Comprehensive Learning Gene Expression Programming for Automatic Implicit Equation Discovery

Yongliang Chen[1], Jinghui Zhong[1] (Corresponding author), and Mingkui Tan[2]

[1] School of Computer Science and Engineering,
South China University of Technology, Guangzhou, China
jinghuizhong@gmail.com
[2] School of Software Engineering,
South China University of Technology, Guangzhou, China

Abstract. Implicit equation is loose in form, which makes it more powerful than explicit equation for data regression. The mainstream method for automatic implicit equation discovery is based on calculating derivatives. However, this derivative-based mechanism requires high time consumption and it is difficult to solve problems with sparse data. To solve these deficiencies, this paper proposes a new mechanism named Comprehensive Learning Fitness Evaluation Mechanism (CL-FEM). The mechanism learns knowledge from *disturbed* information collected from several previously generated stochastic datasets, to check the validity of the equation model. Only the valid equations can be candidates of selection, which is a process to pick out the equation with the smallest output. We integrate the proposed mechanism with the simplified Self-Learning Gene Expression Programming (SL-GEP) and propose the Comprehensive Learning Gene Expression Programming (CL-GEP). The experiment results have demonstrated that CL-GEP can offer very promising performance.

Keywords: Implicit equation, symbolic regression, gene expression programming (GEP), *disturbed* knowledge learning.

1 Introduction

Genetic programming (GP) is a powerful evolutionary computing technique that has been used to solve complicated optimization problems[3, 5]. Due to its high efficacy, GP has aroused people's attention these years, and many enhanced variants of GP [7, 6, 1] have been developed, such as Gene Expression Programming (GEP)[2, 12] and Self-Learning Gene Expression Programming (SL-GEP) [14]. Thus far, GEP has been applied to a number of science and engineering fields [4, 8, 13, 11].

One of the most common applications of GP is symbolic regression, which requires finding proper mathematic equations to fit the given observed data. Symbolic regression problem has profound influence on our life, for finding proper

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

equations to express data can help people predict future incidents or even excavate unknown laws of nature. The existing GPs, however, are mostly focused on finding explicit equation. An explicit equation in an D-dimension space can be described as $y = f(\boldsymbol{x_{D-1}})$, where $\boldsymbol{x_{D-1}}$ is a $(D-1)$-dimension variable vector, the dependent variable y is separated and is expressed by an explicit formula. As y is separated and the form is fixed, it somehow sacrifices the expression capability of y, therefore, implicit equation is proposed. The form of an implicit equation in an D-dimension space can be $f(\boldsymbol{x_{D-1}}, y) = 0$, putting y in the function $f(...)$ can make the equation more expressive.

To find an approximate implicit model seems very direct and concise, however, there exists many perennial and fundamental problems. The most attractive problem is how to guarantee the feasibility of a model. The aim of symbolic regression problems for implicit equation is to find out a mathematical rule $f(...)$ consisting of $x_1, x_2, ..., x_D$, which can fit the target dataset best to make $f(x_1, x_2, ..., x_D) = 0$. Some direct methods which only focus on reducing $f(...)$ to zero will always converge very fast to some meaningless functions, like $x - x$ and $sin^2(x) + cos^2(x) - 1$, which are equivalent to zero. Therefore, how to avoid models converging to these misleading equations is the key to handle implicit symbolic regression problems.

The mainstream method to avoid finding infeasible models is to minimize the differences of gradients between the model and the dataset [9]. The core idea is assigning fitness value by calculating the differences between the partial derivatives of each dimension of the target point and the derivatives calculated using the neighboring point of the target one. This technique has proved to be useful because it considers the shape of the model rather than only concentrating on the output of the function.

We have practiced solving implicit equation tasks with this method, however, we found some weakness of the derivative-based fitness evaluation mechanism (DB-FEM). On the one hand, a large training dataset is needed. Once the data points are not continuous enough, partial derivatives will be lacking in reliability and accuracy. On the other hand, calculating derivatives is time-consuming and inconvenient. When the set is large (contrast to the previous point) and the model is in high dimension space, using this method seems impractical.

