# Avoiding Overfitting in Symbolic Regression Using the First Order Derivative of GP Trees

Samaneh Sadat Mousavi Astarabadi
Computer Engineering and IT Department
Amirkabir University of Technology
Tehran, Iran
s4mousavi@aut.ac.ir

Mohammad Mehdi Ebadzadeh
Computer Engineering and IT Department
Amirkabir University of Technology
Tehran, Iran
ebadzadeh@aut.ac.ir

## ABSTRACT

Genetic programming (GP) is widely used for constructing models with applications in control, classification, regression, etc.; however, it has some shortcomings, such as generalization. This paper proposes to enhance the GP generalization by controlling the first order derivative of GP trees in the evolution process. To achieve this goal, a multi-objective GP is implemented. Then, the first order derivative of GP trees is considered as one of its objectives. The proposed method is evaluated on several benchmark problems to provide an experimental validation. The experiments demonstrate the usefulness of the proposed method with the capability of achieving compact solutions with reasonable accuracy on training data and better accuracy on test data.

## Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming

## Keywords

Genetic Programming; Multi-Objective Optimization; Symbolic Regression; Derivative; Generalization

## 1. INTRODUCTION

Improving generalization ability and avoiding overfitting with concerning the complexity of GP trees has been already considered in previous papers. Researchers proposed different complexity measures, e.g., variance functional [4], estimation of nonlinearity [6], estimation of curvature [5], variance [1] and Tikhonov regularization [3]. In this paper, the first order derivative of GP trees is considered as a simple and effective method of complexity control for improving GP generalization.

## 2. PROPOSED METHOD

Suppose that the data set $D = \left[(x_{i1}, \ldots, x_{im}; y_i)\right]_{i=1}^{n}$ contains n samples, where $x_i = (x_{i1}, ..., x_{im})$ represents all independent variables of dimension m and $y_i$ is the dependent variable. The goal of symbolic regression is to find a function that maps x to y, i.e. $y = f(x)$. However, usually the exact mapping is not possible. So the goal changes to find $\tilde{y} = \tilde{f}(x)$ that minimizes the Equation 1, known as Root Mean Squared Error (RMSE).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{y}_i - y_i)^2} \tag{1}$$

In order to avoid overfitting, when minimizing RMSE, complexity control is one of the suggested techniques. In this paper, an MOGP is used to optimize both the accuracy and the complexity of GP trees. The accuracy is the RMSE between $y$ and $\tilde{y}$, like many other GP papers, but unlike previous research, the complexity is a simple measure, i.e. the RMSE between the first order derivative of the desired solution, $y'$, and the first order derivative of GP tree, $\tilde{y}'$. Although $y'$ is not provided by the data set D, it can be numerically estimated by Equation 2. Evidently, for multivariate functions (functions with more than one independent variable), at every point, the directional derivative in direction of its nearest point is estimated, Equation 3. We can only estimate the derivative in direction of points that are available in the data set D and for other directions we don't have any information.

$$y'_i = \frac{df}{dx} = \lim_{h\to 0}\frac{f(x_i + h) - f(x_i)}{h} \tag{2}$$

$$y'_i = \bigtriangledown_{v_i} f(x_i) = \lim_{h\to 0}\frac{f(x_i + hv_i) - f(x_i)}{h|v_i|} \tag{3}$$

On the other hand, the directional derivative of GP tree at every point, $\tilde{y}'$, in direction of its nearest point is the dot product between the gradient and a unit vector that indicates the direction of its nearest point, Equation 4. As the gradient is the vector of partial derivatives, to calculate $\tilde{y}'$, partial derivatives must be calculated.

$$\tilde{y}'_i = \bigtriangledown_{v_i} \tilde{f}(x_i) = \bigtriangledown \tilde{f}(x_i).\frac{v_i}{|v_i|} \tag{4}$$

$$RMSE_{y'} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{y}'_i - y'_i)^2} \tag{5}$$

The partial derivatives of any GP tree with respect to its variables are simply obtained by the chain rule and a recursive procedure. There is one rule for each function in



*GECCO'15* Companion, July 11–15, 2015, Madrid, Spain.
ACM 978-1-4503-2138-9
http://dx.doi.org/10.1145/2739482.2764662

**Table 1: Recursive rules for computing the partial derivatives of GP trees.**

| Node | Function of Node | Derivative of Node |
|---|---|---|
| ADD | $LST$ [i] + $RST$ [ii] | $LST' + RST'$ |
| SUB | $LST - RST$ | $LST' - RST'$ |
| MUL | $LST \times RST$ | $LST' \times RST + LST \times RST'$ |
| DIV | $\frac{LST}{RST}$ | $\frac{LST' \times RST - LST \times RST'}{RST^2}$ |
| AQ[2] | $\frac{LST}{\sqrt{1+RST^2}}$ | $\frac{LST' \times (1+RST^2)}{(1+RST^2)\sqrt{1+RST^2}} - \frac{LST \times RST' \times RST}{(1+RST^2)\sqrt{1+RST^2}}$ |
| variable | value of variable | 1 or 0 [iii] |
| constant | value of constant | 0 |

[i/ii] the result of left/right subtree

[iii] in multivariate functions, the partial derivative of a variable with respect to itself is 1 and with respect to other variables is 0

function set in order to compute its derivative. These rules are summarized in Table 1 and can be extended as needed. It should be noted that, if a variable does not attend to a GP tree, the partial derivative with respect to it, is zero and does not need to be computed. For example, in the case of Toxicity problem with 626 input variables only 8-10 partial derivatives need to be computed and other partial derivatives are zero. Finally, the complexity of GP tree is controlled with Equation 5. Then these two objectives, Equation 1 and Equation 5, are used in an MOGP for application of symbolic regression.

