Undergraduate Thesis (Design)

DATA-DRIVEN EVOLUTION ALGOTIRHMS FOR SOLVING NON-LINEAR SYSTEMS OF EQUATIONS

Zhao Chenjun

The English version was translated and generated using PDFMathTranslate

Harbin Institute of Technology May 2024

Undergraduate Thesis (Design)

Undergraduate student: Zhao Chenjun

Student ID: 200810428

Instructor: Zhang Xinming

Major: Data Science and Big Data Technology

College: School of Science, Shenzhen Campus

Defense date: May 2024

School: Harbin Institute of Technology

Abstract

The solution of nonlinear systems of equations is a significant research topic in both scientific and engineering domains. This paper addresses the scenario of solving nonlinear systems of equations by introducing data-driven optimization algorithms. Through adjustments and improvements in model structure and data generation algorithms, the RDDEA-ZPS algorithm is proposed, demonstrating excellent accuracy and performance across various test cases.

Initially, an analysis of data-driven optimization algorithms, particularly the WDDEA-DBC algorithm, is conducted, highlighting drawbacks in data generation, proxy model, and optimization processes, elucidating their limitations in solving nonlinear systems of equations.

Subsequently, in response to the limitations of the WDDEA-DBC model, methods such as the ZPS data augmentation algorithm, the inverted RBFNN proxy model framework, and the modified FPA optimization process are proposed to address several error introduction issues of the original algorithm. This culminates in the development of the inverse operator zero-point sampling data-driven optimization algorithm, namely the RDDEA-ZPS model.

For the new model and processes, numerical experiments are conducted on various test cases to search for optimal parameters. By visualizing results through curve plots, box plots, and heatmaps, and by calculating metrics such as mean, variance, and confidence intervals, empirical support is provided for parameter selection.

Finally, employing both algorithms, WDDEA-DBC and RDDEA-ZPS, to solve the same nonlinear system of equations, the accuracy and stability of the RDDEA-ZPS algorithm in solving nonlinear systems of equations are intuitively validated by comparing the Euclidean distance between true and numerical solutions and calculating the mean and variance of errors, thereby proving the effectiveness of the improved model framework.

Keywords: data-driven evolution algorithm, swarm intelligence algorithms, nonlinear systems of equations, radial basis function network, data augmentation algorithm

Chapter 1 Introduction

1.1 Background of the topic

1.1.1 Nonlinear equations

The solution of nonlinear equations has always been an important research topic in mathematics and engineering. In practical problems, the behavior of many systems is often described by nonlinear equations. For example, in engineering fields, such as control system design, signal processing, and optimization problems, the efficient solution of nonlinear equations has important application value. However, these systems of equations are often difficult to solve directly to obtain accurate analytical solutions. Therefore, it has become an urgent need to find methods to obtain solutions to nonlinear equations with low computational cost and high solution accuracy. Therefore, the development of solving algorithms that can effectively handle nonlinear systems of equations in practice is of great significance to improve the performance and efficiency of engineering systems.

Specifically, the structure of a general system of nonlinear equations is shown in Equation (1-1).

$$\begin{cases} f_1(x_1, x_2, \cdots, x_n) = 0 \\ f_2(x_1, x_2, \cdots, x_n) = 0 \\ \cdots \\ f_n(x_1, x_2, \cdots, x_n) = 0 \end{cases} \tag{1-1}$$

The number of independent variables is equal to the number of equations, and for different input vectors $(x_1, x_2, \ldots, x_n)$, each equation in the system of equations corresponds to a value. Therefore, the optimization process of the system of equations That is By finding the best vector $\left(x_1^*, x_2^*, \ldots, x_n^*\right)$ so that these independent variable components are input, the value obtained on the right side of each equation is closer to 0. The subsequent model building process is also based on this premise.

1.1.2 Methods for solving nonlinear equations

The solution of nonlinear systems of equations is of great significance in the fields of mathematics and engineering. However, for complex systems of equations, their analytical solution is often difficult or infeasible. Therefore, researchers turn to numerical methods to approximate the solution of the system of equations.

Iterative methods are a common method for solving systems of nonlinear equations. Among them, Newton's method uses the first-order and second-order derivative information of the function to iterate, and approximates the solution of the system of equations by continuously improving the current solution. However, for complex functions or high-dimensional problems, Newton's method may be slow to converge and require extensive calculations. Quasi-Newton's law reduces calculations by approximating the Hessian matrix or its inverse instead of the exact calculations in Newton's method

quantity, but performance degradation may occur in high-dimensional problems.

In addition to traditional methods, different optimization methods can be used to so lve systems of nonlinear equations. For example, the gradient descent method uses the n egative gradient direction to search to minimize the objective function to solve a system of nonlinear equations. Intelligent optimization method [1] is also a method for solving n onlinear systems of equations. It is based on heuristic search and optimization principles . This type of algorithm usually does not rely on the analytical form of the system of equ ations, but searches the entire solution space to find the optimal solution or approximate solution. However, this type of method often has problems such as uncertain convergenc e and weak local search capabilities.

Therefore, it is of great significance to propose a solution method with small time cost and high solution accuracy for solving nonlinear equations problems.

1.1.3 WDDEA-DBC algorithm

Data-driven optimization algorithm [2] (Data-Driven Evolutionary Algorithm, DDE A) is a method that uses existing data in the problem domain to guide the optimization p rocess. This type of method often uses different data generation methods to generate mo re and more accurate data to improve optimization efficiency. In the optimization proces s, the DDEA algorithm often uses the swarm intelligence evolutionary algorithm for opti mization.

The DDEA algorithm can effectively improve the efficiency and accuracy of prob lem solving by collecting and analyzing input and output data to search and optimize th e solution to the problem in a more intelligent and efficient way.

Among them, the WDDEA-DBC[3] model is an excellent DDEA algorithm. It cons ists of steps such as data generation, agent model training, agent model management and aggregation, and optimization algorithm optimization. First, it uses the clustering metho d to detect the categories that need to generate data and generate data. Then it uses the ra dial basis function network as the agent model, uses a unique and efficient model manag ement framework to aggregate several agent models, and finally uses intelligent optimiz ation algorithms to optimize the aggregate agent model.

By converting complex optimization problems into agent model training, the WDD EA-DBC algorithm further simplifies the problem-solving scenarios and greatly improv es the efficiency of problem-solving. This further inspires us to apply the DDEA algorit hm to the solution of nonlinear equations, and is expected to improve the accuracy and e fficiency of solving equations.

However, limited by the proxy model structure of the WDDEA-DBC model and the application scenarios of the optimization algorithm, it introduces several errors into the s olution steps of the nonlinear equations, resulting in insufficient solution accuracy. There fore, how to reasonably improve this model to make it more suitable for solving multi-di mensional optimization problems such as nonlinear equations is an issue worth exploring .

1.2 Research purpose and significance

Limited by the complex form of nonlinear equations, traditional solving methods often face many challenges, such as

Problems such as high sensitivity to initial conditions, easy falling into local optimal sol utions, high computational complexity, and numerical instability limit its effectiveness i n practical applications. In view of this, this project aims to combine data-driven swarm intelligence algorithms with the solution of nonlinear equations, in order to improve the fit between the algorithm and the problem, so as to obtain solutions to the equations mo re efficiently and accurately.

At present, traditional optimization algorithms such as Newton's method and bisec tion method show extremely high sensitivity in the selection of initial conditions, and it is often difficult to achieve a balance between solution accuracy and computational cost . At the same time, traditional intelligent optimization algorithms such as the flower pol lination algorithm are easily limited by local optimal solutions, and their convergence s peed is slow. To cope with this challenge, some scholars have proposed data-driven opti mization algorithms, which use historical data to build surrogate models and use evoluti onary algorithms to solve optimization problems to achieve a more intuitive, direct and fast nonlinear equation solving process.

This project plans to introduce the data-driven optimization algorithm into new pro blem scenarios based on the DDEA algorithm, especially the WDDEA-DBC model, and conduct in-depth exploration of methods for solving nonlinear equations. Research direc tions and purposes include improving data generation methods, transforming network st ructures, expanding application scenarios of intelligent optimization algorithms, etc., to reduce errors introduced by learning norms and optimize the training process. Through t hese efforts, a model with high accuracy for solving nonlinear equations can be establish ed.

1.3 Development status at home and abroad

The solution of nonlinear equations has always been a key research direction in the f ield of numerical computing and optimization. Classic numerical methods, such as Newt on's method, quasi-Newton's method and conjugate gradient method, have attracted muc h attention due to their effectiveness and wide application in solving nonlinear problems . In addition, global optimization algorithms such as genetic algorithms and simulated an nealing algorithms have also achieved certain results in solving global optimal problems of nonlinear equations. In recent years, with the rapid development of deep learning tech nology, its application in solving nonlinear equations has gradually attracted attention. F or example, using deep learning models such as neural networks to approximate the solu tion of nonlinear functions has become one of the effective methods to solve complex no nlinear system problems.

Data-driven optimization algorithms are a research direction that has attracted muc h attention in recent years. Its core lies in fully mining the information behind the data a nd using the data generated during the optimization process to generate data, establish a gent models, model aggregation, and optimize intelligent algorithms. The academic co mmunity is exploring ways to combine data-driven ideas with traditional optimization a lgorithms, and has proposed a series of innovative research results. In terms of applicati ons, data-driven algorithms have shown broad application prospects in fields such as bi g data analysis and intelligent manufacturing.

Although certain progress has been made in the fields of numerical computing and optimization, applying data-driven optimization algorithms to the solution of nonlinear equations is still an area that needs to be explored in depth.

1.3.1 Current status of research on solving nonlinear equations

The solution of a set of nonlinear equations is a key issue in the field of mathematics and computational mathematics. Its goal is to find the values of a set of variables so that a set of nonlinear equations are satisfied. In order to solve this problem, researchers have proposed two main categories of methods, namely traditional methods and intelligent optimization algorithms.

Traditional methods include Newton's method, quasi-Newton's method, and steepest descent method. These are usually iterative methods [4,5], starting from an initial guess and updating the values of the variables iteratively until the conditions of the system of equations are met. The traditional method is simple and easy to implement, and has been widely used in numerical calculation and other fields.

In recent years, Intelligent Optimization Algorithms (IOAs) have shown significant superiority in solving nonlinear equations [1]. Especially for nonlinear equations with multiple local optimal solutions, intelligent optimization algorithms, such as genetic algorithm [6], flower pollination algorithm [7], simulated annealing algorithm, etc., are widely used in the task of finding the global optimal solution. Compared with traditional iterative methods, these algorithms have stronger global search capabilities, can effectively avoid falling into local optimal solutions, and have a certain degree of adaptability, which can achieve better results when solving complex problems. Its efficiency and reliability in the field of solving nonlinear equations provide powerful tools and methods for solving complex problems.

1.3.2 Research status of solving nonlinear equations based on neural network

With the development of neural networks, the solution of nonlinear equations has become an important scenario for neural network applications. In order to better combine the problem of solving nonlinear equations with data-driven optimization algorithms, this topic is dedicated to exploring neural network solutions to nonlinear equations, so as to facilitate the subsequent establishment of more efficient numerical models. By combining neural network technology with optimization algorithms, we aim to develop a more accurate, robust and efficient method for solving nonlinear equations, providing more possibilities for research and applications in related fields.

In 1997, Li Yinghui et al. proposed a basic architecture [8] for solving nonlinear equations using artificial neural networks. Based on the basic feedforward neural network, for the nonlinear equations $f(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), \dots, f_n(\boldsymbol{x})\right)^T = 0$, the backpropagation (BP) algorithm was used to construct a neural network with vector $f(\boldsymbol{x})$ as input and independent variable vector $\boldsymbol{x}$ as output. By generating several independent variable $\boldsymbol{x}$ data within the solution interval, calculating the value corresponding to the dependent variable vector $f(\boldsymbol{x})$, inverting the two sets of variables, and using them as training data for training, good results were obtained on some nonlinear equations, laying the foundation for the neural network method to solve nonlinear equations.

Since then, scholars such as Zhao Qilin proposed a new neural network architecture [9] in 2000. This architecture is based on a coupled neural network and provides a new network structure for solving nonlinear equations by correlating the weights of the network. However, this method has certain requirements for the form of nonlinear equations and is not suitable for all types of nonlinear equations.

In 2002, scholars such as Zhao Huamin proposed a unique neural network model, which contains the differential form [10], and deduced through rigorous mathematical proof that the network can iterate from any starting point and finally converge to a solution to the nonlinear equation system. This discovery provides new ideas and methods for the neural network method to find a wider range of solutions. Similarly, G. Li and other scholars proposed another neural network model [11], which also uses iterative calculation to optimize the solution. Different from the previous method, this model eliminates complex differential and integral operations, thereby effectively improving the solution efficiency and reducing computational overhead.

Based on the basic neural network architecture for solving nonlinear equations proposed by Li Yinghui in 1997, Sun Yinhui et al. proposed an iterative neural network algorithm [12] for solving nonlinear equations in 2009. This algorithm is based on the basic neural network architecture, establishes an explicit neural network structure, and solves it through an iterative method. During the training process, based on the training results of gradually expanding the search range of the existence interval of the solution, the algorithm is able to find the true solution of the system of equations outside the given initial interval.