To solve these problems, we propose a new mechanism to evaluate symbolic regression problems for implicit equations, named Comprehensive Learning Fitness Evaluation Mechanism (CL-FEM). Traditional GPs only learn from positive training data (i.e., those need to fit), using these data to guide the search process. While in our method, we learn from not only positive training data, but also negative training data, so we call it Comprehensive Learning. Rather than calculating derivatives, we test the model function not only by using the target dataset, but also by using several randomly generated, stochastic datasets produced in advance. Each dataset focuses on one dimension of the equation, and the equation model makes sense only if every dimension makes contribution to solving the problem. Otherwise, the model will be regarded as a failure.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

To test this mechanism, we combine it with a simplified Self-Learning Gene Expression Programming (SL-GEP), and propose a new algorithm for automatic implicit equation discovery, named Comprehensive Learning Gene Expression Programming (CL-GEP). We implement CL-GEP to some typical implicit function models to test the new mechanism. We not only pay attention to the success rate of finding accurate functions, but also the time consumption. Also, we use sparse datasets to test the proposed CL-FEM. To demonstrate the superiority, we make comparisons with the DB-FEM. Results have shown that CL-FEM is promising.

2 Preliminaries

To ensure that readers can have a better understanding of the paper, this section will present some preliminary knowledge about the implicit equation, and the derivative-based fitness evaluation mechanism (DB-FEM).

2.1 Implicit Equation Problem

An implicit equation is a function adopting $\boldsymbol{x_D}$ as input and the output is equivalent to zero, which is expressed as $f(\boldsymbol{x_D}) = 0$.

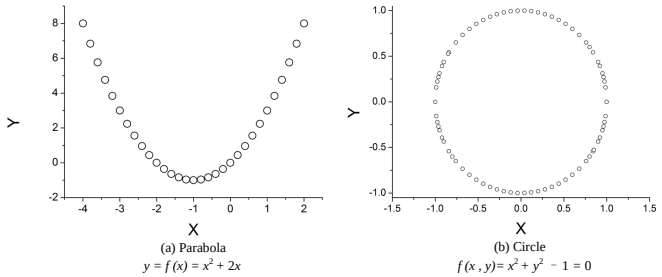

Fig. 1. Examples for explicit equation (parabola) and implicit equation (circle).

The need for implicit equation models arises when there is no independent variable in a dataset, or at least we do not know which variable can be the independent variable. To try to find an independent variable or make assumption is impractical, therefore, implicit equation is the best choice. Transferring an explicit equation to an implicit equation is very easy. For example, $y = f(x) = x^2 + 2x$ (Fig. 1 a)can be translated into an implicit form $g(x, y) = f(x) - y = x^2 + 2x - y = 0$ very easily. On the contrary, if you want to transfer an implicit equation, like $x^2 + y^2 = 1$ (Fig. 1 b), to an explicit equation, it is a piecewise function $y = \pm\sqrt{1 - x^2}$, which is very inconvenient. Therefore, implicit equations are loose in form and are more powerful than explicit equations in expressing data to some extent.

As the advantages of implicit equation are realized, several trials for implicit equation discovery have been made previously. The main challenge is that there are infinite implicit equations that are valid for any dataset, which proves finding

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

$f(\boldsymbol{x_D}) = 0$ directly infeasible, as mentioned above. To make symbolic regression more comprehensive, the need for effective implicit equation evaluation mechanism is highly emergent.

2.2 Derivative-based Fitness Evaluation Mechanism

The mostly used method today to solve implicit equation regression problems is using partial derivatives. Using this mechanism not only can ensure that the target point fits the data, but also can predict the indirect relationships between variables of the system.

For an implicit equation $f(x, y) = 0$, abandon the $= 0$ symbol, we can calculate the partial derivatives $\delta f/\delta x$, $\delta f/\delta y$. Then, from derivative rules, we have

$$\frac{\delta y}{\delta x} = -\frac{\delta f/\delta x}{\delta f/\delta y} \tag{1}$$

where $\frac{\delta y}{\delta x}$ is the implicit derivative which can express the gradient of the implicit equation. When it comes to three or more dimension equations, more derivative values like $\frac{\delta z}{\delta x}$, $\frac{\delta z}{\delta y}$ should be used to describe the information of the equations. Then, we estimate the implicit derivatives $\triangle y/\triangle x$ from the dataset, where $\triangle x = x_a - x_{a+1}$ and $\triangle y = y_a - y_{a+1}$, a is the index of the target point in the dataset, the data need to be in *order* when calculating derivatives. To evaluate an implicit equation model, differences between the model and the dataset are expressed as

