農業工程學報 第 48 卷第 3 期
中華民國 91 年 9 月出版

Journal of Chinese Agricultural Engineering
Vol. 48, No. 3, September 2002

遺傳程式應用於水文資料延伸之研究

A Study of Applying Genetic Programming to Hydrological Data Extension

Li Chen

Department of Civil Engineering,
Chung Hua University,
Hsin Chu, Taiwan 30067, R.O.C.

ABSTRACT

The main purpose of this paper is to present the genetic programming (GP) and apply it to extend the flow records (y) according to the nearby stream-flow station (x). Based on GP, the relationships between x and y can be expressed as parse trees. A fittest function type will be obtained automatically from this method. The most advantage of GP is that provides system identification in a transparent and structured way.

To explore the theoretical capability of the model, a simple mathematical function is identified by GP. The model is then applied to extend the annual stream flow records according to the nearby stream flow station. The results show that the model has better performance than both of the traditional linear regression method and exemplar-based learning model. Apparently, GP demonstrates the power and efficiency for the hydrological data analysis.

Keywords: Genetic programming, Genetic algorithms, System identification, Regression method, Exemplar-based learning model, Hydrological data analysis.

1. Introduction

Natural phenomena are marvelously complex, so we are hardly to estimate stream-flow from experience. There are many hidden and useful relationships between the nearby gages data, which may not be recognized by the analyst. The traditional regression analysis turns into the most common function approximation model. The linear regression method may produce inaccurate results. However, some sophisticated regression models must through time-consuming trail and error procedures. Besides, the correct regression type can't be known ahead of time.

Therefore, many kinds of data mining techniques are developed, such as statistics, memory-based reasoning, artificial neural networks, decision trees, genetic algorithms and so on.

Evolutionary computation techniques, which are based on a powerful principle of evolution: survival of the fittest, are very efficient optimization methods. The well-known algorithms in this domain include genetic algorithms, evolutionary programming, evolution strategies, and genetic programming. Among these methods genetic algorithm (GA) is one of the most popular search algorithms. But there are some kinds of difficulties of GA, such as problem encoding and fixed-length chromosome. On the contrary, genetic program-ming (GP) operates a population of the chromo-some expressed as dynamic tree, which is more flexible to data structure.

Recently, there has been substantial success in the use of GP to evolve pattern recognition. GP works by emulating natural evolution to generate a model structure that maximizes (or minimizes) objective function (Koza 1992) involving an appropriate measure of the level of agreement between the model and system responses. This new model allows us to gain additional information on how the system performs, i.e., gives an insight into the relationship between input and output data. The advantage of GP overcomes the drawback of the artificial neural network (ANN), which owns a weight matrix cannot accessible to human understanding at present.

Early applications of GP were performed by Cramer (1985) and by Fujiki and Dickinson (1987), among others. Recently, GP is an established technique, which has been applied to several nonlinear modeling tasks including the development of signal processing algorithms (Sharman and Esparcia-Alcazar 1996). Gary et. al applied GP to the identification of the nonlinear structure of a dynamic model from experimental response from experimental data (Gary et al. 1997). Savic et al. applied GP to the identification of rainfall-runoff modeling (Savic et al. 1999).

2. Genetic Programming

GP works by emulating natural evolution to generate a model structure that maximizes (or minimizes) objective function involving an appropriate measure of the level of agreement between the model and system responses (Koza 1992). This new model allows us to gain additional information on how the system performs, i.e., gives an insight into the relationship between input and output data. GP builds on methods derived from the genetic algorithm. GP expresses the hierarchical computer programs as parse trees, rather than as the binary strings usually used by GA. Many engineering applications of GP have been developed, such as system modeling, artificial control, optimization and scheduling, design, and signal processing and so on.

There are five major preparatory steps in using GP for a particular problem. These five steps involve determining (Chen 2002)

1. The set of terminals consists of the variables and constants of the program.
2. The set of primitive functions consists of the mathematical functions and other more complex functions.
3. The fitness measure, which is the most important part of GP. For example, it can be used the errors between the measured and the estimated values.
4. The parameters for controlling the run include the population size, crossover rate and mutation rate etc.