Based on this method, in 2013, Zhao Qinglan and other scholars proposed an improved iterative neural network algorithm to solve nonlinear equations [13]. In order to solve the problems of infinite iteration that may occur in the aforementioned algorithm, this algorithm proposes a method of memorizing the minimum error point and limiting infinite iteration, thus significantly improving the effect of the algorithm.

1.3.3 Research status of data-driven optimization algorithm (DDEA)

The research team of Yaochu Jin et al. established the framework of online DDEA and offline DDEA and clarified the general process of the DDEA model [2]. Based on this novel optimization algorithm, they conducted in-depth research on difficult optimization problems such as multi-objective optimization and expensive optimization, and used their research results to successfully solve many practical application problems and achieve remarkable results.

In 2019, Wang Handing et al. introduced the Bagging management strategy [14] of the proxy model for the first time based on the DDEA framework. After processing the noise of the data set, they generated new high-quality data through Bootstrap sampling. Afterwards, they used the integrated learning method in machine learning to correct the errors of the proxy model, further improving the accuracy of the optimization algorithm.

In 2020, scholars such as Li Jianyu introduced the new Boosting integrated learning method to the application of DDEA and established the BDDEA-LDG model [15]. This model generates new data where the data fitting effect is not good, and then uses the Boosting method, which is faster than Bagging, to fit the surrogate model.

Guo Zongliang and others innovatively proposed the use of a data generation method based on clustering information, using $\frac{1}{MSE}$ as the weight information of the RBFNN integrated boosting framework, and using the flower pollination algorithm for solution. This method is referred to as WDDEA-DBC[3]. This model shows strong performance in the process of optimizing several functions, but there is still room for improvement in solving this optimization problem of nonlinear equations. Therefore, this is also the main Baseline model of this article. It is expected that this model will be improved to achieve better solution results.

1.4 The main research content of this article

1.4.1 Research on data generation methods

The quality of data has a direct impact on the generalization ability and performan ce of the model. A uniform data set can ensure that the model has enough representativ e samples in different categories or features, thereby avoiding model bias caused by dat a skew. For the solution of nonlinear equations, it is also necessary to generate evenly distributed pairs of independent variables and dependent variables. Such data sets are c rucial for subsequent network searches to solve the system of equations because they c an better reflect the characteristics of the system of equations, thereby improving the ac curacy and reliability of the solution process. Therefore, ensuring the uniformity and re presentativeness of the data set is a key step that cannot be ignored in solving nonlinear equations.

Since the purpose of this study is to solve a system of nonlinear equations, the model needs to focus on learning the correspondence between the data distribution and the inde pendent variables and dependent variables close to the solution. How to generate unifor m data that can cover the solution interval and at the same time enhance the generation o f key parts of the data is also one of the important research goals of this topic.

1.4.2 Construction of the neural network architecture of the agent model

For the DDEA algorithm, the choice of surrogate model is crucial. In the WDDEA-DBC model, RBFNN is selected as the surrogate model to fit the norm of the nonlinear equation system and optimize the model through the optimization algorithm. However, f or this architecture, the fitting of the norm further introduces errors, which may lead to a decrease in the quality of the data.

Therefore, this paper needs to re-adjust the structure of the surrogate model and fin d a surrogate model suitable for solving problems of nonlinear equations. This step is cr ucial because the choice of surrogate model directly affects the performance of the mod el and the accuracy of the solution results. By carefully designing and adjusting the arch itecture of the surrogate model, the algorithm can better adapt to the characteristics of th e nonlinear equation system, thereby improving the model's fitting ability and solution e ffect.

1.4.3 Establishment of optimization problem

In the original WDDEA-DBC, data augmentation is used to enhance the data set, a nd the original RBFNN model is optimized to obtain solutions to the nonlinear equation s. However, this step introduces errors and is equivalent to directly using intelligent opti mization algorithms to solve nonlinear equations to a certain extent, and its effect needs to be further improved.

Therefore, after improving the agent model, the application method of the optimiza tion algorithm also needs to be adjusted to achieve better optimization results. By redesi gning and adjusting the optimization algorithm, we can make more effective use of the improved surrogate model and further improve the accuracy and efficiency of solving n onlinear equations.

1.4.4 Solution of nonlinear equations

After the model is constructed, the solution to the system of nonlinear equations is finally obtained. This requires structural integration of the model, data processing, and finally using the model to solve the solution to the system of equations, that is, directly obtaining the solution $\boldsymbol{x}$ such that $f(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\right)^T = 0$ is obtained based on the model.

Chapter 2 Analysis Application of WDDEA-DBC

2.1 Introduction to WDDEA-DBC algorithm

2.1.1 Model structure

As the baseline of the improved model, the WDDEA-DBC model has many innovations in its data processing methods and structure, and can be applied to the solution of nonlinear equations.

In order to successfully extend the DDEA algorithm to the scenario of solving nonlinear equations, analyze the advantages and disadvantages of the model in specific scenarios, and propose improvement plans, this chapter will briefly explain the process of the Baseline model, analyze its limitations in this scenario, and provide comparison and basis for the establishment of subsequent improved models.

The WDDEA-DBC model is mainly divided into several parts: data sampling, DBC clustering to expand data, training and aggregation of proxy models, and optimization using flower pollination algorithms. Specifically, the structural process of the WDDEA-DBC model is shown in Figure 2-1.

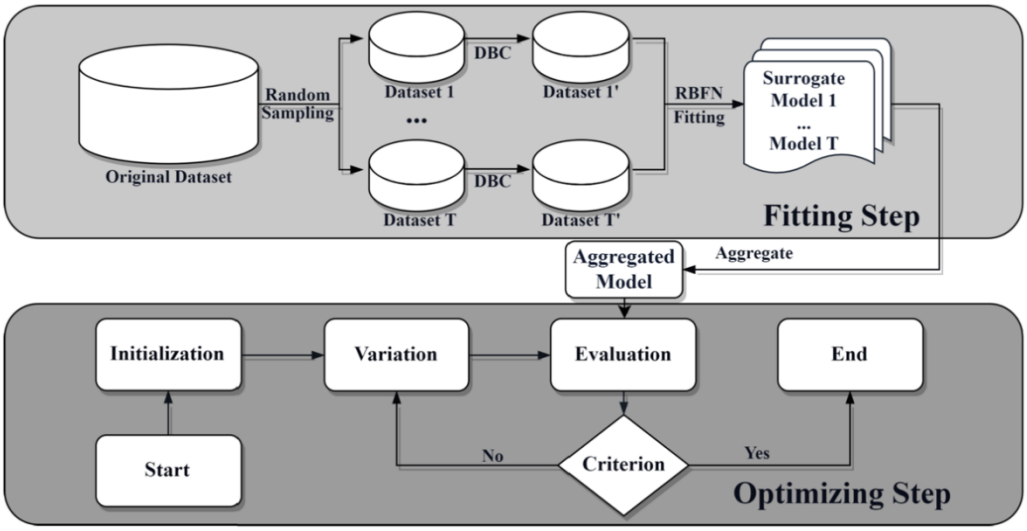


Figure 2-1 WDDEA-DBC model structure [3]

At the beginning of the algorithm, Latin Hypercube Sampling is first used to generate data that is relatively evenly distributed in the data definition domain. Afterwards, the data set will be expanded through the data generation [3] method (Data Generation Based on Clustering, DBC) based on clustering information to further evenly distribute the data and reasonably expand the data set. Further, we sample and train several surrogate models on the expanded data set. The surrogate model here chooses the radial basis function network (Radial basis function neural network).

network, RBFNN), each independent agent model is aggregated using a management fra mework with MSE (Mean Squared Error) weights. Finally, the Flower Pollination Algor ithm (FPA) is used to optimize the aggregated model to achieve optimization of the orig inal problem.

2.1.2 The innovation of the algorithm

The innovation of the WDDEA-DBC model is mainly that it creatively proposes an algorithm for DBC to generate data, which can enhance the quality of data in weak areas in the data space. It is especially effective in expanding the data set in scenarios where d ata is difficult to obtain. In addition, the MSE reciprocal weighting method cleverly redu ces the prediction weight of models with larger errors (MSE Loss), thereby further erodi ng the impact of large error results on the final problem solution.

Since this algorithm has a wide range of application scenarios and is a single-obje ctive optimization algorithm, applying it to the solution of nonlinear equations require s model adjustment and reasonable processing of components.

2.2 Application and expansion in nonlinear equation solving scenarios

2.2.1 Data sampling

The solution of nonlinear equations is a very important type of optimization proble m. How to reasonably adjust the algorithm so that it can be applied to specific scenarios is a type of problem that needs to be studied.

The first is the data sampling part of the WDDEA-DBC algorithm. The purpose of s ampling is to generate sufficient and uniform data sets to provide high-quality data supp ort for subsequent agent model training. In the solution scenario of nonlinear equations, the sampling space is the independent variable $\boldsymbol{x}$ space of the nonlinear equations. By pe rforming Latin hypercube sampling on the domain space, an initial independent variable data set is generated, and the dependent variable data set $\boldsymbol{y}$ corresponding to the indepen dent variable data set is obtained through the nonlinear equations. The DBC algorithm is then used on this data set to generate new data where the distribution is sparse. Samplin g several times as above will yield several groups of generated data sets, which will be u sed for subsequent training of several proxy models.

Specifically, the structure of the nonlinear system of equations is shown in the form ula (1-1) in Chapter 1. First, the $\boldsymbol{x}$ space is sampled through the Latin hypercube, and th en the corresponding right-hand side value y_i is calculated through each equation of (1-1), which is the value of the ith equation, and is assembled into the vector $\boldsymbol{y} = (y_1, y_2, \dots, y_n) = \big(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_n(\boldsymbol{x})\big)$, which is used as the data set of the over all algorithm.

Since this algorithm is a single-objective optimization algorithm, in subsequent optimi zation, the value of a single variable can only be optimized by searching the multi-dimensi onal $\boldsymbol{x}$ space. Therefore, the $\boldsymbol{y}$ vector on the right is normed, that is, $F_1(\boldsymbol{x}) = \|f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_n(\boldsymbol{x})\|_1$, and the norm value is used as fitness for optimization. The type of norm is determined by the original algorithm on the test function. The process of minimizing the norm is equivalent to minimizing the value of the right-hand side of the equation of the system of equations. thus generating data

When , the $\boldsymbol{x}$ vector space is sampled, and the sampled $\boldsymbol{x}$ is substituted into the system o f equations to obtain the norm of the $\boldsymbol{y}$ vector. The format of the sampled data points can be denoted as $\{(\boldsymbol{x}, \boldsymbol{F}_1(\boldsymbol{x}))\}$.

2.2.2 Establishment and optimization of agent model

In the proxy model establishment part, RBFNN is used as the proxy model, $\boldsymbol{x}$ is the input of the network, $\boldsymbol{F}_1(\boldsymbol{x})$ is the output of the network, and the implicit correspondence relationship of $\boldsymbol{x} \rightarrow \boldsymbol{F}_1(\boldsymbol{x})$ is obtained through training. The optimization algorithm of th e network adopts the common Adam method.

In the model aggregation part, after training several groups of models, the algorith m obtains the mean square error (MSE) value of each independent model. Then, the alg orithm uses the reciprocal of MSE as a weight to weight the model to weaken the large error model, enhance the influence of the prediction results of the small error model, an d obtain a more accurate function correspondence. The final model will use these weig hts to combine the performance of individual models to improve overall prediction acc uracy.

After the model construction is completed, the algorithm uses the Flower Pollination Algorithm (FPA) for optimization. Specifically, the function is passed into the FPA algor ithm, and a better solution $\boldsymbol{x}$ is found through global or local search, natural mutation, Le vy flying and other methods, so that $\boldsymbol{F}_1(\boldsymbol{x})$ becomes closer and closer to the minimum val ue 0, that is, the solution gradually approaches the solution $\boldsymbol{x}^*$ of the nonlinear equation s ystem. After the operation of the above model program, the approximate solution $\boldsymbol{x}^*$ of t he nonlinear equations can finally be obtained.

In order to maintain the coherence of the description of the improved model, the proc ess here is only briefly described. Some of the methods used in the WDDEA-DBC algorit hm will be elaborated and noted in the improved algorithm later.

2.3 Disadvantages and limitations of WDDEA-DBC

The original algorithm has excellent results in several optimization scenarios, but i f it needs to be applied to nonlinear equations, it will have several limitations.

First, the training process of the surrogate model belongs to the step of approximati ng the mapping relationship between independent variables and dependent variables of the system of equations. Compared to directly using the correspondence relationship fro m left to right (i.e., the norm of $\boldsymbol{x}$ to $\boldsymbol{y}$) of the original system of equations as the optimi zation function of the FPA algorithm, the introduction of the neural network causes the original precise mapping relationship to become fuzzy and error-prone, which is the firs t step of error introduced.

Secondly, the algorithm is limited to single-objective optimization scenarios and in troduces norms as the output of the neural network. The introduction of norm makes th e neural network unable to specifically learn the correspondence between independent variables and dependent variables. This conversion causes some information to be swal lowed up and introduces a second step error.

