Received: 2 May 2020 | Revised: 29 July 2020 | Accepted: 14 August 2020

DOI: 10.1002/ail2.9

LETTER

WILEY

Multiobjective genetic programming for reinforced concrete beam modeling

Amirhessam Tahmassebi[1] | **Behshad Mohebali**[1] | **Anke Meyer-Baese**[1] | **Amir H. Gandomi**[2]

[1]Department of Scientific Computing, Florida State University, Tallahassee, Florida

[2]Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia

Correspondence
Amir H. Gandomi, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia.
Email: gandomi@uts.edu.au

Abstract

This paper presents the application of multiobjective genetic programming (MOGP) in engineering issues. An evolutionary symbolic implementation was developed based on a case study on prediction of the shear strength of slender reinforced concrete beams without stirrups including 1942 set of published test results. In the implementation of the MOGP model, the nondominated sorting genetic algorithm II with adaptive regression by mixing algorithm with considering the optimization of mean-square error as the fitness measure and the subtree complexity was used. The developed MOGP model was compared to previously developed genetic programming models, different building codes, and additional machine learning based approaches. It is clearly shown that the MOGP model outperformed the other algorithms applied on this database and can be a general solution on any engineering problems with the main advantage of prediction equations without assuming prior form of the relevance among the input predictor variables.

KEYWORDS

multiobjective genetic programming, RC beam, symbolic regression

1 | INTRODUCTION

Since its conception in 1992 by John Koza,[1] genetic programming (GP) has been an attractive tool for researchers to identify models and systems. GP and its close predecessor genetic algorithm (GA),[2] are two most significant members of a group of methods, called evolutionary algorithms (EA), which use natural selection principles to pursue better solutions to various types of engineering problems. One important difference between GP and GA is that GA generally does not allow for variation of the model structure during the computation,[3] while in GP the structure of the model is a part of the problem and will be a part of the solution as well. In better words, GA is more concerned with the model parameters and inputs and assumes the behavior of the to-be-optimized system is known, while GP does not make such assumption about the internals of the system beforehand, and there lies its true power. GP is specifically effective for the problems in which the structure of the solution—in terms of how the variables inside the model are related to each other (interaction among variables) and to the output—is unknown, or expected to be poorly known.[4] A good example for such problems is the synthesis of electronic circuits such as amplifiers, voltage-current converter, low-voltage balun circuit, and so forth[1,5,6] that was achieved by merely describing the higher level behavior of the circuit. Streeter et al[6] managed to duplicate the functionality of five patented circuits using a GP approach, which outputs the topology of the circuit and the numerical value of its components. Other than producing novel topologies for electronic solutions, GP

has been used for synthesizing topologies of PID controllers.[7] Not only was the proposed GP approach able to produce with the controller topology, but also it managed to come up with tuning rules that would outperform Ziegler-Nichols and Astrom-Hagglund tuning rules, which have been extensively in use for the majority of the twentieth century. Another area of strength for GP based approaches is where there are powerful simulation tools for a given problem but no concrete approach,[4] especially when the problem is somehow related to mimicking a natural system or phenomenon. Evolutionary robotics (ER) is a field in which EA are applied to solve robot design problems. For instance, something as rudimentary as walking has been posed as a great challenge to robotics researchers for decades.[8] A multitree GP approach was adopted in Reference 9 to optimize the way a swarm of cubic shaped robots assemble themselves into a larger unit. Others used GP approach to create controllers for mobile robots.[8-10] Due to the complex nature of walking locomotion, GP is proven very effective in producing satisfactory results. GP has been utilized to model complex processes in other engineering issues where the internal parts of the model are unknown or time consuming to attain.[11,12] In Reference 12, the authors used GP to model the strength of the concrete cylinders after they are enhanced by carbon-fiber reinforced polymer composites. In this paper, an evolutionary symbolic implementation for the prediction of the shear strength of slender reinforced concrete (RC) beams based on ACI 318-08 (2008)[13] without stirrups was investigated. In this implementation, various models including different characteristics via the combination of multi-objective genetic programming (MOGP) and adaptive regression by mixing algorithm were proposed. The proposed models were compared to the benchmark machine learning algorithms along with the model proposed by Gandomi et al[14] (Data S1). Additionally, the proposed model has shown better accuracy than models based on several building codes such as ACI building code, EC2 building code, Canadian Standard Code, and New Zealand Standard.[13]

Algorithm 1 NSGA-II

Inputs: Generations N, Population P
Output: Best model

(1) Initialize population P
(2) Generate random population size M
(3) Evaluate objective values
(4) Assign ranking based on Pareto sort
(5) Generate child population
(6) Binary tournament selection
(7) Recombination and mutation

for $i \in \{1, ..., N\}$ **do**
for each parent and child in population **do**
(1) Set ranking based on Pareto, (2) Generate sets of nondominated solutions, (3) determine crowding distance, (4) Loop inside by adding solutions to next generation starting from the first front until N individuals.

(1) Select points on the lower front with higher crowding distance, (2) Create next generation, (3) Binary tournament selection, (4) Recombination and mutation.

