World Academy of Science, Engineering and Technology
International Journal of Environmental and Ecological Engineering
Vol:8, No:6, 2014

Genetic Programming: Principles, Applications and Opportunities for Hydrological Modelling

Oluwaseun K. Oyebode, Josiah A. Adeyemo

***Abstract*—**Hydrological modelling plays a crucial role in the planning and management of water resources, most especially in water stressed regions where the need to effectively manage the available water resources is of critical importance. However, due to the complex, nonlinear and dynamic behaviour of hydro-climatic interactions, achieving reliable modelling of water resource systems and accurate projection of hydrological parameters are extremely challenging. Although a significant number of modelling techniques (process-based and data-driven) have been developed and adopted in that regard, the field of hydrological modelling is still considered as one that has sluggishly progressed over the past decades. This is majorly as a result of the identification of some degree of uncertainty in the methodologies and results of techniques adopted. In recent times, evolutionary computation (EC) techniques have been developed and introduced in response to the search for efficient and reliable means of providing accurate solutions to hydrological related problems. This paper presents a comprehensive review of the underlying principles, methodological needs and applications of a promising evolutionary computation modelling technique – genetic programming (GP). It examines the specific characteristics of the technique which makes it suitable to solving hydrological modelling problems. It discusses the opportunities inherent in the application of GP in water related-studies such as rainfall estimation, rainfall-runoff modelling, streamflow forecasting, sediment transport modelling, water quality modelling and groundwater modelling among others. Furthermore, the means by which such opportunities could be harnessed in the near future are discussed. In all, a case for total embracement of GP and its variants in hydrological modelling studies is made so as to put in place strategies that would translate into achieving meaningful progress as it relates to modelling of water resource systems, and also positively influence decision-making by relevant stakeholders.

***Keywords*—**Computational modelling, evolutionary algorithms, genetic programming, hydrological modelling.

I. Introduction

GENETIC programming (GP), developed by Koza [1] belongs to the class of evolutionary algorithms (EA), which is based on the "principle of survival of the fittest", adopted from the process of natural evolution and genetics. GP is however a relatively new addition to the group of other EA techniques such as evolutionary programming (EP), genetic algorithms (GA) and evolution strategies (ES) [2]. GP however differs from GA in that its solution is a computer program or an equation as against a set of binary strings in the GA [3]. GP is a data-driven modelling technique which performs its operation via a population-based search, in which computer programs or equations that are perceived to be candidate solutions to a given problem are randomly generated and bred using specific genetic operators. GP differs from other data-driven models (DDMs) such as artificial neural networks (ANNs), fuzzy rule-based systems (FRBS) and model trees (MTs) in that it provides mathematically meaningful structures (with optimum parameters) while relating input-output variables of the system [4]. GP has found application mostly in the area of symbolic regression, where the aim is to find a functional relationship between input and output variables. These functional relationships may either be linear, quadratic or higher order polynomial. GP however defines the relationships by optimizing the model structure and the numerical coefficients of the model simultaneously. The GP algorithm is characterized by two major components, which are the terminal and function sets. These two sets contain the major building blocks used to construct the population members of the GP search space.

A. Representation of GP

GP programs are usually expressed as syntax trees, consisting of terminal and function sets. The terminal set, generally referred to as "Terminals", consists of the independent variables and constants (known as the "tree leaves") which are inputs to the problem [5]. For instance, the terminal set may simply consists of causative variables of a particular hydrological process. Following that the GP algorithm is being used as a regression technique, the numerical constants (coefficients) that match the chosen model structure to the target output are thereafter determined. Thus, the input variables and numerical constants constitute the terminal set. This is achieved with the aim of searching for a formula that uses the input variables to produce the target output [2]. The function set consists of a number of domain-specific functions that are combined with the terminal set to enable the algorithm generate candidate solutions to the problem. The function set may consist of basic arithmetic operators, mathematical functions (sin, cos, tan, log, ln, ex), Boolean operators (AND, OR, NOT), logical expressions (IF-THEN-ELSE), iterative functions (DO-UNTIL), and any other user-defined function. The program can however be represented linearly or in an explicit tree representation. Fig. 1 demonstrates a syntax tree notation of a mathematical model, $y' = \sin(x_2) + x_3\sqrt{x_1}$ which is a combination of function and

O. K. Oyebode is with the Department of Civil Engineering and Surveying, Faculty of Engineering and the Built Environment, Durban University of Technology, PO Box 1334, Durban, 4000, South Africa (phone: +27 84 807 3576; e-mail: oluwaseun.oyebode@gmail.com).