$$\frac{1}{N}\sum_{n=1}^{N} log(1 + |\frac{\triangle y}{\triangle x} - \frac{\delta y}{\delta x}|) \tag{2}$$

where N is the size of the dataset. When the number of dimensions is more than two, the estimation of $f(x_1, x_2, ..., x_D)$ becomes

$$\frac{1}{N}\sum_{n=1}^{N} log(1 + |\frac{\triangle x_2}{\triangle x_1} - \frac{\delta x_2}{\delta x_1}| + |\frac{\triangle x_3}{\triangle x_1} - \frac{\delta x_3}{\delta x_1}| + ... + |\frac{\triangle x_D}{\triangle x_{D-1}} - \frac{\delta x_D}{\delta x_{D-1}}|) \tag{3}$$

From Eq. 3, we can easily find that the examination cost of DB-FEM in computational time is $O(ND^2)$, where D is the number of dimensions. The $O(ND^2)$ time consumption is high when the estimated mechanism should be repeated thousands of times during GP. Besides, finding the nearest point of the target point in the dataset is another time consuming task. What's more, using derivatives as criterion to evaluate a model requires the dataset to be large and crowded enough, otherwise, derivatives may convey wrong information (see Fig. 2).

3 The Proposed Mechanism and Algorithm

This section will first describe the proposed CL-FEM, which is a new direction to evaluate an implicit function model. Then the proposed CL-GEP will be introduced.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

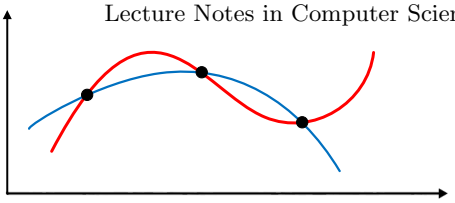

Fig. 2. Example for a dataset with few data points. The red curve represents the target equation, while the DB-FEM tends to find the equation represented by the blue curve, for it is smoother and fits the derivatives better.

3.1 Comprehensive Learning Fitness Evaluation Mechanism

In contrast to the existing mechanism like calculating derivatives, the application of CL-FEM is more direct and concise. As mentioned in previous sections, the main difficulty of finding proper implicit equation model is how to avoid being trapped in equations which are equivalent to zero. Instead of concentrating on partial constitution, we focus on the output of $f(\boldsymbol{x_D})$, but selectively.

The CL-FEM regards a model meaningful under a premise that each dimension of $\boldsymbol{x_D}$ contributes to solving the problem. To be more specific, the model is illegal when some dimensions are missed, like $f(x, y) = x^3 + x$, or some dimensions are useless, like $f(x, y) = y + x^2 - y$. We estimate outputs of the functions by implementing data and computing the mean square error (MSE) of the results with zero, which is defined as

$$MSE = \frac{1}{N}\sum_{j=1}^{N} s_j^2 \tag{4}$$

Most of all, we need to ensure that a model is not misleading. To justify the validity of a model, CL-FEM uses the *disturbed* data.

Step 1-Form stochastic datasets

For each dimension of the variable vector $\boldsymbol{x_D}$, we generate a stochastic dataset. Data in the stochastic datasets are generated at the beginning of GP, the dataset for the kth dimension is produced following the rule of

$$x_{ij} = \begin{cases} randval(L_k, U_k) & j = k \\ t_{ij} & otherwise \end{cases} \quad i = 1, 2, ..., N \quad j = 1, 2, ..., D \tag{5}$$

where N is the volume of the dataset, D is the dimension number for each point, $randval(a, b)$ returns a random value in the range of (a, b), L_k and U_k are the upper bound and the lower bound of the kth dimension, and t_{ij} is variable from the target dataset. Points are distributed randomly in one dimension, which can be a tool for us to judge whether every dimension is meaningful.

Step 2-Estimate target models

We use the datasets generated in *Step 1* to estimate whether every dimension contributes to the target model.