2 | MULTIOBJECTIVE GENETIC PROGRAMMING

One area of use for GP[1] as a symbolic optimization technique is to find a model that best represents the dataset, in the simplest and most accurate form. This is done by searching the space of the mathematical expressions. GP was originally formulated based on functional programming language as an evolutionary method to use Darwin's theory of natural selection to create computer programs aimed at a specific problem. Instead of using one candidate, GP uses a group of individuals (known as population), formed by randomly combining mathematical building blocks such as constants, mathematical operators, analytic functions, state variables, and genetic operators, to make new individuals

(generations) guided by fitness and complexity as objective functions that are meant to gauge the quality of each individual. The regression model[15,16] in Algorithm 1 (flowchart available in Reference 17) is implemented as an MOGP approach based on the work of Deb et al on nondominated sorting genetic algorithm II (NSGA-II).[18] Instead of fitness sharing (which was used in NSGA-I), NSGA-II uses the concept of crowding distance. In other words, in NSGA-II, parents and offspring are combined to form one set and a nondominated sorting is used to classify the entire population based on an estimate of the density of solutions using the crowding distance. There are two values that the algorithm will minimize; first, the error, which can be the mean squared error (MSE) or mean absolute error (MAE), and second, the subtree complexity measure.[15,16,19] GP has great potential in predicting complex patterns.[20-22] It is also a suitable approach to be implemented using common parallelization frameworks such as message passing interface (MPI) and open multiprocessing (OpenMP).[23] The proposed model forms a Pareto front of models based on the fitness and subtree complexity measures. In this study, the most accurate model, and the model at the knee of the Pareto front (the knee of the 2D space of the objectives: fitness-complexity space, which presents the maximal trade-offs between objectives) were presented. Additionally, a fused model of the Pareto front obtained with adaptive regression by mixing (ARM) algorithm (shown in Algorithm 2) introduced by Yang[24] was presented.

Algorithm 2 ARM

Inputs: Input variables X_i, Target variables Y_i, $i \in (1,N)$, Function $\hat{f}$ Output: Best model
(0) Random permutation the order of the observations M

for $m \in \{1, ..., M-1\}$ **do**
(1) Randomly permute the order of the observations, (2) Split the data into two parts, (3) $Z^{(1)} = (X_i, Y_i)_{i=1}^{\frac{N}{2}}$,
(4) $Z^{(2)} = (X_i, Y_i)_{i=\frac{N}{2}+1}^{N}$

for $j \in \{1, ..., J\}$ **do** (a) Estimate $\hat{f}_{j,\frac{N}{2}}\left(xZ^{(1)}\right)$ of f, (b) Estimate the variance function $\sigma^2(x)$ by $\hat{\sigma}^2_{j,\frac{N}{2}}(x)$

for $i \in \left\{\frac{N}{2}+1, ..., N\right\}$ **do** (c) Predict Y_i by $\hat{f}_{j,\frac{N}{2}}(X_i)$,

(d) $E_j = \frac{\prod_{i=\frac{N}{2}+1}^{N} h\left(\left(Y_i - \hat{f}_{j,\frac{N}{2}}(X_i)\right)/\hat{\sigma}_{j,\frac{N}{2}}(X_i)\right)}{\prod_{i=\frac{N}{2}+1}^{N} \hat{\sigma}_{j,\frac{N}{2}}(X_i)}$; (e) Compute the current weight $\hat{W}_j = \frac{E_j}{\sum_{l=1}^{J} E_l}$.
(5) The final estimate is $\hat{f}_N(x) = \sum_{j=1}^{J} \hat{W}_j \hat{f}_{j,N}(x)$

3 | PERFORMANCE ANALYSIS

As an example of engineering problems, a case study on prediction of the shear strength of slender RC beams without stirrups including 1942 set of published test results was employed which are gathered from different studies by Gandomi et al.[14] The input predictor variables are (1) web width (b_w in mm), (2) effective depth (d in mm), (3) shear span to depth ratio ($\frac{a}{d}$), (4) concrete compressive strength (f_c in MPa), and (5) the amount of longitudinal reinforcement (ρ_l in %), as well as shear strength of the RC beam without stirrups (V in kN) as the output. Figure 1 illustrates the kernel density distribution of the predictor variables in the database. As seen, all of the variables have a Gaussian distribution. The database randomly splitted into training and testing sets including 1458 and 486 instances, respectively. An MOGP model based on NSGA-II employing the most important factors commonly used in the previous building codes was developed. In order to find the optimized mutation and crossover rates of the model, various rates ranging from 0.1 to 0.9 were employed for the developed MOGP model for 200 generations. The 3-dimensional surface plot of the mutation rates and the crossover rates based on the fitness and subtree complexity measures are illustrated in Figure 2A and B, respectively. The highest fitness value (0.9389) was obtained with a value of 0.6 as the crossover rate, and a value of 0.4 as the mutation rate. Additionally, the lowest subtree complexity (153) was obtained with a value of 0.2 for the crossover rate, and a value of 0.8 for the mutation rate. However, the crossover and the mutation rates of 0.1 and 0.9 showed a fitness value of 0.9337 and a subtree complexity value of 181, accordingly. Therefore, the aforementioned