## 3. EXPERIMENTAL RESULTS

Here, the experimental parameters are: # of runs = 30, population size = 200, termination condition = $2 \times 10^4$ fitness evaluations, crossover probability = 0.9, probability of point mutation = 0.02, tournament size = 10, initialization depth limit = 4, depth limit = 20, and function set = $\{+, -, \times, AQ[2]\}$. Instead of division operator, analytic quotient (AQ) operator [2] is used in order to ensure removing protected/unprotected division discontinuities and achieving the differentiability. The terminal set consists of variables and 100 random real constants. The fitness function is Equation 1 in single objective approach and Equation 1 and Equation 5 in multi-objective approach. For MOGP, an algorithm like NSGA-II with considering the characteristic of GP is implemented.

The proposed method has been tested on some real world regression problems that are also used in [1]. The data set of each benchmark problem is divided to the 50% training data and 50% test data. In Table 2, the training/test error of the best GP trees found after $2 \times 10^4$ fitness evaluations and the average size of the individuals in population (avg size) for both the standard GP and the proposed method are demonstrated. The solutions achieved by the proposed algorithm are smaller in size for all benchmark problems. The training/test error of the proposed method in the case of Concrete and Pollen benchmark problems are better than the standard GP. In the case of Toxicity and Bioavailability benchmark problems, the test error of the proposed method is better than the test error of the standard GP. It indicates that in comparison to the standard GP, the proposed algorithm have better generalization ability and the standard GP is more at the risk of overfitting. The results show that Equation 5 is a useful measure to select smaller solutions and to prevent overfitting.

**Table 2: Comparison between the proposed method and the standard GP in terms of average training/test error (RMSE) over 30 independent runs and the average size of the individuals in population after $2 \times 10^4$ fitness evaluations.**

| Benchmark | training error | test error | avg size |
|---|---|---|---|
| **Toxicity** | | | |
| StGP | $1726.3 \pm 136.6$ | $2198.9 \pm 54.86$ | 277.27 |
| proposed method | $1846.67 \pm 50.42$ | $2191.14 \pm 77.21$ | 89.39 |
| **Bioavailability** | | | |
| StGP | $27.38 \pm 1.81$ | $30.67 \pm 1.83$ | 257.8 |
| proposed method | $30.69 \pm 0.76$ | $29.52 \pm 0.91$ | 65.5 |
| **Pollen** | | | |
| StGP | $2.31 \pm 0.59$ | $2.33 \pm 0.58$ | 162.03 |
| proposed method | $1.52 \pm 0.23$ | $1.55 \pm 0.22$ | 105.23 |
| **Concrete** | | | |
| StGP | $12.73 \pm 3.23$ | $12.96 \pm 2.91$ | 164.7 |
| proposed method | $11.31 \pm 1.73$ | $11.69 \pm 1.51$ | 105.57 |

## 4. CONCLUSION

Avoiding overfitting using the first order derivative of GP trees is a simple and effective idea. By adding this objective in the evolution process, the solutions (GP trees) that overfit to the training data, are rejected. The results of the test error on the benchmark problems support the effectiveness of the proposed method. They have shown that the proposed method could find solutions with better generalization ability than the standard GP. Moreover, the proposed method achieves smaller solutions.

## References


[1] M. A. Haeri, M. M. Ebadzadeh, and G. Folino. Improving gp generalization: a variance-based layered learning approach. *Genetic Programming and Evolvable Machines*, 16(1):27–55, 2015.

[2] J. Ni, R. H. Drieberg, and P. I. Rockett. The use of an analytic quotient operator in genetic programming. *Evolutionary Computation, IEEE Transactions on*, 17(1):146–152, 2013.

[3] J. Ni and P. I. Rockett. Tikhonov regularization as a complexity measure in multiobjective genetic programming. *Evolutionary Computation, IEEE Transactions on*, 19(2):157–166, 2015.

[4] N. Y. Nikolaev and H. Iba. Regularization approach to inductive genetic programming. *Evolutionary Computation, IEEE Transactions on*, 5(4):359–375, 2001.

[5] L. Vanneschi, M. Castelli, and S. Silva. Measuring bloat, overfitting and functional complexity in genetic programming. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 877–884. ACM, 2010.

[6] E. J. Vladislavleva, G. F. Smits, and D. Den Hertog. Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming. *Evolutionary Computation, IEEE Transactions on*, 13(2):333–349, 2009.