processes

MDPI

Article

A Moving Window Double Locally Weighted Extreme Learning Machine on an Improved Sparrow Searching Algorithm and Its Case Study on a Hematite Grinding Process

Huating Liu, Jiayang Dai * and Xingyu Chen

Guangxi Key Laboratory of Intelligent Control and Maintenance of Power Equipment, School of Electrical Engineering, Guangxi University, Nanning 530004, China
* Correspondence: daijiayang@gxu.edu.cn; Tel.: +86-185-7439-5495

Abstract: In this paper, a double locally weighted extreme learning machine model based on a moving window is developed to realize process modeling. To improve model performances, an improved sparrow-searching algorithm is proposed to optimize the parameters of the proposed model. The effectiveness of the proposed model and algorithm are verified by data from a hematite grinding process. The experimental results show that the proposed algorithm can significantly improve the global search ability and convergence speed in the parameter optimization of the proposed model. The proposed model can correctly estimate the grinding particle size which is expected to be applied to other complex industries.

Keywords: extreme learning machine; double locally weighted model; sparrow searching algorithm; moving window; hematite grinding process

Citation: Liu, H.; Dai, J.; Chen, X. A Moving Window Double Locally Weighted Extreme Learning Machine on an Improved Sparrow Searching Algorithm and Its Case Study on a Hematite Grinding Process. *Processes* **2023**, *11*, 169. https://doi.org/10.3390/pr11010169

Academic Editors: Guoqing Zhang, Zejia Zhao and Wai Sze YIP

Received: 25 November 2022
Revised: 29 December 2022
Accepted: 31 December 2022
Published: 5 January 2023

1. Introduction

Iron is the most widely used metal, and hematite is one of the most common raw materials for iron smelting. In hematite iron-smelting, the grinding process is the most critical production unit, as it connects the previous crushing and the following separation operations. Thus, useful minerals can be separated from gangue and different minerals are dissociated to offer raw material for the subsequent beneficiation works [1,2]. Moreover, the grinding particle size in the grinding process is a very important parameter that is directly related to the grade of concentrated ore and metal recovery rate [3–5]. Thus, accurate and real-time particle size information is the key to improve the grinding efficiency and product quality. The existing instruments for measuring the grinding particle size are expensive, complicated in structure, prone to failure and time-consuming [6,7]. At the same time, the strong magnetic particles in hematite have the phenomenon of "Magnetic agglomeration" [8]. Therefore, it is difficult for the instruments to measure the grinding particle size online [9]. In addition, the grinding process has the characteristics of a complex mechanism, non-linearity and time variance, making it difficult to establish an accurate prediction model through mechanism analysis. In recent years, a large amount of data collected in the industrial field has been useful for predicting the grinding particle size. It is critical to investigate the data-driven modeling of the grinding particle size for the hematite grinding process.

Many studies have been working on data-driven modeling for the grinding process to address the limitations of mechanism modeling methods, such as the radial basis function (RBF) neural network (NN) method [10,11], support vector machine (SVM) [12,13], extreme learning machine (ELM) [14] and so on. Du et al. [15] and Del Villar et al. [16] used the data of Dynafrag which established a NN model for the grinding particle size. Sun et al. [17] created a model for the grinding particle size of the hydrocyclone overflow using a back propagation NN, but there are large errors. Although the modeling method of NN can

deal with the strong nonlinearity of the grinding process, there are problems of numerical metamorphosis in output weight calculation and weak network generalization ability, which lead to the decline of model accuracy. Ding et al. [18] used multiple least-squares SVM to model the clustered data and then perform multiple weighted combinations of models to establish a hematite productivity forecast model. However, SVM has problems, such as difficulty in selecting kernel function and overfitting [19]. Tang Jian et al. [20] proposed a selective integrated modeling method to obtain the soft-sensor model of the mill load by multi-sensor information. Although the real-time prediction of the grinding particle size is possible, the accuracy of the model needs to be further improved. In recent years, extreme learning machine(s) (ELM) [21] have been used in soft-sensor modeling for industry process. Deepika et al. [22] proposed an ELM combined PCA method for boiler output prediction in thermal power plants. Compared with other methods, the ELM has fewer parameters. In ELM, the hidden layer's threshold, as well as the connection weights of the input layer and the hidden layer, can be set at random and do not need to be adjusted after setting. Therefore, the parameter identification of the ELM is relatively easy, which is suitable for the process modeling that requires real-time learning [7]. However, ELM has the problem of poor non-linear representation ability and less stable results, which needs to be further studied.

Aside from complex mechanisms and nonlinear problems, process time variation also should be taken into account when modeling the hematite grinding process. The traditional modeling methods such as NN, ELM and SVM are often adapted to the global model, so it is easy to lead to more accurate models with sufficient data and poor accuracy in models with insufficient data or large changes. Hence, just-in-time learning (JITL) [23–25] and moving window (MW) [26,27] have been proposed to deal with the time-varying process problems. The MW technology is often applied to process time-varying adaptive modeling, and the prediction model is constantly reconstructed in a certain step length in the process sampling direction. Therefore, the MW technology can track the state changes of the modeling process and improve the real-time performance of the model. In the JITL framework, when the prediction of a query sample is required, online local models are built to track the process state. [28–30]. Yuan et al. [31] have proposed a new ensemble JITL (E-JITL) framework. For sample selection in the E-JITL, different similarity measurements are used. The local models are then built and trained to estimate the query data output as relevant samples with different groups corresponding to the similarity measures. Finally, an ensemble strategy was used on each local model to obtain the final prediction. In Pan et al. [32], Ding et al. [33] and Peng et al. [34], JITL is combined with ELM. The JITL-ELM model has better online adaptive ability compared to traditional ELM methods. The JITL technique is very effective in dealing with the rapid change of process state. However, hematite grinding is typically a time-varying problem, which may reduce the performance of the JITL model because samples far from the current state may be selected for local modeling. To address the process time-varying and the change of process state, JITL is combined with MW technology for process modeling [27,30]. These works provide a good foundation for building the model of grinding particle size in the hematite grinding process. However, the parameter identification of the model which combined JITL with MW is difficult and time-consuming. It is important to further study the parameter identification of the model.