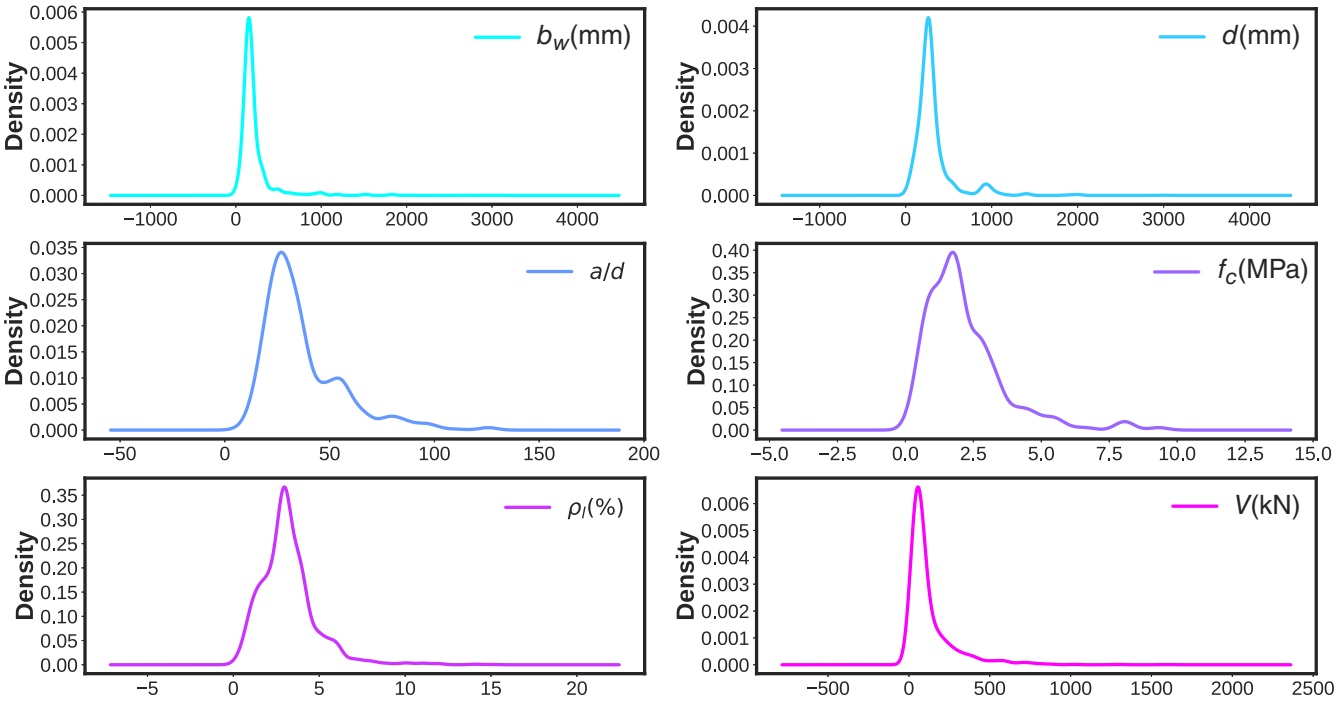
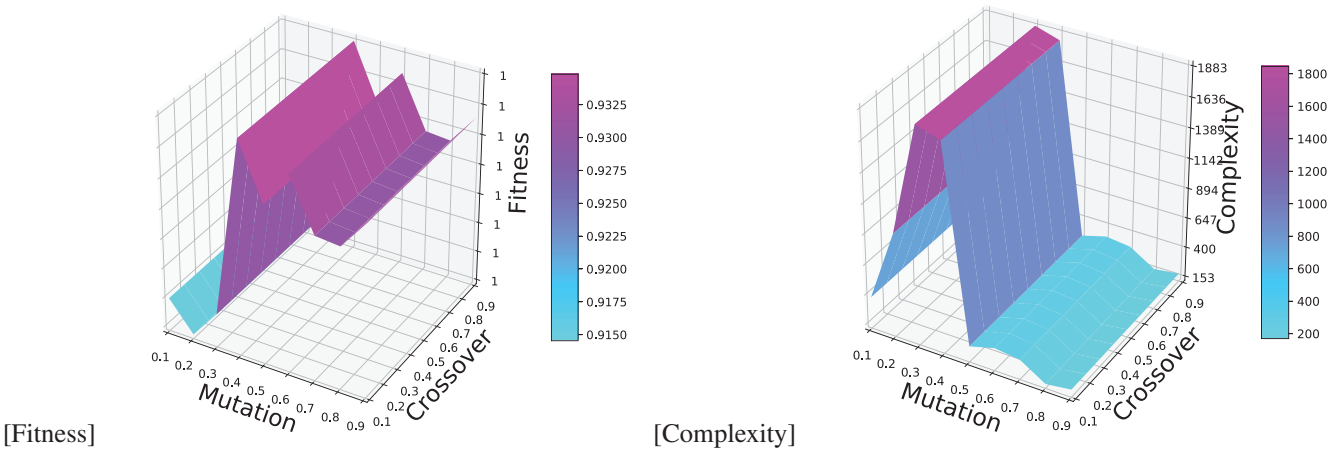


