# Assignment2
# Genetic Programming

Xiyao Huang
*Department of Computer Science*
*St. Francis Xavier University*
Antigonish, Canada
x2023fkv@stfx.ca
202306631

## I. Introduction

The main task of this assignment is to implement genetic programming for symbolic regression. Genetic programming is the use of genetic algorithms to evolve programs or functions, which usually use tree-structured genotypes to represent program/function phenotypes. In other words, it is a genetic algorithm that uses genetic operators that operate on tree encodings to search the space of possible programs or functions. Discovering relationships in experimental data and formalizing them into symbolic expressions is a key task in scientific research. To achieve this process, symbolic regression is usually used. [1] Neural Networks (NN) and Genetic Programming (GP) are the mainstream methods often used to implement SR in the literature. [2] Oh, Hongsup, et al(2023) proposed a machine learning method for discovering analytical solutions to differential equations. The method exploits symbolic regression with genetic programming. They demonstrated the ability of the method to recover the true analytical solution (rather than numerical approximation). They compared it with the traditional purely data-driven genetic programming symbolic regression algorithm and showed the reliability of successfully evolving the true solution or the equivalent algebraic solution. [3] Mundhenk, T. Nathan, et al. (2021) proposed a hybrid method combining neural guidance and genetic programming to solve symbolic regression and other combinatorial optimization problems. This method uses a neural guidance module to initialize the initial population of random restart genetic programming and gradually learn a better starting population. [4] Through scientists' continuous exploration of these aspects, we can see that GP for SR has broad potential and prospects, and has extremely high practical significance.

## II. Problem description

### A. Symbolic Regression

One of the grand goals of artificial intelligence is to automatically discover scientific laws from experimental data. Symbolic regression, that is, learning symbolic expressions from data, is an important step in achieving this grand goal. [5] The problem of this assignment is to complete a valid GP and pass the provided test case, and perform interesting evolutionary computation operations on this GP to improve the algorithm. At the same time, realize visualization. The difficulty is that many such examples exceed two or three dimensions. This will make it difficult to visualize and draw data.

### B. The Typed Problem

The typing problem in symbolic regression is to introduce specific data type constraints. Thus, GP can be applied to a specific problem, such as electric power forecasting, email filtering and other interesting and practical problems. In my homework, I will apply it to electric power forecasting. I think the difficulty lies in the high complexity of the algorithm for predicting electric power. There is a lot of room for improvement in efficiency, but it is also very difficult.

## III. Algorithm Description

### A. Symbolic Regression

#### 1) Implementation:

- Set Operator Set: The mathematical operations for symbolic regression are defined, and the number of operators and their parameters are set through the gp function imported from the deap package, namely gp.PrimitiveSet. Also for functions like exponential and logarithmic, safe versions are defined to prevent overflow or invalid input
- Initialization: First, create individuals. Use the creator module of DEAP to create the individual class Individual, which is a gene expression tree that conforms to the design logic of gp. Then initialize the population: create a population of the specified size, each individual is randomly initialized and generated by the symbolic expression tree, and use the gp.genHalfAndHalf method to ensure depth and diversity.
- Fitness Evaluation: The fitness function is written to calculate the mean square error (MSE) and use it as the fitness value. This function compiles an individual expression tree and uses it to predict the data. The predicted result is controlled within a reasonable range so that the value will not be too large. At the same time, the function also handles some abnormal situations to ensure that the error is reported.

- Selection: In this GP, Tournament Selection is used, and the tournament size is 3. This selection method provides a certain diversity while also selecting individuals with higher fitness.
- Mutation: The mutation operation randomly replaces certain nodes in the tree, which allows changes in individual structures and thus ensures genetic diversity.
- Cross: A crossover function is designed to exchange part of the structure between two expression trees. The parameter in this function makes the crossover more biased towards leaf nodes, which can keep the expression relatively simple.
- Visualization: A function is written to visualize the true value and the predicted value. This is useful for evaluating whether GP can find the best individual. One point represents the true value and the other point represents the predicted value, and then the overlap between the two is observed in the image. If the overlap is high, it means that the prediction is relatively accurate.

*2) Enhancements:*

- Elitism: Elitism can improve the convergence speed of the algorithm by retaining the current best individual in each generation, ensuring that the optimal solution will not be lost due to crossover or mutation. First, define the object and add it to the DEAP algorithm so that the best individual in the evolution process is recorded. Then, in the evolution process of each generation, the current best individual is automatically stored so that it will not be lost due to random mutation or crossover. Finally, the best individual is extracted as the final output
- Leaf-Biased Crossover: In symbolic regression, the crossover operation may generate a very complex expression tree, making the model difficult to interpret. However, adopting a leaf-biased crossover strategy can reduce the complexity, keep the model simple, and improve the interpretability of the expression.The crossover function is biased towards selecting crossover points close to leaf nodes when crossing over. Setting the parameter termpb to 0.1 controls the degree to which crossover is biased towards terminal nodes, which can preserve variables and constants in expressions while generating too many nodes. This enhancement ensures that simpler offspring are generated, reducing the risk of overfitting.
- Safe Operation: In symbolic regression, when using exponential and logarithmic operations, overflow or calculation errors often occur because the input value is too large or too small. Therefore, designing safe operations can effectively enhance the stability of the algorithm. First, define a function for safe operations to limit the input value to a reasonable range and avoid returning an exponential value that is too large. Then define another function to check whether the input value is positive to avoid errors caused by negative numbers. Then add it to the operator set to ensure that these operations can be selected during the generation of the expression tree.