Parameter identification of the hematite grinding process model is an important factor to improve the model performance. The parameter identification problem is usually described as a nonlinear optimization problem. In recent years, the swarm intelligent optimization algorithm is an effective tool for nonlinear optimization problem. Scholars have put forward a series of swarm intelligent optimization algorithms through ants, wolves, birds, moths, whales, sparrows and other biological behaviors. The sparrow searching algorithm (SSA) is a new swarm intelligent optimization algorithm proposed by Xue et al. [35] in 2020. Compared to other algorithms, the Sparrow searching algorithm has a higher solution efficiency. However, it is still possible to get stuck in the local extremum

at the end of the algorithm iteration. Many scholars have proposed various improvement strategies in order to improve local and global search ability. Oliva et al. [36] applied the logistic chaos mapping to the particle swarm algorithm to enhance the diversity of understanding and reduce the probability of the algorithm being trapped in the local space to some extent. Hezagy et al. [37] and Wang et al. [38] accelerated the convergence of the algorithm and successfully applied to the feature selection problem by adding an adaptive weight factor, effectively balancing the global and local search abilities. Li et al. [39] applied a reverse learning strategy to PSO, which not only increased population diversity but also improved global exploration ability. Wang et al. [40], Xu et al. [41] and Pappula et al. [42] added Gauss and Cauchy mutations to improve both local and global search capabilities. Farah et al. [43] proposed a new teaching–learning-based optimization algorithm (TLBO) for optimizing the tuning of power system stabilizers (PSS) and controllers based on static var compensators (SVC). Dong et al. [44] used niche the optimization technique to improve the optimization effect of multi-objective SSA and introduced Levy flight strategy to enhance the ability of the multi-objective sparrow search algorithm to jump out of the local optimum. Ding et al. [45] proposed a transformer fault diagnosis method based on an improved SSA for optimal SVM. To a certain extent, the improvement of the swarm intelligence algorithm reduces the possibility of the algorithm falling into the local extremum, but it still has some flaws, such as low convergence precision and insufficient development ability.

To solve the problems of modeling and parameter identification in the hematite grinding process, this paper studies the data-driven modeling method of the grinding particle size and the optimization method of the model. A double locally weighted extreme learning machine based on a moving window (MW-DLW-ELM) modeling method is proposed for the hematite grinding process. In this method, an on-line model of grinding particle size is established by double locally weighted technique. The moving window technique is used to track the process's time variation. Furthermore, to optimize the parameters of the MW-DLW-ELM model, an improved sparrow searching algorithm (ISSA) is developed.

2. The Grinding Particle Size in Hematite Grinding Process

Hematite is the main ore source in iron smelting. Most hematite is characterized by low grade, complex mineral composition, low magnetic susceptibility of iron minerals, uneven size distribution of useful minerals and close intergrowth of minerals. Hematite grinding process refers to the process in which the crushed raw ore is treated by a series of equipment, and the minerals are separated to the maximum extent, and finally, the particle size of the ore conforms to the industrial requirements. The closed-circuit grinding with two-stage ball mills and hydrocyclones are widely used in the hematite grinding process when the grinding particle size is required to be less than 0.15 mm. Figure 1 depicts a two-stage closed-circuit grinding process that includes No. 1 and No. 2 ball mills, a spiral classifier, a pump sump, and a hydrocyclone. First, ore, water and a certain number of steel balls are fed into the No. 1 ball mill for grinding. After grinding in the No. 1 ball mill, the ore pulp is discharged into the spiral classifier for the ore particles that do not meet the process specifications to be sent back to No. 1 ball mill for re-grinding. The smaller particles of ore are fed into the pump sump. Through the underflow pump, ore pulp is sent to the hydrocyclone for classification. The hydrocyclone separates the coarse and fine ore particles. The overflow fines will flow to the next process. The coarse particles are ground in the No. 2 ball mill. No. 1 ball mill and No. 2 ball mill work in recycling to complete the two-stage closed-circuit grinding process.

Grinding particle size is a key technical index to test the final product quality of the grinding operation. The grinding particle size is usually expressed as mass proportion of a specific particle size (such as 75 μm) content in the product [7]. The shape of ore particles is generally irregular, the diameter of the selected grains indicates the particle size value, units generally adopt millimeter or micron. In the actual grinding process, the grinding particle size is an important basis for the field staff to determine the ore feed

and water feed of the ball mill. However, the grinding process is complex so that it is difficult to use measuring instruments to monitor the particle size online. The particle size of the grinding process is generally obtained by manual analysis, which cannot meet the requirements of real-time production. Thus, it is impossible to describe them accurately. Secondly, the grinding particle size is related to many variables, such as ore feed rates of the ball mills and feed water rate of the spiral classifier [7]. It has the characteristics of strong non-linearity and multi-variable coupling. To deal with the problems in the hematite grinding process, a data-driven ELM model for the grinding particle size is proposed in this paper. Because of the change of operating conditions during the grinding process, the accuracy of the model may be reduced by the new data. A strategy based on moving window technology combined with double locally weighted modeling is introduced to make the model parameters track the process state changes.