J. A. Adeyemo is with the Department of Civil Engineering and Surveying, Faculty of Engineering and the Built Environment, Durban University of Technology, PO Box 1334, Durban, 4000, South Africa (e-mail: josiaha.dut.ca.za).

terminal points (where y' is the predicted output).

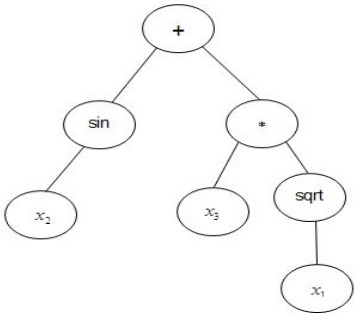

Fig. 1 Tree-representation of the mathematical expression

The set of permissible functions and terminals together constitute the primitive set and a major part of the search space of the GP system. Babovic and Keijzer [2] noted that the primitive set of a GP program must exhibit two properties namely, a "closure" property and a "sufficiency" property. A closure property is a property of a function set which enables it to swap any sub-tree to another location in the tree. The closure property also ensures that all functions can accept all possible inputs from the terminals, so as to return a well-defined value. For example, the use of a division function was presented with "illegal inputs" such as a zero value will return an undefined value, which may distort the operation of the algorithm [1]. Thus, protection is usually given to such functions by allowing them to return a specific value when confronted with such illegal inputs. Following the above example, the division function may be set to return a value of 1 when confronted with a zero value. i.e. division (x) = 1. The sufficiency property, however, ensures that elements of the primitive set are able to express a solution to the given problem. Poli et al. [5] stated that "when a primitive set is insufficient, GP can only develop programs that approximate the desired one". It was however added that such an approximation, in many cases, can be very close and good for the user's purpose.

Having established the process of constructing the primitive set of the GP program, it is important to determine how good the elements of the population space are. Thus, a "fitness function" (also referred to as "objective function") is assigned to each population member in order to measure the performance of each individual program. The objective function measures the accuracy of each computer program by computing the difference between the actual and the predicted values. The error measures may be in form of root mean-square error (RMSE), mean square error (MSE), etc. [2]. This function can either be minimized or maximized depending on the objective of the modelling problem. Furthermore, a number of "hits" can also be employed to evaluate performance of each program. The number of hits is a function of the number of data points correctly predicted within some frequency interval of tolerance. A major limitation to this approach is that each data point is classified as a "hit" or a "miss", thus, making it possible for one model to be inaccurate than another but still share the same objective function.

B. Initialization of GP

The first step in implementing GP is to randomly create an initial population for a given population size. The initial population is often randomly created to provide for a satisfactory initial coverage of the search space. In addition, it is a cheap computational process that requires no a priori knowledge from the modeller. Different methods are employed to randomly generate the initial population of the search space.

These methods include the full method, the grow method and the ramped half-and-half method. In the full method, the tree nodes are generated only from the function set until a pre-assigned maximum tree depth is reached, where only terminals can be chosen. The method is referred to as full method, because it generates full trees, with all the leaves at the same depth. Unlike the full method, the grow method gives room for creation of trees with variable sizes or shapes. The grow method also allows for selection of nodes from the whole primitive set, until the pre-defined depth limit is reached, where only terminals can be selected.

The ramped half-and-half method however is a combination of the full and grow method, which allows for the creation of trees with various sizes and shapes, but also with equal number of trees for the specified depth. The ramped half-and-half method was preferred by Koza [1] due to its easy implementation and usage, as well as its good coverage of the search space. Poli et al. [5], however, noted that whenever the ramped half-and-half method is employed, high difficulty is often experienced in controlling the statistical distribution of the important properties such as size and shape of the generated syntax trees.