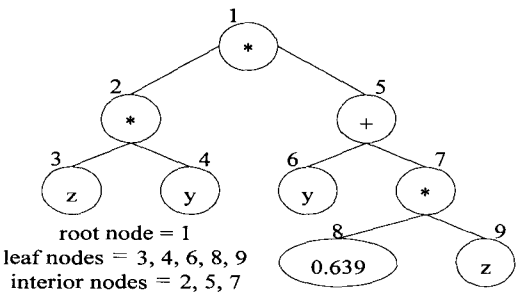

Figure 1. Parse tree representation of an expression.

5. The criterion for terminating a run generally is set by a predefined number of generation, or the amount of variation of individuals between different generation, or a target value of fitness.

(1) Representation Schemes

GP uses parse trees instead of lines of code to represent programs. Thus, for example, the simple algebraic expression $zy(y+0.639z)$ would be represented as the tree in Figure 1. The 'root node' is the first element of the tree, the 'interior nodes' are the functions and the 'leaf nodes' are the constants and /or the variables. If the set of functions used is sufficient, tree structures are capable of representing any kind of hierarchical programs.

(2) Reproduction (Selection)

Reproduction is a process in which individual trees are set according to their fitness function values. Baker (1987) presented three measures of performance of the selection algorithms, bias, spread and efficiency. The reproduction operator may be implemented in algorithmic form in a number of ways. Tournament selection is computationally more efficient and more amenable to parallel implementation (Goldberg and Deb 1991).

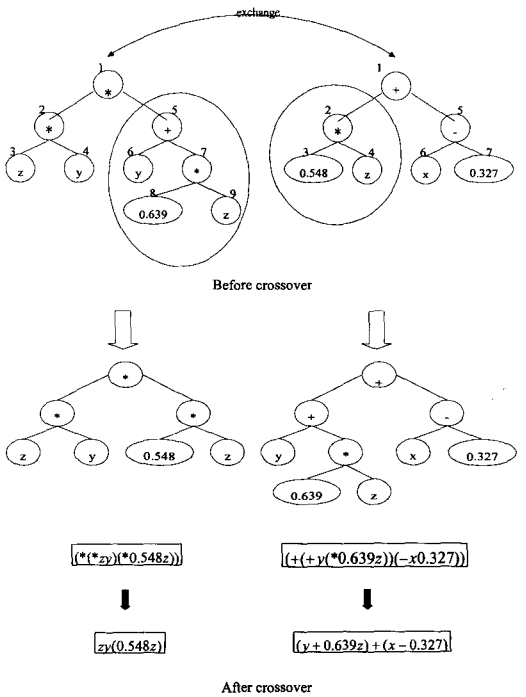

Figure 2. Crossover scheme of two parse trees.

(3) Crossover (Recombination)

After reproduction, the algorithm uses a crossover operator that exchanges arbitrary subtrees between two individuals with probability Pc. The crossover operator used in GP must ensure that programs obey the syntax of the representation scheme. So it creates new offspring that consist of genetic material taken from the parents. Figure 2 shows how this operator works.

(4) Mutation

Simple GA mutation is the occasional random alteration, with small probability (Pm), of the value of a string position. By itself, mutation is a random walk through the search space. When used sparingly with reproduction and crossover, it is an insurance policy against genetic drift that will lose important notions. The mutation of GP simply consists of randomly

exchanging a node in the tree with another node or a sub-tree.

3. Stream-flow Extension Using GP

The system identification problem may be viewed as a search for a function type, which maps input values onto an output value. This method consists the calibration (training) period and the validation (testing) period. For both of theses periods some objective criterion is needed to compare the observed and modeled outputs. The mean absolute error (MAE) is used as the objective function in this case.

3.1 Tutorial Example

To illustrate applications to system identification the following simple mathematic model is introduced (Cousin and Savic 1997, Babovic and Abbott 1997, Savic et al. 1999):

$$y = C \times x^{D}$$

Where C = 4.5, D = 2.5 are the constants.

The first step of this study is to build a proper function library. The GP can automatically select elements from this library and generate a model structure, which will best represent the training data. This function library is very important and should be flexible enough to be able to represent a broad range of systems. In this study, we choose the basic functions, including algebraic functions {+, - *, /, Sin, Cos, Log, x^y, Exp} as well as terminals (variables and constants). Compared with the past study (Savic et al. 1999), we have obtained a perfect match function. According to their results, the expression was $y = 1.26605 \times (\sin(x) \times x^2) + 0.0916853$. Owing to the key operator, x^y, we can easily acquire the correct function type, $y = (1.82509 * x)^{2.5}$ (see Figure 3), which is equal to $y = 4.5 * x^{2.5}$.