First, calculate the results using the target dataset and all stochastic datasets, forming a set consisting $(D + 1)$ result vectors $\{\boldsymbol{S}, \boldsymbol{S^1}, \boldsymbol{S^2}, ..., \boldsymbol{S^D}\}$. $\boldsymbol{S}$ is the result vector gained from the target dataset, and the others are result vectors

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

gained from the D stochastic datasets. Then, we estimate the model by calculating MSE between $\boldsymbol{S}$ and $\boldsymbol{S^i}$s

$$m_i = \frac{1}{N}\sum_{j=1}^{N}(s_j - s_j^i)^2, s_j^i \in \boldsymbol{S^i}, \quad i = 1, 2, ..., D \tag{6}$$

It is easy to justify that if the i-dimension makes sense in the model, there must be $m_i > 0$. We set $1E-4$ to be the *tolerance degree*, and set the fitness value V of the obtained model by

$$V = \begin{cases} MSE & (m_1 > 1E-4) \wedge (m_2 > 1E-4) \wedge ... \wedge (m_D > 1E-4) = TRUE \\ 1E10 & otherwise \end{cases} \tag{7}$$

where $1E10$ is a huge constant which means the model is obsolete. Like the direct methods, a smaller V is preferred during the evolution. When there is a D-dimension equation model, the examination time consumption of CL-FEM is $O(ND)$, and the calculation processes are convenient and practical.

3.2 The Comprehensive Learning Gene Expression Programming Algorithm

The CL-GEP adopts a simplified SL-GEP [14], using a novel representation in chromosomes. What's more, we make some fine-tunings for the convenience of use.

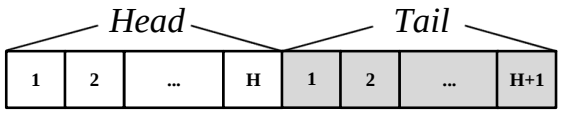

Fig. 3. The chromosome representation of an individual.

Chromosome Representation The proposed CL-GEP removes the mechanism of using subfunctions, for we want to get more concise and readable results, however, it keeps the basic chromosome representation of SL-GEP. In CL-GEP, each chromosome consists of a *Head* and a *Tail*. The *Head* and the *Tail* are stored in a chromosome continuously. The *Head* can contain both functions (e.g., + or -) and terminals (e.g., x, y, or 1), while the *Tail* can only store terminals. Suppose that the length of the *Head* is H, then the length of the *Tail* should be $H + 1$, to ensure that the chromosome can be decoded successfully even if the *Head* only contains functions.

The form of the chromosome is presented in Fig. 3 and an example of decoding an implicit equation is presented in Fig. 4. Notice that, as the length of a chromosome is fixed, some information in the chromosome may not be used, however, as the length of the used chromosome is flexible, it can improve the diversity of population to some extent.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

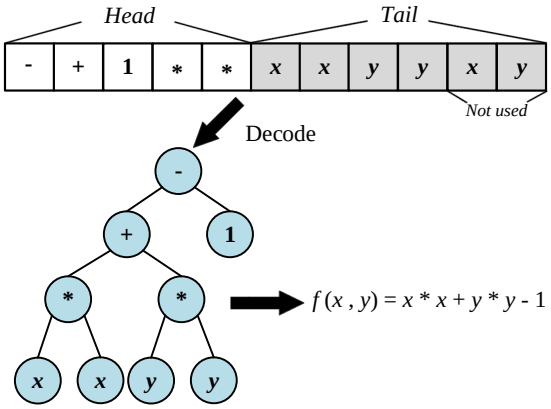

Fig. 4. Decode a chromosome to an implicit equation.

Function Set and Terminal Set We make some fine-tunings of the terminal set and function set of SL-GEP to construct CL-GEP's. The function set of CL-GEP is much simpler

$$\Psi = \{+, -, *\} \tag{8}$$

We abandon the function / in CL-GEP, for implicit equations are equivalent to zero, which is only affected by numerator in a fraction. Therefore, adding / to the function set is unnecessary, and it will make the output more difficult to comprehend.

As for the terminal set, we add constant 1 to be a terminal, and the set can be expressed as

$$\Gamma = \{x_1, x_2, ..., x_D, 1\} \tag{9}$$

Because we remove / from the function set, constant 1 or other constant can not be decoded as x/x in CL-GEP. In order to improve the capability of precise expression, we make 1 a terminal choice. It is proved to be necessary, for 1 is always an important component of an implicit equation.

Mutation and Crossover Mutation and Crossover in CL-GEP are the same as SL-GEP. The mutation process is a variant of the mutation in traditional Differential Evolution (DE) [10], making three main improvements, *Distance measuring*, *Distance adding* and *Distance scaling*, to ensure that it is feasible in the unique chromosome representation. Crossover is performed after mutation to make offsprings vary in diversity.