Once the initial population is randomly created, the next step is to create the next generation of population, which involves selecting better program solutions.

C. Selection

From the principles of genetics, better individuals are more likely to produce more offspring than inferior individuals. Thus, GP employs a selection process (an optimization force in EA), to focus on worthwhile regions of the search space by mapping the objective function values to the number of offspring produced [2]. Selection involves the use of genetic operators to probabilistically pick individuals based on their performance, as evaluated using the objective function. The better the fitness of an individual, the greater the chance of that individual being carried over into the next generation. Although, different selection methods are being used within the EA domain such as truncation selection and fitness proportionate selection; the most popular selection method employed in GP is the "tournament" selection method.

Tournament selection involves choosing two individual programs randomly from the population, and comparing them to one another. The "fitter" program wins the tournament. A major characteristic of the tournament method is that it only looks at which program is better than another, and not how much better [5]. This is done in order to ensure that extra-ordinary programs do not outshine others throughout the

selection process, and thereafter populate the entire search space with its offspring. Hence, it gives programs with average quality the opportunity to produce offspring. The advantages of the tournament selection over other methods are: (i) it is easier to implement, and (ii) it provides automatic fitness rescaling, and hence, it's wide acceptance in GP.

D. Population Structures in GP

In GP, different structural configurations are used to populate the search space with candidate solutions. According to [2], the population structures can be classified as: (i) *Steady-state population*; (ii) *Generational population* (iii) *Panmictic and distributed populations*.

In the steady-state population structure, the population size remains unchanged during the GP run. However, in a generational population structure, an intermediate mating pool is created for the selected programs to occupy so as to form the next generation of offspring. The generation population structure may or may not ensure the reproduction of some programs into the next generation. When a population structure ensures that some programs in the mating pool are reproduced to the next generation, such a structure is referred to as an "Elitist" population model [2]. Elitism ensures that the best program individuals survive the selection process in order to preserve the best-so-far individual programs in the new generation.

A panmictic population structure entails an extremely mixed population which considers the fitness of the entire population. Therefore, selection process is performed on a global scale. A distributed population structure involves the spatial distribution of the population of candidate solutions into multiple semi-independent subpopulations called "demes" [5]. This gives room for the execution of selection process on a local scale. The distributed structure also allows for occasional migration of individuals majorly between adjacent demes for exchange of genetic material as illustrated in Fig. 2.

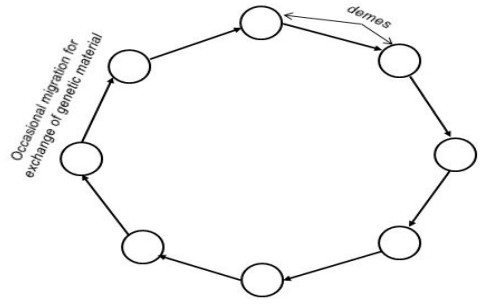

Fig. 2 Distribution of the population space into demes, allowing for exchange of genetic material

This genetically inspired mechanism is usually referred to as "Island model" [2], [6]. A major advantage of the distributed population over other methods is that the distribution of the population space into demes facilitates the occurrence of parallel evolution. The parallel evolution consequently makes the GP algorithm less susceptible to convergence to local optima [2], as a local optima in one deme might be overcome by other demes with better search direction. Furthermore, the distributed population structure ensures that the evolution progresses faster than in a single population structure [6].

E. Genetic Operators

In GP, two genetic operators namely, crossover and mutation are often used to transform selected best programs into a new generation of programs. These genetic operators operate by applying slight modifications to the structure of selected programs in order to achieve better or fitter programs (Fig. 3). The purpose of the crossover operator is to generate new programs which did not exist in the old population, to allow for thorough sampling of the search space. Crossover is performed by selecting two parent programs from the mating pool and swapping some corresponding sections across a randomly chosen point to produce two different offspring programs of different characteristics. The number of programs experiencing crossover is dependent on a predefined probability of crossover, P_c.

The mutation operation however involves random modification of a structural member of a selected parent program to create a new offspring program. The modification is also performed based on a probability of mutation, P_m. The evolution process is performed over successive generations until a preset termination criterion is met, and the program generated upon termination of the run is finally selected as the best program that gives the most accurate description of the modelled system.

