Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105
Volume **10**, Number **2**, May **2025**

Research on Interpolation Algorithm of NC Machining Based on Optimized RBF Neural Network

Li-Min Song*

Shanxi Institute of Mechanical & Electrical Engineering, Changzhi 046011, P. R. China
slmwxt123@163.com

Sofia Suen

School of Educational Management and Development
Mahasarakham University, Mahasarakham 44150, Thailand
vq2082@163.com

*Corresponding author: Li-Min Song

Received May 8, 2024; revised September 2, 2024; accepted December 11, 2024.

Abstract. *In the context of intelligent manufacturing, CNC machining technology is crucial for the precision manufacturing field. However, traditional interpolation algorithms face many challenges when dealing with complex surfaces, such as large data volume, poor machining accuracy and large feed acceleration. To overcome these limitations, this work proposes a CNC machining interpolation algorithm based on an optimised Radial Basis Function (RBF) neural network, which incorporates the Snake Optimizer (SO) algorithm to improve global search and fast convergence. Firstly, the basic principles of the SO algorithm and the RBF neural network are introduced. Then, a Modified Snake Optimizer (MSO) algorithm is proposed, and the improvements introduced include population initialisation of orthogonal matrices, adversarial learning strategy and centre-of-mass variation operation. Secondly, based on the principle of CNC interpolation, the RBF structure for solving the free curve interpolation problem is designed. The input layer has 4 neuron nodes corresponding to coordinate position, and velocity of each axis. The output layer has 4 nodes corresponding to coordinate position, tangent angle and radius of curvature. MSO is used to optimise the weight selection and network training process of the RBF neural network. The waveform data generated by sine function is used to simulate the actual situation, and the performance of the algorithm is tested accordingly. The results show that the proposed MSO-RBF neural network interpolation algorithm can significantly improve the interpolation accuracy, and the maximum error is less than 1.5×10-5 mm, which meets the machining requirements of complex shaped surface parts, and provides a new interpolation control method for the field of precision machining, which is of great theoretical significance and application value.*
Keywords: CNC machining; Interpolation algorithm; RBF neural network; Snake optimizer algorithm; Intelligent manufacturing

1. **Introduction.** With the rapid development of intelligent manufacturing, CNC machining technology plays an increasingly important role in the field of precision manufacturing [1, 2]. As the key equipment to achieve high-precision and high-efficiency machining, the machining accuracy and efficiency of CNC machine tools depend largely on the performance of interpolation algorithms. Traditional interpolation algorithms have many limitations when dealing with complex surfaces, such as large data storage [3], low

machining accuracy [4], and large feed acceleration [5, 6], which not only affect the machining quality, but also limit the machining capability of CNC machine tools. Therefore, the development of a new algorithm that can improve the interpolation accuracy and speed is of extremely important to promote the progress of CNC machining technology.

In recent years, neural network techniques have become important tools for solving complex optimisation problems. In the field of CNC machining, these techniques are especially suitable for dealing with nonlinear and multivariate problems that are difficult to be solved by traditional algorithms [7, 8]. Radial Basis Function (RBF) neural networks are widely recognised as an effective means of achieving complex surface interpolation due to their excellent performance in function approximation [9, 10]. RBF networks are able to perform accurate nonlinear mapping of the input data through the radial basis function neurons in their hidden layer [11], thus generating interpolated points that are highly compatible with the actual surface. However, RBF neural networks have some limitations in the selection of weights and network training process. The network is prone to fall into local optimal solutions and the training process may take a long time to reach convergence [12]. These problems are particularly prominent when dealing with real-time interpolation tasks in CNC machine tool machining. To overcome these limitations, this study proposes a novel RBF neural network optimisation strategy incorporating the Snake Optimizer (SO) algorithm [13]. The Snake Optimizer algorithm, as a meta-heuristic algorithm that simulates the behaviour of snakes in nature, has been shown to have significant advantages in terms of global search and fast convergence. By simulating the social behaviours of snakes in searching for food, competing for mates and fighting, the SO algorithm is able to effectively explore and exploit the solution space.

The research objective of this thesis is to construct a CNC machining interpolation algorithm based on Snake Optimizer RBF neural network, and to improve the machining accuracy and efficiency of complex parts through the improved algorithm. The thesis firstly introduces the basic principles of the Snake Optimizer algorithm and RBF neural network, then proposes the improvement strategy of the algorithm, and verifies the effectiveness of the proposed algorithm through simulation experiments. The research results are expected to provide a new interpolation control method for the field of ultra-precision machining, which has important theoretical significance and application value.

1.1. **Related work.** CNC machining technology is increasingly used in modern manufacturing, especially in the field of precision machining. However, traditional CNC interpolation algorithms face many challenges when dealing with complex surfaces, such as large amount of data, low machining accuracy and high feed acceleration. These problems limit the machining capability and efficiency of CNC machine tools [14]. Du et al. [15] proposed a Non-Uniform Rational B-Splines (NURBS) curve interpolation algorithm with real-time monitoring of interpolation error and feed acceleration, and demonstrated high-speed and high-precision characteristics in CNC machining. Chu et al. [16] reviewed the related methods and advances in tool path planning and speed interpolation in CNC machining, noting that although existing methods simplify computation and improve efficiency, there remains room for overall optimisation and higher accuracy. Li et al. [17] proposed a method for predicting contouring error in multi-axis machining based on interpolated data, highlighting the importance of contour error in high-precision CNC machining and proposing a data-based prediction method, though its robustness under complex conditions requires further verification.