Selection In this step, good offsprings are selected to replace their parents to become new individuals in the new population. The result of selection is based on CL-FEM.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

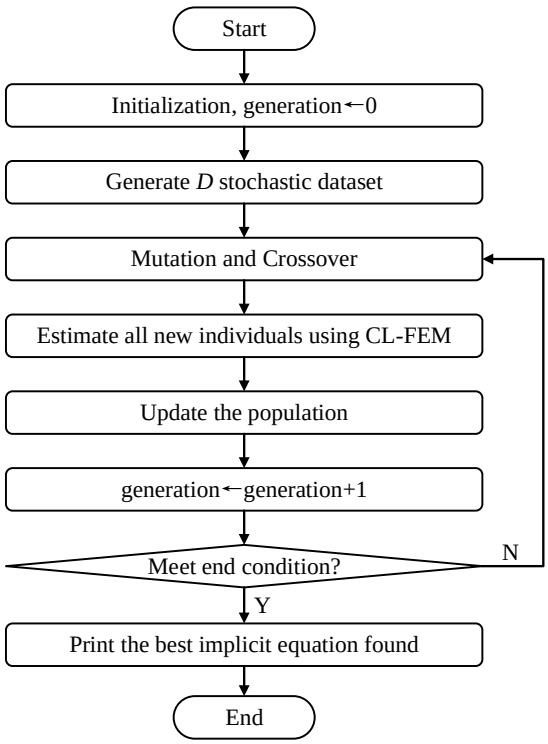

Fig. 5. The flowchart of the proposed CL-GEP algorithm.

The flowchart of the algorithm is shown in Fig. 5. We set limits to the number of generations as the terminal condition. At last, the best-so-far implicit equation model, which has the smallest fitness value V, will be regarded as the final solution to the problem.

4 Experiments

This section investigates the performance of the proposed CL-GEP algorithm. First, the experimental settings are presented. We also implement the DB-FEM to the simplified SL-GEP and generate a testing algorithm named DB-GEP, whose fitness value V is calculated by

$$V = MSE + diff \tag{10}$$

where $diff$ is calculated by Eq. 3.

The settings for the two algorithms are the same and they are competed with each other. Then, we will present the experiment results and have some discussions.

4.1 Experimental Settings

We experiment with four typical implicit equation problems with varying difficulty (Fig. 6). The first two equations, *Circle* and *Hyperbola*, are the simplest

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

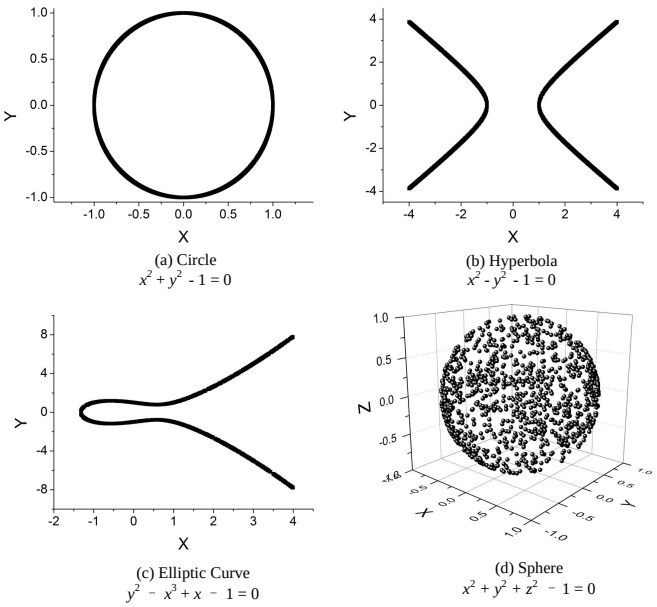

Fig. 6. Data sampled from the four target implicit equation systems.

implicit equations for the feature of the target curves are easy to capture and the combining rule for x and y is concise. The third equation is *Elliptic Curve*, which is more complex in form and harder to predict. For these three equations, we take 500 sample points randomly from each system. The forth one is a 3-dimension equation *Sphere*, which is the most difficult one in the four tasks. For this equation, We sample 1000 points from the space.