Third, for the final generated model, the algorithm uses the FPA algorithm for opti mization. The weighted integrated model is already an approximation of the correspond ing relationship of the original set of equations. Based on the error in this step, an intelli gent optimization algorithm is applied, and the third step optimization error is further in troduced.

In addition, theoretically speaking, the data of the nonlinear equation system can be obtained endlessly, and the DBC data generation algorithm works best when the data source is lacking. Therefore, in this scenario, the data generation algorithm seems to be of little use. So how to generate data more reasonably to generate more and higher-quality data support is also an important aspect to consider when improving the model.

Based on the above issues, subsequent model improvements and experimental designs will be carried out to further improve the adaptability and accuracy of the DDEA algorithm in solving nonlinear equations.

2.4 Summary of this chapter

To sum up, this chapter mainly introduces the application of the WDDEA-DBC model as the baseline of the improved model in solving nonlinear equations, and conducts a detailed analysis and discussion of it. By analyzing the DDEA algorithm represented by WDDEA-DBC, it can be clearly seen that the data processing methods and structures of this type of model have many innovations and can be applied to the solution of nonlinear equations.

Although the WDDEA-DBC model shows good results in several optimization scenarios, it has some limitations when applied to solving nonlinear systems of equations. Mainly including agent model training errors, single-objective optimization scenario restrictions, errors introduced by intelligent optimization algorithms, etc. In addition, the DBC data generation algorithm has limited effect when the data source is sufficient, and more effective data generation methods are needed to support the application and improvement of the model.

Thereafter, starting from Chapter 3, the improved model construction process will be described. From data generation to final results, they are all components of the improved model, including the extension of the WDDEA-DBC model and the sequential modification of its shortcomings.

Chapter 3 RDDEA-ZPS model establishment

3.1 Establishment of agent model

3.1.1 Agency model of WDDEA-DBC algorithm

For the DDEA algorithm, the selection of the surrogate model is crucial. In the WD DEA-DBC model, RBFNN[16] is selected as the surrogate model to fit the norm of the n onlinear equation system, and the optimization algorithm is used to optimize the model. The structure of the agent model used in the original algorithm is shown in Figure 3-1.

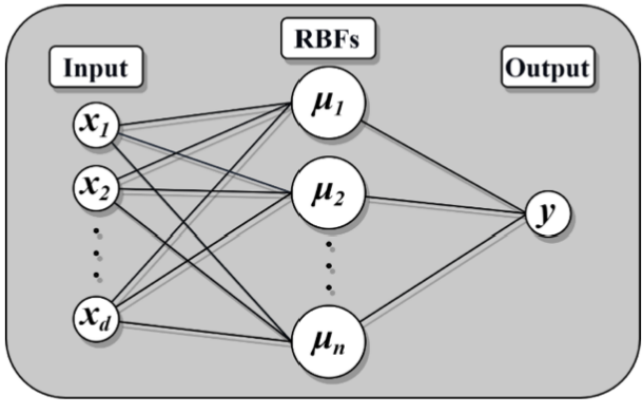


Figure 3-1 Agency model structure of WDDEA-DBC algorithm [3]

For this architecture, as mentioned in the previous chapter, since the original model is solved using the Flower Pollination Algorithm (FPA), this optimization algorithm is a single-objective optimization algorithm, and the output is the norm of $\boldsymbol{y}$. However, such an algorithm has major limitations.

First of all, the introduction of norms makes it impossible to specifically learn the c orrespondence between independent variables and dependent variables. This conversio n causes some information to be swallowed up and introduces the first step error. Secon dly, for the final generated function, the FPA algorithm is used for optimization. The fu nction is already an approximation of the original system of equations. Based on the err or in this step, an intelligent optimization algorithm is applied, which further introduces errors. Therefore, the final result is not very ideal.

The surrogate model is the main structure of the overall model, and reasonable mod ifications to the surrogate model can greatly improve the accuracy of the overall model. Therefore, consider modifying the agent model framework in the original algorithm to i mprove the solution accuracy.

3.1.2 Modified proxy model

In response to the above problems, this project improves the structure of the surroga te model, from fitting the norm of the system of nonlinear equations to fitting the invers e operator of the system of equations, that is, the correspondence between $\boldsymbol{y}$ and $\boldsymbol{x}$.

In order to facilitate comparison and control variables, the new model still uses RB FNN as the basic structure of the surrogate model. The modified RBFNN structure is sh own in Figure 3-2.

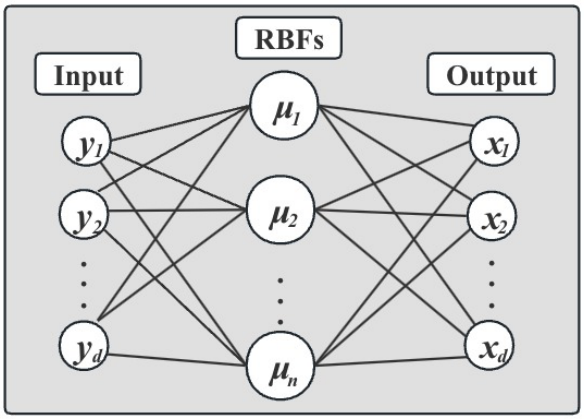


Figure 3-2 Structure of the proxy model in the improved model

In the structure diagram, the left side is the various categories of the dependent vari able vector, that is, the value on the right side of each equation in the nonlinear equation system, and the right side is the value of the independent variable vector $\boldsymbol{x}$ correspondin g to the dependent variable. In addition to the input and output layers of this RBFNN, th e middle layer is the radial base layer. The calculation formula from the input layer to th e radial base layer is shown in formula (3-1):

$$\mu_i = \sum_{j=1}^{d} w_{ji}\,\phi\left(\left\|\boldsymbol{y} - \boldsymbol{c_j}\right\|\right) \tag{3-1}$$

Among them, μ_i is the calculation result of the radial basis function and is also the a ctivation value of the intermediate neuron; w_{ji} is the weight from the input layer neuron to the intermediate layer neuron, which is often recorded as 1 in use; $\boldsymbol{y}$ is the input vecto r, and $\boldsymbol{c_i}$ is the clustering center of the ith neuron. In order to facilitate the comparison of effects, the radial basis function used in this project is consistent with the original algorit hm. It is a Gaussian function, and the expression is given by (3-2):

$$\phi(x) = exp\left(\frac{x^2}{-2\sigma^2}\right) \tag{3-2}$$

In the specific calculation, x in the formula is replaced by the distance between the i nput vector $\boldsymbol{y}$ and the center point $\boldsymbol{c_i}$ of each neuron. σ is a parameter that controls the w idth, determined by the intermediate results in subsequent data generation steps.

Compared with the original agent model, two changes have been made here. The fi rst is to restore the norm in the data set to a vector, so that the mapping relationship bet ween independent variables and dependent variables can be learned more accurately an d the problem of introducing norms can be eliminated.

Information loss caused by one step. The second is to reverse the input and output stat us of independent variables and dependent variables in the neural network. Here, the d ependent variable $\boldsymbol{y}$ vector is used as input and the independent variable $\boldsymbol{x}$ is used as o utput to train the network.

After such a transformation, the mapping relationship from vector $\boldsymbol{y}$ to $\boldsymbol{x}$ can be obtai ned, which greatly simplifies the solution step while eliminating the norm step error. Sinc e the original model needs to use RBFNN approximation and FPA optimization to solve t he nonlinear system of equations, both steps introduce errors. The error produced by the modified proxy model is only the error that may be introduced by the fitting neural netwo rk. When solving, there is no need to use an intelligent optimization algorithm to search f or the optimal solution, but only the *d*-dimensional zero vector $\boldsymbol{y} = (0,0, \dots ,0)$ needs to b e input to the trained proxy model, and the output solution $\boldsymbol{x}^*$ vector is the algorithm solut ion result.

3.2 Data generation

3.2.1 Latin hypercube sampling

Building and using high-quality data sets is critical to improving model algorithm performance. For the special scenario of solving nonlinear equations, a uniform data se t can ensure that the training set contains and locates the interval where the solution is l ocated. In addition, focusing on generating data sets near the solution also plays a great role in improving the model's optimization accuracy around the solution. Therefore, in this scenario, how to generate a uniform and focused data set is crucial to the establish ment of the model.

Drawing on the data generation method of the WDDEA-DBC model, Latin hypercu be sampling is first performed on the domain space to ensure the uniformity and breadth of the data.

Compared with traditional random sampling methods, Latin hypercube sampling can make more effective use of parameter space, avoid correlation between samples, and thus better explore the global characteristics of parameter space.

In Latin hypercube sampling, the value range of each parameter is first divided into equal sub-intervals, and then a sample point is randomly selected in each sub-interval. This design ensures that the sample points are evenly distributed and independent of ea ch other.

In the application scenario of solving a system of nonlinear equations, the model will sample each dimension of the independent variable $x = (x_1, x_2, \dots , x_d)$, and the mark nu mber here is changed to *d* to facilitate the understanding of subsequent variable settings. The algorithm flow is as follows.

Algorithm 3-1 Latin Hypercube Sampling

Input:

Dimension d; number of sampling points N_{data}

Output:

Sampled dataset $\mathbf{X}$

Begin

For $j = 1$ to d Do

Divide the j-th dimension evenly into N_{data} intervals.

For $i = 1$ to N_{data} Do

Generate a random number within the corresponding interval, denoted as N_{ij}.

Add this random number to the sampling matrix.

End For

End For

End

The final sampling result is a matrix, as shown in equation (3-1):

$$\mathbf{X} = \begin{pmatrix} X_{11} & X_{21} & \cdots & X_{N1} \\ X_{12} & X_{22} & \cdots & X_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ X_{1d} & X_{2d} & \cdots & X_{Nd} \end{pmatrix} \tag{3-3}$$

Each column is a data point $\boldsymbol{x}$, and each row is the same dimension of different data points. Such a method

It proves that uniform and representative sampling is carried out in each dimension, so that the sampling results can better reflect the characteristics of the parameter space. Afterwards, the corresponding $\boldsymbol{y}$ vector is calculated based on the $\boldsymbol{x}$ data points of each column above, and several data pairs are obtained as the preliminary data set $\{(\boldsymbol{x}, \boldsymbol{y})\} = \{(\boldsymbol{x}, F_1(\boldsymbol{x}))\}$.

3.2.2 DBC sampling

After generating the Latin hypercube sampling data set, also inspired by the WDDEA-DBC algorithm, the DBC algorithm will be further used to further generate the data. The DBC algorithm is a data generation method based on clustering information. The purpose is to expand the size of the data set. The specific way is to generate synthetic data in places where the data distribution in the data set is sparse to improve the training effect.

Specifically, the original data set after Latin hypercube sampling can be expressed as $\boldsymbol{TD} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) | i = 1,2, \ldots, N\}$.

Since the responsibility of DBC is to generate new synthetic data based on the original data of TD, the first task of the DBC algorithm process is to determine the area scope of data generation. This process involves calling the K-Means algorithm for clustering and obtaining information about clusters and their center points.

Since in the improved model, the input becomes $\boldsymbol{y}$, $\boldsymbol{y}$ needs to be expanded so that the model can learn sufficient distribution of $\boldsymbol{y}$ to facilitate further optimization. For each cluster, sort according to its compactness (CP). Among them, the calculation method of tightness is:

$$CP_j = \sum_{i=1}^{n_j} \frac{\|\boldsymbol{y_i} - \boldsymbol{c_j}\|}{n_j}, j = 1,2,\dots,m \tag{3-4}$$

Where n_j is the number of data points in the jth cluster, $\boldsymbol{c_j}$ is the center point of the jth cluster. Afterwards, data will be generated for the m clusters with the largest CP values. The specific algorithm for data generation will be given in subsequent chapters.

However, since the problem currently being solved is a system of nonlinear equations, directly expanding $\boldsymbol{y}$ after switching the input and output of the surrogate model will result in the inability to calculate the corresponding independent variable $\boldsymbol{x}$. Therefore, by finding $\boldsymbol{x}$ corresponding to $\boldsymbol{y}$, $\boldsymbol{x}$ is generated, and then the corresponding $\boldsymbol{y}$ is calculated from the new $\boldsymbol{x}$.

The flow of the DBC algorithm is illustrated in Figure 3-3.

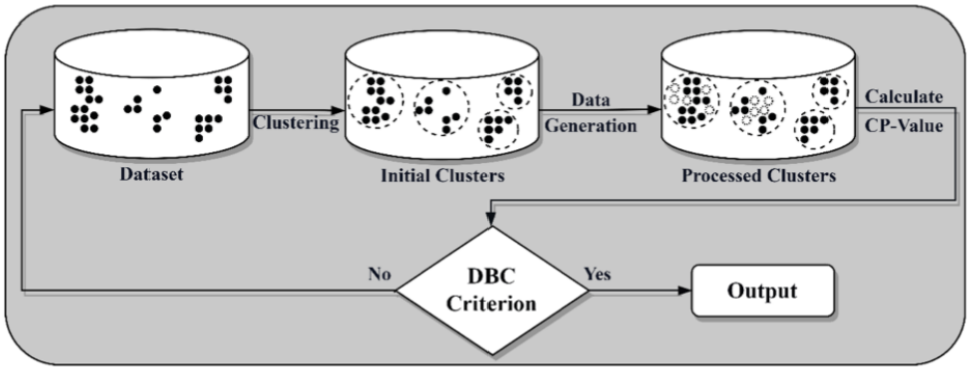


Figure 3-3 DBC algorithm process[3]

That is, perform K-Means clustering on the original data set, calculate clusters with low CP values, generate data for them, and then continue to detect the CP values of each cluster until convergence or stopping conditions are reached, and the DBC algorithm stops.