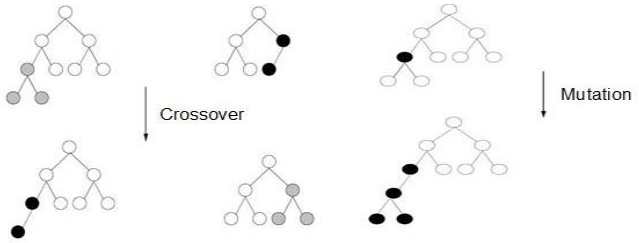

Fig. 3 Crossover and mutation being performed on parent programs.

Generally, GP evolve models in two different forms. These models can either be evolved in form of computer programs or in form of equations [7]. Program-based models consists of codes that may be written in different programming languages such as Assembly, Java, C or C++, which enables the source files to be called and new data sent to them. On the other hand, in equation-based approach, the input-output relationships are defined by evolving models in form of mathematical formulae. Some good examples of program-based and equation-based GP implementation kits include Discipulus™ [8] and GPKernel respectively.

II. Application of GP in Hydrological Modeling

Regardless of its status of being a relatively young and growing branch of EA, GP has been successfully applied to solve a wide range of water-related problems. Thus, literature on its applications is multifaceted. Presented in Table I below are some representative examples of major applications of GP.