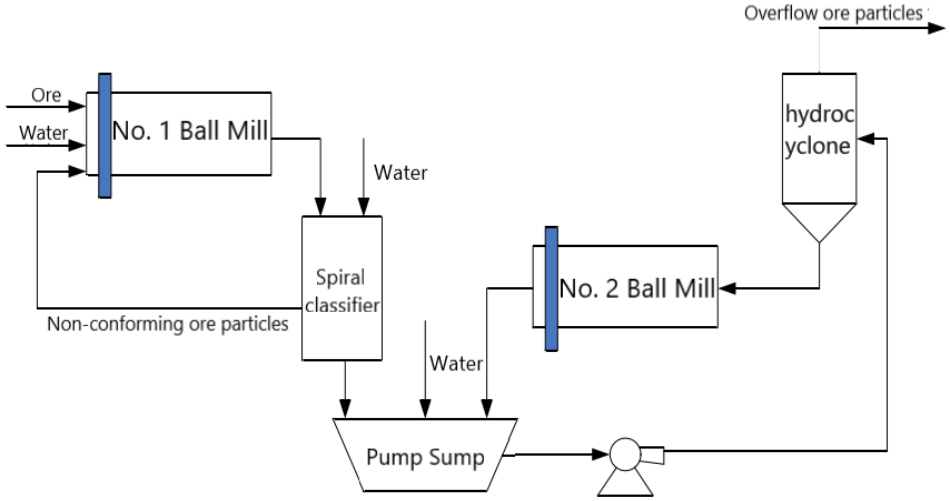

Figure 1. Two-stage closed-circuit grinding process.

The right input variables must be chosen in order to build a data-driven model of the grinding particle size. From the hematite grinding process, four main factors affecting the grinding particle size can be selected. First, the ore feed rate of the No. 1 ball mill is very important, which influences all subsequent parameters, including the grinding particle size. Second, the pulp % solids by volume have a significant impact on the No. 1 ball mill's grinding efficiency. The pulp % solids by volume can be stabilized by adjusting the feed water rate of the No. 1 ball mill. Third, the overflow pulp % solids by volume of the spiral classifier affects the No. 2 ball mill efficiency and capacity. The overflow pulp % solids by volume of the classifier can be controlled by the feed water volume of the spiral classifier. Fourth, the feed flow rate of the hydrocyclone plays an important role in the final product quality. By reasonably adjusting the feed flow rate, the grinding particle size can be controlled directly. Apart from the four factors, the size distribution and the bond work index of the ore are main characteristics when feeding the circuit. However, they are difficult to obtain online. For convenience, the size distribution and the bond work index of the ore are regarded as constants rather than variables for modeling. Based on the above analysis, taking into account the characteristics of the hematite grinding process and available process variables, the ore feed rate of the No. 1 ball mill, the feed water rate of the No. 1 ball mill, the feed water volume of the spiral classifier and the feed flow rate of the hydrocyclone are taken as the input variables of the grinding particle size model.

3. A Double Locally Weighted Extreme Learning Machine Based on Moving Window for the Grinding Particle Size Modeling

3.1. The Basic Principle of the Extreme Learning Machine

The extreme learning machine is distinguished by its rapid training speed, ease of realization and strong generalization ability. ELM's input weights and hidden layer

thresholds are set at random, and the hidden layer activation function is typically chosen as "RBF function" or "Sigmoidal function". The ELM is described in more detail below.

For any the ith random samples (x_i, y_i), where $x_i = [x_{i1}, x_{i2}, \ldots, x_{i\overline{n}}]^T \in R_n$, $y_i \in R_{1\times 1}$, $\overline{n}$ is the number of input layer nodes. The hidden layer activation function is denoted by $g(x)$, and the ELM model is depicted in Equation (1):

$$\sum_{i=1}^{\overline{m}} \beta_i g_i(x_i) = \sum_{i=1}^{\overline{m}} \beta_i g(\overline{w}_i \cdot x_i + b_i) = y_j \tag{1}$$

where j = 1, 2, ... , N, N is the number of training samples, $\overline{w}_i = [\overline{w}_{i1}, \overline{w}_{i2}, \ldots, \overline{w}_{i\overline{n}}]^T$ is the weight between the ith hidden layer node and the input node, b_i is the ith hidden layer node's threshold, $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{i\overline{m}}]^T$ is the weight of the connection between the output layer node and the ith hidden layer node, $\overline{m}$ is the number of hidden layer nodes.

The mentioned Equation (1) can be abbreviated:

$$H\beta = Y \tag{2}$$

$$H = \begin{bmatrix} g(\overline{w}_1 x_1 + b_1) & \cdots & g(\overline{w}_{\overline{m}} x_1 + b_{\overline{m}}) \\ \vdots & \ddots & \vdots \\ g(\overline{w}_1 x_{\overline{n}} + b_1) & \cdots & g(\overline{w}_{\overline{m}} x_{\overline{n}} + b_{\overline{m}}) \end{bmatrix} \tag{3}$$

$$\beta = [\beta_1, \beta_2, \cdots, \beta_{\overline{m}}]^T \tag{4}$$

$$Y = [y_1, y_2, \cdots, y_{\overline{n}}]^T \tag{5}$$

where H is the hidden layer's output matrix, β is the output layer's weight matrix, Y is the expected output. The least squares norm solution of $H\beta = Y$ is $\hat{\beta} = H^+Y$, where H^+ is the generalized inverse matrix of H.

The following are the steps of the extreme learning machine:

Step 1: Initialize the weight $\overline{w}_i$ and the threshold b_i at random;

Step 2: Using Equation (3), compute the output matrix H;

Step 3: Calculate the weight matrix $\hat{\beta} = H^+Y$.

3.2. A Double Locally Weighted Extreme Learning Machine Based on Moving Window Technology

By incorporating the merits of ELM, local modeling methods [16,17] and moving window technology [18–20], a MW-DLW-ELM algorithm is proposed in this paper. To obtain the weight coefficients of historical samples and local modeling samples, the distance between each historical sample and the query sample is first calculated. Second, different weights are assigned to variables based on the correlation between each input variable and output variable of local modeling samples. The ELM model is built to estimate the output variable by simultaneously weighting samples and variables.