Recently, researchers have applied intelligent optimisation algorithms and neural network techniques to CNC machining. Er et al. [18] explored an image recognition method based on RBF neural networks for nonlinear problems and large-scale computation, but its

application in interpolation algorithms remains under-explored. Manrique et al. [19] proposed an evolutionary system for automatic construction and tuning of RBF networks, optimising network structure via an evolutionary algorithm to improve generalisation. Gillebaart et al. [20] investigated adaptive radial basis functions in neural networks, extending RBF applications to varied input shapes, yet their effectiveness in CNC interpolation demands further study. Jiao and Jin [21] introduced a GA-RBF neural network algorithm using a genetic algorithm to optimise weights and structure, showing enhanced efficiency and accuracy, and offering new possibilities for CNC interpolation research.

1.2. **Motivation and contribution.** Analysing the above studies reveals the impact of traditional RBF parameters (e.g. weights and biases) on prediction accuracy; traditional optimisation methods may fail to reach global optima, limiting network performance. The Snake Optimizer algorithm, as a global optimisation method, can effectively search the parameter space to optimise hidden node centres and network parameters, thus improving prediction accuracy and performance. Therefore, to enhance CNC machining accuracy, this paper proposes:

(1) A Modified Snake Optimizer (MSO) algorithm to address insufficient population diversity and poor convergence of the traditional SO. We introduce an orthogonal matrix initialisation to enhance population uniformity and diversity for better global search, and employ an adversarial learning strategy based on variational centre of mass to increase population diversity and avoid local optima.

(2) A CNC interpolation model based on RBF neural network optimised by MSO, adjusting centre positions, scale parameters, weights, and biases to improve accuracy and efficiency. The model considers tool path position and axis velocities to further refine interpolation through multi-information control.

2. Analysis of relevant principles.

2.1. **Snake optimizer algorithm.** The Snake Optimizer algorithm is an emerging metaheuristic inspired by snake mating and feeding behaviours [22, 23]. It simulates snake population social behaviours in search, competition, and reproduction. In SO, each solution is treated as a snake operating in the solution space. Snakes undergo three main phases: exploration, exploitation, and fighting/mating, adapting behaviours based on environmental conditions [24].

2.1.1. *Search phase.* In the search phase, the snakes randomly search for food in the environment, which is modelled as follows:

$$X_i(t+1) = X_{\text{rand}} + \Delta X \tag{1}$$

where $X_i(t)$ denotes the position of the i-th snake at time t, X_{rand} is the randomly chosen position of another snake, and ΔX is a random displacement, usually determined by Equation (2).

$$\Delta X = c \cdot (X_{\max} - X_{\min}) \cdot rand() \tag{2}$$

where c is a positive small constant; $X_{\max}$ and $X_{\min}$ are the maximum and minimum values of the solution space, respectively; $rand()$ is a random number between 0 and 1.

2.1.2. *Exploitation phase.* When the snake discovers food, it enters the exploitation phase, when the behaviour of the snake shifts to exploiting the resources near its current position [25]. The update method for this phase is shown as follow:

$$X_i(t+1) = X_{\text{food}} - \Delta X \tag{3}$$

where X_{food} represents the location of the food, usually the currently found optimal solution.

2.1.3. *Fighting/mating phase.* During the fighting/mating phase, snakes fight or mate to produce a new generation of snakes depending on ambient temperature and food availability. In the fighting mode, male snakes compete for the opportunity to mate with females. The mating process can be simulated by

$$X_i(t+1) = X_{\text{best}} + \alpha \cdot (X_i(t) - X_{\text{worst}}) \tag{4}$$

where X_{best} is the current optimal female snake position, X_{worst} is the worst male snake position, and α is mating strength, a positive random number.

As a newly proposed algorithm, the snake optimizer algorithm may show better performance on some specific optimisation problems, especially on the types of problems it is designed to specifically address. By simulating the natural behaviour of snakes, snake optimizer algorithms may exhibit better robustness when faced with complex optimisation problems, especially in areas such as path planning and robot navigation.

2.2. **Overview of RBF Neural Networks.** RBF neural network is a feed-forward neural network which is widely used in function approximation, pattern recognition and classification problems. RBF networks are valued for their local response properties and good generalisation ability. The typical structure of an RBF network is shown in Figure 1.