FIGURE 1 Kernel density distribution of the predictor variables in the database



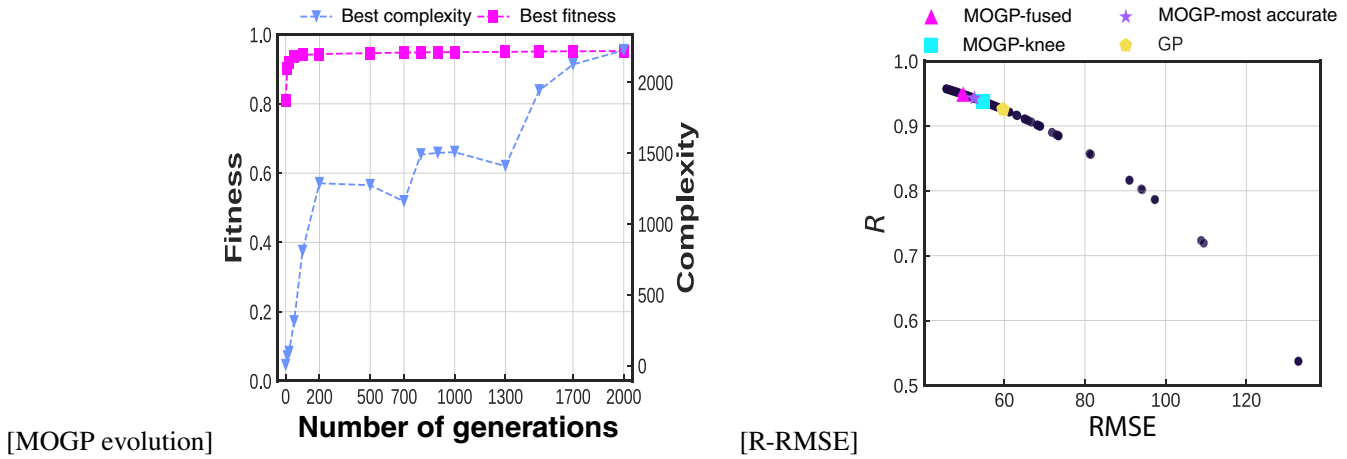
[Fitness] [Complexity]

FIGURE 2 Exploring the optimized values for mutation and crossover rates for the developed genetic programming model based on the fitness and complexity measures

rates were considered for running the developed MOGP model with 2000 generations and 1000 population for the regression task. The details of the parameters setting for the MOGP model are presented in Table 1. The developed MOGP model was trained on the training dataset with 1458 instances and tested on the testing dataset with 486 instances. This would decrease the chance of over-fitting of the model. The fitness and the complexity values at each generation were reported for the best individuals. Figure 3A depicts the evolution of the best individuals in the Pareto front of the MOGP model through generations for both fitness and complexity measures. The left y-axis was set to fitness measure, the right y-axis to complexity, and x-axis to number of generations. As seen, after roughly about 100 generations the model reached the fitness with 94% accuracy. After 2000generations, the fitness reached a value of 96% with the complexity value of 2227 which is about three times more than the value that was obtained for the 94% fitness

TABLE 1 Parameters setting for the multiobjective genetic programming function regressor

Parameter	Setting
Population size	1000
Number of generations	2000
Tournament size	20
Number of inputs	5
Crossover rate	0.1
Mutation rate	0.9
Number of instances in training set	1458
Number of instances in testing set	486
Number of central processing unit threads	8
1st objective	Mean squared error
2nd objective	Subtree complexity
Population initialization	Ramped-half-and-half
Function set	$+, -, \times, /, \sqrt{}, (\ )^2, (\ )^3, (\ )^4$ log, exp, sin, cos



[MOGP evolution] [R-RMSE]

FIGURE 3 A, The evolution of the employed objective functions, fitness, and complexity measures for the developed multiobjective genetic programming (MOGP) model through different numbers of generations. B, Scatter plots of coefficient of determination vs root-mean-square error (R vs RMSE) of the all models in the Pareto front (dark violet circles) with indicating the most MOGP-accurate model (dark orchid star), the MOGP-fused model (purple triangle), the MOGP-knee model (cyan square), and the genetic programming (GP) model by Gandomi et al[14] (yellow pentagon)

measure. By increasing the number of generations, both fitness and complexity values increased as well, however the rate of improvement of the fitness value was almost constant after 200 generations.