3.2.3 Zero point sampling

When data sources are limited, the DBC algorithm can generate data similar to the original data, improving the quantity and quality of data. However, in the application scenario of solving a system of nonlinear equations, every time an independent variable $\boldsymbol{x}$ is sampled and generated, the corresponding $\boldsymbol{y}$ can be obtained. This allows the model to obtain a steady stream of data, and the advantages of DBC are no longer obvious. Therefore, we need a data generation method that is more efficient and conducive to model optimization.

Since the goal of solving a system of nonlinear equations is to find a combination of $\boldsymbol{x}$ that makes each equation equal to zero, the data points where $\boldsymbol{y}$ is close to 0 are of mos t research value, and the closeness of $\boldsymbol{y}$ to 0 is measured by the norm. Additional samplin g of this type of data is called zero-point sampling (ZPS).

The specific process is as follows. For the data set after DBC sampling, calculate the $\|\boldsymbol{y}\|$ of the points in the data set and sort them, select the data point at m with the smallest $\|\boldsymbol{y}\|$, and generate surrounding data for its corresponding $\boldsymbol{x}$. Specifically, for the selected part of the data set $\boldsymbol{X}_m$, the same data generation method as the DBC algorithm is used, th at is, for each point x_i in $\boldsymbol{X}_m$, a new data point $\boldsymbol{x_i} + \boldsymbol{\delta_i}$ is generated in its neighborhood. T he distribution and parameters of $\boldsymbol{\delta_i}$ are given by (3-5) and (3-6):

$$\boldsymbol{\delta_i} \sim N_d(\boldsymbol{0_d}, \boldsymbol{L_d}), \boldsymbol{L_d} = l \cdot \boldsymbol{E_d} \tag{3-5}$$

$$l = \sqrt{\frac{\sum_{j=1}^{d}\left(U_j - L_j\right)^2}{d}} \cdot 10^{-6} \tag{3-6}$$

lThe specific value of is given in the literature [15], which is used to control the fluctuati on range of the generated data. In the future, parameter optimization experiments and dis cussions will also be conducted on the parameters involved in the DBC and ZPS algorith ms. After data expansion sampling, the newly generated data set set is $K = \{\boldsymbol{x}_{new}, \boldsymbol{F}(\boldsymbol{x}_{new}) | \boldsymbol{x}_{new} = \boldsymbol{x}_i + \boldsymbol{\delta}_i\}$, where $\boldsymbol{F}(\cdot)$ is a nonlinear equation system function . The original data is combined with the new data to obtain the final data set used for train ing.

3.2.4 Data expansion algorithm process and results

After DBC sampling and ZPS data enhancement, the data set is more suitable for so lving scenarios of nonlinear equations. Under basic Latin hypercube sampling, the overa ll process description of the data expansion algorithm is given by the pseudo code of Al gorithm 3-2.

Select low-dimensional calculation examples for two-dimensional and three-dimen sional visualization. By drawing the data after Latin hypercube sampling and DBC and ZPS processing, different data distributions are compared, and the quality of the data se t obtained by the algorithm in the data generation and expansion steps is initially tested.

Specifically, in the simple system of equations

$$\begin{cases} x_1^{\,2} - x_2^{\,2} = 0 \\ 1 - |\,x_1 - x_2\,| = 0 \end{cases} \tag{3-7}$$

Conduct data visualization experiments and draw three-dimensional scatter diagrams to further analyse the distribution of data points after running the data generation and augmentation algorithm.

Algorithm 3-2 DBC and ZPS data expansion

Input:

TD: Original dataset

N_{centers}: Number of clusters

w: Number of clusters to be selected for data generation

m: Number of samples to be generated

Output:

Generated dataset TD

Begin

$CPs = KMeans(TD, N_{\text{centers}})$

Select the w clusters with the highest **CP** values and denote them as S.

For each y_i in S Do

Find the corresponding x_i.

Generate a new data point according to $x_{\text{new}} = x_i + \delta_i$, and add it to TD.

Compute y_{new}, and add the pair $(x_{\text{new}}, y_{\text{new}})$ to TD.

End For

Select m samples with the smallest y-range values and denote them as M.

For each y_i in M Do

Find the corresponding x_i.

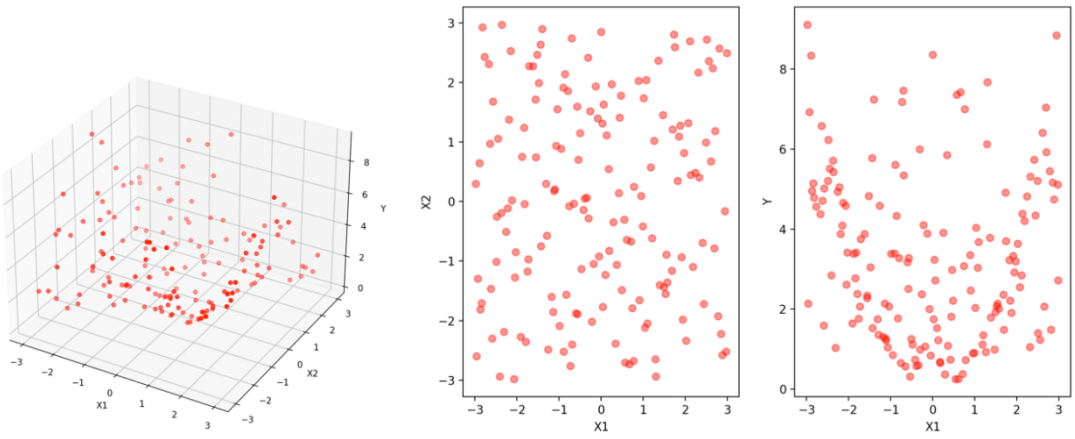
Generate a new data point according to $x_{\text{new}} = x_i + \delta_i$, and add it to TD.

Compute y_{new}, and add the pair $(x_{\text{new}}, y_{\text{new}})$ to TD.

End For

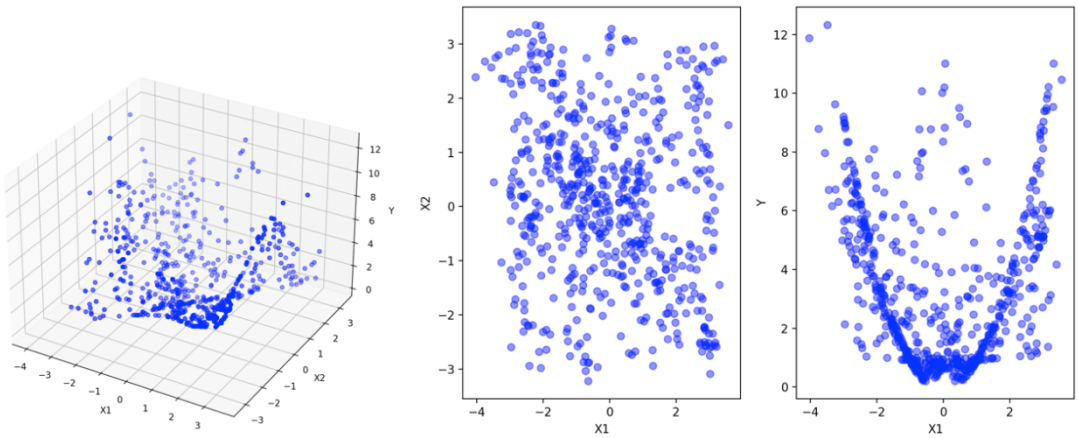
End

The result is shown in Figure 3-4.



（a）Data diftribution after DBC sampling

Figure 3-4 Visualization of low-dimensional equation data generation



(b) Data distribution after DBC and ZPS

Figure 3-4 (continued)

A set of pictures in each row shows the sampling results after one step. The first ro w is the data distribution after only Latin hypercube sampling, and the second row is the data distribution after DBC and ZPS data expansion. The first picture in each row is the point spatial distribution diagram drawn with $\boldsymbol{x} = (x_1, x_2)$ as the bottom two coordinates and $\|\boldsymbol{y}\|$ as the longitudinal bearing. The second picture is the distribution diagram of $\boldsymbol{x} = (x_1, x_2)$ on the plane. The third picture is a longitudinal section of the three-dimensi onal graph, and we can see the longitudinal value distribution of $\|\boldsymbol{y}\|$.

It can be seen that after Latin hypercube sampling, $\boldsymbol{x}$ is relatively evenly distributed in the definition domain and can basically cover the definition domain space. After DB C sampling and sampling at zero points, $\boldsymbol{y}$ points near 0 are denser than only Latin hype rlegislative sampling, which verifies that the data sampling and generation algorithm ha s a certain effect. In the future, we will continue to build and aggregate the model and so lve the equations based on the generated data to more accurately verify the effect of the model.

3.3 Optimization and integration of agent models

3.3.1 Conversion of optimization algorithms

Following the improvement plan of the model in Chapter 3, we can see that the mod el has changed from a single output to a multi-dimensional output. The FPA algorithm us ed by the original WDDEA-DBC to finally solve the system of equations is no longer sui table for direct solution of the equations.

Specifically, after the surrogate model is trained, the original model uses it to appr oximate the original system of equations, search its input space through the FPA algorit hm, minimize the output norm, and gradually approach the solution of the nonlinear sy stem of equations. and

After the model is reversed, the quasi-operator of the nonlinear system of equations is directly fitted. In the solution process, only all 0 vectors are input to obtain the numerical solution of the nonlinear system of equations without the use of intelligent optimization algorithms. In addition, after the improvement, the output of the model becomes multi-dimensional, and the output of the new model becomes the input of the original model, that is, the vector $\boldsymbol{x}$. It is no longer possible to define a new fitness value, and it is no longer possible to put it into the FPA algorithm to directly optimize the definition domain. In summary, the application of intelligent optimization algorithms needs to be adjusted to adapt to new problem solving scenarios.

Therefore, this topic shifts the application scenario of the optimization algorithm from directly optimizing the fitted function to optimizing the neural network. This conversion can bring many benefits.

First of all, intelligent optimization algorithms usually have excellent global search capabilities, which help neural networks get rid of the constraints of local optimal solutions and discover better global optimal solutions. Secondly, these algorithms also show good adaptability and robustness, and can flexibly adjust the search strategy based on feedback information during the optimization process to cope with the complexity of the optimization problem and the uncertainty of the environment. In addition, intelligent optimization algorithms are also suitable for discontinuous and non-smooth optimization problems, and are less dependent on parameters and hyperparameters, making it easier to adjust and optimize parameters.

3.3.2 Implementation of optimization algorithm

After determining the application conversion method of the intelligent optimization algorithm, it is worth thinking about how to apply it to the optimization of neural networks. For the convenience of comparison, this project also uses the FPA algorithm as the choice of intelligent optimization algorithm. Referring to the FPA algorithm [17] and its related information, the optimization process is roughly as follows.

First define the problem, use the parameters of RBFNN as optimization variables, and use the performance indicators of the neural network as the optimization objective function. Here is the fitness value of the data set calculated on the network corresponding to the parameters.

After that, a certain number of flower individuals are initialized, and each individual represents a parameter combination of the RBFNN network, including the network structure and the weight and bias of each neuron. Then, for each candidate parameter individual, this paper uses the RBFNN network to calculate the output under the parameter setting, and obtains the output result $\hat{\boldsymbol{x}}$ of the neural network under the current parameters, and calculates the MSE with the accurate value $\boldsymbol{x}^*$ as the fitness value of the optimization problem.

Then, this topic uses mathematical expressions of natural laws such as self-pollination, cross-pollination, Levy flight, and natural variation to update the position of each individual, expand the scope of the search, and increase the randomness and diversity of the algorithm.

After updating the position, this project updates the position of each individual according to the pollen dispersal strategy, so that individuals with better fitness values are more likely to spread their information to neighboring individuals. Finally, through step-by-step iteration, the stopping threshold is set to obtain the best RBFNN parameter combination.

3.3.3 Model integration

Inspired by ensemble learning, after obtaining each trained RBFNN model, how to perform ensemble aggregation to obtain more accurate solution results is a question wor th exploring. Here we still refer to the integration method of the WDDEA-DBC model, by calculating the MSE of a single model and giving it a weight that is the reciprocal of the MSE. Specifically, assume that T models are obtained through training, the expressi ons of each model are $\widehat{g_1}(\boldsymbol{y}), \widehat{g_2}(\boldsymbol{y}), \dots, \widehat{g_T}(\boldsymbol{y})$, and the MSEs are $MSE_1, MSE_2, \dots, MSE_T$, respectively, and are weighted by Equation (3-8).

$$\hat{g}(\boldsymbol{y}) = \frac{\sum_{i=1}^{T} \frac{g_i(\boldsymbol{y})}{MSE_i}}{\sum_{i=1}^{T} \frac{1}{MSE_i}} \tag{3-8}$$

Since the higher the MSE, the greater the deviation of the model, the reciprocal w eighting operation can reduce the weight of the large deviation model, further enhanci ng the accuracy of the model solution.