The four equations are estimated by the two algorithms, CL-GEP and DB-GEP. For each equation, we run 30 times, 20,000 generations in each time, then calculate the success times during the 30 runs and the average time consumption for one run. Programs are run on a computer with INTEL I7-6500U CPU. The detailed parameter settings for the algorithms are listed in Table 1.

Table 1. Parameter settings.

Parameter	*Value*	*Summary*
H	15	The length of Heads
F	$rand(0,1)$	Scaling factor of mutation
CR	$rand(0,1)$	Crossover rate
$MAXGENS$	$20,000$	Maximum generations
$POPSIZE$	200	Size of the population

To test the algorithms' practicability when faced with datasets of different densities, experiments are implemented for sparse data on the first three equations, *Circle*, *Hyperbola* and *Elliptic Curve*. Each equation is run for 30 times,

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

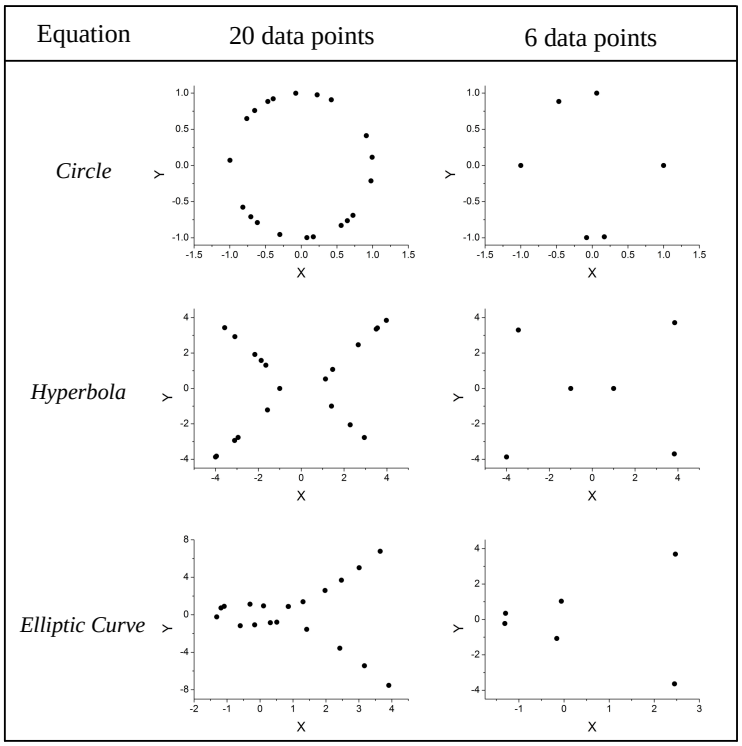

Fig. 7. Sparse datasets of *Circle*,*Hyperbola* and *Elliptic Curve*.

20,000 generations in each time. We calculate the success times when there are respectively 500, 100, 20 and 6 data in the set, and make comparisons between the two algorithms.

To show the sparse datasets more intuitively, we mark the points in the space for sets containing 20 and 6 data points, which are shown in Fig. 7. It can be observed that when the datasets contain 20 data points, we can still recognize the general features of the curves. When it comes to datasets containing 6 data points, however, it is almost impossible to tell out what they refer to. If the algorithm is able to help find out the relationships between variables from these sparse datasets, it will be of great significance.

4.2 Experimental Results and Discussion

We conduct experiments from two directions. The first one focuses on the capability of the algorithms to find the right equation models, where data are crowded enough in the datasets. The second one aims to test how the two algorithms perform when the datasets are sparse.

Experiments on the Four Equations Table 2 shows the results of the first experiment, the success times and average time consumption to find a successful equation are presented.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

Table 2. Results on running four implicit equations.

Problem	CL-GEP		DB-GEP	
	Suc	*Time*(s)	*Suc*	*Time*(s)
Circle	30	231.57	30	371.13
Hyperbola	30	91.62	30	405.03
Elliptic Curve	29	148.78	20	467.59
Sphere	0	\	0	\

The results have shown that the proposed CL-GEP algorithm has gained promising performance against DB-GEP. It can be observed that both of the algorithms can perfectly solve the first two problems, which are concise in forms. When it comes to *Elliptic Curve*, however, the superiority of CL-GEP reveals distinctly, with the DB-GEP only getting the success rate of 66.7%, while the CL-GEP only fails in one run, holding the success rate of 96.7%.