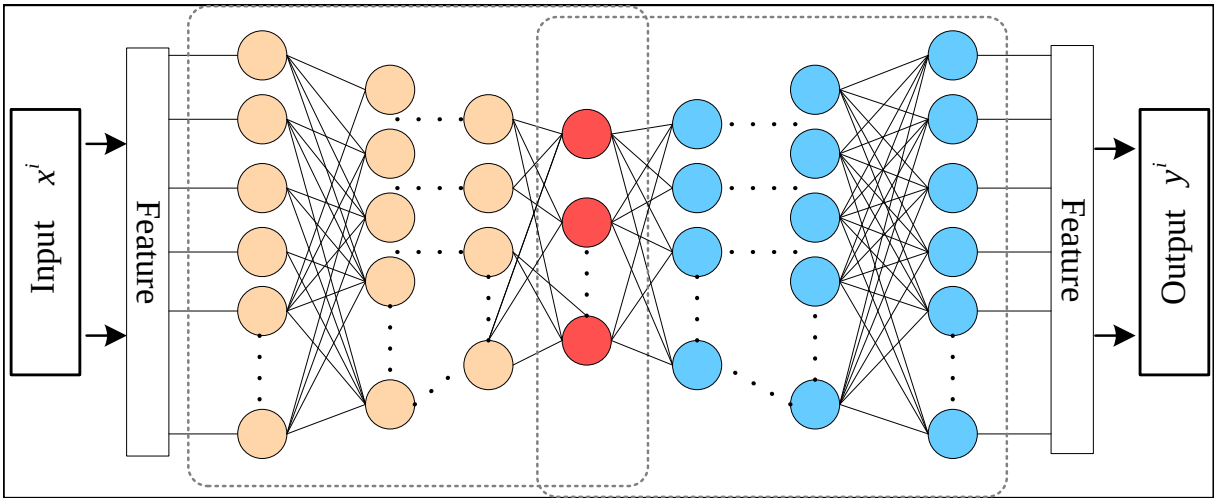

Figure 1. Typical structure of an RBF network

The key feature of an RBF network is the activation function of the neurons in the hidden layer, which is usually a Gaussian function used as the radial basis function. For a given input vector $\mathbf{x} \in \mathbb{R}^n$, the output of the hidden layer neuron $\phi(\mathbf{x})$ can be expressed as

$$\phi(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{2\sigma_i^2}\right) \tag{5}$$

where $\mathbf{c}_i$ is the centre of the neuron and σ_i is the scale parameter, also known as the scale factor, that affects the width of the function.

The output of the RBF network is a linear combination of the outputs of the implicit layers and can be expressed as:

$$\mathbf{y} = \mathbf{W}^T \phi(\mathbf{x}) + \mathbf{b} \tag{6}$$

where **W** is the weight matrix from the implicit layer to the output layer and **b** is the bias term.

Receive input data over the network and calculate the activation value of the implicit layer. Calculate the value of the output layer based on the activation value of the implicit layer. Calculate the error between the network prediction **y** and the target value **t**. A commonly used error function is the mean square error (MSE), defined as:

$$E = \frac{1}{2}\sum_{j=1}^{m}(y_j - t_j)^2 \tag{7}$$

where y_j is the network prediction output and t_j is the target value.

The weights **W** and bias **b** are adjusted according to the error. The weight update formula is based on gradient descent and can be expressed as.

$$W_{ij}^{\text{new}} = W_{ij}^{\text{old}} - \eta\frac{\partial E}{\partial W_{ij}} \tag{8}$$

where η is the learning rate.

RBF neural networks have been successfully applied in many fields due to their simple structure, fast training and good generalisation. However, the selection of appropriate centre and scale parameters is crucial for network performance.

3. **Modified snake optimizer.**

3.1. **Improvement of initialised population.** In SO algorithms, the initialisation of the population is crucial for the global search ability and convergence performance of the algorithm. The traditional random initialisation method may lead to insufficient diversity of the population, thus affecting the search efficiency of the algorithm. To resolve this matter, this paper proposes an orthogonal matrix-based population initialisation method to enhance the uniformity and diversity of the population.

The orthogonal matrix initialisation method [26] uses the properties of orthogonal tables to generate the initial population. Orthogonal tables can provide uniformly distributed combinations of positions, which helps the algorithm to explore potential optimal solutions uniformly in the solution space. The specific steps are as follows:

(1) Select Orthogonal Table: Based on the dimensions of the problem and the population size, select a suitable orthogonal table $L_{Q(N-1)}$, where Q is the number of rows in the orthogonal table and $N-1$ is the number of columns.

$$L_{Q(N-1)} = \begin{bmatrix} l_{11} & l_{12} & \cdots & l_{1(N-1)} \\ l_{21} & l_{22} & \cdots & l_{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ l_{Q1} & l_{Q2} & \cdots & l_{Q(N-1)} \end{bmatrix} \tag{9}$$

(2) Generate the initial solution: use each row of the orthogonal table to generate the individuals in the snake population. For each row j, calculate the value of the j-th dimension of the initial solution x_i.

$$x_{ij} = l_{ij} \cdot (x_{\max} - x_{\min}) + x_{\min} \tag{10}$$

where l_{ij} is the value of the element in the orthogonal table, $x_{\max}$ and $x_{\min}$ is the maximum and minimum value of the solution space.

(3) Adjusting the initial solutions: based on the specific constraints of the problem, the generated initial solutions are adjusted to ensure that they satisfy all the constraints of

the problem. The population diversity measure is shown below:

$$D = \frac{1}{Q(N-1)} \sum_{i=1}^{Q} \sum_{j=1}^{N-1} d_{ij} \tag{11}$$

where d_{ij} is the difference between the value of individual i on dimension j.

Individual fitness is calculated as shown below:

$$f(x_i) = \sum_{k=1}^{m} \left(x_{ik} - x_{\text{opt}}^{k}\right)^2 \tag{12}$$

where $f(x_i)$ is the fitness of individual i, x_{ik} is the value of individual i in the k-th dimension, x_{opt}^{k} is the optimal value in the k-th dimension, and m is the dimension of the problem.

With the improved initialisation population strategy, the algorithm is able to explore the solution space with higher efficiency and better quality, laying a solid foundation for the subsequent optimisation process.

3.2. **Oppositional learning strategy based on variational centre of mass.** In the Modified Snake Optimizer (MSO), an adversarial learning strategy based on the variational centre of mass is introduced in order to solve the problem of slow convergence and the tendency to fall into local optimums in the late iteration.

Opposition-based learning (OBL) is a search strategy [27, 28] that increases the diversity of a population by calculating the opposition point of each individual in the current population. Opposite points are points in the solution space that are opposite to the position of the current individual. By evaluating the fitness of the original individual and its opposing point, a better individual can be selected for subsequent iterations. The centre of mass (centroid) is the weighted average of the positions of all individuals in the population. In MSO algorithm, the concept of centre of mass is used to generate opposing points to enhance the diversity of the population.