The ball mill's ore feed rate, feed water rate, spiral classifier's feed water volume and hydrocyclone's feed flow rate are all used as input factors in the model for the grinding particle size, and the output variable is the grinding particle size. Assume that N historical input samples are $\{x_i\}_{i=1}^N$, the corresponding output samples are $\{y_i\}_{i=1}^N$. Local modeling samples are chosen and assigned different weights based on their distance to the query sample after evaluating the distance and similarity between historical samples and the query sample. The query sample is designated as x_q. To calculate the similarity between each historical sample and the query sample, the common Euclidean distance is used, which is shown in Equation (6):

$$d_i = \sqrt{(x_i - x_q)^T (x_i - x_q)}, i = 1, 2, \ldots, N \tag{6}$$

where d_i is the Euclidean distance between x_q and x_i. The assigned weight to x_i is calculated as:

$$w_j = \exp(\frac{-d_j^2}{\sigma^2}), j = 1, 2, \ldots, N \tag{7}$$

where σ is the adjust parameter. The larger the weight value, the closer the historical sample is to the query sample. For local modeling, the closest samples $\overline{N}$ are chosen. The input matrix is $X = [x_1, x_2 \ldots, x_{\overline{N}}]^T$ and the output matrix is $Y = [y_1, y_2 \ldots, y_{\overline{N}}]^T$. Then, the weighted training sample in the nonlinear characteristic space is $x_j^w = w_j x_j, j = 1, 2 \ldots, \overline{N}$.

In order to improve prediction performance, consider the correlation between input variables and output variable. As a result, the Pearson coefficient is used to calculate the weight of each input variable, which is defined as follows:

$$r = \frac{E(xy) - E(x)E(y)}{\sqrt{E(x^2) - E^2(x)}\sqrt{E(y^2) - E^2(y)}} \tag{8}$$

where $E(x)$ represents a single variable's expectation. The greater the r is, the more relevant the two variables are. As a result, correlation coefficients are used to assess the significance of variables. Assuming that the original input variables have a dimension of $\overline{L}$, then the weight of each dimension of the input variable is defined by a correlation coefficient:

$$\lambda_s = \frac{|r_s|}{\sum_{k=1}^{\overline{L}} |r_k|}, s = 1, 2, \ldots, \overline{L} \tag{9}$$

The weighted input sample is:

$$x_j^p = x_j \cdot diag(\lambda_1, \lambda_2, \ldots, \lambda_{\overline{L}}) = \left[\lambda_1 x_{j1}^p, \lambda_2 x_{j2}^p, \ldots, \lambda_L x_{j\overline{L}}^p\right]^T \tag{10}$$

where x_{js}^p is the sth input of the j sample; $\lambda_s, s = 1, 2, \ldots, \overline{L}$ is the weight value corresponding to each input variable.

Finally, the double locally weighted input sample can be described as:

$$x_j^{\omega p} = \omega_j x_j^p \tag{11}$$

Therefore, the model based on double locally weighted extreme learning machine is shown in Equation (12):

$$\sum_{i=1}^{\overline{m}} \beta_i g_i(x_i^{\omega p}) = \sum_{i=1}^{\overline{m}} \beta_i g(\overline{w}_i \times (\omega_i x_i^p) + b_i) = y_j, j = 1, 2, \ldots, \overline{N} \tag{12}$$

The DLW-ELM is a just-in-time learning method capable of dealing with nonlinearity in the hematite grinding process. However, the hematite grinding process, like the ore raw batches, frequently undergoes slowly time-varying working conditions. As a result, the gap between historical data and new working conditions will gradually widen, resulting in model performance degradation. Thus, the moving window technique is used to track the most recent state of the grinding process in order to ensure the model's long-term prediction accuracy. As a result, the DLW-ELM models are gradually built in the moving windows alongside the process running time for the prediction of grinding particle size.

Figure 2 depicts an illustration of the MW-DLW-ELM model. Assume W_m for the window length and D_m for the step size. Denote $\{X_{t1}, Y_{t1}\}$ the input dataset and output dataset in the first window. Firstly, DLW-ELM models are built to estimate the grinding particle size for each of the subsequent D_m query samples in the first window. Then, the window is moved forward by step size D_m according to the sampling direction, while the datasets in the window are updated to $\{X_{t2}, Y_{t2}\}$. In these datasets, the MW-DLW-ELM

models are also constructed to predict the output of D_m query samples after the second window. By repeating the above procedures, the datasets in the sth window can be represented as $\{X_{ts}, Y_{ts}\}$. The MW-DLW-ELM model trained in the moving window can predict the output of the query samples until the predicted output for all query samples is obtained.

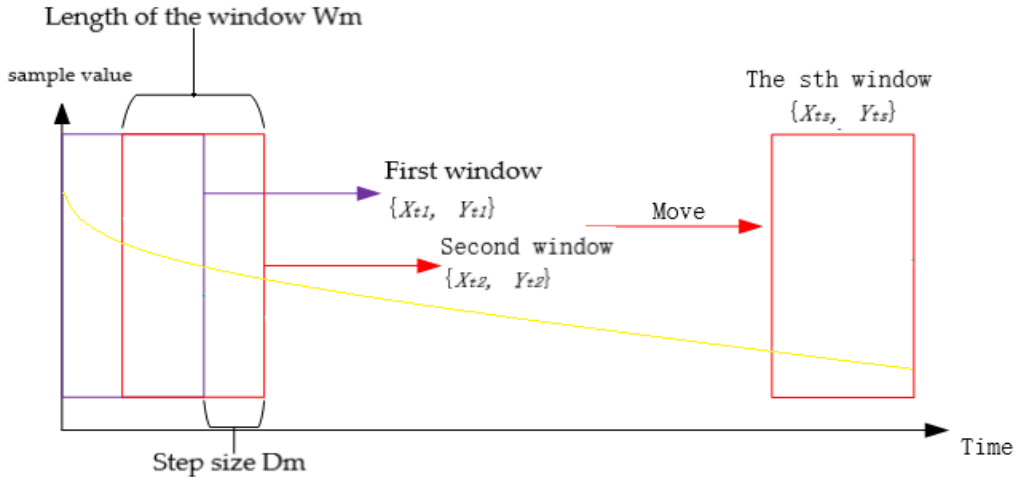

Figure 2. Illustration of the MW-DLW-ELM.