After the above processes of data generation, data expansion, proxy model training, and weighting, the final model $\hat{g}(\boldsymbol{y}) = (x_1^*, x_2^*, \dots, x_d^*)$ is obtained. When it is necessary to solve the nonlinear system of equations, only the all-zero vector $\boldsymbol{y_d}$ needs to be input to the model to obtain the vector $\hat{g}(\boldsymbol{y_d})$, which is the numerical solution of the system of equations solved by the final algorithm.

3.4 Summary of this chapter

This chapter mainly analyzes and makes modifications to the original WDDEA-DBC model.

This article first introduces the surrogate model structure of the algorithm and its role in fitting nonlinear equations, and then analyzes the limitations of the original algorithm in surrogate model selection and optimization, as well as the problems of information loss an d error accumulation. Next, this paper proposes to transform the surrogate model from the fitting norm to the inverse operator of the fitting system of equations, and improves the ac curacy and efficiency of the algorithm by modifying the surrogate model structure and trai ning methods. These improvements provide new methods and foundations for solving pro blems of nonlinear equations.

At the same time, this article introduces the importance of data generation methods, d escribes two methods of Latin hypercube sampling and data expansion: DBC sampling an d zero-point sampling (ZPS), and shows the process of the data expansion algorithm and it s effect verification, further providing a reliable foundation for model establishment and so lution.

Finally, the overall architecture of the RDDEA-ZPS algorithm is obtained, as shown in Figure 3-5.

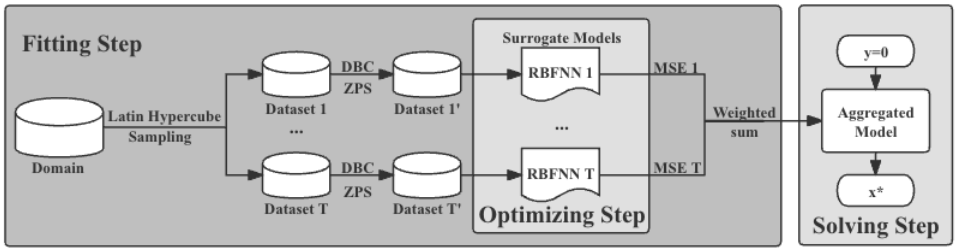


Figure 3-5 RDDEA-ZPS algorithm architecture

Based on the above discussion, this paper proposes an improved RDDEA-ZPS mod el by modifying the RBFNN agent model and innovatively proposing the ZPS method. T his model can effectively deal with the error problem of the original model, simplify the solution process, and improve the efficiency and accuracy of solving nonlinear equations.

Chapter 4 Numerical Experiments and Results

4.1 Parameter optimization experiment

4.1.1 RDDEA-ZPS parameters and calculation example equations

The process of data generation, proxy model establishment and optimization involv es several important parameter selection issues. These hyperparameters have a great im pact on the operating efficiency and results of the algorithm. By adjusting the hyperpara meters, the complexity of the model can be controlled, the performance of the model ca n be optimized, and the training process of the model can be accelerated, thereby impro ving the generalization ability and practicality of the model.

Therefore, after the model framework is built, how to set parameters reasonably ne eds further exploration. Here, several parameters that have an important impact on the e xperimental results are selected and explained. Subsequent numerical experiments will be conducted on these parameters to find the optimal settings.

The selected parameters are shown in Table 4-1.

Table 4-1 Parameter values and meanings that need to be optimized

Notation	Meanings
N_centers	The number of centres of K-Means, also the number of neurons in the hidden layer of the radial basis function network.
w	The proportion of data that needs to be generated when ZPS expands data
T	The number of basic proxy models in the integrated model
l	When expanding the data, the fluctuation range of new data points

Following the calculation examples in the original WDDEA-DBC algorithm, this pr oject selected the following six nonlinear equations from the paper [18] as the objects of s tudy and comparison. The equations are listed in Table 4-2.

It includes low-dimensional equation systems such as 2 dimensions, which facilitat es visual analysis during data generation and parameter optimization, and also includes large-scale nonlinear equation systems with dimensions of 10 and above to test the oper ating performance of the equation system in high-dimensional situations.

After checking the solution range of the system of equations in the original paper, i t was found that there are multiple solutions in the original domain. Considering that it is difficult to find multiple solutions for the two current models, the solution range was further refined.

Table 4-2 Example of nonlinear equation system

Number	System of equations	Solution range	dim
N1	$\begin{cases} x_1^2 - x_2^2 = 0 \\ 1 - \lvert x_1 - x_2 \rvert = 0 \end{cases}$	$x_1 \in [0,3]$ $x_2 \in [-3,0]$	2
N2	$\begin{cases} x_1^2 - x_2 - 2 = 0 \\ x_1 + \sin(\frac{\pi x_2}{2}) = 0 \end{cases}$	$x_1 \in [0.71,3]$ $x_2 \in [-3,0]$	2
N3	$\begin{cases} x_1 + x_2 - 3 = 0 \\ x_1^2 + x_2^2 - 9 = 0 \end{cases}$	$x_1 \in [0,1.5]$ $x_2 \in [1.5,3]$	2
N4	$\begin{cases} x_1 - \sin(2x_1 + 3x_2) - \cos(3x_1 - 5x_2) = 0 \\ x_2 - \sin(x_1 - 2x_2) + \cos(x_1 + 3x_2) = 0 \end{cases}$	$\boldsymbol{x} \in [-1,0]^2$	2
N5	$\begin{cases} x_1 - 0.25428722 - 0.18324757x_4x_3x_9 = 0 \\ x_2 - 0.37842197 - 0.16275449x_1x_{10}x_6 = 0 \\ x_3 - 0.27162577 - 0.16955071x_1x_2x_{10} = 0 \\ x_4 - 0.19807914 - 0.15585316x_7x_1x_6 = 0 \\ x_5 - 0.44166728 - 0.19950920x_7x_6x_3 = 0 \\ x_6 - 0.14654113 - 0.18922793x_8x_5x_{10} = 0 \\ x_7 - 0.42937161 - 0.21180486x_2x_5x_8 = 0 \\ x_8 - 0.07056438 - 0.17081208x_1x_7x_6 = 0 \\ x_9 - 0.34504906 - 0.19612740x_{10}x_6x_8 = 0 \\ x_{10} - 0.42651102 - 0.21466544x_4x_8x_1 = 0 \end{cases}$	$\boldsymbol{x} \in [-2,2]^{10}$	10
N6	$\begin{cases} x_i + \sum_{j=1}^{D} x_j - (D+1) = 0 \quad i = 1, \cdots, D-1 \\ \left[\prod_{j=1}^{D} x_j\right] - 1 = 0 \end{cases}$	$\boldsymbol{x} \in [-1,1]^{20}$	20

In the subsequent process of parameter optimization, different equations will be used to conduct numerical experiments according to the actual situation.

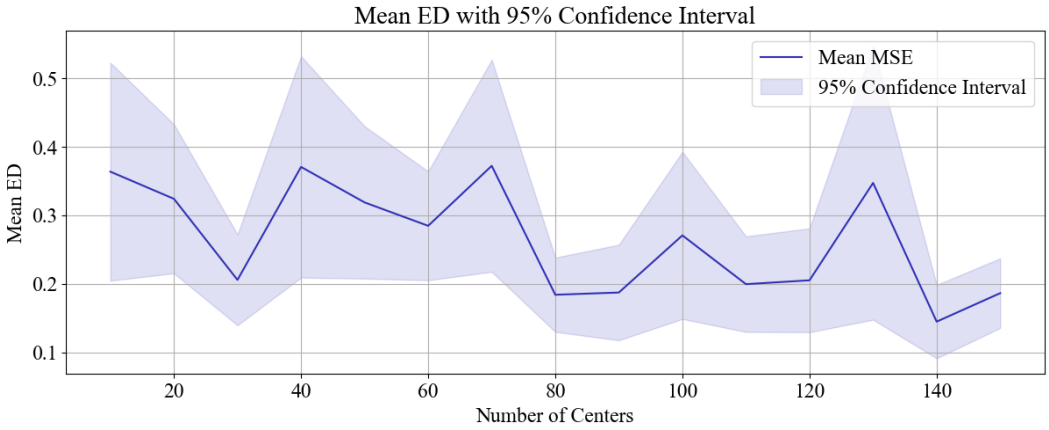
4.1.2 Number of clustering centers

This parameter is the number of categories *N_centers* in the clustering algorithm K-Means. In order to save time and overhead, the cluster center obtained in this step is also used as the center point of the radial base layer, so it involves the number of hidden layer neurons.

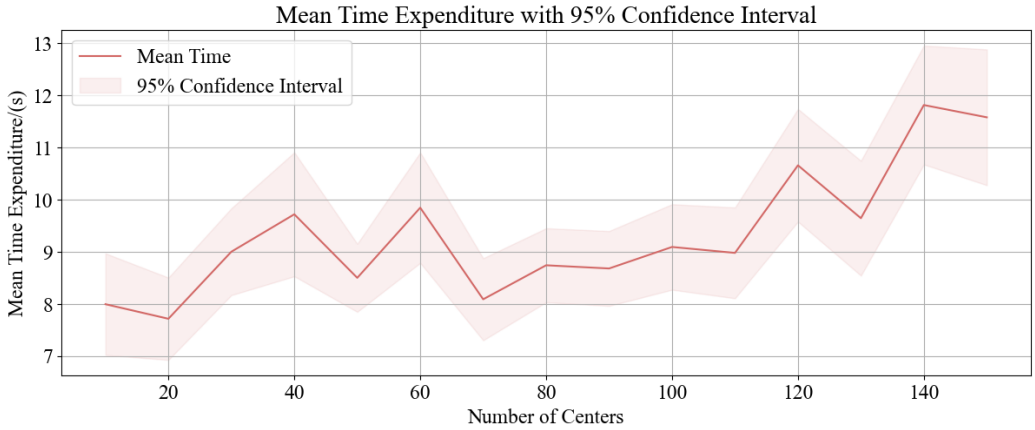
If the number of neurons is too small, the model is too simple and cannot fully expr ess the characteristics of the training data, resulting in underfitting problems. In this case , the model may not capture the complex relationships of the data well, resulting in high training and test errors. However, if there are too many neurons, the model will become too complex, and it will be easy to remember the noise and details in the training data, w hile ignoring the real patterns and laws in the data.

At this time, excessive time overhead may occur. Therefore, how to select the clusteri ng center, that is, the number of neurons, requires numerical experiments to explore.

Select the N1 equation, and while other parameters remain consistent, set the number of test centers to 10, 20, …, and 150 to conduct numerical experiments. For each differen t center value, repeat the complete experiment 20 times, and record each parameter, error and time cost of each experiment. The trend graph of the experimental results drawing lin e is shown in Figure 4-1. The error is measured by the Euler Distance (ED) between the p redicted value and the true solution.



（a）N1 Loss

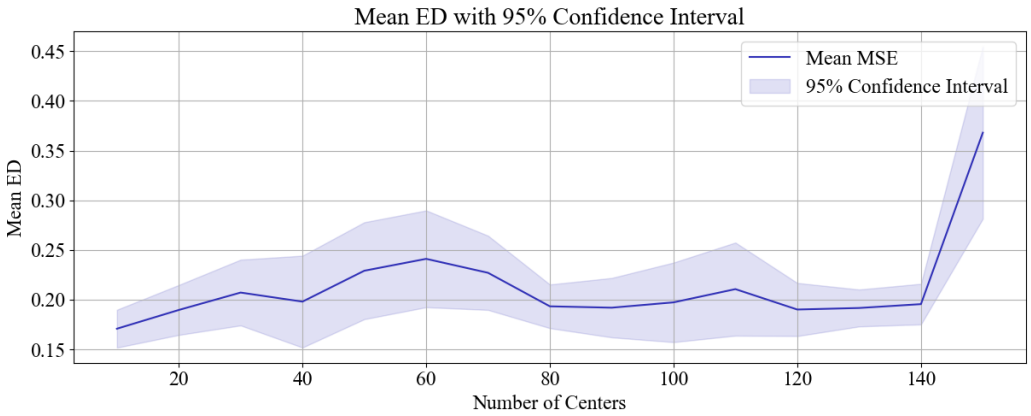


（b） Averaged Time Expenditure of Equation N1

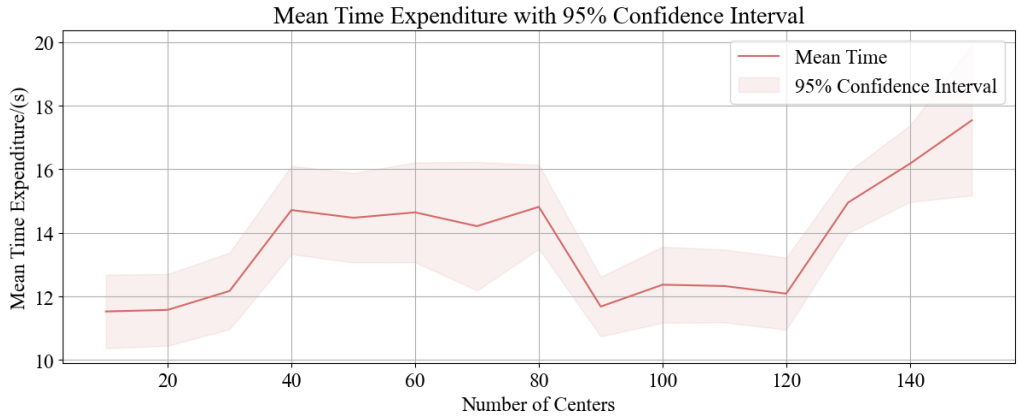
Figure 4-1 Optimization results for the number of different center points on the N1 example

It can be seen that in repeated experiments on low-dimensional examples, as the nu mber of center points increases, the average ED, that is, the error, fluctuates but graduall y decreases. At the same time, the time consumed in solving the equation is gradually in creasing.