In the context of statistical modeling with the main purpose of prediction of future outcomes, coefficient of determination (R^2) provides a measure of how well observed outcomes are replicated by the model based on the proportion of total variation of outcomes explained by the model. Additionally, root-mean-square error (RMSE) is frequently used to measure the differences between values predicted by a model and the values actually observed. To explore the performance of the proposed models, both regression metrics were computed for all of the models in the Pareto front (shown in dark violet circles) as shown in Figure 3B. Furthermore, the specified models including the most MOGP-accurate model (dark orchid star), the MOGP-fused model (purple triangle), the MOGP-knee model (cyan square), and the GP model by Gandomi et al[14] (yellow pentagon) were indicated. It is clearly shown that the ARM algorithm found a model even better than the most accurate model in the Pareto front. The MOGP-fused model with a value of 0.9487 as R, a

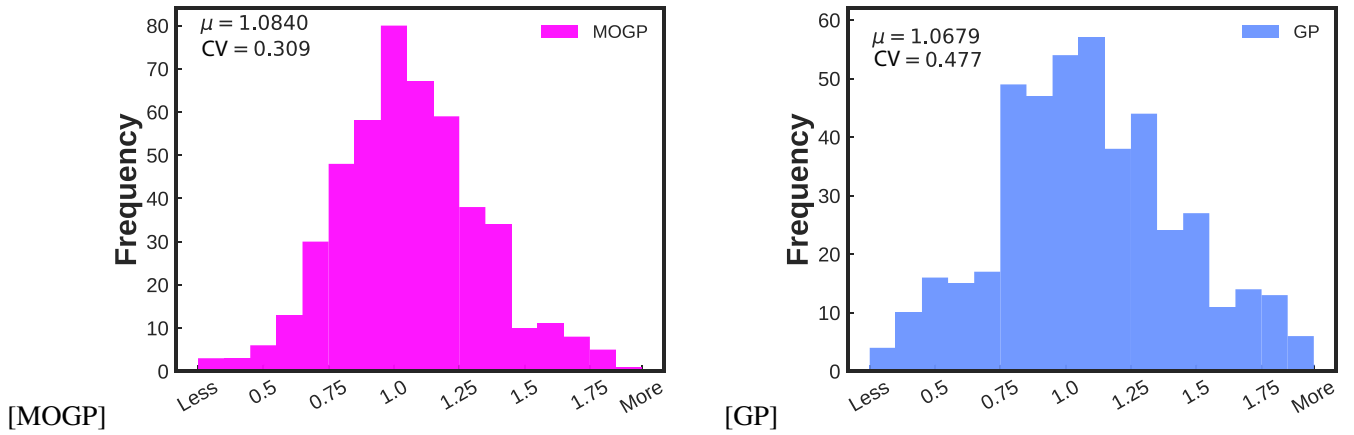


FIGURE 4 Histograms of the ratio of the predicted and the measured V(kN) for A, the multiobjective genetic programming (MOGP)-fused model, and B, the genetic programming (GP) model by Gandomi et al.[14] Mean value and coefficient of variation of this ratio are also reported for each model

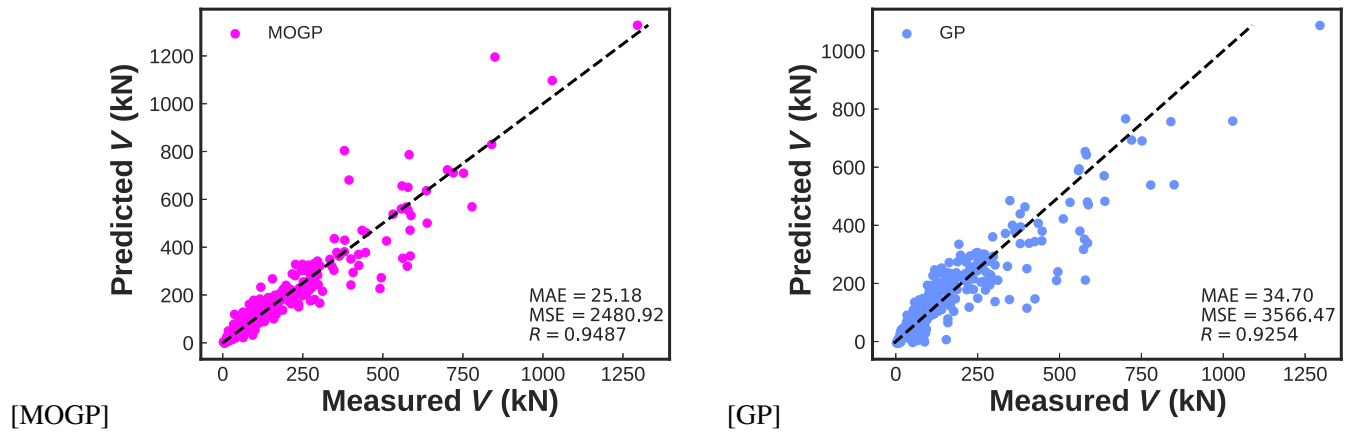


FIGURE 5 Measured vs predicted V(kN) values using A, the multiobjective genetic programming (MOGP)-fused model and B, the genetic programming (GP) model by Gandomi et al.[14] The black dashed line indicates the ideal fit

value of 2480.92 as MSE, and a value of 25.18 as MAE outperformed the other proposed models in the Pareto front and the GP model proposed by Gandomi et al[14] with a value of 0.9254 as R, a value of 3566.47 as MSE, and a value of 34.70 as MAE.