TABLE I
APPLICATION OF GP IN VARIOUS FIELDS OF HYDROLOGICAL MODELLING

S/N	Authors & Year	Application Area	Input Variables	Location	Forecast Interval	Comparisons	Remarks on GP
1.	Khu et al. [9]	Real-time runoff forecasting	Rainfall	Orgeval Catchment, France	+1	AR, Kalman Filter	GP was found to be a better updating tool when compared to AR and Kalman filter method
2.	Sheta and Mahmoud [10]	River flow forecasting	Previous flow	Nile River, Sudan	+1	AR	GP produced simpler models with less error estimates
3.	Babovic and Keijzer [11]	Hourly Rainfall-runoff modeling	Rainfall & Runoff	Orgeval Catchment, France	+1	NAM Conceptual model	Better forecasting accuracy produced by GP. Integration of both models suggested
4.	Whigham and Crapper [12]	Rainfall-runoff modelling	Rainfall	Teifi & Namoi Basin (Wales & Australia)	Function Approx.	IHACRES (Deterministic model)	GP produced an impressive performance just as IHACRES, but without requiring any causal relationship
5.	Liong et al. [13]	Rainfall-runoff modelling	Runoff & Rainfall intensity	UBT Catchment, Singapore	+15, +30, +45 (minutes)	Naïve Rainfall-runoff model	GP induced R-R relationship was seen as viable alternaive to the naïve model
6.	Dorado et al. [14]	Rainfall-runoff prediction	Rainfall	Old Victoria Catchment, Spain	N/A	Traditional SCS Unit Hydrograph, NARMAX	Combination of GP & ANN performed better than the ARMAX & traditional models
7.	Jayawardena et al. [15]	Rainfall-runoff modelling	Rainfall & Runoff	Hok Tau Basin, Hong Kong; Shanqio & Shunhan Basin, China	+1	N/A	GP performance was not very satisfactory, as it was unable to capture high peak discharge magnitudes.
8.	Muttil and Lee [16]	Daily prediction of algal blooms	Chlorophyll, dissolved oxygen, water temp, sola radiation & wind speed	Kat O Station & Tolo Harbour, Hong Kong	+1	ANN, ARMA	GP results was in agreement with that of ANN, but was able to identify key input variables. The GP performance was also better than that of the ARMA models
9.	Bautu and Bautu [17]	Weather prediction	Temperature, Precipitation & Pressure	Rennes, France & Froson Sweden	+2	AR, ANN	GP exhibited its self-adaptive nature when unsupervised; producing better input-output relationships than other models
10.	Makkeasorn et al. [18]	Daily Streamflow forecasting	NEXRAD rainfall, SST & Met. data	Choke Canyon Watershed	+30, +7, +3 days	ANN	GP-derived models performed better than the ANN models
11.	Elshorbagy and El-Baroudy [19]	Prediction of Soil moisture content	Net radiation, Precipitation, Air temp. & Soil temp.	Mildred Lake Mine Site, Canada	N/A	Evolutionary Polynomial Regression (EPR)	A program-based GP tool (Discipulus™) performed comparably with EPR, but better than an equation-based GP tool (GPLAB). Tool uncertainty was identified.
12.	Maity and Kashid [20]	Streamflow prediction	Streamflow, ENSO & EQUINOO indices	Narmada River Basin	+1	N/A	Satisfactory performance was recorded by GP-derived models, as they were able to provide significant impacts of various input combinations
13.	Wang et al. [21]	Forecasting of monthly discharge time series	Previous flows	Lancangjiang River, Asia	N/A	ARMA, ANN, ANFIS, SVM	Best performance was obtained by GP, ANFIS & SVM models with GP producing better results in the validation phase
14.	Londhe and Charhate [3]	Daily streamflow forecasting	Previous streamflows	Narmada Basin, India	+1	ANN, MTs	GP performed marginally better than ANN & MTs, especially in normal & extreme events
15.	Ni et al. [22]	Annual streamflow prediction	Precipitation & ET	West Malian River	Function Approx.	MLP, Grey system theory, MLR	GP performed better than other models under limited availability of datasets
16.	Selle and Muttil [23]	Testing structure of hydro. Models	N/A	Southeastern Australia	N/A	Conceptual model	GP predictions supported the dominant processes contributing to deep percolation in conceptual models
17.	Guven [24]	Modelling of daily flow rate	Previous flows	Schuylkill River, USA	+1	ANN (MLP, GRNN), AR	GP results were fairly better than the ANN & AR models
18.	Guven and Kişi [25]	Daily susp. sediment modelling	Streamflow & Suspended Sediment	Two stations in the Tongue River, USA	+1	GEP, ANN	LGP performed better than ANN & GEP models, producing simple & explicit models
19.	Sivapragasam et al. [26]	Long term inflow forecasting	Rainfall & Inflows	Tamarabarami Basin, India	Multiple	N/A	GP-derived models generated improved inflow forecasts, especially when rainfall values from neighbouring stations were included as inputs
20.	Oyebode et al. [27]	Monthly Streamflow Prediction	Rainfall, Temperature & Streamflows	Upper Mkomazi River, South Africa	+1	N/A	GP-derived models provided accurate representation of streamflows in the river, while also producing adequate generalization with limited datasets.
21.	Zahiri and Azamathulla [28]	Flow discharge Prediction	Depth ratio, coherence parameter & discharge rate	Mulitple number of Rivers	N/A	M5 model trees, Vertical divided channel method	Better accuracy produced by GP compared to other methods

A. Areas of Concern

Researchers have pointed out some important issues pertaining to the application of GP in water-related studies. Some criticisms have trailed the generation of mathematical formulations by GP, as its combination of multiple elementary functions often result in extremely complex models which may be too difficult to interpret [29], [30]. Keijzer and Babovic [29] argued that the complexity of these formulae may result in GP producing models with accurate syntax but meaningless semantics. A case was thereafter made for the introduction of dimension into the GP paradigm.

Another major challenge faced by modellers in the use of GP is the selection of appropriate parameter settings to control the algorithm run. The convergence of the GP algorithm to global optimum is dependent on the parameters that govern the evolution process [4]. However, precaution needs to be taken while setting these parameters, as optimum settings vary from application to application. Thus, for a given problem, multiple runs using different parameter settings are often carried out and the solutions compared. This task is often considered as highly laborious, and also demands for higher computational resources.

The ability of GP to find the optimal solution at higher forecast horizons have also generated some concern to GP users, as forecast accuracy tends to deteriorate with increase in forecast horizon. Jayawardena et al. [15] in their study, observed the inability of GP to capture complex rainfall-runoff transformation when applied to a steep-sloped catchment, characterized by high peak discharge magnitudes with steep rising and recession limbs. However, they acknowledged the satisfactory performance of GP when applied to two other catchments at smaller time intervals. Furthermore, Babovic and Keijzer [2] earlier noted that the ability of GP to find the optimal solution depend on the magnitude of the numerical values and its dimensions. Thus, difference in the magnitude and dimensions of input variables often makes it difficult for GP to scale the variables to workable values.