Similarly, the same numerical test was performed on the higher-dimensional example N5, and the results are shown in Figure 4-2 below.



（a）N5 Loss



（b）Average Time Expenditure of Equation N5

Figure 4-2 Optimization results for the number of different center points on the N5 example

In high-dimensional examples, for different numbers of center points, the error difference of model optimization is not as large as that of low-dimensional examples. Excluding the influence of the extreme value at 150 centers due to unstable calculations, as the number of centers increases, the overall trend still shows that when there are few centers, the error is large and the time is small, and when there are many centers, the error is small for a long time.

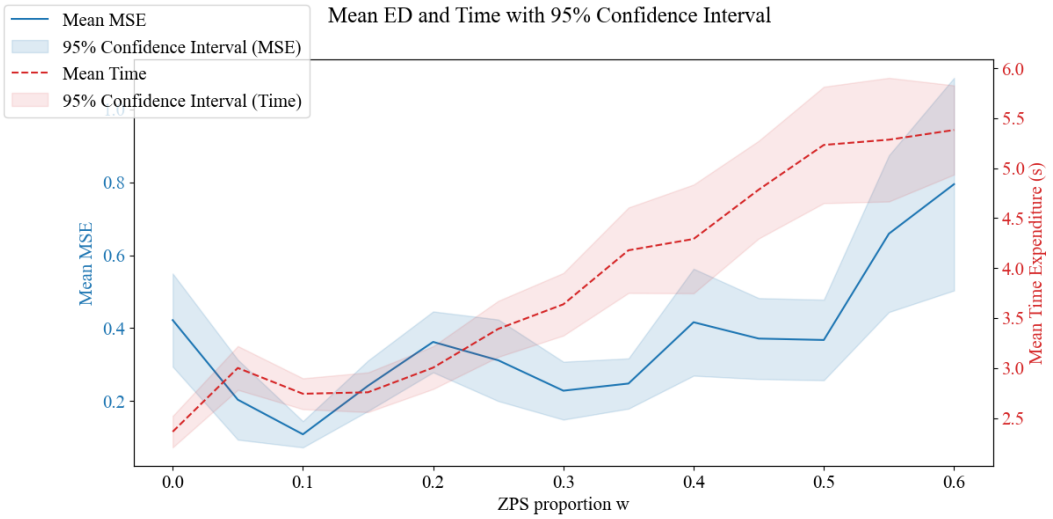
Based on the curve trends of the two calculation examples, *N_centers* was selected as 80 after weighing it, which is the number of centers for subsequent solution of the equation.

4.1.3 ZPS data expansion ratio

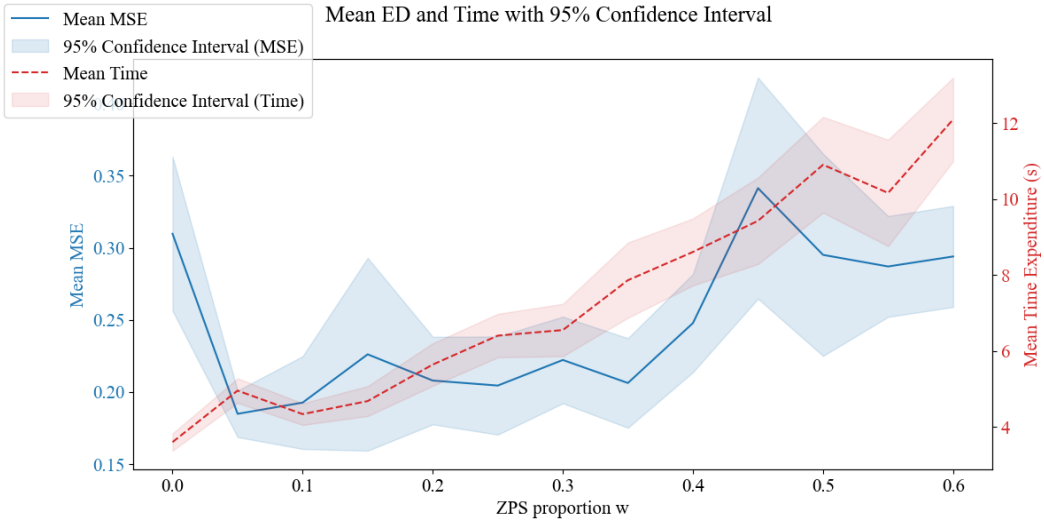
Zero-point sampling is one of the important ways to expand the effective data set, and the value of its sampling ratio w will also affect the accuracy and effect of the model in solving nonlinear equations.

When the number of samples is small, only a small amount of data is sampled, and t he data near the zero point is still not enough for the model to capture the variable corre spondence near the solution. When more data is sampled by ZPS, most of the data will be expanded, which is not essentially different from ordinary DBC. Therefore, the select ion of w is also an important parameter that needs to be considered through actual nume rical experiments in this article.

Similarly, conduct numerical experiments on two equation systems with different di mensions, N1 and N5. For the setting of w, select a series of sequences of 0, 0.05, 0.1, …, 0.55, 0.6 with 0.05 as the arithmetic difference, and conduct experiments in sequenc e. The experimental results on the N1 and N5 equations are shown in Figure 4-3.



（a）Optimisation result of N1



（b）Optimisation result of N5

Figure 4-3 Optimization results of w with different sampling ratios on the calculation examp

The variables and horizontal and vertical coordinates shown in Figure 4-3 and Figure 4-2 are the same. Figure 4-3 moves the two variables into the same coordinate system at the same time, making it easier to select parameters more intuitively through intersection points, trends, etc.

It can be seen from the optimization result graphs of the two equations that the errors of the solutions obtained first decrease and then increase as the sampling ratio increases. The errors of the two graphs reach a minimum value near $w = 0.1$; at the same time, the time cost of solving the two equations increases with the increase of the sampling ratio. Therefore, considering the error and time, $w = 0.1$ was finally selected as the proportion of ZPS data expansion, that is, zero-point sampling was performed on 10% of the entire data set for expansion.

4.1.4 Number of proxy models

Since the RDDEA-ZPS algorithm also uses the Bagging framework for the management of proxy models, that is, integration, the selection of the number of proxy models T also has a certain impact on the solution results of the algorithm.

The effect of ensemble learning depends on the differences between individual models. When the number of proxy models is too small, it may cause the integrated model to have a large deviation and fail to capture the complexity of the data, thus affecting the final results. However, increasing the number of base models also means increasing the complexity of the integrated model. If the underlying model is of low quality, increasing the number may cause the variance of the ensemble model to increase, thereby reducing overall performance. Therefore, there is a need to balance model quantity and quality.

In this section, we also use repeated experiments and conduct numerical experiments on three calculation examples of N1, N2 and N5 while keeping other parameter settings the same. Optimization was carried out in three cases when the number of models T was as 10, 20 and 30. The results of the optimization are drawn as box plots 4-4 for comparison.

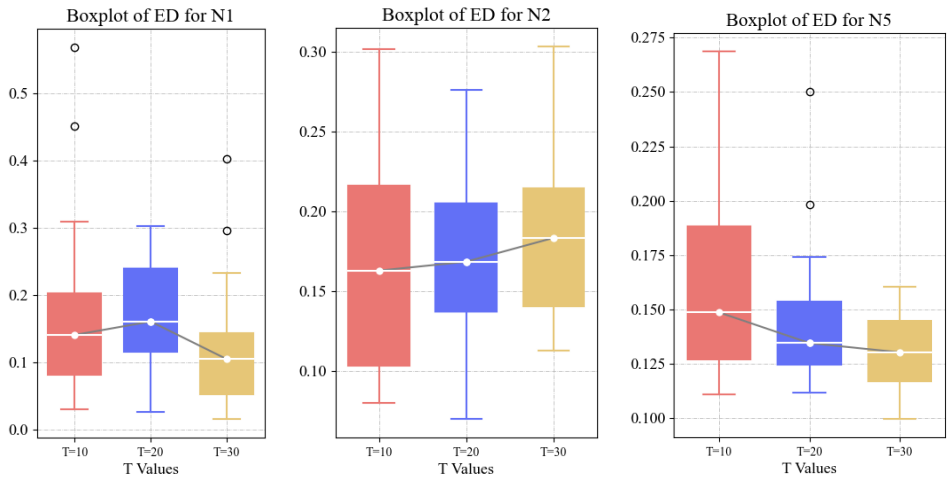


Figure 4-4 Optimizing the number of agent models

It can be seen that in the N1 and N2 low-dimensional calculation examples, the ED l oss of the model under the conditions of T=10 and T=20 is similar, and the loss distributi on at T=20 is more concentrated; at T=30, the performance of the two low-dimensional c omputing power is unstable. In N1, the ED loss is lower, while in N2, the loss is larger a nd the distribution is more dispersed.

In high-dimensional examples, the ED loss decreases as the number of surrogate m odels increases. At T=10, the loss distribution is highly dispersed, showing an unstable s tate, and the ED and mean value are significantly higher than the other two cases; in the case of T=20 and 30, the ED losses of the two are close.

Based on the optimization of the above three equations and factors such as progra m operating efficiency, the subsequent model selects T=20 as the number of proxy mo dels to solve the equation system.

4.1.5 The fluctuation range and proportion combination of ZPS expanded data

For the data expansion step, in addition to the proportion of data expansion w menti oned in Section 3.2, the coefficient l that controls the variation range of $\boldsymbol{\delta_i}$ during data g eneration also needs to be determined through experiments. This is because l directly aff ects the degree of deviation between the new data points generated around the target dat a point and the original data. The specific expressions are given by (3-5) and (3-6).

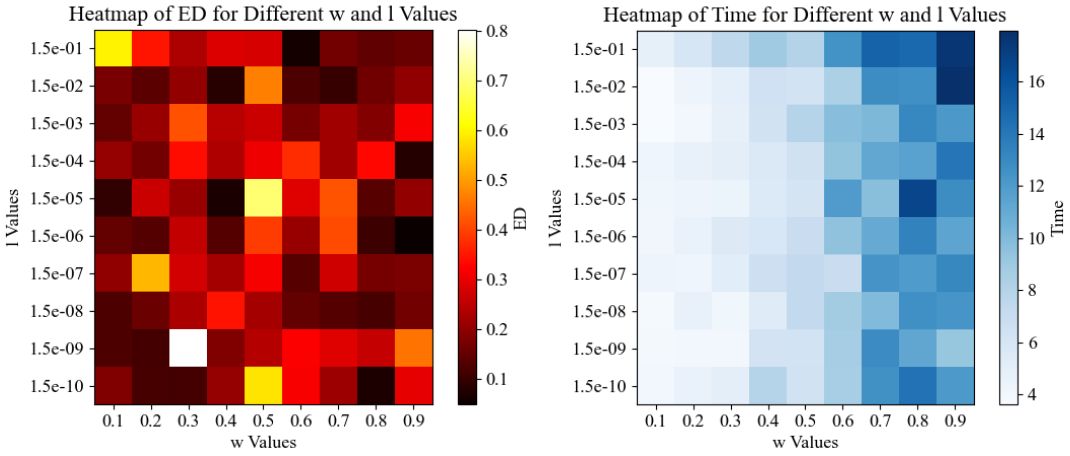
When l is small, the distance between the original data points and the new data point s is small, and the degree of deviation is small, so the generated data may be very close t o the original data, so that the best effect of the data expansion algorithm cannot be exert ed; when l is large, the distribution between the new data and the original data points is f ar away, and the difference is large, so the new data may fall into the neighborhood of ot her points, causing the similarity between the new and old data to be too poor, and the ge nerated data cannot be a good approximation of the original data points. Features. Theref ore, how to choose the range value l is another hyperparameter that needs to be weighed.

In addition, the matching of l and w is also crucial. w controls the proportion of ne wly generated data, and l controls the degree of fluctuation of generated data. They joint ly affect the similarity and degree of fluctuation between the new data points generated by the data expansion algorithm and the original data points.

Specifically, when the selected parameter l is small, the generated new data points a re more similar to the original data points because the distance between them is closer. However, if a larger parameter w is used to control the proportion of data expansion at t his time, it may cause a large number of new data points to overlap with the original dat a points, thus losing the advantages of the data expansion algorithm. When the selected parameter l is larger, the degree of fluctuation between the new data point and the origin al data point will increase because the distance between them is farther. In this case, if a larger parameter w is still used to control the proportion of data expansion, the generate d new data points may be too scattered and lose a good approximation of the original dat a distribution.