Assuming that the population $P = \{X_1, X_2, \ldots, X_N\}$ consists of N individuals in a D-dimensional search space, the centre of mass M of the population can be defined as:

$$M = \frac{X_1 + \cdots + X_N}{N} \tag{13}$$

Using the centre of mass of the current population as a reference point, calculate the opposite individual for each individual in the population:

$$\overline{X_i} = 2 \times M - X_i \tag{14}$$

When SO algorithm is developed locally in the late iteration, individuals will gradually gather around the optimal individuals, which will lead to the inconspicuous change of the centre of mass position during the iteration process, and thus the diversity of the population cannot be maintained. Therefore, the variation operation of the centre of mass is implemented to avoid the aggregation of individuals in the population, so as to improve the diversity of the population. The variation operation of the centre of mass is shown in Equation (15).

$$\overline{M} = M \times (1 + \gamma) \tag{15}$$

where γ denotes the mutation operator, and $\overline{M}$ is the mutated centre of mass, i.e. the mutated centre of mass.

The variational operation of the centre of mass is finally represented by Equation (16):

$$\overline{M} = M \times \left(1 + L(\beta)\right) \tag{16}$$

Since there is no guarantee that the fitness value of the post-mutation centre of mass is better than that of the pre-mutation centre of mass, a greedy strategy is used to compare the fitness values of the pre and post-mutation centres of mass. When the fitness value of the post-mutation centre of mass is better, replace the pre-mutation centre of mass M with the post-mutation centre of mass $\overline{M}$, and use Equation (12) to compute the opposing individuals of each individual; otherwise, use Equation (12) to compute the opposing individuals of each individual directly.

4. **Neural network models for CNC interpolation.**

4.1. **Principle of CNC interpolation.**

Numerical Control Interpolation (NCI) is a core technology in CNC machining, which involves calculating the trajectory of each axis of the machine tool based on given input data (e.g., tool path, speed, etc.) [30]. The purpose of interpolation is to ensure that the tool is able to move precisely in accordance with a predetermined contour, thus achieving high precision machining.

In CNCs, interpolation is usually divided into two categories: linear interpolation and curvilinear interpolation. Linear interpolation refers to the movement of the machine along a straight line path, whereas curvilinear interpolation involves the movement of the machine along a non-straight line (e.g., arc, parabola, etc.) path.

For a straight line path between two points, interpolation can be achieved by simple linear equations:

$$P(t) = (1-t)P_0 + tP_1 \tag{17}$$

where P_0 and P_1 are the position vectors of the start and end points, respectively, and t is an interpolation parameter that takes values from 0 to 1.

For circular arc paths, interpolation needs to be calculated from the parametric equations of the circle.

$$x(t) = x_c + r \cdot \cos(\theta t + \phi) \tag{18}$$

$$y(t) = y_c + r \cdot \sin(\theta t + \phi) \tag{19}$$

where (x_c, y_c) is the coordinates of the centre of the circle, r is the radius of the circle, θ is the ratio of the angle of the centre of the circle corresponding to the point on the circumference to the radius, and ϕ is the phase angle of the start of the arc.

For complex curves, such as free curves, interpolation is often performed using spline curves. NURBS is a commonly used tool and its mathematical model is shown as follow:

$$P(t) = \sum_{i=0}^{n} R_i(t) \cdot P_i \tag{20}$$

where $R_i(t)$ is the basis function, P_i is the control point, n is the number of control points, and t is a parameter.

In the actual CNC system, interpolation is usually done by the Numerical Control Unit (NCU), which calculates the movement instructions for each axis of the machine in real time according to the input programme code and the current machine status. By optimising the interpolation algorithm, the machining efficiency and quality of the machine can be improved to meet the machining requirements of complex parts.

4.2. **Design of RBF structure for solving free curve interpolation problem.**

CNC machine tools in the processing of parts, the feed rate of the actuator is given, but the direction of the feed speed is constantly changing, so that the speed in each axis is constantly changing, in order to complete the interpolation, not only to calculate the position information of the interpolation point must be calculated also must calculate the

speed of each axis. In order to improve the interpolation accuracy, multiple interpolation information can be introduced when the network model is built.

In this paper, an RBF network model is established as an example of solving free curve interpolation. According to the analysis, the input layer has 4 neuron nodes corresponding to the coordinate positions x_{i-1} and y_{i-1}, the velocity of each axis $v(x_{i-1})$ and $v(y_{i-1})$. The output layer has 4 nodes corresponding to coordinate positions x_i and y_i, tangent angle θ_i, and radius of curvature R_i. The control by more than one information can improve the interpolation accuracy. The number of nodes in the hidden layer is chosen as 9 according to Kolmogomv's theorem ($2m+1, m$ is the number of input nodes).

In RBF neural networks, the node parameters of the hidden layer mainly include the centre c_i and width σ_i of the radial basis function, which have a decisive influence on the prediction performance of the network. When initialising the network, h training samples are randomly selected as the initial clustering centre $c_i^{(u)}$, where u denotes the number of iterations.

For each input sample x_j, compute its Euclidean distance from each cluster centre $c_i^{(u)}$.

$$d_{ij}^{(u)} = \sqrt{\sum_{k=1}^{n} \left(x_{jk} - c_{ik}^{(u)} \right)^2} \tag{21}$$

where x_{jk} is the kth feature of the input sample x_j and n is the total number of input features.

Based on the distance calculation, the position of each cluster centre is adjusted by Equation (19):

$$c_i^{(u+1)} = \frac{1}{n_i^{(u)}} \sum_{x_j \in \Theta_i^{(u)}} x_j \tag{22}$$

where $\Theta_i^{(u)}$ denotes the set of all input samples assigned to the clustering centre c_i in the uth iteration, and $n_i^{(u)}$ is the number of samples in that set.