Finally, the generalization ability of GP solutions is influenced by the phenomena of bloat and overfittting, which means over-growing of programs without limits and without any improvement in the fitness of the population. The increase in code size is an effect of so-called *introns,* parts of the tree that do not affect the solution's functionality. Towards the end of a GP run introns grow rapidly and comprise almost all of the code while the optimization process stagnates [31]. This consequently leads to an excessive use of CPU time and memory coupled with inadequate generalization. This is an important issue which is of concern to the GP research community and continues to steer up further studies.

B. Performance Improvement Methods

A major improvement to the GP approach was suggested by Babovic and Keijzer [2]. They proposed a "dimensionally aware GP", which requires the scaling of inputs and outputs to cast in dimensionless and proportionate terms. A dimension-based brood selection method was also introduced, which involves the use of a "culling function" to measure the goodness-of-dimension of candidate solutions. This ensures the selection of best solutions in terms of goodness-of-dimension, and thus results in the development of solutions with improved goodness-of-fit and less complexity.

Deschaine and Francone [32] also introduced an improved version of a program-based GP toolkit which ranks and combines the best individual program solutions called "program models" into teams of solutions referred to as "team models". This combination ensures better predictive accuracy in the team models than any of the individual program models. Thus, the ability of the GP model to generalize and converge towards global optimum is improved.

Bleuler et al. [31] in a study aimed at evolving compact solutions and to reduce the effects caused by bloating proposed an approach with objective not only based on program functionality, but also on program size. The program size was incorporated as an additional, but independent objective. This approach was used conjunctively with a multi-objective optimization algorithm for develop a Strength Pareto Evolutionary Algorithm (SPEA2). The ability of the SPEA2 in evolving compact programs in fewer generations was investigated comparatively against existing approaches such as Standard GP with tree depth limitation, Constant Parsimony Pressure, Adaptive Parsimony Pressure, and a Two-stage ranking method. Results found that SPEA2 produced average tree size lower than any of the other methods, while also evolving more compact solutions. Furthermore, SPEA2 was slightly faster in finding solutions than any other of the tested methods, although the Constant Parsimony approach provided the overall best performance. It was concluded that a Pareto-based multi-objective approach is a promising way of reducing bloat in GP.

In a recent study, Naik and Dabhi [33] tested the performance of four bloat control techniques namely Tarpeian, double tournament, lexicographic parsimony pressure with direct bucketing and ratio bucketing on six different problems. The double tournament (selection method) and Tarpeian method were combined and results obtained compared with the individual performance of other methods. It was found that the combination of the two methods was able to avoid bloated solutions and thus produced better generalization than the individual methods.

C. Advantages and Disadvantages

GP has been found to exhibit several advantages over other DDMs. Its major advantage is in its ability to generate programs that can efficiently simulate complex processes using symbolic expressions [19]. Another advantage of GP over other robust methods such as ANN is that it generates a transparent and structured representation of the system being modelled, without requiring a priori identification of the model structure [34]. This is unlike the ANN approach where the structure of the network and training algorithm have to be defined in advance, and only the optimization of the network parameters (weights and biases) are performed. However, in GP both the model structure and its parameters are being optimized, as they are both part of the search process [11]. This gives GP the ability to automatically identify the input variables that contribute beneficially to the model and disregard those that do not [15], thus reducing the

dimensionality of the model. Besides, GP evolves models capable of giving physical insight into the input-output interactions inherent in the modelled system, in contrast to the ANNs where difficulty still exists in extracting knowledge from the network parameters [35].

On the other hand, GP has its own limitations. Principally, GP is not very powerful in finding constants, and more importantly, it tends to produce more complex functions as the forecast horizon increases [34].

