in figures 5 and 6 respectively. It is obviously seen that the GEP and FFBP models compute the floods successfully. The forecasted hydrographs generally agree well with the observed hydrographs that included all storms.

The best of generation individual, by setting 30 chromosomes and 3 gene, has fitness 721. The best model was found after 37000 generations. After putting the corresponding values, the best model that describes the dynamics of the Juniata river flow using GEP is presented as;

$$Q_t = \frac{(Q_{t-2}+Q_{t-3})}{(Q_{t-1}+Q_{t-2})}\left(\frac{Q_{t-1}^2}{Q_{t-2}}\right) + 3.07Q_{t-2}^{-0.5}(Q_{t-1}-Q_{t-3}) + 0.000213(Q_{t-1}-7.025)\cdot(Q_{t-2}-Q_{t-1}). \tag{8}$$

Table 4. *The MSE and determination coefficients of AI models for flow estimation in test period.*

	GEP				FFBP		GRNN	
Input combinations	Min m^3s^{-1}	Max. m^3s^{-1}	MSE m^6s^{-2}	R^2	MSE m^6s^{-2}	R^2	MSE m^6s^{-2}	R^2
(i) Q_{t-1}	20.62	902.05	3540	0.833	3551	0.833	**3901**	**0.818**
(ii) Q_{t-1}, Q_{t-2}	23.56	1048.42	3420	0.839	3568	0.835	4128	0.809
(iii) $Q_{t-1}, Q_{t-2}, Q_{t-3}$	17.37	1258.74	**3389**	**0.858**	3537	0.839	4476	0.790
(iv) $Q_{t-1}, Q_{t-2}, Q_{t-3}, Q_{t-4}$	10.04	999.72	3145	0.852	2800	0.871	5028	0.765
(v) $Q_{t-1}, Q_{t-2}, Q_{t-3}, Q_{t-4}, Q_{t-5}$	20.09	1008.11	3545	0.834	**2779**	**0.872**	4992	0.767
(vi) $Q_{t-1}, Q_{t-2}, Q_{t-3}, Q_{t-4}, Q_{t-5}, Q_{t-6}$	23.32	1232.91	3492	0.839	3087	0.864	5131	0.762

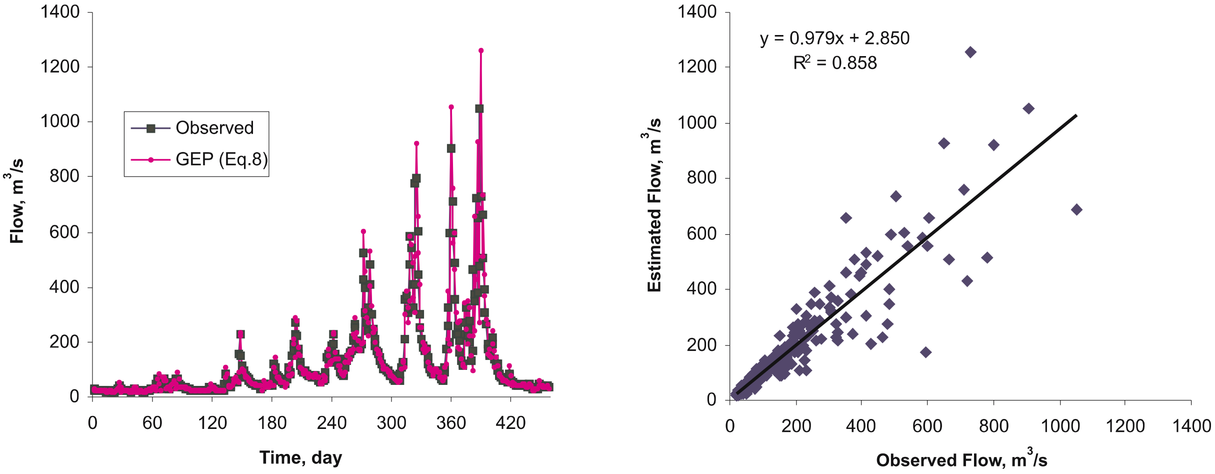

Figure 5. The observed and estimated flow in the validation period for the GEP model (equation 8).