Use the maximum distance $c_{\max}$ between cluster centres to determine the width of the radial basis function σ_i.

$$\sigma_i = \frac{c_{\max}}{\sqrt{h}} \tag{23}$$

where h is the total number of clustering centres.

After the centre and width are determined, the least squares method is used to compute the weights w_{lj} between the implicit layer and the output layer.

$$w_{lj} = \left(R^T R \right)^{-1} R^T y_j \tag{24}$$

where R is the implicit layer output matrix and y_j is the vector of the jth output sample.

4.3. **Improvement of RBF neural network using MSO algorithm.** The MSO algorithm optimises the weight selection and training process of the RBF network by introducing mechanisms such as orthogonal matrix initialisation and adversarial learning strategies.

The MSO algorithm optimises the search by modelling the predatory behaviour and mating strategy of snakes. In the context of an RBF network, the MSO algorithm can be used to optimise the centroid position c_i and the scale parameter σ_i of the neurons in the hidden layer, as well as the weights W and biases b of the network.

The centroid position c_i and scale parameter σ_i of the RBF network are first initialised using an orthogonal matrix.

$$c_i = c_{\min} + (c_{\max} - c_{\min}) \cdot l_{ij} \tag{25}$$

$$\sigma_i = \sigma_{\min} + (\sigma_{\max} - \sigma_{\min}) \cdot l_{ij} \tag{26}$$

where l_{ij} is the element in the orthogonal table, $c_{\min}$ is the minimum of the centre position, and $c_{\max}$ is the maximum of the centre position, $\sigma_{\min}$ is the minimum of the scale parameter, and $\sigma_{\max}$ is the maximum of the scale parameter.

The network parameters are updated by calculating the opposing learning points for each parameter and comparing their fitness.

$$c_i' = c_i + \alpha \cdot (c_{\text{rand}} - c_i) \tag{27}$$

$$\sigma_i' = \sigma_i + \alpha \cdot (\sigma_{\text{rand}} - \sigma_i) \tag{28}$$

where c_{rand} and σ_{rand} are randomly selected individual parameters and α is the learning rate.

Use the fitness evaluation mechanism in the MSO algorithm to select better network parameters.

$$F(c_i, \sigma_i) = \frac{1}{2} \sum_{j=1}^{N} (y_j - \hat{y}_j)^2 \tag{29}$$

where y_j is the target value, $\hat{y}_j$ is the network prediction, and N is the number of samples.

Update the weights and biases of the network according to the search strategy of the MSO algorithm.

$$W^{\text{new}} = W^{\text{old}} + \beta \cdot \left(W_{\text{best}} - W^{\text{old}}\right) \tag{30}$$

$$b^{\text{new}} = b^{\text{old}} + \beta \cdot \left(b_{\text{best}} - b^{\text{old}}\right) \tag{31}$$

where W_{best} and b_{best} are the weights and biases of the most adapted individual and β is the update step.

Through the above steps, the MSO algorithm is able to improve the parameter selection of the RBF neural network, thus improving the prediction accuracy and generalisation ability of the network. The pseudo-code of the proposed MSO-RBF is shown in Algorithm 1.

5. **Experiments and analysis of results.**

5.1. **Experimental environment.**

In this section, we present the experimental environment used to test and validate the proposed MSO-RBF for CNC interpolation applications. The setup of the experimental environment is crucial to ensure the reproducibility of the experiments and the reliability of the results. The hardware and software configuration of the experimental environment is shown in Table 1. This hardware configuration ensures that the algorithm has sufficient computational power and data storage space during training and testing. To implement and test the MSO-RBF neural network, we used the MATLAB neural network toolbox and the Python NumPy library.

Table 1. Main parameters of the experimental environment

Configure	**Descriptions**
CPU	Intel Core i7-9700K, 3.6GHz, 8-core
RAM	16GB DDR4
Stockpile	512GB SSD
GPU	NVIDIA GeForce RTX 2070
Operating system	Windows 10 Professional, 64-bit
Numerical calculation software	MATLAB R2020b
Programming language	Python 3.8.5

Algorithm 1 Pseudo-code for MSO-RBF neural network optimization

Input: Training data set D, initial parameters $\Omega = (c_i, \sigma_i, W, b)$
Output: Optimized RBF network parameters Ω^*

```
Initialize the population of snakes with random positions based on Ω
while termination criterion not met do
    Sort the snake population by fitness evaluated on D
    for each snake i in the population do
        Calculate distance between snake i and the best snake
        if distance > threshold then
            Exploration phase: search for food (optimize in search space)
            Apply opposition-based learning to generate a new position for snake i
            if new position has better fitness then
                Update position of snake i
            end if
        else
            Exploitation phase: move towards best solution found so far
            Adjust position of snake i based on best snake's position and distance
        end if
    end for
    Perform diversity maintenance for next iteration
    Update RBF parameters Ω based on sorted snake population:
    for each parameter p in Ω do
        Select best value p_best from top snakes
        Update p using p_best and MSO-specific rule
    end for
    Check convergence criteria (e.g., max iterations or fitness threshold)
end while
Ω* ← Ω                                   ▷ Optimized RBF network parameters
return Ω*
```

5.2. **Simulation test parameters.** In order to test the above CNC interpolation MSO-RBF neural network model, the sinusoidal curve $y = \sin(x)$, $x \in [-2\pi, 2\pi]$ is taken as an example, and the simulation verification is carried out on the MATLAB software. As shown in Figure 2, on the sinusoidal curve $y = \sin(x)$ randomly selected points as the sample data of the network, the sample data should be selected to fully reflect the shape of the curve features, such as the starting point of the curve coordinates, the end of the curve of the previous point of the coordinates of the curve curvature of the curvature of the curve has undergone a large change in some of the coordinates of the point. Table 2 shows the selected sample data. The population size is 20, the number of neurons in the hidden layer of the RBF network is 9. The initial scale parameter σ_i is 0.1, the learning rate η is 0.1, and the maximum number of iterations is 200. The threshold for judging the exploratory or developmental stage in the MSO algorithm is 0.25.