### *B. The Typed Problem*

*1) Implementation:*

- Set Operator Set: The basic mathematical operations of symbolic regression are defined to form expression trees, and safety functions are set to prevent numerical overflow or invalid input in exponential and logarithmic operations.
- Fitness evaluation: First, the fitness function and individual type are created, and then the individuals are initialized by randomly generated expression trees through a population generator. Calculate the mean square error (MSE) between the predicted value and the true value of the expression tree. Limit all predicted values to a reasonable range to prevent excessive values.
- Selection: The tournament selection is also adopted with a tournament size of 3 to ensure that individuals with better fitness are more likely to be selected.
- Mutation: Use the mutUniform operation to uniformly mutate the nodes of an expression tree.
- Crossover: Perform single-point crossover while ensuring that crossover is biased towards leaf nodes, thereby retaining more terminal nodes.
- Termination: The genetic programming evolution process was executed through a function, with a total of 40 generations of iterations set and an elitism strategy used to retain the best individuals.
- Visualization: Similar to SR, a comparison chart of the true value and the predicted value is drawn to intuitively evaluate the prediction effect of the model.

## IV. Results

Because there are a lot of test data, the results are displayed using d06 as an example. First, the results are shown in two parts: 1. The blue dots represent real data points. 2. the yellow dots represent the data predicted by the algorithm. The way to judge whether the prediction is accurate in this visualization is to see whether two different points overlap and how much overlap there is. If most of them overlap, it means that the prediction is relatively accurate. From the results, we can say that the prediction of the GP algorithm for SR has a relatively high accuracy. Because almost all points are overlapped, only one true value has a significant deviation from the pre-test, but overall the prediction accuracy is quit high.

The instances contained in d06 are three dimensions. However, there are still four-dimensional data such as d16, so we still need to observe the performance of the algorithm in the case of instances with more than two or three dimensions. From the results, we can see that although the effect of each run may be slightly different, overall, its accuracy remains at a high level. Therefore, the GP can be successfully applied to all the test examples provided and achieve a good result.

Next, I will use statistical methods to analyze its results.

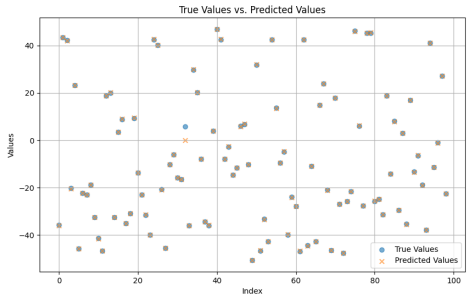


Fig. 1. Result-d06

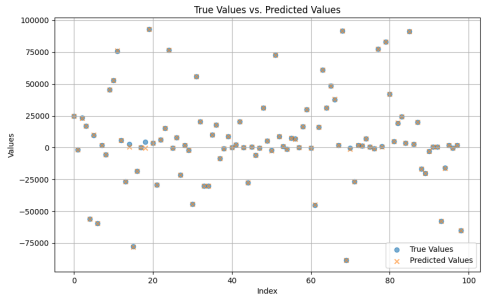


Fig. 2. Result-d16

### A. *Best fitness values for d06*

From this result, we can see that the best fitness values in the 30 runs are relatively concentrated. Most of the best fitness values are concentrated between 0.30 and 0.35, which shows that the algorithm performs basically the same in these runs and reaches a fitness value close to the best. And perhaps because of the better results produced by chance or due to other factors, there are also a few runs with fitness values around 0.05. In short, most runs have similar fitness values, indicating that the algorithm can stably achieve nearly the same performance level in most cases.

I also wrote a program to calculate the best fitness mean, it is about 0.333 and the standard deviation is about 0.0788 For the three-dimensional data d06, the closer the mean is to 0, the lower the average error that the model can achieve. The model fits the objective function well, but there will still have some error.

In addition, the p-value of the t-test is 0.091. Usually a value less than 0.05 proves that there is a significant difference. However, the current value can show that there is no significant difference and it is relatively stable. The p-value of the Mann-Whitney U test was 0.065. This p-value is also greater than 0.05. Therefore, the results of the Mann-Whitney U test also indicate that there is no significant difference theoretically, but because the value is close to 0.05, there may still be a difference worthy of attention in reality. The value of Cohen's D test was 0.451, this value should usually be considered as a medium effect size, meaning that there is some difference but it is not significant.