To study the quality assurance, mean value (μ) and coefficient of variation (CV) of this ratio are also reported. Figure 4 presents the comparison of the histograms of the MOGP-fused model and the GP model presented by Gandomi et al.[14] As seen, the MOGP-fused model has a mean value of 1.0840 and a coefficient of variation of 0.309 which is better than the GP model with a mean value of 1.0679 and a coefficient of variation of 0.477. By using the coefficient of variation, we achieve more clear understanding of the SD in light of the data mean value. Figure 5 shows the measured vs predicted shear strength values resulted from the developed models. Multistage genetic programming (MSGP) can be served as a substitute for the NSGA-II based model that was developed to decrease error decomposition.[11,25] There are two main stages in the MSGP algorithm:

1. Incorporating the individual effect of the input variables;
2. Incorporating the interactions among the input variables.

The MSGP formulates these two terms in an efficient procedure to optimize the error among predicted and actual values. This procedure can be parallelized to decrease overall computation time.

There are three machine learning regression models compared with the developed MOGP model[26]:

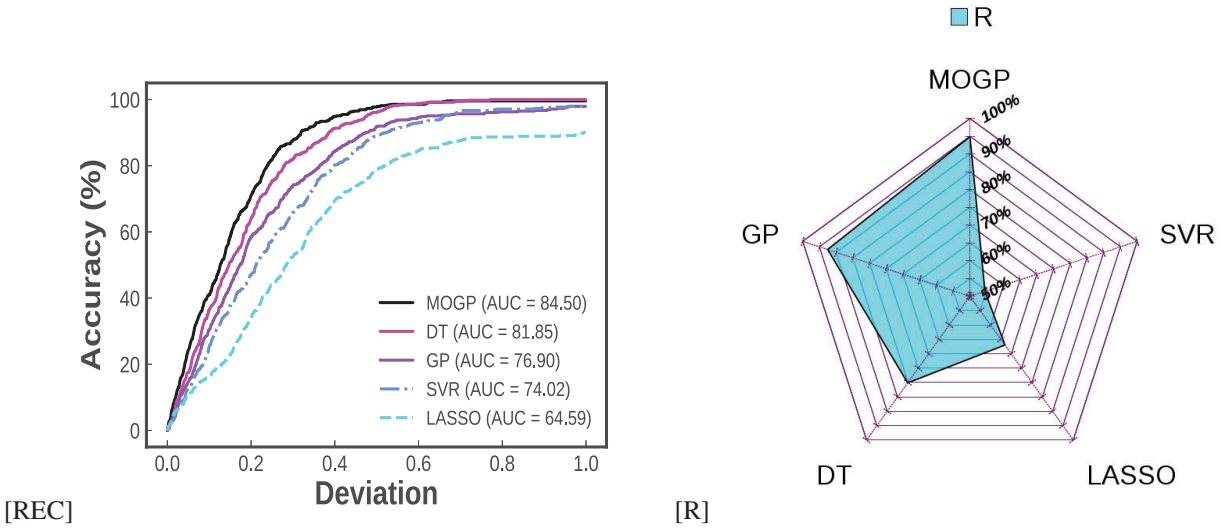


FIGURE 6 A, REC curves of various regression methods along with calculated AUC compared to the developed symbolic regression model using MOGP. B, Radar plot representation of the coefficient of determination for various regression methods compared to the developed symbolic regression model using MOGP. AUC, area under the curve; DT, decision tree; GP, genetic programming; LASSO, least absolute shrinkage and selection operator; MOGP, multiobjective genetic programming; REC, regression error characteristic; SVR, support vector regressor

TABLE 2 External validation results of the developed MOGP models

		MOGP			
	Condition	**Accurate**	**Fused**	**Knee**	**GP**
R	$R \geq 0.8$	0.94341	0.95071	0.94075	0.92806
$K = \frac{\sum_{i=1}^{n} h_i t_i}{h_i^2}$	$0.85 < K < 1.15$	0.90652	0.98390	0.87762	0.86914
$K' = \frac{\sum_{i=1}^{n} h_i t_i}{t_i^2}$	$0.85 < K^{'} < 1.15$	1.02946	0.95670	1.05910	1.05295
$R_m = R^2\left(1 - \sqrt{\lvert R^2 - R_0^2 \rvert}\right)$	$R_m \geq 0.5$	0.61620	0.62421	0.62295	0.57937
$R_o^2 = 1 - \frac{\sum_{i=1}^{n}\left(t_i - h_i^0\right)^2}{\sum_{i=1}^{n}\left(t_i - \bar{t}_i\right)^2}$	$h_i^0 = K \times t_i$	0.98467	0.99956	0.97270	0.96845
$R_o^{'2} = 1 - \frac{\sum_{i=1}^{n}\left(h_i - t_i^0\right)^2}{\sum_{i=1}^{n}\left(h_i - \bar{h}_i\right)^2}$	$t_i^0 = K' \times h_i$	0.99856	0.99690	0.99424	0.99537
$\lvert m \rvert = \frac{\lvert R^2 - R_0^2 \rvert}{R^2}$	$\lvert m \rvert < 0.1$	0.10634	0.10590	0.09907	0.12439
$\lvert n \rvert = \frac{\lvert R^2 - R_0^{'2} \rvert}{R^2}$	$\lvert m \rvert < 0.1$	0.12195	0.10295	0.12340	0.15565

Abbreviations: GP, genetic programming; MOGP, multiobjective genetic programming.