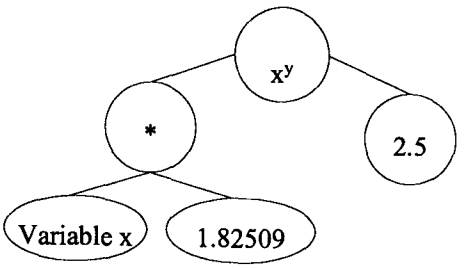

Figure 3. Parse tree of tutorial example.

3.2 Stream-flow Extension for the Potomac Basin

Stream flow records have been extensively used in a wide variety of water resources studies. Unfortunately, we often face the problem that the records of stream flow are too short to contain a sufficient range of hydrological conditions or have periods of missing data. To solve this problem, we may transfer information from nearby stream gage, that is, use the historic records and extend them in time by the correlation between flow at the site of interest and concurrent flow at nearby gage (Alley & Burns 1983).

The GP provides a feasible alternative in this circumstance. In the following, we used stream flow gages in the Potomac basin, U.S.A. to demonstrate the usefulness of the algorithm for annual stream flow extension. The data of annual stream flows from 1931 to 1960 were obtained from Salas et al. (1980) and the training data are shown in Table 1. For this case, one station was assumed as a short-record gage so that its first 10 years annual flow needs be estimated and the other station was taken as long-record gage. In other words, the last 20 years of concurrent annual flow of the short-record gage and the long-record gage were used as a training set to build up the GP model. The root mean square error (RMSE) is used as the objective function in this case. The final fittest

Table 1. Training data

Near-by gage	378	228	192	257	231	181	357	242	270	280	251	209	336	340	251	173	344	168	125	229
Estimated gage	1440	1228	750	1172	1439	1134	1652	986	1087	1175	1113	1218	1574	1570	1356	760	1480	1060	852	799

Table 2. Verification results for these three models

Near-by Station	Estimated Station	Regression Model	Hyper Rectangle	Genetic Programming
158	731	896.55	806	900.18
309	1314	1371.05	1535.75	1334.14
268	1192	1242.21	1109.38	1246.18
268	1344	1242.21	1109.38	1246.18
242	1649	1160.51	1109.38	1181.50
127	643	799.14	806	753.04
383	1237	1603.59	1492.88	1461.46
479	1336	1905.26	1492.88	1587.03
423	1609	1729.29	1492.88	1518.13
278	1231	1273.64	1175	1269.07
Correlations		R = 0.659	R=0.681	R = 0.767
RMSE		280	233	199

relationship was

$$y = (-85.9121 + 97.1002) \times 27.5606 \times \left[\frac{\ln x}{\exp\left(\frac{86.9818}{x}\right)} \right],$$

shown in Figure 4. This equation can be reduced to the form of $y = 308.351 \times \left[\frac{\ln x}{\exp\left(\frac{86.9818}{x}\right)} \right]$. As the model is set, its structure and parameters would not be modified. Then it is directly used to predict the first ten years' stream flow for the short record gage. This procedure is by using the concurrent records from the other one gage. The results are shown in Table 2. The linear correlation coefficient of GP is larger than that of linear regression, $y = 400.05 + 3.14 \times x$. Obviously, the GP has better performance than the linear regression.

In order to demonstrate the high accuracy of GP results, another artificial machine learning model called nested hyper-rectangle (NHR) was used to be compared with GP too. The strategy of this model is based on storing points (or examples)

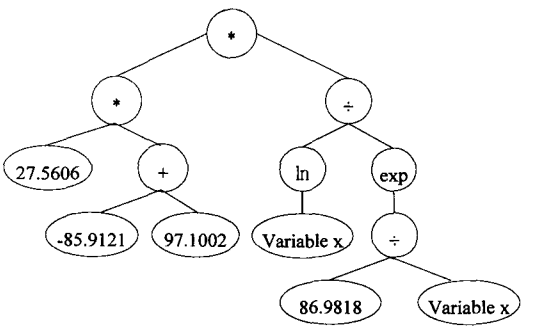

Figure 4. Parse tree of stream-flow extension problem in Potomac basin.