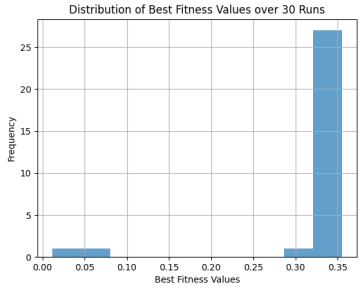


Fig. 3. Best-fitness-values-d06

After implementing the Genetic Programming (GP) for Symbolic Regression, I further explored Typed Problem for City Power Forecast. The visualization method is similar to the SR problem. Two different points are used to represent the original data and the predicted data respectively, so as to check the overlap between the two to determine whether the prediction is accurate or not. From the results, we can see that most of the real values and predicted values are overlapped, which means that the GP can basically meet the accuracy of prediction. However, there are still many deviations, and some deviations are very obvious, and the predicted values are relatively discrete. These parts reflect that the stability of the algorithm in practical applications is not very high, so there is still room for improvement. The following is the result graph (The test data comes from online resourse):

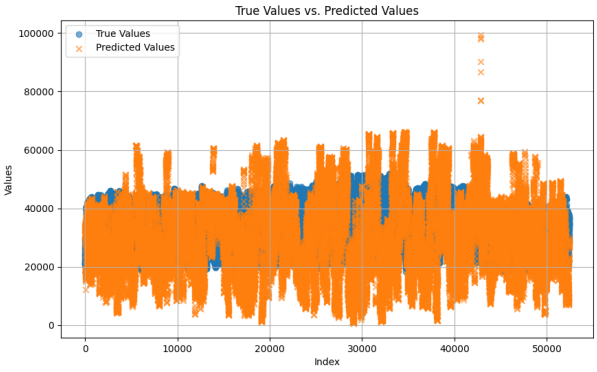


Fig. 4. City Power Forecast

## V. Discussion

The test results of Genetic Programming for Symbolic Regression show high accuracy, with most real data points highly coincident with the predicted values. It can be seen that the model is able to predict most of the real data well, with only a few points deviating significantly. Therefore, GP could be treated as a relatively stable and reliable solution to the SR problem.

GP performs significantly better compared to random methods. Randomness tends to produce irregular and inconsistent

results. GP through evolutionary operations such as selection, crossover, mutation, and elitism should theoretically achieve a higher accuracy than random methods. Although GP also has possibilities to make random results, which means that the results of running GP each time may be slightly different. For example, although most test data have a high overlap with real data, there are a few times where large deviations occur, but overall there are few deviations and the probability is not high, so it is more stable than random methods.

GP is easier to explain than other algorithms such as neural networks. Neural networks have relatively high accuracy, but their processes are difficult to understand. For example, I learned in a DL course that neural networks have hidden layers, this part can be also called as Black Box, so their processes are sometimes not analyzable. GP can provide clearer mathematical relationships for understanding.

Comparing the results of this GP with the best known techniques in symbolic regression shows that, while the overall performance is stable, it is often easy to have anomalies in certain data, especially in four dimensions, resulting in large outliers. This is also a common problem of evolutionary computation, nobody can guarantee that the algorithm will perform best in every case.

The modifications I made in this GP, such as elitism and leaf node bias crossover, makes the algorithm more explainable and have higher convergence. Elitism retains the best individuals in each generation, ensuring the quality of the population. Leaf node bias crossover keeps the expression relatively simple. These improvements have made my results more stable and reliable.

## VI. Conclusion

The result of this assignment is to implement a relatively stable and efficient method of genetic programming in symbolic regression tasks. The GP algorithm successfully predicted the Symbolic Regression, and the overlap with the true value in the test data was high, showing that GP can play the role of an effective solution in the SR problem. Although the algorithm performs well, there is still a lot of room for improvement. For example, when predicting four-dimensional data, although the results are relatively accurate, they are not as stable as when predicting three-dimensional data. The probability of abnormal data is high. For example, the value of the best fitness mean will be very high, because sometimes there will be very different predictions. The stability and accuracy can be improved by exploring more advanced crossover and mutation techniques or combining them with technologies such as neural networks. In the City Power Forecast Typed Problem, most of the predictions are relatively accurate, which is a relatively successful case, but there are still some data with large differences, and there is still a lot of room for improvement.

## References


[1] Zeng, Peng, et al. Differentiable Genetic Programming for High-Dimensional Symbolic Regression. 2023, https://doi.org/10.48550/arxiv.2304.08915.

[2] Koza, John R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, 1992.

[3] Oh, Hongsup, et al. Genetic Programming Based Symbolic Regression for Analytical Solutions to Differential Equations. 2023, https://doi.org/10.48550/arxiv.2302.03175.

[4] Mundhenk, T. Nathan, et al. Symbolic Regression via Neural-Guided Genetic Programming Population Seeding. 2021, https://doi.org/10.48550/arxiv.2111.00053.

[5] Jiang, Nan, and Yexiang Xue. Symbolic Regression via Control Variable Genetic Programming. 2023, https://doi.org/10.48550/arxiv.2306.08057.