III. Conclusion

An extensive review of the principles and techniques for implementing the GP technique has been presented in this paper. Diverse applications of GP as it relates to the hydrological domain have also been showcased. It is evident from the applications presented that GP is capable of the solving complex hydrological modelling problems. GP has also been found to produce better prediction performance when investigated comparatively with other DDMs. As a result, the hydrological modelling community stands to benefit from the ample opportunities GP presents. Such opportunities include easy integration into other DDMs (model hybridization) for the purpose of complimentary modelling, coupling with process-based models to achieve uncertainty reduction and mitigation against sensitivity challenges as well as improvement of generalization ability of models under limited availability of datasets. Although some areas of concern have been identified in the implementation of GP in this review, the advantages derivable from its application, however, outweighs the drawbacks. These areas of concern will however give research direction to the hydrological modelling community and will continue to attract discussion in various spheres of soft computing; from the perspectives of IT experts to that of hydrological modellers.

References

[1] J. R. Koza, Genetic programming: on the programming of computers by means of natural selection vol. 229. Cambridge, Massachusetts, London, England: The MIT Press, 1992.

[2] V. Babovic and M. Keijzer, "Genetic programming as a model induction engine," Journal of Hydroinformatics, vol. 2, pp. 35-60, 2000.

[3] S. Londhe and S. Charhate, "Comparison of data-driven modelling techniques for river flow forecasting," Hydrological Sciences Journal, vol. 55, pp. 1163-1174, 2010.

[4] V. Babovic and R. Rao, "Evolutionary Computing in Hydrology," in Advances in data-based approaches for hydrologic modeling and forecasting, B. Sivakumar and R. Berndtsson, Eds., ed Singapore: World Scientific, 2010, pp. 347-369.

[5] R. Poli, W. W. B. Langdon, and N. F. McPhee, A Field Guide to Genetic Programming: Lulu Enterprises UK Limited, 2008.

[6] M. F. Brameier and W. Banzhaf, "A comparison of linear genetic programming and neural networks in medical data mining," Evolutionary Computation, IEEE Transactions on, vol. 5, pp. 17-26, 2001.

[7] M. C. Deo, "Recent Data Driven Methods and Applications in Coastal and Hydrologic Data Analysis," ISH Journal of Hydraulic Engineering, vol. 15, pp. 310-327, 2009.

[8] F. D. Francone, "Discipulus™ Software Owner's Manual, version 3.0 DRAFT," Register Machine Learning Technologies, Inc, 1998.

[9] S. T. Khu, S. Y. Liong, V. Babovic, H. Madsen, and N. Muttil, "Genetic Programming and Its Application in Real-Time Runoff Forecasting," JAWRA Journal of the American Water Resources Association, vol. 37, pp. 439-451, 2001.

[10] A. F. Sheta and A. Mahmoud, "Forecasting using genetic programming," in System Theory, 2001. Proceedings of the 33rd Southeastern Symposium on, 2001, pp. 343-347.

[11] V. Babovic and M. Keijzer, "Rainfall runoff modelling based on genetic programming," Nordic hydrology, vol. 33, pp. 331-346, 2002.

[12] P. Whigham and P. Crapper, "Modelling rainfall-runoff using genetic programming," Mathematical and Computer Modelling, vol. 33, pp. 707-721, 2001.

[13] S. Y. Liong, T. R. Gautam, S. T. Khu, V. Babovic, M. Keijzer, and N. Muttil, "Genetic Programming: A New Paradigm in Rainfall Runoff Modeling," JAWRA Journal of the American Water Resources Association, vol. 38, pp. 705-718, 2002.

[14] J. Dorado, J. R. RabuñAL, A. Pazos, D. Rivero, A. Santos, and J. Puertas, "Prediction and modeling of the rainfall-runoff transformation of a typical urban basin using ANN and GP," Applied Artificial Intelligence, vol. 17, pp. 329-343, 2003.

[15] A. Jayawardena, N. Muttil, and T. Fernando, "Rainfall-runoff modelling using genetic programming," in Proceedings of the MODSIM 2005 international congress on modelling and simulation: advances and applications for management and decision making. Melbourne, Australia, 2005, pp. 1841-1847.

[16] N. Muttil and J. H. Lee, "Genetic programming for analysis and real-time prediction of coastal algal blooms," Ecological modelling, vol. 189, pp. 363-376, 2005.