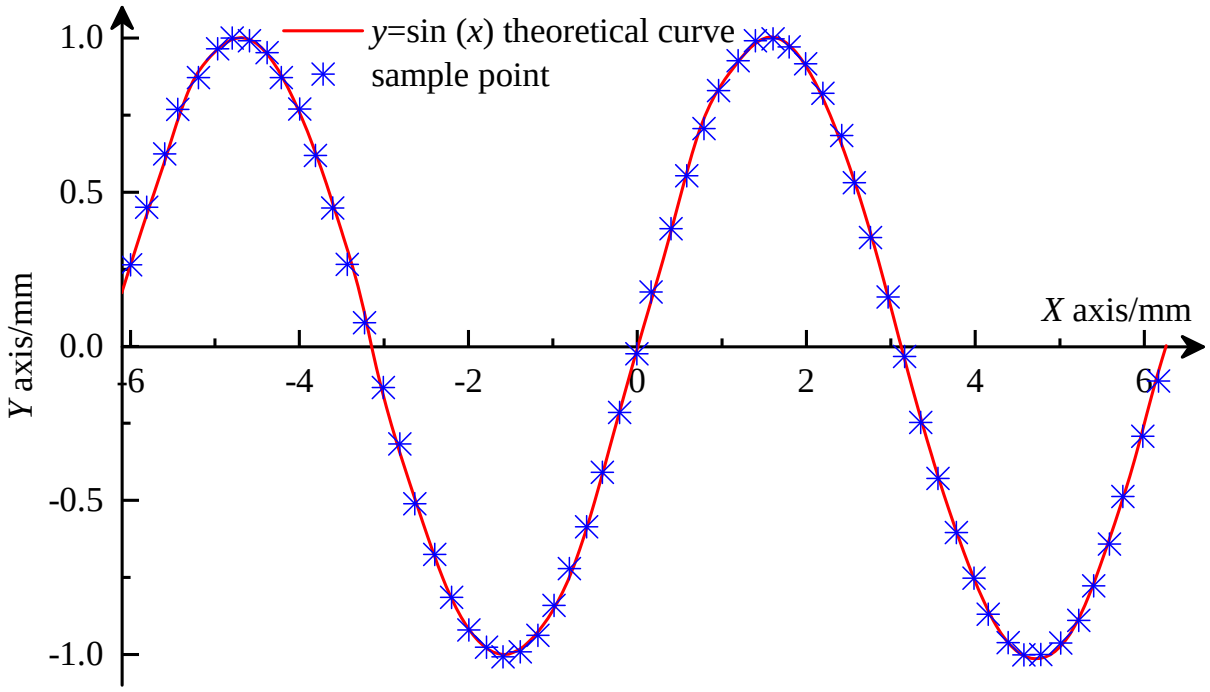

Figure 2. Random sample data

Table 2. Partial sample data

No.	Sample data		Desired output	
	X-axis/mm	Y-axis/mm	X-axis/mm	Y-axis/mm
1	-6.08319	0.198669	-6.0536	0.113671
2	-5.88319	0.389418	-5.86543	0.301186
3	-5.68319	0.564642	-5.67027	0.475574
4	-5.48319	0.717356	-5.47275	0.627963
5	-5.28319	0.841471	-5.27416	0.751924
6	-5.08319	0.932039	-5.07497	0.842415
...	-4.88319	0.98545	-4.87538	0.895789
46	-4.68319	0.999574	-4.67547	0.909905
47	-4.48319	0.973848	-4.47526	0.884197
48	-4.28319	0.909297	-4.27471	0.819698
49	5.916815	-0.35823	5.935747	-0.27024
50	6.116815	-0.16563	6.149906	-0.08191

5.3. **Contour error analysis.** Figure 3 shows the network output of the MSO-RBF neural network model. It can be seen that the predicted points of the neural network interpolation model output basically coincide with the real points. The predicted points of the network can be used as both CNC interpolation tool trajectory points. To make the model prediction results more intuitive, the fitting error curve is given in Figure 4.

The simulation results show that the maximum error of MSO-RBF neural network model interpolation is less than 1.5×10^{-5} mm, which indicates that the method can perform high-precision interpolation of complex profiles with versatility, and therefore can meet the machining requirements of ultra-precision complex profile parts.