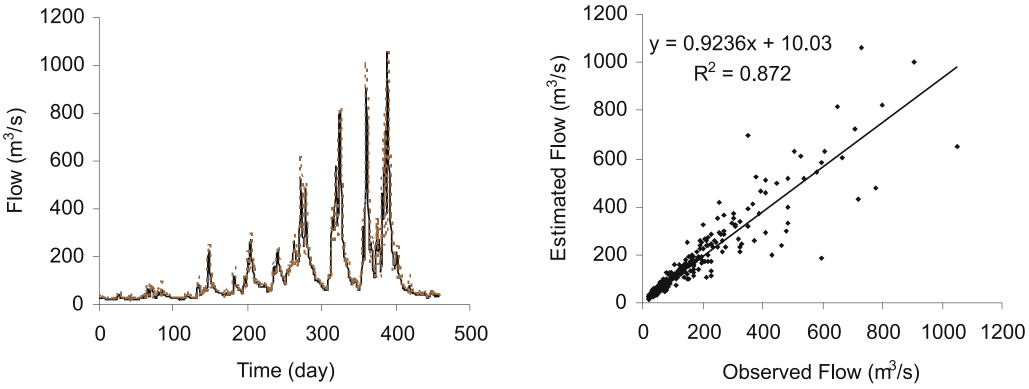

Figure 6. The observed and estimated flow in the validation period for the FFBP model.

6. Conclusions

This study indicates the ability of genetic programming (GP) technique to model the rainfall–runoff modeling. The GP model is explicit and simple that can be used by anyone not necessarily being familiar with GP. The model gives a practical method for rainfall–runoff estimation to obtain accurate results and encourages the use of GP in other aspects of water resources engineering studies. The rainfall–runoff estimations based on GP models are compared with two different ANN-based models. The results obtained with GP models are as successful as those obtained using the ANN techniques and confirm the ability of this approach to provide a useful tool in solving specific problems in hydrology, such as rainfall–runoff estimation.

Acknowledgements

We thank the reviewers for their constructive remarks. The first author would like to thank the Research Foundation of Gaziantep University for the support provided during this study.

References

Agarwal A and Singh R D 2004 Runoff modeling through back propagation artificial neural network with variable rainfall–runoff data; *Water Resources Management* **18** 285–300.

Antar M A, Elassiouti I and Alam M N 2006 Rainfall–runoff modeling using artificial neural networks technique: a Blue Nile catchment case study; *Hydrological Process* **20** 1201–1216.

Aytek A and Kişi Ö 2008 A genetic programming approach to suspended sediment modeling; *J. Hydrol.* **351** 288–298.

Babovic V and Keijzer M 2002 Rainfall runoff modelling based on genetic programming; *Nordic Hydrology* **33(5)** 331–346.

Banzhaf W, Nordin P, Keller R E and Francone F D 1998 Genetic Programming; Morgan Kaufmann, San Francisco, CA.

Chiang Y M, Chang L C and Chang J F 2004 Comparison of static-feed forward and dynamic-feedback neural networks for rainfall–runoff modeling; *J. Hydrol.* **290** 297–311.

Cigizoglu H K and Alp M 2004 Rainfall–runoff modeling using three neural network methods; Lecture Notes in Artificial Intelligence (Lecture Notes in Computer Science); Springer-Verlag, 166–171.

Cousin N and Savic D A 1997 A rainfall–runoff model using genetic programming; Centre for Systems and Control Engineering, Report No. 97/03, School of Engineering, University of Exeter, Exeter, United Kingdom, p. 70.

Darwin C 1864 On the origin of species by means of natural selection or the preservation of favoured races in the struggle for life; Cambridge, UK, Cambridge University Press, sixth edition, originally published in 1859.

Dawson C W and Wilby R 1998 An artificial neural network approach to rainfall–runoff modeling; *Hydr. Sci.* **43** 47–66.

Dorado J, Rabunal J R, Pazos A, Rivero D, Santos A and Puertas J 2003 Prediction and modeling of the rainfall–runoff transformation of a typical urban basin using ANN and GP; *Applied Artificial Intelligence* **17** 329–343.

Drecourt J P 1999 Application of neural networks and genetic programming to rainfall–runoff modeling, D2K Technical Report 0699-1-1, Danish Hydraulic Institute, Denmark.

Eberhart R C and Dobbins R W 1990 Neural network PC tools: A practical guide; Academic Press, San Diego, 414p.

Fernando D A K and Jayawardena A W 1998 Runoff forecasting using RBF networks with OLS algorithm; *J. Hydrol. Engg.* **3** 203–209.

Ferreira C 2001a Gene expression programming in problem solving; 6th online world conference on soft computing in industrial applications (invited tutorial).

Ferreira C 2001b Gene expression programming: A new adaptive algorithm for solving problems; *Complex Systems* **13** 87–129.

Ferreira C 2002 Gene expression programming: mathematical modeling by an artificial intelligence, Springer-Verlag, Germany.