[17] A. Bautu and E. Bautu, "Meteorological data analysis and prediction by means of genetic programming," in Proceedings of the 5th Workshop on Mathematical Modeling of Environmental and Life Sciences Problems Constanta, Romania, 2006, pp. 35-42.

[18] A. Makkeasorn, N. B. Chang, and X. Zhou, "Short-term streamflow forecasting with global climate change implications – A comparative study between genetic programming and neural network models," Journal of Hydrology, vol. 352, pp. 336-354, 2008.

[19] A. Elshorbagy and I. El-Baroudy, "Investigating the capabilities of evolutionary data-driven techniques using the challenging estimation of soil moisture content," Journal of Hydroinformatics, vol. 11, pp. 237-251, 2009.

[20] R. Maity and S. S. Kashid, "Hydroclimatological Approach for Monthly Streamflow Prediction Using Genetic Programming," ISH Journal of Hydraulic Engineering, vol. 15, pp. 89-107, 2009.

[21] W.-C. Wang, K.-W. Chau, C.-T. Cheng, and L. Qiu, "A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series," Journal of hydrology, vol. 374, pp. 294-306, 2009.

[22] Q. Ni, L. Wang, R. Ye, F. Yang, and M. Sivakumar, "Evolutionary modeling for streamflow forecasting with minimal datasets: a case study in the West Malian River, China," Environmental Engineering Science, vol. 27, pp. 377-385, 2010.

[23] B. Selle and N. Muttil, "Testing the structure of a hydrological model using Genetic Programming," Journal of Hydrology, vol. 397, pp. 1-9, 2011.

[24] A. Guven, "Linear genetic programming for time-series modelling of daily flow rate," Journal of earth system science, vol. 118, pp. 137-146, 2009.

[25] A. Guven and Ö. Kişi, "Estimation of suspended sediment yield in natural rivers using machine-coded linear genetic programming," Water Resources Management, vol. 25, pp. 691-704, 2011.

[26] C. Sivapragasam, N. Muttil, and V. Arun, "Long term flow forecasting for water resources planning in a river basin," in Proceedings, International Congress on Modelling and Simulation, Perth, Australia, 2011, pp. 4078-4084.

[27] O. Oyebode, J. Adeyemo, and F. Otieno, "Monthly streamflow prediction with limited hydro-climatic variables in the upper Mkomazi River, South Africa using genetic programming," Fresenius Environmental Bulletin, vol. 23, pp. 708-719, 2014.

[28] A. Zahiri and H. M. Azamathulla, "Comparison between linear genetic programming and M5 tree models to predict flow discharge in compound channels," Neural Computing and Applications, vol. 24, pp. 413-420, 2014.

[29] M. Keijzer and V. Babovic, "Dimensionally aware genetic programming," in *Proceedings of the Genetic and Evolutionary computation Conference*, 1999, pp. 1069-1076.

[30] D. Solomatine and A. Ostfeld, "Data-driven modelling: some past experiences and new approaches," *Journal of hydroinformatics,* vol. 10, pp. 3-22, 2008.

[31] S. Bleuler, M. Brack, L. Thiele, and E. Zitzler, "Multiobjective genetic programming: Reducing bloat using SPEA2," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, 2001, pp. 536-543.
[32] L. M. Deschaine and F. D. Francone, "Comparison of Discipulus™ Linear Genetic Programming Soft-ware with Support Vector Machines, Classification Trees, Neural Networks and Human Experts," *Register Machine Learning Technologies Inc*, 2002.
[33] T. R. Naik and V. K. Dabhi, "Improving Generalization Ability of Genetic Programming: Comparative Study," *Journal of Bioinformatics and Intelligent Control*, vol. 2, pp. 243-252, 2013.
[34] O. Giustolisi and D. Savic, "A symbolic data-driven technique based on evolutionary polynomial regression," *Journal of Hydroinformatics*, vol. 8, pp. 207-222, 2006.
[35] K. Sudheer, "Knowledge extraction from trained neural network river flow models," *Journal of Hydrologic Engineering*, vol. 10, pp. 264-269, 2005.