The MW-DLW-ELM steps are summarized as follows:

Step 1: The number $\overline{N}$ of local modeling samples, the window length W_m and the step size D_m are determined by trial and error;

Step 2: The distance between each historical sample and the query sample is calculated in the window. The local modeling samples are then arranged in descending order of distance from small to large;

Step 3: The first $\overline{N}$ samples are chosen for local modeling, and their weights are calculated using Equation (7);

Step 4: The Pearson coefficients of the $\overline{L}$ input variables are calculated by Equation (8). Then, the input variables are weighted by Equations (9) and (10);

Step 5: The double locally weighted samples are obtained by Equation (11). Then, using the least squares norm solution of Equation (12), the output of the query sample can be obtained;

Step 6: After the subsequent D_m query samples in the window is estimated by DLW-ELM, the window is moved forward by step size D_m. The historical samples in the window are updated. Then, repeat Step 2 until all of the query samples receive the predicted output.

The $\overline{w}_i$ and b_i in Equation (12) are the key parameters that affect the performance of the MW-DLW-ELM. However, the range of manually set parameters is limited, and they are not the optimal parameters. Thus, in Section 4, an improved sparrow optimization algorithm is developed to optimize the parameters of the MW-DLW-ELM.

4. Parameter Optimization Based on Improved Sparrow Optimization Algorithm

4.1. Parameter Optimization for the MW-DLW-ELM Model

To obtain optimal performance, MW-DLW-ELM parameters identification is described as an optimization problem of solving the minimum objective function $\overline{J}$:

$$\begin{aligned} &\min_{\theta} \overline{J} = \sum_{i=1}^{N_t} (y_i - \hat{y}_i(\theta))^2 \\ &s.t.\ \theta_{\min} \le \theta \le \theta_{\max} \end{aligned} \tag{13}$$

where $\theta = [\overline{w}_i, b_i]^T$ is the parameter vector, y_i denotes the actual grinding particle size sample value, $\hat{y}_i$ denotes the corresponding model prediction value, N_t is the number

of samples used for training. The optimal parameters for the MW-DLW-ELM model are obtained when all of the parameters are within the expected range $[\theta_{\min}, \theta_{\max}]$ and the objective function $\overline{J}$ is globally minimized. The algorithm proposed in this paper can generate the parameter vector.

4.2. Original Sparrow Searching Algorithm

After determining the parameters needed to be optimized for the MW-DLW-ELM model, the sparrow searching algorithm is used to find optimal parameters. The original SSA is proposed in 2020, inspired by the predatory and anti-predatory behavior of the sparrow in the biological world [35]. In order to complete the foraging, sparrows are usually divided into producers and followers. In the natural state, sparrows will monitor each other. The followers in the swarm usually compete for the food resources of their high intake companions in order to improve their own predation rate. While foraging, all sparrows will keep alert to the surrounding environment to prevent the arrival of natural enemies. Sparrow set matrix is as follows:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n_p,1} & x_{n_p,2} & \cdots & x_{n_p,d} \end{bmatrix} \tag{14}$$

where n_p is the number of sparrows, d is the dimension of variables. Sparrows have the following fitness matrix:

$$F_x = \begin{bmatrix} f\left(\left[x_{1,1} & x_{1,2} & \cdots & x_{1,d}\right]\right) \\ f\left(\left[x_{2,1} & x_{2,2} & \cdots & x_{2,d}\right]\right) \\ \vdots \\ f\left(\left[x_{n_p,1} & x_{n_p,2} & \cdots & x_{n_p,d}\right]\right) \end{bmatrix} \tag{15}$$

where the value of each line in F_x represents the fitness value of sparrows.

In the SSA, the producer typically has a large energy reserve and is responsible for searching the entire population for areas with abundant food, providing all participants with the area and direction of foraging. The energy reserve is determined by the sparrow's fitness value. Sparrows with higher fitness values have first priority in obtaining food, driving the entire population to seek food as producers. The producer's location update mode is as follows:

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} \cdot \exp\left(\frac{-i}{\alpha \cdot iter_{\max}}\right) & R_2 < ST \\ X_{i,j}^{t} + Q \cdot L & R_2 \geq ST \end{cases} \tag{16}$$

where t represents the number of current iterations, j = (1,2, . . . , d), $X_{i,j}^{t}$ represents the jth dimension position of the ith sparrow at iteration t. $iter_{\max}$ represents the maximum number of iterations. α is a random number of (0,1), $R_2 \in$ [0, 1] and $ST \in$ [0.5, 1], R_2 and ST represent warning value and safety value, respectively. Q is a random number subject to [0, 1] normal distribution. L stands for a matrix of $1 \times d$, and each element inside is 1. When $R_2 < ST$, it means that there is no natural enemy around, and the producer conducts an extensive search mode. If $R_2 \geq ST$, it means that some sparrows have found natural enemies, and all sparrows need to fly to other safe areas quickly.

Follower refers to a kind of sparrow with low energy and poor fitness value in the population. The following is the follower's position update formula:

$$X_{i,j}^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{X_{worst}^{t} - X_{i,j}^{t}}{i^2}\right) & i > \frac{n_p}{2} \\ X_{p}^{t+1} + \left|X_{i,j}^{t} - X_{p}^{t+1}\right| \cdot A^{+} \cdot L & other \end{cases} \tag{17}$$

where X_P is the best position for producers, X_{worst} represents the worst position, A represents a matrix of $1 \times d$, each element in the matrix is randomly assigned with a value of 1 or -1, $A^+ = A^T (AA^T)^{-1}$. When $I > n_p/2$, it means that the ith follower with poor fitness value does not get food and is in a very hungry state. At this time, it needs to fly to other places for food to obtain more energy.