Ferreira C 2006 Gene expression programming; mathematical modeling by an artificial intelligence; Springer-Berlin: Heidelberg: Newyork.

Garbrecht J D 2006 Comparison of three alternative ANN designs for monthly rainfall–runoff simulation; *J. Hydrol. Engg.* **11** 502–505.

Giustolisi O 2004 Using genetic programming to determine Chezy resistance coefficient in corrugated channels; *J. Hydroinformatics* 157–173.

Guven A, Aytek A, Yuce M I and Aksoy H 2008 Genetic programming-based empirical model for daily reference evapotranspiration estimation; Clean-Soil Air Water (accepted for publication).

Harris E L, Babovic V and Falconer R A 2003 Velocity predictions in compound channels with vegetated floodplains using genetic programming; *Intl. J. River Basin Management* **1** 117–123.

Hsu K, Gupta H V and Sorooshian S 1995 Artificial neural network modeling of the rainfall–runoff process; *Water Resources Research* **31** 2517–2530.

Koza J R 1992 Genetic programming: on the programming of computers by means of natural selection, Cambridge, MA: The MIT Press.

Loke E, Warnaar E A, Jacobsen P, Nelen F and Almeida D C M 1997 Artificial neural networks as a tool in urban storm drainage; *Water Science and Technology* **36** 101–109.

Mason J C, Price R K and Temme A 1996 A neural network model of rainfall–runoff using radial basis functions; *J. Hydraulic Res.* **34** 537–548.

Minns A W and Hall M J 1996 Artificial neural networks as rainfall–runoff models; *J. Hydrol. Sci.* **41** 399–417.

Palit A K and Popovic D 2005 Computational intelligence in time series forecasting; theory and engineering applications; Springer-Verlag, London.

Rabunal J R, Puertas J, Suarez J and Rivero D 2007 Determination of the unit hydrograph of a typical urban basin using genetic programming and artificial neural networks; *Hydrological Processes* **21** 476–485.

Rajurkar M P, Kothyari U C, Chaube U C 2002 Artificial neural network for daily rainfall–runoff modeling; *J. Hydrol. Sci.* **47(6)** 865–877.

Rajurkar M P, Kothyari U C and Chaube U C 2004 Modeling of the daily rainfall–runoff relationship with artificial neural network; *J. Hydrol.* **285** 96–113.

Riad S and Mania J 2004 Rainfall–runoff model using an artificial neural network approach; *Mathematical and Computer Modeling* **40** 839–846.

Savic A D, Walters A G and Davidson J W 1999 A genetic programming approach to rainfall–runoff modeling; *Water Resources Management* **13** 219–231.

Sajikumara N and Thandaveswara B S 1999 A non-linear rainfall–runoff model using an artificial neural network; *J. Hydrol.* **216** 32–55.

Shamseldin A Y, O'Connor K M and Liang G C 1997 Methods for combining the outputs of the different rainfall–runoff models; *J. Hydrol.* **197** 203–229.

Shamseldin A Y 1997 Application of a neural network technique to rainfall–runoff modeling; *J. Hydrol.* **199** 272–294.

Smith J and Eli R N 1995 Neural-network models of rainfall–runoff process; *J. Water Resour. Plng. and Mgmt.* **121** 499–508.

Specht D F 1991 A general regression neural network; *IEEE Transactions on Neural Networks* **2** 568–576.

Srinivasulu S and Jain A 2006 A comparative analysis of training methods for artificial neural network rainfall–runoff models; *Applied Soft Computing* **6** 295–306.

Tayfur G and Singh V P 2006 ANN and fuzzy logic models for simulating event-based rainfall–runoff; *J. Hydrol. Engg.* **132** 1321–1330.

Tayfur G, Moramorco T and Singh V P 2007 Predicting and forecasting flow discharge at sites receiving significant lateral inflow; *Hydrol. Process.* **21** 1848–1859.

Tokar A S and Johnson P A 1999 Rainfall–runoff modeling using artificial neural networks; *J. Hydrol. Engg.* **4** 232–239.

Tokar A S and Marcus M 2000 Precipitation-runoff modeling using artificial neural networks and conceptual models; *J. Hydrol. Engg.* **5** 156–161.

Whigham P A and Crapper P F 1999 Time series modeling using genetic programming: An application to rainfall–runoff models, Advances in Genetic Programming (eds) L Spector *et al*, 89–104. Cambridge, MA: The MIT Press.

Whigham P A and Crapper P F 2001 Modeling rainfall–runoff using genetic programming; *Mathematical and Computer Modeling* **33** 707–721.

MS received 15 August 2007; revised 5 December 2007; accepted 11 January 2008