1. Decision tree (DT);
2. Support vector regressor (SVR);
3. Least absolute shrinkage and selection operator (LASSO).

Here complex models (eg, deep neural networks) were not used, first, since this study looks for explicit models, not black box models. Second, this is not a large-scale problem. Therefore, complex model are not good candidates due to the high over-fitting potential. A receiver operating characteristic (ROC) curve for a binary classifier is a way to visualize how the trade-off between true positive rate and true negative rate can affect the performance of the system. The area under the curve (AUC) is a measure of expected performance for that classifier.[27] In the case of a regressor,

regression error characteristic (REC) curve is used as a way to visualize the performance. It shows the percentage of the data instances predicted within tolerance (on the y-axis) vs the tolerance level (on the x-axis). The resulting curve estimates the cumulative distribution function of the error. The area over the ROC and REC curve (AOC), which is just 1 − AUC is a biased estimate of the expected error. The coefficient of determination R^2 can be also calculated with respect to the AOC.[27] In addition, The shape of the REC curve can also be used to provide insight for the analysts about the data modeling. The REC curve was implemented in Python[1] and the details of the error metrics and scaling of the residuals are also available.[28] Figure 6A compares the REC curves of the proposed MOGP-fused model along with the machine learning algorithms and Gandomi et al's model proposed in Reference 14. All of the models predicted all the instances with a normalized deviation of 0.55 correctly. By decreasing the deviation tolerance, the fraction of the correct predictions decreased as well. A value of 0.3 can be the critical tolerance for the employed models. The developed MOGP-fused model with an AUC of 84.50% outperformed the other models. In fact, the developed MOGP model has the ability of prediction of the instances with a deviation of less than 0.3. Also, shows the radar plot of the performance of the GP model proposed by Gandomi et al[14] and the regression models are shown in Figure 6B. In terms of coefficient of determination R. By comparing the coefficient of determination, it is clear that the MOGP model with an R value of 0.9487 outperformed the other models. The closest R score to the proposed model is the GP model presented by Gandomi et al[14] with an R score of 0.9254.

According to the external verification criteria suggested by Golbraikh and Tropsha[29] for a proposed model: the slope (K or K') of the regression line between the actual data (h_i) and the predicted data (t_i) should be close to 1, and the performance indices $|m|$ and $|n|$ should be lower than 0.1. In 2008, Roy et al[30] introduced an index (R_m) for external predictability evaluation of models. Their validation criterion is satisfied for $R_m \geq 0.5$. As shown in Table 2, the statistical parameters of the variations of the MOGP model are externally validated. Additionally, the external verification results of the proposed GP model by Gandomi et al[14] are also compared with the MOGP models. As seen, the MOGP-fused model outperformed the other models.

4 | CONCLUSIONS

This paper shows the application of robust solutions ($R = 0.9487$, MAE = 25.18, MSE = 2480.92, $\mu = 1.0840$, CV = 0.309, and REC − AUC = 84.50) employing MOGP specifically in engineering problems. In this regard, an evolutionary symbolic implementation for shear strength of slender RC beams was developed. The model uses NSGA-II to optimize subtree complexity as the complexity measure and the MSE as the fitness function. The final result was obtained after running the model for 2000 generations with the population size of 1000. Training-testing split was used to monitor and mitigate any possible over-fitting. Additionally, the mutation and the crossover rates were optimized over 200 generations based on the fitness and complexity measures. A total of five features from shear strength database were employed as the predictor input variables of the MOGP regression model. The developed MOGP model outperformed other machine learning algorithms that used this database for training.

ACKNOWLEDGMENT
The authors would like to thank Eitan Lees for the final revision of the manuscript.

CONFLICT OF INTEREST
No conflict of interest has been declared by the authors.

DATA AVAILABILITY STATEMENT
The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID
Amirhessam Tahmassebi https://orcid.org/0000-0003-4677-6907
Behshad Mohebali https://orcid.org/0000-0001-8013-0312
Anke Meyer-Baese https://orcid.org/0000-0001-6363-2687
Amir H. Gandomi https://orcid.org/0000-0002-2798-0104