Situations such as the above two types pose challenges to exploring the combinatio n of l and w. This section will select a series of candidate values of l and w, and perfor m grid optimization based on their different combinations. Each combination will be tra ined 20 times, and

Find the average of the indicators and draw a heat map. The results are shown in Figure 4-5. The value options of l and w are marked in the horizontal and vertical coordinates.



（a）ED Loss Heatmap　　（b）Time Expenditure Heatmap

Figure 4-5 w and l parameter combination ED and time consumption heat map

It can be seen from the figure that for the ED loss, compared with the thermal value s at other positions, the color on the diagonal from the upper left corner to the lower righ t corner is lighter, which means that when a value in w or l is larger, it will bring a large r ED error to the experiment; while the thermal values at the upper left and lower right c orners are lower, which means that when the two parameters increase or decrease to the same extent, the optimization effect is better. However, considering that when both are l arge, the range of data generation cannot be accurate to the target range, which may red uce the effect and increase program running overhead. This suggests that we should sele ct parameter combinations from the lower left of the heat map.

Regarding time consumption, you can see that the contrast between the left and righ t sides of the heat map is more obvious. From a horizontal perspective, for different w v alues, the time consumption increases significantly as its value increases; from a vertical perspective, for different l values, there is no obvious difference in time consumption. T his phenomenon is also consistent with the actual situation, because l only controls the r ange of generated data rather than the amount, but w controls the amount of data that ne eds to be generated and is directly related to the algorithm running time.

Based on the above two indicators, in order to maintain the accuracy and breadth of the data generation range and save running time to the greatest extent, $w = 0.1$ and $l = 1.5 \times 10^{-3}$ were finally selected as the final parameter values. This selection is consi stent with the parameters in 4.1.3.

4.2 Comparison of model solution results

4.2.1 Error comparison

Under the optimal parameter settings, several experiments were conducted on the ori ginal model WDDEA-DBC and the improved model RDDEA-ZPS, and the ED loss was compared. For each algorithm, run it 20 times on the N1～N6 example, record the ED err or of each model in Table 4-3, and calculate the mean and variance.

Table 4-3 Comparison of ED errors of two algorithms

Number	RDDEA-ZPS	WDDEA-DBC
N1	**0.0693±0.0427**	0.8200±0.8502
N2	**0.1896±0.0359**	0.6857±0.4842
N3	**0.1896±0.673**	1.3836±0.3781
N4	**0.0425±0.0235**	0.3507±0.1502
N5	**0.1426±0.0185**	0.4752±0.1778
N6	**0.3794±0.0043**	1.6690±0.4905

The results show that in all six calculation examples, the average error of the RDDE A-ZPS algorithm is significantly smaller than the WDDEA-DBC algorithm, and it can b e seen from the size of the standard deviation that the stability of RDDEA-ZPS is much stronger than the WDDEA-DBC algorithm. This directly demonstrates the powerful abil ity of the improved algorithm to solve.

4.2.2 Comparison of solution results

Finally, several experiments were conducted using the original model WDDEA-DB C and the improved model RDDEA-ZPS. The two models gave solution results in the n ew definition domain, as shown in Table 4-4.

Table 4-4 Comparison of the solution results of the two algorithms

number	Independent Variable	Solution	RDDEA-ZPS Result	RDDEA-ZPS Error	WDDEA-DBC Result	WDDEA-DBC Error
N1	x_1	0.5000	0.5004	**0.0530**	0.6620	0.2149
	x_2	-0.5000	-0.4469		-0.6412	
N2	x_1	1.0000	0.9301	**0.1737**	1.1477	0.1741
	x_2	-1.0000	-1.1590		-0.9078	
N3	x_1	0.0000	0.1359	**0.1365**	0.5755	0.5864
	x_2	3.0000	3.0134		2.8875	

Table 4-4（Continued）

number	Independent Variable	Solution	RDDEA-ZPS Result	RDDEA-ZPS Error	WDDEA-DBC Result	WDDEA-DBC Error
N4	x_1	-0.1733	-0.1809	**0.0124**	-0.4115	0.2401
	x_2	-0.2561	-0.2659		-0.2264	
N5	x_1	0.2578	0.1739	**0.1621**	0.1294	0.2983
	x_2	0.3811	0.3708		0.1551	
	x_3	0.2787	0.3433		0.2451	
	x_4	0.2007	0.2651		0.2067	
	x_5	0.4453	0.4795		0.4550	
	x_6	0.1492	0.1556		0.1412	
	x_7	0.4320	0.4801		0.3043	
	x_8	0.0734	0.0796		0.0389	
	x_9	0.3460	0.3776		0.3184	
	x_{10}	0.4273	0.5064		0.4703	
N6	x_1	1.0000	1.0378	**0.1889**	0.9255	0.3752
	x_2	1.0000	0.9774		0.9088	
	x_3	1.0000	1.0391		0.9426	
	x_4	1.0000	1.0000		0.9147	
	x_5	1.0000	0.9035		0.8805	
	x_6	1.0000	1.0245		0.9265	
	x_7	1.0000	1.0119		0.9207	
	x_8	1.0000	1.0285		0.9368	
	x_9	1.0000	0.9911		0.9099	
	x_{10}	1.0000	1.0681		0.9433	
	x_{11}	1.0000	0.9975		0.9223	
	x_{12}	1.0000	0.9120		0.8889	
	x_{13}	1.0000	1.0406		0.9398	
	x_{14}	1.0000	0.9913		0.9109	
	x_{15}	1.0000	1.0057		0.9248	
	x_{16}	1.0000	0.9284		0.8855	
	x_{17}	1.0000	0.9935		0.9161	

number	Independent Variable	Solution	RDDEA-ZPS Result	RDDEA-ZPS Error	WDDEA-DBC Result	WDDEA-DBC Error
	x_{18}	1.0000	0.9865		0.9101	
	x_{19}	1.0000	0.9874		0.9143	
	x_{20}	1.0000	1.0402		0.9376	

The results show that the RDDEA-ZPS model is significantly improved compa red with the original model in all six examples, verifying the effectiveness of the n ew model framework.

4.3 Ablation experiment

4.3.1 Model structure experiment

The RDDEA-ZPS algorithm has made major changes in the structure of the agent model and the optimization process during the improvement. How to verify the effectiv eness of such modifications can be done by changing part of the model structure to verif y the overall effectiveness of the new model.

In this section, the model will be dismantled, and the models used for comparison a re divided into the following categories: only using FPA to optimize the system of equat ions, RDDEA-ZPS only using the Adam optimization algorithm, and RDDEA-ZPS usin g only a single agent model without weighting.

For the first comparison model, that is, the model that only uses FPA for optimizati on, specifically the case where the subsequent RBFNN, a neural network fitting the inve rse operator of the nonlinear system of equations, is removed. This model is to verify th e impact of the fitting inverse operator neural network proposed by the algorithm on the effectiveness and accuracy of solving the system of equations.

For the second RDDEA-ZPS model, which only uses the Adam optimization algori thm, the FPA neural network optimization step is removed based on the original algorith m, and only the traditional neural network optimization algorithm Adam algorithm is us ed to find network parameters, which is the same as the method of optimizing the neural network in the WDDEA-DBC model. This step is based on the WDDEA-DBC algorith m to ensure that no new parameter optimization algorithm is introduced. At the same ti me, it mainly tests the efficiency of the FPA algorithm in neural network optimization a nd its impact on the final equation solution results.

For the third model, RDDEA-ZPS only uses a single agent model without weightin g, which means that the integration method of MSE reciprocal weighting is removed an d only a single agent model is used for solution. The purpose of this step is to verify the role of the model integration module in the RDDEA-ZPS algorithm.

For the three models mentioned above, and the complete model, respectively recor ded as FPA, Adam, Single and RDDEA-ZPS in the subsequent tables, run the above diff erent models 20 times on six calculation examples, and get

The results are shown in Table 4-5.

Table 4-5 Model ablation ED error comparison

Number	FPA	Adam	Single	RDDEA-ZPS
N1	0.1080±0.0631	0.5132±0.4043	0.4003±0.3772	**0.0693±0.0427**
N2	1.2970±0.0060	**0.1284±0.0594**	0.2158±0.0959	0.1896±0.0359
N3	1.6770±0.0010	0.2945±0.1725	0.3576±0.3286	**0.1896±0.673**
N4	0.0584±0.0428	0.6986±0.0605	0.0852±0.0867	**0.0425±0.0235**
N5	0.8898±0.0536	0.7853±0.0321	0.2963±0.1264	**0.1426±0.0185**
N6	0.9186±0.1548	**0.2175±0.0297**	0.3859±0.0194	0.3794±0.0043

It can be seen from the experimental results of different models that the complete RDDEA-ZPS model has the smallest overall error and the best effect on the six calculation examples.

For the nonlinear system of equations solution method that only uses FPA without applying the neural network surrogate model method, it can be seen that the results on the six examples are poor. Except for the errors on N1 and N4, which are close to the complete model, the Euler distance errors on the remaining examples are large. This proves that the method of using neural network to fit the inverse operator has an obvious effect on solving the problem of nonlinear equations.

For the algorithm that only uses one surrogate model without integrating multiple models, that is, Single in the table, the overall performance on the six calculation examples is better than the algorithm that only uses FPA, but the accuracy is still generally lower than that of the complete model. In addition, due to the lack of integration, the model shows greater instability, which is specifically reflected in the large variance of the 20-time error. This ablation model proves that the integrated model method of MSE reciprocal weighting can effectively reduce errors and improve the accuracy of the algorithm. It also proves the stability of the integrated model algorithm.

For the model that only uses Adam to optimize the neural network parameters, it can be seen that its performance is slightly better than the complete model on the two calculation examples of N2 and N6, while the complete model has smaller errors on other calculation examples. The comparison in this step can show that compared with the traditional Adam optimization method, FPA can search for better parameter combinations near the effective parameters in a more precise manner in parameter optimization to a certain extent, improving the accuracy of the algorithm.

Through the algorithm comparison of the above four different architectures, it can be proved as a whole that the necessity of each architecture of the RDDEA-ZPS model in solving nonlinear equations problems.

4.3.2 Data sampling experiment

The data sampling method mentioned in this model can also improve the accuracy of the algorithm to a certain extent. In order to verify the role of each part, this project also conducted ablation experiments on each module of the data generation part.

As mentioned in Chapter 4, the generation of the data set mainly consists of two pa rts: Latin hypercube sampling and data expansion. The latter is divided into DBC and Z PS data expansion methods. The following will conduct ablation experiments on the DB C and ZPS parts respectively to detect the role of the two new algorithms in data expans ion.

In the following, the model using only Latin hypercube sampling is denoted as Lati n, the model without ZPS is denoted as Latin+DBC, the model without DBC module is denoted as Latin+ZPS, and the complete sampling model is denoted as Full. While keep ing other parameters the same, perform different model designs 15 times, and record the average ED error and its standard deviation. The results are shown in Table 4-6.

Table 4-6 Data sampling ablation ED error comparison

编号	Latin	Latin+DBC	Latin+ZPS	Full
N1	0.2247±0.2041	0.2914±0.0857	0.1633±0.1069	**0.0769±0.0456**
N2	0.1690±0.0705	**0.1154±0.0494**	0.1633±0.0657	0.1948±0.0377
N3	0.1389±0.0805	0.2642±0.1087	**0.0827±0.1232**	0.1814±0.0627
N4	0.2335±0.0612	0.1386±0.0503	0.0589±0.0377	**0.0451±0.0233**
N5	0.1582±0.0454	0.2549±0.7901	0.1991±0.1035	**0.1458±0.0200**
N6	0.4177±0.0669	0.4051±0.0237	0.4007±0.0054	**0.3787±0.0044**

It can be seen from the results that the algorithm using the complete data sampling process performs better on most calculation examples. On the two calculation examples N2 and N3, the performance is not the best but the corresponding error data are not ver y different. The error standard deviation of the complete model is the lowest and the m odel is the most stable.

Comparing the complete models in the first and last columns, the algorithm using o nly Latin hypercube sampling has larger errors in all six calculation examples. This sho ws that subsequent data augmentation steps including DBC and ZPS play an important r ole in improving the accuracy of the overall algorithm.

For the algorithm using Latin hypercube sampling and DBC sampling, compared wi th the data using only Latin hypercube sampling, the accuracy has not been greatly impr oved, and the ED error and error variance are close. This proves that for the problem of s olving nonlinear equations, although DBC sampling can make the data distribution more uniform, it cannot effectively enhance the most important part of the data. The experime ntal results are consistent with the analysis in Chapter 4.

For the algorithms using Latin hypercube sampling and ZPS sampling, compared w ith the above two algorithms, the accuracy has been greatly improved, and the results in some examples even exceeded the complete model using the DBC expansion method. T his proves that the ZPS newly proposed in this topic is very effective in enhancing impo rtant data and improving the accuracy of the algorithm.

After the above experiments and analysis, the effectiveness of each module of the d ata generation and expansion steps can be verified. Then, the 15 experimental results of each of the above models were drawn into box plots to more intuitively display the diffe rences between the models.

The difference is shown in Figure 4-6.