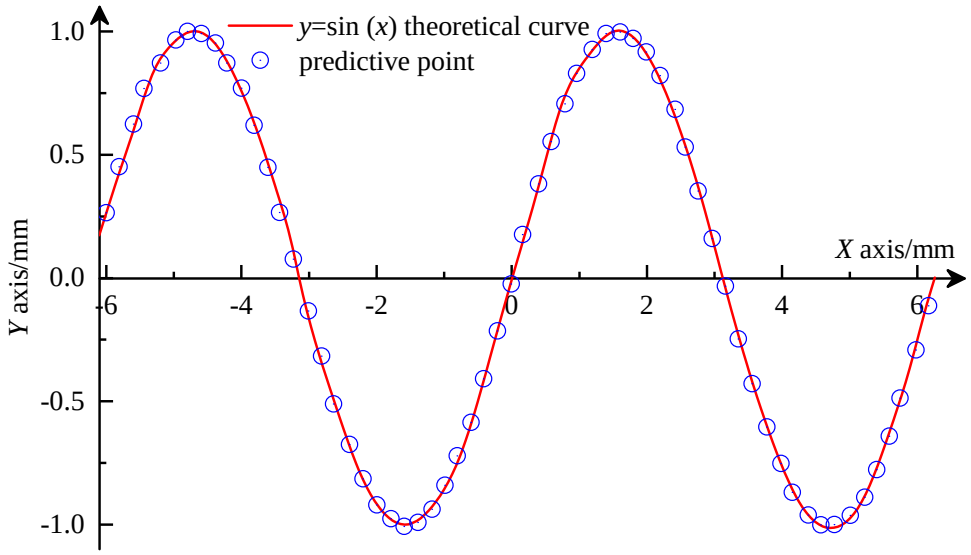

Figure 3. Comparison of interpolated curves with original curves

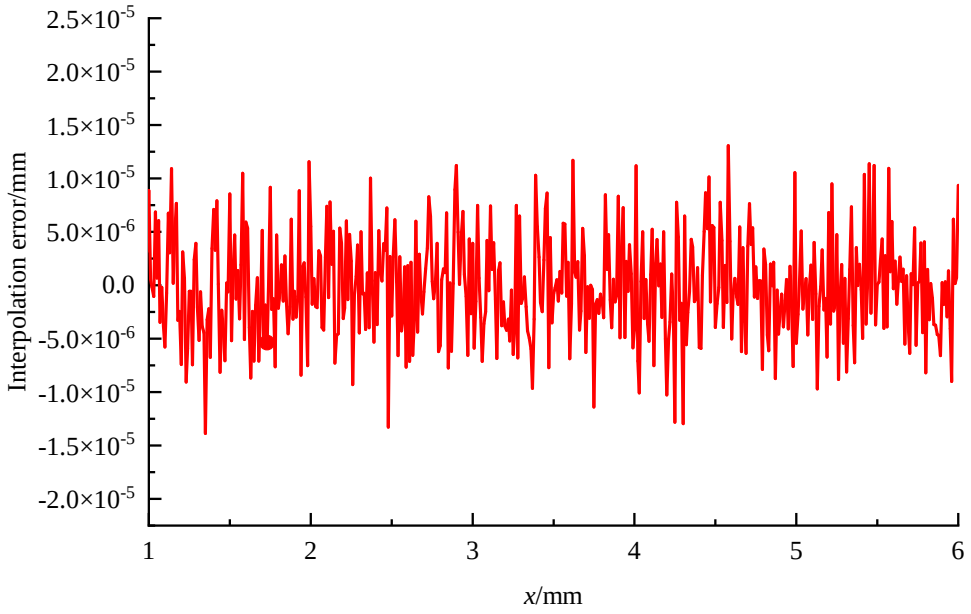

Figure 4. Interpolation Errors

6. **Conclusion.** In this paper, a novel optimization algorithm for MSO-RBF neural networks is proposed. The MSO algorithm effectively enhances the performance of RBF neural networks in the process of weights selection and network training by simulating the behaviour of snakes in nature and introducing orthogonal matrix initialization and adversarial learning strategies. This innovation significantly improves the global search capability and fast convergence of the network, and solves the problem that RBF neural networks are prone to fall into local optimums and long training time. The prediction performance of the network is optimised by using the least squares method to compute the weights between the implicit layer and the output layer. A highly accurate interpolation algorithm for CNC machining is implemented by the proposed MSO-RBF neural network model. The algorithm is able to accurately map the input data to the actual surface and generate interpolation points that highly match the actual surface, thus significantly

improving the machining accuracy and efficiency of complex parts. Simulation experiments show that the maximum error of the interpolation algorithm is less than 1.5×10-5 mm, which meets the machining requirements of ultra-precision complex shaped parts and provides a new interpolation control method for ultra-precision machining. Although the simulation experiment verifies the effectiveness of the algorithm, more complex and variable situations may be encountered in actual industrial applications. Follow-up studies need to be tested on real CNC machine tools to verify the robustness and adaptability of the algorithm.

Acknowledgment. This work is supported by the·2021 Annual Project·of the 14th Five-Year Plan of·Educational·Science in·Shanxi Province (No. GH-21238).

REFERENCES

[1] X. W. Xu and S. T. Newman, "Making CNC machine tools more open, interoperable and intelligent—a review of the technologies," *Computers in Industry*, vol. 57, no. 2, pp. 141–152, 2006.

[2] S.-H. Suh and S.-U. Cheon, "A framework for an intelligent CNC and data model," *The International Journal of Advanced Manufacturing Technology*, vol. 19, pp. 727–735, 2002.

[3] C.-H. Yeung, Y. Altintas, and K. Erkorkmaz, "Virtual CNC system. Part I. system architecture," *International Journal of Machine Tools and Manufacture*, vol. 46, no. 10, pp. 1107–1123, 2006.

[4] A. Keller, A. Kamath, and U. Perera, "Reliability analysis of CNC machine tools," *Reliability Engineering*, vol. 3, no. 6, pp. 449–473, 1982.

[5] L. Wang, P. Orban, A. Cunningham, and S. Lang, "Remote real-time CNC machining for web-based manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 20, no. 6, pp. 563–571, 2004.