ENDNOTE

[1] https://github.com/amirhessam88/Regression-Error-Characteristic-Curve

REFERENCES

1. Koza JR. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Vol 1. Cambridge, MA: MIT Press; 1992.
2. Holland JH. Genetic algorithms. *Sci Am*. 1992;267(1):66-73.
3. Dabhi VK, Chaudhary S. Empirical modeling using genetic programming: a survey of issues and approaches. *Nat Comput*. 2015;14(2):303-330.
4. Poli R, Langdon WB, NF MP, Koza JR. *A Field Guide to Genetic Programming*. North Carolina, UK: Lulu.com; 2008.
5. Koza JR, Bennett FH III, Andre D, Keane MA, Dunlap F. Automated synthesis of analog electrical circuits by means of genetic programming. *IEEE Trans Evol Comput*. 1997;1(2):109-128.
6. Streeter MJ, Keane MA, Koza JR. *Automatic Synthesis Using Genetic Programming of both the Topology and Sizing for Five Post-2000 Patented Analog and Mixed Analog-Digital Circuits*. New York, NY: IEEE; 2003:5-10.
7. Koza JR, Keane MA, Streeter MJ, Mydlowec W, Yu J, Lanza G. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Vol 5. Heidelberg, Germany: Springer Science & Business Media; 2006.
8. Silva P, Santos CP, Matos V, Costa L. Automatic generation of biped locomotion controllers using genetic programming. *Robot Auton Syst*. 2014;62(10):1531-1548.
9. Lazarus C, Hu H. Using genetic programming to evolve robot behaviours. In: Proceedings of the 3rd British Conference on Autonomous Mobile Robotics & Autonomous Systems, Manchester, 5 April 2001.
10. Lewis MA, Fagg AH, Solidum A. Genetic programming approach to the construction of a neural network for control of a walking robot. In: Proceedings 1992 IEEE International Conference on Robotics and Automation. Nice, France, Vol 3; 1992:2618-2623.
11. Gandomi AH, Alavi AH. Multi-stage genetic programming: a new strategy to nonlinear system modeling. *Inform Sci*. 2011;181(23):5227-5239.
12. Jalal M, Ramezanianpour AA, Pouladkhan AR, Tedro P. Application of genetic programming (GP) and ANFIS for strength enhancement modeling of CFRP-retrofitted concrete cylinders. *Neural Comput Appl*. 2013;23(2):455-470.
13. ACI. Standardization fIO. In: Building Code Requirements for Structural Concrete (ACI 318-08) and Commentary. American Concrete Institute; 2008.
14. Gandomi AH, Alavi AH, Kazemi S, Gandomi M. Formulation of shear strength of slender RC beams using gene expression programming, part I: without shear reinforcement. *Autom Construct*. 2014;42:112-121.
15. Veeramachaneni K, Arnaldo I, Derby O, O'Reilly UM. FlexGP. *J Grid Comput*. 2015;13(3):391-407.
16. Veeramachaneni K, Derby O, Sherry D, O'Reilly UM. *Learning Regression Ensembles With Genetic Programming at Scale*. New York: ACM; 2013:1117-1124.
17. Alimirzaloo V, Sadeghi M, Biglari F. Optimization of the forging of aerofoil blade using the finite element method and fuzzy-Pareto based genetic algorithm. *J Mech Sci Technol*. 2012;26(6):1801-1810.
18. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput*. 2002;6(2):182-197.
19. McDermott J, White DR, Luke S, et al. *Genetic Programming Needs Better Benchmarks*. New York: ACM; 2012:791-798.
20. Gandomi AH, Alavi AH, Ryan C. *Handbook of Genetic Programming Applications*. Heidelberg, Germany: Springer; 2015.
21. Tahmassebi A, Gandomi AH. Building energy consumption forecast using multi-objective genetic programming. *Measurement*. 2018;118:164-171.
22. Tahmassebi A, Gandomi AH, McCann I, et al. *An Evolutionary Approach for fMRI Big Data Classification*. New York: IEEE; 2017:1029-1036.
23. Jin H, Jespersen D, Mehrotra P, Biswas R, Huang L, Chapman B. High performance computing using MPI and OpenMP on multi-core parallel systems. *Parallel Comput*. 2011;37(9):562-575.
24. Yang Y. Adaptive regression by mixing. *J Am Stat Assoc*. 2001;96(454):574-588.
25. Tahmassebi A, Gandomi AH. *Genetic Programming Based on Error Decomposition: A Big Data Approach*. Heidelberg, Germany: Springer; 2018:135-147.
26. Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: machine learning in Python. *J Mach Learn Res*. 2011;12:2825-2830.
27. Bi J, Bennett KP. *Regression Error Characteristic Curves*; 2003:43-50). Palo Alto, CA: Association for the Advancement of Artificial Intelligence (AAAI).
28. Tahmassebi A. *iDeepLe: Deep Learning in a Flash*. Bellingham WA: [10652]. International Society for Optics and Photonics; 2018.
29. Golbraikh A, Tropsha A. Beware of q2! *J Mol Graph Model*. 2002;20(4):269-276.
30. Roy PP, Roy K. On some aspects of variable selection for partial least squares regression models. *QSAR Comb Sci*. 2008;27(3):302-313.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Tahmassebi A, Mohebali B, Meyer-Baese A, Gandomi AH. Multiobjective genetic programming for reinforced concrete beam modeling. *Applied AI Letters*. 2020;1–10. https://doi.org/10.1002/ail2.9