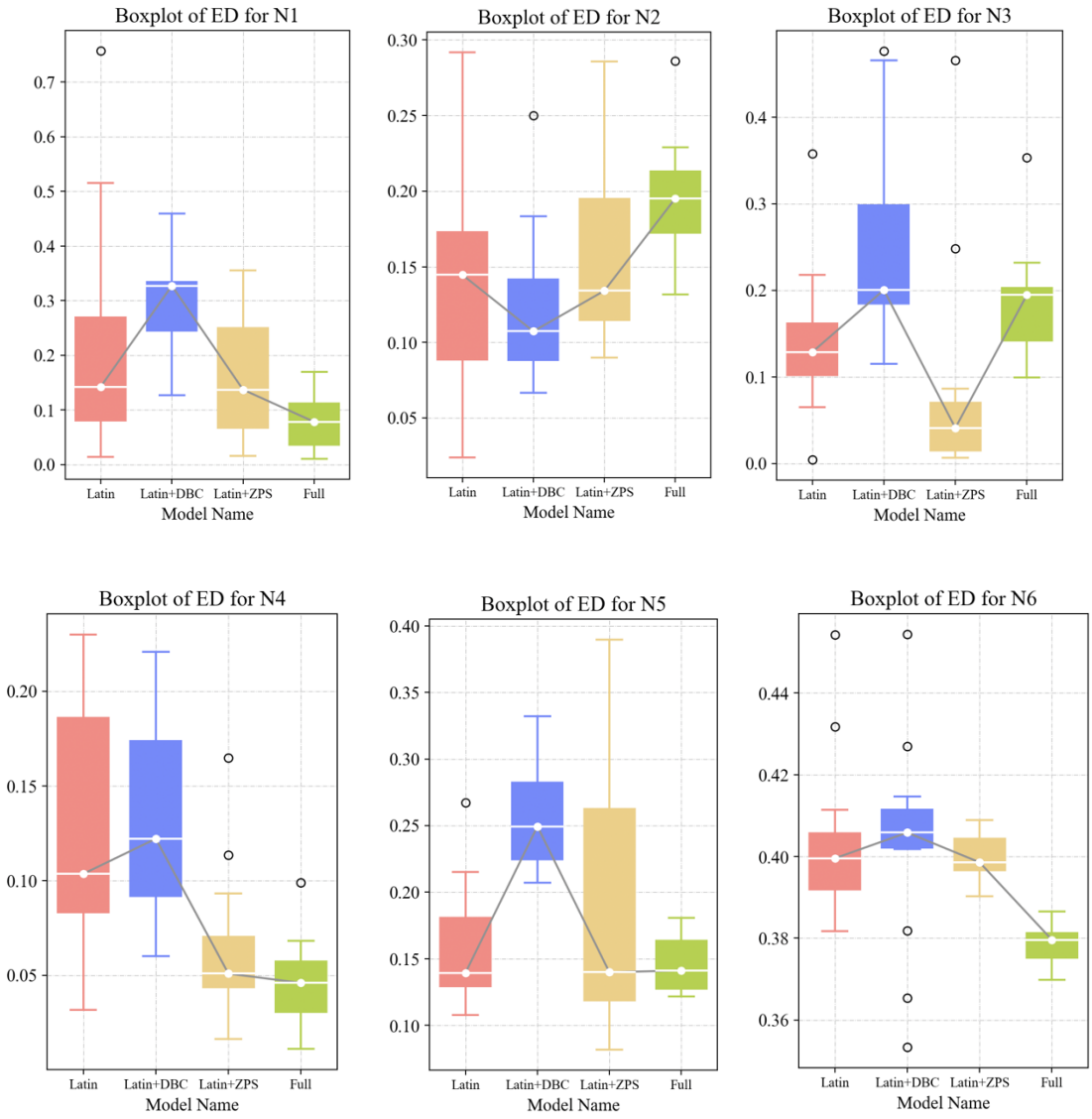


Figure 4-6 ED error box plots of different models

As can be seen from the figure, the complete model has the smallest error in most c alculation examples, followed by the algorithm using Latin hypercube sampling and ZP S sampling model. This is consistent with the previous analysis and further proves the e ffectiveness of data sampling and expansion algorithms including the ZPS method.

4.4 Summary of this chapter

This chapter improves the performance of the algorithm through the optimal selectio n and verification experiments of the four key parameters in the RDDEA-ZPS algorithm. This paper studies the number of clustering centers (*N_centers*), ZPS data expansion rati o (*w*),

The impact of the selection of the number of agent models (T) and the data generation fluctuation range (l) on the performance of the algorithm was determined, and the optimal parameter combination was determined. Optimizing these parameters improves the expression and generalization capabilities of the model and ensures the effectiveness of the algorithm.

Furthermore, by comparing the performance of the two algorithms in solving nonlinear equations, the results show that the RDDEA-ZPS model is better than the original model in all calculation examples, verifying the effectiveness of the new model framework. During the improvement process, this paper adjusted the structure and optimization process of the agent model, and verified the necessity of each improvement measure through ablation experiments. Experimental results show that the complete RDDEA-ZPS model performs best in terms of overall error and stability.

In summary, these studies and experiments have verified the superiority and necessity of the RDDEA-ZPS model and its data sampling and expansion algorithms in solving nonlinear equations.

Conclusion

This topic mainly focuses on the characteristics of nonlinear equations. By extendin g the DDEA algorithm to new solution scenarios, a new data-driven intelligent optimiza tion algorithm, RDDEA-ZPS algorithm, adapted to the solution of nonlinear equations i s proposed.

The topic mainly revolves around the WDDEA-DBC model and its improved version RDDEA-ZPS, and systematically discusses the application of this model in solving nonlin ear equations and key issues and solutions in the optimization process.

First, the WDDEA-DBC model as the baseline of the improved model is introduced and analyzed in detail. By analyzing the WDDEA-DBC algorithm, it demonstrates the in novation of its data processing methods and structure, which can be used to solve nonlin ear equations. After that, this article focuses on analyzing the problems of applying this model to the scenario of nonlinear equations. In view of the problems existing in the orig inal algorithm, such as proxy model selection, data set format, and optimization algorith m limitations, a plan to modify the proxy model structure and training method was propo sed.

After that, this article began to construct an improved RDDEA-ZPS algorithm for th e WDDEA-DBC algorithm problem. In the construction of the proxy model, this paper r everses the input and output of the original RBFNN network and restores the original da ta set, reducing errors caused by introducing norms and using the FPA algorithm. In the process of data generation, based on Latin hypercube sampling and DBC sampling, the zero-point sampling (ZPS) method was proposed, which expanded the data generation method, generated high-value data more accurately, and provided a reliable data basis fo r model establishment and solution. Subsequently, this article discusses in detail the opti mization and weighting process of the RDDEA-ZPS model, as well as its advantages an d improvements compared with the original model, providing new ideas and methods fo r solving nonlinear equations problems.

After proposing a complete improved algorithm framework, this article determined the selection of key parameters in the RDDEA-ZPS algorithm by conducting numerical simulation experiments on examples, and improved the performance and effect of the al gorithm in the subsequent solution process. Finally, the optimal parameters were used to solve the nonlinear equations and the effects were compared. It was found that the impr oved model had higher solution accuracy and obvious effects.

Finally, through ablation experiments, the different architectures of the model were disassembled, further verifying the effectiveness of each part of the model.

The main creative work of this paper is summarized as follows:

1. Introducing algorithms in a new field - data-driven optimization algorithms into the solution of nonlinear equations, broadening the application space of the algorithm and prov iding new ideas for solving nonlinear equations.

2. A new data generation algorithm is proposed, which improves the solution accur acy in the scenario of solving nonlinear equations.

3. In view of the solving characteristics of nonlinear equations, the main framework of the agent model was creatively modified, and the introduced errors were greatly reduced by reversing the input and output and converting the application scenarios of the intelligent optimization algorithm.

4. The new algorithm transforms from single-objective optimization to multi-objective optimization, providing ideas for the DDEA algorithm in multi-objective optimization scenarios.

5. The optimized model greatly simplifies the steps and difficulty of solving nonlinear equations.

Due to time, personal ability and other limitations, the current work still lacks a certain depth. In the future, we will continue to conduct in-depth research in the following aspects:

1. The current optimization algorithm can still only locate one solution. For the situation where there are multiple solutions within the definition domain interval, the algorithm cannot solve it well.

2. The current algorithm can only solve nonlinear equations with the same number of independent variables and equations, and cannot handle cases where the numbers do not match.

3. There is still room for improvement in model performance. In the future, we will continue to improve the optimization algorithm by modifying the model framework and studying and comparing time overhead.

References

[1] GONG W, LIAO Z, Mi X, et al. Nonlinear Equations Solving with Intelligent Optimization Algorithms: A Survey［J］. Complex System Modeling and Simulation, 2021, 1(1): 15-32.

[2] JIN Y C, WANG H D, CHUGH T, GUO D, MIETTINEN K. Data-Driven Evolutionary Optimization: An Overview and Case Studies [J]. IEEE Transactions on Evolutionary Computation, 2019, 23(3): 442-458.

[3] GUO Z L, LIN S K, SUO R Z, ZHANG X M. An Offline Weighted-Bagging Data-Driven Evolutionary Algorithm with Data Generation Based on Clustering［J］. Mathematics, 2023, 11(2): 431.

[4] BROYDEN C G. A class of methods for solving nonlinear simultaneous equation［J］. Mathematics of Computation, 1965: 577-593.

[5] RAMOS H, MONTEIRO M. A new approach based on the newtons method to solve systems of nonlinear equations［J］. Journal of Computational and Applied Mathematics, 2017: 3-13.

[6] Wu Long, Ren Hongmin, Bi Weihong. A review of research on genetic algorithms for solving nonlinear equations [J]. Electronic Technology, 2014 (4): 173-178.

[7] YANG X S. Flower Pollination Algorithm for Global Optimization［J］. Unconventional Computation and Natural Computation, 2012: 240-249.

[8] Li Yinghui, Qiu Wanhua. A method of solving nonlinear equations using artificial neural network [J], Systems Engineering and Electronic Technology, 1997, 19(5): 3.

[9] Zhao Qilin, Zhuo Jiashou. Coupled neural network algorithm for nonlinear equations [J]. Journal of Hohai University (Natural Science Edition), 2000, (5): 38-40.

[10] Zhao Huamin, Chen Kaizhou. Neural network method for solving multivariate nonlinear equations [J]. Journal of Xi'an University of Electronic Science and Technology, 2000, (04): 480-482.

[11] LI G M and ZENG Z Z. A Neural-Network Algorithm for Solving Nonlinear Equation Systems［C］. 2008 International Conference on Computational Intelligence and Security, Suzhou, China, 2008: 20-23.

[12] Sun Yinhui. Iterative neural network algorithm for solving nonlinear equations [J]. Computer Engineering and Applications, 2009, 45(6): 55-56.

[13] Zhao Qinglan, Li Wen, Dong Xiaoli. Improved iterative neural network algorithm for solving nonlinear equations [J]. Modern Electronic Technology, 2013, 36 (08): 20-22.

[14] WANG H D, JIN Y C, SUN C L, DOHERTY J. Offline Data-Driven

Evolutionary Optimization Using Selective Surrogate Ensembles［J］. IEEE Transactions on Evolutionary Computation, 2019, 23(2): 203-216.

［15] LI J Y. Boosting Data-Driven Evolutionary Algorithm with Localized Data Generation［J］. IEEE Transactions on Evolutionary Computation, 2020, 24(5): 923-937.

［16] WANG M. Offline data-driven evolutionary optimization algorithm using k-fold cross [C]. In Advances in Swarm Intelligence, Proceedings of the International Conference on Sensing and Imaging; Springer: Cham, Switerlands, 2022: 305–316.

［17] CAO H Q, NGUYEN H

［18] LIAO Z, GONG W, YAN X, WANG L and HU C. Solving Nonlinear Equations System with Dynamic Repulsion-Based Evolutionary Algorithms［J］. IEEE Transactions on Systems Man and Cybernetics Systems, 2018: 1-12.

Thanks

As I write the last character, I am filled with gratitude for my four years in college.

Thanks to my mentor for giving me the inspiration and inspiration of algorithms, so that I still have a way forward when encountering difficulties. Every communicatio n with him is an opportunity for me to learn and grow. He is a beacon and inspiration on my academic path. Without his careful teaching, this article would not have been c ompleted.

Secondly, I would like to sincerely thank my roommates. They are not only my cl assmates, but also my closest partners and pillars in college life. I see friendship and tol erance in them. They share the joys and sorrows with me, and I benefit a lot from every time we get along with them. Their companionship and understanding made me feel wa rm and secure while studying in a foreign country, and they are a precious asset in my li fe.

In addition, I would also like to thank all the friends I made in college. They walk ed with me and witnessed every detail of my four years in college. When I encounter di fficulties, they are always by my side to give me selfless support and encouragement, a nd they are an indispensable part of my life.

Finally, I want to give a special thanks to Taylor Swift. Her music accompanied me through every important moment in my college years. Her songs were like a clear spring , giving me spiritual nourishment and strength. Her music is not only a part of my life, b ut also my motivation and belief in pursuing my dreams.

Here, I would like to sincerely thank all those who have supported and helped m e. Your love and support have made me what I am today. I will always be grateful an d remember it in my heart.