[6] S. T. Newman et al., "Strategic advantages of interoperability for global manufacturing using CNC technology," *Robotics and Computer-Integrated Manufacturing*, vol. 24, no. 6, pp. 699–708, 2008.

[7] S. T. Newman, A. Nassehi, R. Imani-Asrai, and V. Dhokia, "Energy efficient process planning for CNC machining," *CIRP Journal of Manufacturing Science and Technology*, vol. 5, no. 2, pp. 127–136, 2012.

[8] Z.-Y. Jia, J.-W. Ma, D.-N. Song, F.-J. Wang, and W. Liu, "A review of contouring-error reduction method in multi-axis CNC machining," *International Journal of Machine Tools and Manufacture*, vol. 125, pp. 34–54, 2018.

[9] M. J. Er, S. Wu, J. Lu, and H. L. Toh, "Face recognition with radial basis function (RBF) neural networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 3, pp. 697–710, 2002.

[10] Q. Jiang, L. Zhu, C. Shu, and V. Sekar, "An efficient multilayer RBF neural network and its application to regression problems," *Neural Computing and Applications*, pp. 1–18, 2022.

[11] Y. Zhou, X. Zhang, and F. Ding, "Hierarchical estimation approach for RBF-AR models with regression weights based on the increasing data length," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 12, pp. 3597–3601, 2021.

[12] J. Wang, S. Zhang, and X. Hu, "A fault diagnosis method for lithium-ion battery packs using improved RBF neural network," *Frontiers in Energy Research*, vol. 9, p. 702139, 2021.

[13] F. A. Hashim and A. G. Hussien, "Snake Optimizer: A novel meta-heuristic optimization algorithm," *Knowledge-Based Systems*, vol. 242, p. 108320, 2022.

[14] W. Wang, Q. Guo, Z. Yang, Y. Jiang, and J. Xu, "A state-of-the-art review on robotic milling of complex parts with high efficiency and precision," *Robotics and Computer-Integrated Manufacturing*, vol. 79, p. 102436, 2023.

[15] D. Du, Y. Liu, C. Yan, and C. Li, "An accurate adaptive parametric curve interpolator for NURBS curve interpolation," *The International Journal of Advanced Manufacturing Technology*, vol. 32, pp. 999–1008, 2007.

[16] C.-H. Chu, P.-H. Wu, and W.-T. Lei, "Tool path planning for 5-axis flank milling of ruled surfaces considering CNC linear interpolation," *Journal of Intelligent Manufacturing*, vol. 23, pp. 471–480, 2012.

[17] X. Li, H. Zhao, X. Zhao, and H. Ding, "Interpolation-based contour error estimation and component-based contouring control for five-axis CNC machine tools," *Science China Technological Sciences*, vol. 61, pp. 1666–1678, 2018.

[18] M. J. Er, W. Chen, and S. Wu, "High-speed face recognition based on discrete cosine transform and RBF neural networks," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 679–691, 2005.

[19] D. Manrique, J. Ríos, and A. Rodríguez-Patón, "Evolutionary system for automatically constructing and adapting radial basis function networks," *Neurocomputing*, vol. 69, no. 16-18, pp. 2268–2283, 2006.
[20] T. Gillebaart, D. Blom, A. Van Zuijlen, and H. Bijl, "Adaptive radial basis function mesh deformation using data reduction," *Journal of Computational Physics*, vol. 321, pp. 997–1025, 2016.
[21] H. Jiao and H. Jin, "Developing a visual prediction program for residual stress in girth butt welds using GA-RBF neural network," *The International Journal of Advanced Manufacturing Technology*, pp. 1–14, 2024.
[22] T.-Y. Wu et al., "A Spectral Convolutional Neural Network Model Based on Adaptive Fick's Law for Hyperspectral Image Classification," *Computers, Materials & Continua*, vol. 79, no. 1, pp. 19–46, 2024.
[23] T.-Y. Wu, A. Shao, and J.-S. Pan, "CTOA: Toward a Chaotic-Based Tumbleweed Optimization Algorithm," *Mathematics*, vol. 11, no. 10, p. 2339, 2023.
[24] T.-Y. Wu, H. Li, and S.-C. Chu, "CPPE: An Improved Phasmatodea Population Evolution Algorithm with Chaotic Maps," *Mathematics*, vol. 11, no. 9, p. 1977, 2023.
[25] R. Yousri, M. Elbayoumi, A. Soltan, and M. S. Darweesh, "A power-aware task scheduler for energy harvesting-based wearable biomedical systems using snake optimizer," *Analog Integrated Circuits and Signal Processing*, vol. 115, no. 2, pp. 183–194, 2023.
[26] R. Dudeja, M. Bakhshizadeh, J. Ma, and A. Maleki, "Analysis of spectral methods for phase retrieval with random orthogonal matrices," *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 5182–5203, 2020.
[27] A. A. Ewees, M. Abd Elaziz, and D. Oliva, "A new multi-objective optimization algorithm combined with opposition-based learning," *Expert Systems with Applications*, vol. 165, p. 113844, 2021.
[28] X. Yu, W. Xu, and C. Li, "Opposition-based learning grey wolf optimizer for global optimization," *Knowledge-Based Systems*, vol. 226, p. 107139, 2021.
[29] S. Ji, L. Lei, J. Zhao, X. Lu, and H. Gao, "An adaptive real-time NURBS curve interpolation for 4-axis polishing machine tool," *Robotics and Computer-Integrated Manufacturing*, vol. 67, p. 102025, 2021.
[30] B. Li, H. Zhang, P. Ye, and J. Wang, "Trajectory smoothing method using reinforcement learning for computer numerical control machine tools," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101847, 2020.