At the same time, the sparrow population has the behavior of detection and early warning:

While searching for food, some sparrows will serve as scouts, alerting others. When danger approaches, they will abandon their current food source. Whether the producer or follower will abandon their current food and take on a new role. *SD* (typically 10–20%) sparrows are chosen at random from the population for early warning behavior. Its location update equation is as follows:

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^t + \overline{\beta} \cdot \left| X_{i,j}^t - X_{best}^t \right| & f_i > f_g \\ X_{i,j}^t + K \cdot \left(\frac{\left| X_{i,j}^t - X_{worst}^t \right|}{(f_i - f_w) + \varepsilon} \right) & f_i = f_g \end{cases} \tag{18}$$

where X_{best} is the global optimal position, $\overline{\beta}$ is a step control parameter, a random number of normal distribution with mean value of 0 and variance of 1, $K \in [-1, 1]$ is a uniform random number. f_i is the fitness value of the current sparrow. f_g and f_w are the current global best and worst fitness values, respectively. ε is the minimum constant to avoid a zero in the denominator. When $f_i > f_g$, it means that sparrows are at the edge of the population and are extremely vulnerable to natural enemies. $f_i = f_g$ indicates that sparrows in the middle of the population are aware of this danger, so they need to be close to other sparrows. K represents the direction of movement of the sparrow and is also a step control parameter.

4.3. Improved Sparrow Searching Algorithm

The original sparrow searching algorithm is slow to converge and easily falls into a local optimum. This paper improves the original algorithm by incorporating aspects of the evolutionary mechanism.

Introducing dynamic weight factor for producer location update:

The dynamic weight factor is introduced for the producer location update using the concept of inertial weight [38]. With the weight that decreases adaptively with the number of iterations, it indicates that the producer is affected by the global optimal solution, and it can also effectively balance the local and global search abilities and improve the convergence speed. The weight coefficient is denoted as follows:

$$\omega_d = \frac{e^{2(1-t/iter_{\max})} - e^{-2(1-t/iter_{\max})}}{e^{2(1-t/iter_{\max})} + e^{-2(1-t/iter_{\max})}} \tag{19}$$

The improved producer location is updated as follows:

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^t + \left[(\omega_d \cdot f_{j,g}^t - X_{i,j}^t) \right] \cdot rand & R_2 < ST \\ X_{i,j}^t + Q & R_2 \geq ST \end{cases} \tag{20}$$

where $f_{j,g}^t$ is the global optimal solution of the jth dimension in the previous generation, and rand is a random number of (0,1).

Introducing Gaussian distribution strategy for follower location update:

To avoid sparrows blindly following the producer and to improve global search ability, the Gaussian distribution strategy [41] is introduced to the follower update equation. The improved follower's updated equation is as follows:

$$X_{i,j}^{t+1} = X_{best}^t + N(0,1) \cdot \left| X_{best}^t - X_{i,j}^t \right| \tag{21}$$

where $N(0,1)$ is the standard Gaussian distribution.

Introducing teaching–learning-based optimization for SSA:

Students are viewed as search points distributed in the decision variable space in the teaching–learning-based optimization (TLBO) algorithm [43], and the best student is defined as the class teacher. Different from the traditional evolutionary algorithm and swarm intelligence algorithm, the iterative evolution process of the teaching optimization algorithm includes the teaching stage and learning stage, as shown in the Figure 3. During the teaching stage, students will improve their own level by learning from teachers in order to raise the average level of the class; later, in the learning stage, the students will improve their own level through interactive learning with another randomly selected student.

Figure 3. Diagram of teaching–learning-based optimization.

The update equation for students at the learning stage is:

$$newX_i = \begin{cases} X_i + rand \cdot (X_i - X_k) & if\ f(X_i) < f(X_k) \\ X_i + rand \cdot (X_k - X_i) & otherwise \end{cases} \tag{22}$$

The SSA is improved by the mechanism that students learn from each other in the learning stage of the TLBO. Thus, the producers can learn from one another, improving the producer's search ability and broadening the search scope. The final update equation of the improved producer and follower is as follows:

Producers:

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^t + rand \cdot (X_{i,j}^t - X_{k,i}^t) + \left[\left(\omega_d f_{j,g}^t - X_{i,j}^t\right) \cdot rand\right] & if\ f(X_i^t) \le f(X_k^t), R_2 < ST \\ X_{i,j}^t + rand \cdot (X_{k,i}^t - X_{i,j}^t) + \left[\left(\omega_d f_{j,g}^t - X_{i,j}^t\right) \cdot rand\right] & if\ f(X_i^t) > f(X_k^t), R_2 < ST \\ X_{\mathrm{i,j}}^t + Q & if\ R_2 \ge ST \end{cases} \tag{23}$$

Followers:

$$X_{i,j}^{t+1} = X_{best}^t + N(0,1) \cdot \left| X_{best}^t - X_{i,j}^t \right| \tag{24}$$

Improved sparrow renewal equation for detection and early warning (scout):

In the SSA algorithm, the step control parameter $\overline{\beta}$ plays an important role in balancing global search ability and local search ability [38]. However, $\overline{\beta}$ is a random number that cannot satisfy the searching ability of the algorithm, which may lead SSA to local optimum. Larger step control parameters $\overline{\beta}$ can facilitate global search ability, and smaller step control parameters can facilitate local search ability. Thus, the step length control parameter $\overline{\beta}$ need to be adjusted dynamically. The update equation for $\overline{\beta}$ is shown as follows:

$$\overline{\beta} = f_g - (f_g - f_w) \cdot \left(\frac{iter_{\max} - t}{iter_{\max}}\right)^{1.5} \tag{25}$$

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^t + \overline{\beta}(X_{i,j}^t - X_{best}^t) & f_i \ne f_g \\ X_{best}^t + \overline{\beta}(X_{worst}^t - X_{best}^t) & f_i = f_g \end{cases} \tag{26}$$

The final update equation of the scout is as follows:

According to the improved scout update equation, if the sparrow is in the optimal position, it will escape to the random position between the optimal and worst positions; otherwise, it will escape to the random position between itself and the optimal position.

Algorithm 1 depicts the improved ISSA algorithm's pseudocode.