in Euclidean n-space, E^n, where n is the number of variables or features in an example, then comparing the new example to those, and finding the most similar example in memory. The NHR, proposed by the author of this article, is taking the training data as the points and generalized them into hyper-rectangles (Chang and Chen 1995). As the generalizations grow large, there may exist some exceptions, which create "holes" in the hyper-rectangles. These may in turn give holes inside them, resulting in a nested structure of hyper-rectangles. The basic algorithm NHR is that it uses some existing events as a foundation to predict the outcomes of other events by building the structure of hyper-rectangles. This process includes adjusting the weights of the model's parameters in time and making the system learn. The same data running through these three different methods were listed in the Table 2. It can be observed from Table 2 that GP were better in preserving correlation than the traditional linear regression model and NHR learning model described above.

4. Conclusions

GP and NHR are two good models for system identification problems and GP can acquire more information on the detail of insight

relationships between input and output data. Unlike the traditional regression techniques, GP automates the trial and error process of system identification and can be used to build a model structure that best fits training data. The most important step of building the GP model is based on the proper function library types. However, if it is too general, that the GP tends to produce an empirical 'best-fit' rather a meaningful model structure. The Potomac basin model developed using GP may be improved by using a different objective function or even several of them. Further research should be undertaken to use various near-by stations to construct a multivariable GP model.

References

1. Alley, W. M. and Burns, A. W.,: 1983, Mixedstation extension of monthly streamflow records. ASCE J. Hydraul. Enh., 109 (10): 1272-1284.
2. Babovic, V. and Abbott, M. B.: 1997, The equations from hydraulic data, Part II: Applications, J. Hydraulic Res. 35, 411-430.
3. Baker, J. E., 1987, Reducing bias and inefficiency in the selection algorithms, Proc. 2nd Int. Conf. Genetic Algorithms. Lawrence Erlbaum Associates, Hillsdale, NJ, 14-21.
4. Chang, F. J., and Chen, L.: 1995, An exemplarbased learning model for hydrosystems prediction and categorization, *Journal of Hydrology*, 169, pp.229-241.
5. Chen, L. 2002. A Study of Applying Genetic Programming to Reservoir Trophic State Evaluation Using Remote Sensor Data. International Journal of Remote Sensing. (Accepted)
6. Cousin, N. and Savic, D. A.: 1997, A rainfallrunoff 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.
7. Cramer, N. L.: 1985, A representation for the adaptive generation of simple sequential programs. In J. J. Grefenstette, ed., Proceedings of the First International Conference on Genetic Algorithms and Their Applications. Erlbaum.
8. Fujiki, C., and Dickinson, J.: 1987, Using the genetic algorithm to generate Lisp source code to solve the Prisoner's dilemma. In J. J. Grefenstette, ed., Proceedings of the First International Conference on Genetic Algorithms and Their Applications. Erlbaum.
9. Gary, G. J., Murray-Smith, D. J., Li, U., & Sharman, K. C.: 1997, Nonlinear structural system identification using Genetic Programming, In Troch, I. And Breitenecker, F., (Eds), Proceedings of the Second International Sym-posium on Mathematical modeling (MAT-HMOD), volume 1 of ARGESIM Report, pp. 301-306, Vienna. ARGESIM.
10. Goldberg, D. E. and Deb, K.: 1991, A comparitive analysis of selection schemes used in genetic algorithms. In G. Rawlins, ed., Foundations of Genetic Algorithms. Morgan Kaufmann.
11. Koza, J. R.: 1992, Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press.
12. Salas, J. D., J, W. Delleur, V. Yevjevich, and W. L. Lane,: 1980, Applied Modeling of Hydro-logic Time Series., Water Resource Publications.
13. Savic, D. A., Walters, G. A. and Davidson, J. W.: 1999, A Genetic Programming Approach to Rainfall-Runoff Modeling, Water Resources Management 13: 219-231.
14. Sharman, K., & Esparcia-Alcazar, A.: 1996, Some applications of genetic programming in digital processing. In Late breaking papers at the Genetic Programming '96 Conference, pp. 473-480, Stanford, CA.

收稿日期：民國 91 年 3 月 18 日
修正日期：民國 91 年 9 月 8 日
接受日期：民國 91 年 9 月 16 日