Consider the DB-GEP, concentrating on the partial feature of the model can guarantee the validity of the model to some extent, and it is certainly an effective angle of solving symbolic regression problems, however, it may give little attention to the original definition of an implicit equation, which is $f(\boldsymbol{x_D}) = 0$. On the contrary, CL-GEP focuses only on the output of $f(\boldsymbol{x_D})$. It can adopt the output as fitness value because it can ensure that all equations are valid and meaningful. Previous works have doubts on the reliability and feasibility of the output, experiments have proved that it works in CL-GEP.

As for another index of the experiment, CL-GEP has gained outstanding performance in time consumption. The gap of time consumption between the two algorithms is apparently huge. In equation *Circle*, the time cost by DB-GEP is 150% of CL-GEP. What's more, the differences of *Elliptic Curve* and *Hyperbola* between the two algorithms are triple and quadruple. Regarding the time consumption cost of the fitness evaluation mechanisms, DB-FEM mostly spends time in calculating partial derivatives, the quotients of derivatives and finding the nearest points, while CL-FEM spends time only when calculating the output of $D+1$ datasets. The results prove effectively that the CL-FEM has evident advantages.

Nevertheless, problems arise when solving the 3-dimension equation *Sphere*. To our surprise, neither of the algorithms can find out the right equation. As for DB-GEP, the DB-FEM performs weakly when handling complex and high-dimension datasets. It is easy to be trapped in local optima, and the low converging speed can be another factor that accounts for the results. We analyse the outputs of CL-GEP, and the outputs are in high similarity. We take one of them for an example

$$f(x,y,z) = (((z*x)*(((z*z)-y)*(y*(x+x))))*((x*(x*x))*(x*y))) \quad (11)$$

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

It can be observed from function (11) that the outputs of CL-GEP are in the multiple multiplication. The sampled data for *Sphere* are ranged in $[-1, +1]$, which determines that the multiple multiplication function can finally produce very small outputs. As all variables are contributing to the problem based on CL-FEM, these wrong functions are regarded as favorable by mistakes.

Experiments on Sparse Datasets The results of the experiments on sparse datasets are shown in Table 3, the numbers are success times in 30 runs. To provide a more direct view, the variational curves are presented in Fig 8.

Table 3. Results of experiments on sparse datasets.

Problem	CL-GEP				DB-GEP			
	data quantity				data quantity			
	500	**100**	**20**	**6**	**500**	**100**	**20**	**6**
Circle	30	30	30	30	30	30	22	0
Hyperbola	30	30	30	30	30	30	29	0
Elliptic Curve	29	28	29	26	20	20	12	0

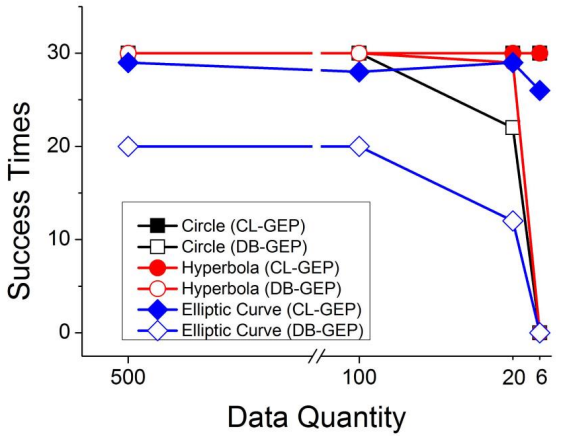

Fig. 8. The curves of the results on sparse datasets.

From the results, we can find that DB-GEP performs weakly in handling implicit symbolic regression problems with sparse datasets. For datasets containing 500, 100 and 20 data, the relationships between different points are still able to be recognized, therefore, it is still possible for it to find out the right solution. When the quantity of data is reduced to 6, however, it fails in all the 90 runs.

From our perspective, the feature of sparse data accounts for the results. On the one hand, if the sparse data are crowded in one specific region, they somehow

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

will be related, however, they can not represent the whole equation model in the space. On the other hand, if the sparse data are separated randomly in the space, the neighboring points can not provide useful information to construct the model, it may sometimes even bring out misleading information (Fig. 2). These two factors seriously restrict the performance of DB-FEM, which makes it incapable of solving sparse data problems.