Algorithm 1: Improved Sparrow Searching Algorithm

Input: Generate random population of sparrows $X_i(t)$
Output: $X_i(t+1)$ and $F(X_i(t+1))$(Fitness value)
1: Initialize population parameters, such as population number, the maximum number of iterations $Iter_{\max}$, discoverers PD, number of early warning sparrows SD, warning threshold R_2, etc.
2: Calculate the fitness value of each sparrow, find the current optimal individual fitness values, the worst fitness and the corresponding location.
3: The producers were selected from the sparrows with better position, and the producer updates the position by Equation (23).
4: The remaining sparrows act as followers and update the position by Equation (24).
5: Select some sparrows randomly among the sparrows as scout and update the position by Equation (26).
6: Calculate the updated fitness of the entire sparrow population and find the global optimal sparrow.
7: Determine if the end condition is met, and if so, proceed to the next step, otherwise jump to step 2.
8: The program ends and the optimal result is output.

5. Experiments and Analysis

In this part, three experiments are carried out to validate the proposed method. Firstly, the ISSA's performance is validated with other three algorithms by 11 benchmark functions. Secondly, the final grinding particle size model optimized by ISSA is compared with four other models to verify the effectiveness of the algorithm and model. The samples used in this paper are collected on a physical simulation experiment platform. There are a total of 300 samples collected for modeling, with 200 serving as historical data for training and 100 serving as testing data. The detailed results are shown as follows.

5.1. Experiment of Benchmark Function

Eleven benchmark functions F1~F11 are selected to test the original SSA [35], particle swarm optimization (PSO) [46], grey wolf optimizer (GWO) [47] and the ISSA. The benchmark functions F1~F5 are unimodal functions and F6~F11 are multimodal functions. The unimodal functions F1~F5 mainly test the convergence speed of the algorithm. The multimodal functions F6~F11 have many local extreme points, which are difficult to jump out, and they are used to test the global search ability of the algorithms. The specific test functions are shown in Table 1. By trial and error, in the four algorithms, the population numbers are set as $n_p = 30$, and the numbers of iteration are set as $Iter_{\max} = 1000$. In the PSO algorithm, the acceleration factors are set as $c_1 = c_2 = 2$, the minimum inertia factor $\omega_{\min}$ is 0.4, the maximum inertia factor $\omega_{\max}$ is 0.9. The convergence factor in the GWO algorithm is set from 2 to 0. The alarm values of SSA and ISSA are 0.8, the numbers of producers PD are 0.2, and the numbers of sparrows who perceive the danger SD are 0.2. Each test function runs 200 times independently to eliminate randomness. The calculated best value (Best), worst value (Worst), average value (Mean), standard deviation (STD) are used as indices. The results of F1 to F5 are shown in Table 2.

Table 2 shows that ISSA achieves most satisfactory results for five test functions. Especially in the F1~F4 functions, ISSA can find the optimal value 0 in both F1 and F3. There is a certain high requirement for the algorithm's ability to jump out of the local optimal on F5. It can be seen that in the F5 test results, ISSA still has certain advantages over other algorithms. In terms of global search ability, the simulation results show that the ISSA algorithm outperforms the SSA algorithm and the other two algorithms. Figure 4 depicts the convergence curve of the F1 function to demonstrate the performance and speed of the ISSA algorithm more intuitively.

It can be seen from the convergence curve in Figure 4 that ISSA is superior to PSO, GWO and SSA algorithms in convergence speed. From Table 2 and Figure 4, In terms of

global search capability and convergence speed in low peak functions, the ISSA algorithm clearly outperforms the competition.

Table 1. Benchmark functions F1~F11.

Test Function Expression	Symbol	Range of Values	Value of Optimal Solution
$f_1 = \sum_{i=1}^{n} x_i^2$	F1	$[-100, 100]$	0
$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	F2	$[-10, 10]$	0
$f_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_i \right)^2$	F3	$[-100, 100]$	0
$f_4 = \max_i \{ \lvert x_i \rvert, 1 \leqslant i \leqslant n \}$	F4	$[-100, 100]$	0
$f_5 = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	F5	$[-1.28, 1.28]$	0
$f_6 = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	F6	$[-5.12, 5.12]$	0
$f_7 = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + \mathrm{e}$	F7	$[-30, 30]$	0
$f_8 = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	F8	$[-600, 600]$	0
$f_9 = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	F9	$[-65.536, 65.536]$	1
$f_{10} = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^{\mathrm{T}} + c_i \right]^{-1}$	F10	$[0, 10]$	$1/c_i$
$f_{11}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2 \right)$	F11	$[0, 1]$	-3.8628

Table 2. Experimental results of test functions F1 to F5.

Functions		PSO	GWO	SSA	ISSA	Optimal Value
F1	Best	0.058933	3.396×10^{-62}	1.714×10^{-158}	0	0
	Worst	0.38873	7.5113×10^{-58}	2.3176×10^{-75}	0	
	Mean	0.19735	6.8668×10^{-59}	7.7423×10^{-77}	0	
	STD	0.092988	1.5269×10^{-58}	4.231×10^{-76}	0	
F2	Best	1.3475	4.0876×10^{-36}	2.9828×10^{-84}	7.0841×10^{-250}	0
	Wort	7.7625	3.8835×10^{-34}	8.8405×10^{-41}	2.6869×10^{-223}	
	Mean	3.4671	8.7556×10^{-35}	3.2694×10^{-42}	8.9563×10^{-225}	
	STD	1.5434	8.292×10^{-35}	1.6118×10^{-41}	0	
F3	Best	6.6321	3.061×10^{-19}	0	0	0
	Worst	32.7164	2.8866×10^{-13}	7.7326×10^{-35}	0	
	Mean	17.1663	1.1034×10^{-14}	2.7878×10^{-36}	0	
	STD	7.6494	5.2766×10^{-14}	1.4122×10^{-35}	0	
F4	Best	1.0528	5.2743×10^{-16}	9.227×10^{-166}	6.6986×10^{-243}	0
	Worst	4.7375	2.3154×10^{-14}	1.2678×10^{-31}	1.6359×10^{-222}	
	Mean	2.8452	8.8794×10^{-15}	4.245×10^{-33}	5.7244×10^{-224}	
	STD	1.1379	5.607×10^{-15}	2.3144×10^{-32}	0	
F5	Best	0.043813	0.00019499	0.0001679	1.9965×10^{-5}	0
	Worst	5.3953	0.0020989	0.0013119	0.00040725	
	Mean	0.28151	0.0007823	0.0033844	0.0016521	
	STD	0.96657	0.00048393	0.0010085	0.00034993	