Take a look at the results provided by CL-GEP, the performance is stable and outstanding. For the two simple equations *Circle* and *Hyperbola*, the success times are all 30 regardless of different datasets. When it comes to the *Elliptic Curve*, the failure times when solving the 6-data set are only 4, the reducing trend is weak.

Compared with the DB-FEM, CL-GEP wins the comparison because it only concentrates on the target data. The information conveyed by a certain point is only the position represented by coordinates, which is precise and authentic. Information collected from the neighboring points may be disturbances when solving sparse data problems, therefore, the CL-FEM has the superiority over the DB-FEM.

5 Conclusions

In this paper, we have proposed a new evaluation mechanism for implicit equation model estimation, named CL-FEM. The proposed CL-FEM improves the previously arisen direct methods of subtracting the function output to zero, by embedding a new mechanism of learning knowledge from *disturbed* data. The core idea of the mechanism is to compare the outputs calculated from stochastic datasets with outputs from the target set. If there are no differences between the two sets of outputs, then the equation will be regarded as invalid. This mechanism can effectively avoid the evolution being trapped in misleading equations, which is the main trouble of solving implicit symbolic regression problems.

We use a simplified SL-GEP to generate a new algorithm with CL-FEM named CL-GEP. To demonstrate the superiority of CL-GEP, we make comparisons with the algorithm named DB-GEP, which embeds the mainstream fitness evaluation mechanism DB-FEM. The proposed CL-GEP not only is more convenient and practical, but also gets more promising results. When solving equations with complex features, the CL-GEP can get a higher success rate. What's more, the $O(ND)$ examination time consumption makes the proposed CL-FEM much faster than the DB-FEM. Most significantly, the proposed algorithm can handle problems with sparse datasets, which can not be tackled by the DB-FEM.

Nevertheless, there exists some weaknesses of the proposed CL-GEP, one of which is the deficient capability of solving problems with datasets consisting of small value data, which means avoiding being trapped in local optima caused by multiple multiplication. There are also some other interesting research directions. One direction is to enhance the capability of finding proper constant by combining other evolutionary algorithms. Another direction is to reduce the complexity of the output to make it more readable and comprehensible. We can

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9

consider incorporating the multiobjective optimization or other techniques to make it more practical in applications.

6 Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61602181), and by the Fundamental Research Funds for the Central Universities (Grant No. 2017ZD053).

References

1. Brameier, M.F., Banzhaf, W.: Linear genetic programming. Springer Science & Business Media (2007)
2. Ferreira, C.: Gene expression programming in problem solving. In: Soft computing and industry, pp. 635–653. Springer (2002)
3. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection, vol. 1. MIT press (1992)
4. Lee, Y.S., Tong, L.I.: Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming. Knowledge-Based Systems **24**(1), 66–72 (2011)
5. McPhee, N.F., Poli, R., Langdon, W.B.: Field guide to genetic programming (2008)
6. Miller, J.F., Thomson, P.: Cartesian genetic programming. In: European Conference on Genetic Programming. pp. 121–132. Springer (2000)
7. ONeil, M., Ryan, C.: Grammatical evolution. In: Grammatical Evolution, pp. 33–47. Springer (2003)
8. Sabar, N.R., Ayob, M., Kendall, G., Qu, R.: Automatic design of a hyper-heuristic framework with gene expression programming for combinatorial optimization problems. IEEE Transactions on Evolutionary Computation **19**(3), 309–325 (2015)
9. Schmidt, M., Lipson, H.: Symbolic regression of implicit equations. In: Genetic Programming Theory and Practice VII, pp. 73–85. Springer (2010)
10. Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization **11**(4), 341–359 (1997)
11. Zhong, J., Cai, W., Lees, M., Luo, L.: Automatic model construction for the behavior of human crowds. Applied Soft Computing **56**, 368–378 (2017)
12. Zhong, J., Feng, L., Ong, Y.S.: Gene expression programming: a survey. IEEE Computational Intelligence Magazine **12**(3), 54–72 (2017)
13. Zhong, J., Luo, L., Cai, W., Lees, M.: Automatic rule identification for agent-based crowd models through gene expression programming. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems. pp. 1125–1132. International Foundation for Autonomous Agents and Multiagent Systems (2014)
14. Zhong, J., Ong, Y.S., Cai, W.: Self-learning gene expression programming. IEEE Transactions on Evolutionary Computation **20**(1), 65–80 (2016)

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93698-7_9