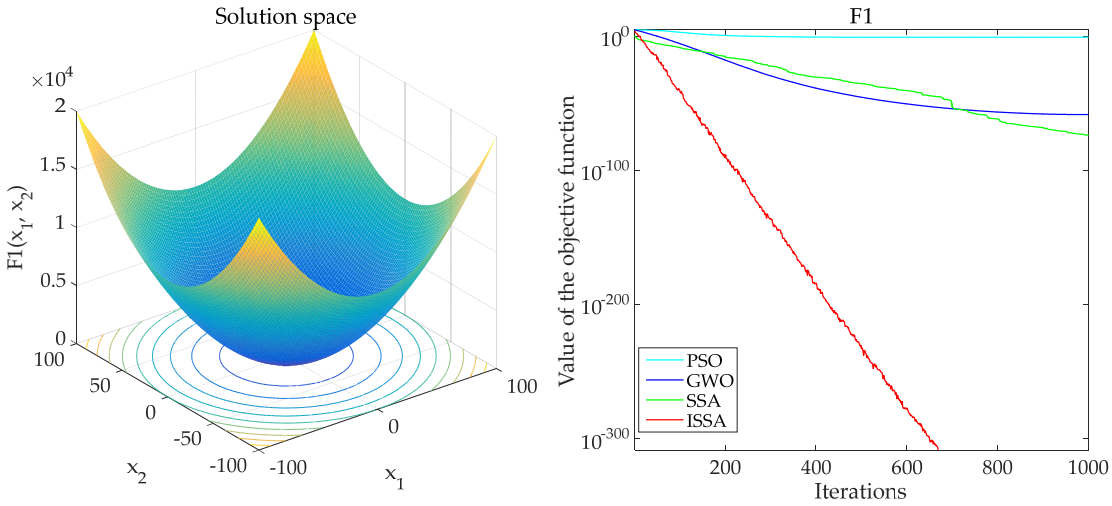

Figure 4. F1 Convergence curve.

To verify the performance of ISSA algorithm on multi-dimensional complex functions, multi-modal test functions F6~F11 are adopted. The convergence curve of the F7 test function is shown in Figure 5. From Table 3 and Figure 5, SSA and ISSA have obvious advantages over other algorithms. ISSA can find the optimal solution 0 on F6 and F7 while SSA and ISSA have the same optimization precision on F7, but ISSA has a faster convergence speed. On the functions F9~F11 with more local extremum, ISSA performs well in standard deviation and achieves high convergence accuracy. The simulation results show that the ISSA algorithm performs well in solving complex multi-extremum problems.

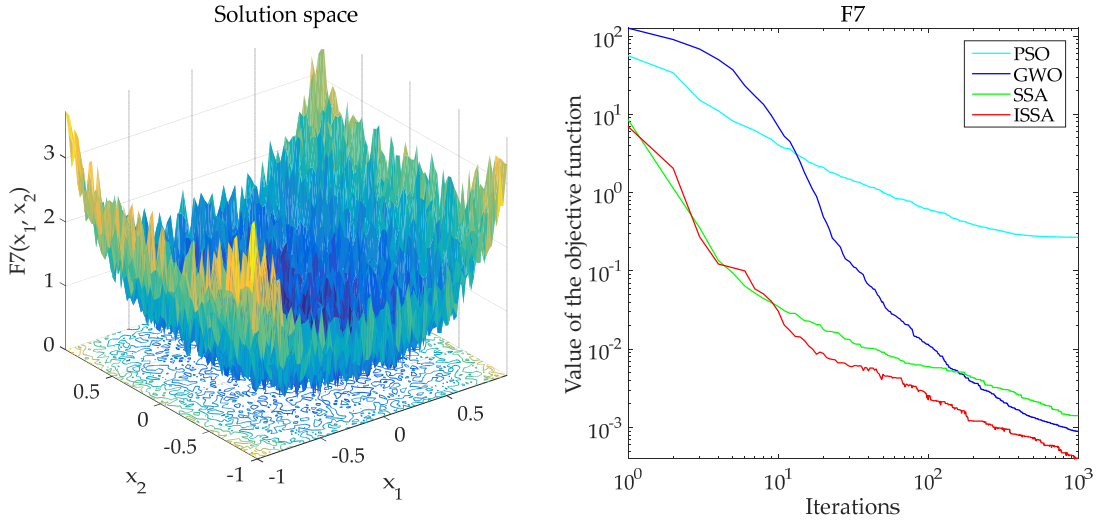

Figure 5. F7 Convergence curve.

5.2. Experiments on the Parameters Optimization of the ELM Models

To examine the effectiveness and efficiency of the proposed method further, the ISSA algorithms will be applied to the ELM model and the moving window double locally weighted extreme learning machine (MW-DLW-ELM) model. The ELM model optimized by PSO (PSO-ELM), ELM model optimized by GWO (GWO-ELM), ELM model optimized by ISSA (ISSA-ELM) and MW-DLW-ELM model optimized by ISSA (denoted as Model 1) are used to predict the grinding particle size. There are 300 samples sorted by time collected for the experiments, with the first 200 samples serving for model training and the last 100 samples serving for testing. The parameters in PSO, GWO and ISSA are the same as in Section 5.1. By trial and error, the length of moving window W_m is set as 20, and the

step size D_m is set as 5. The maximum error (MAX), the mean square error (MSE) and the mean absolute error (MAE) are used as the performance indices. The comparison results are shown in Table 4, the convergence curves of each algorithm are shown in Figure 6, the prediction results of each model are shown in Figure 7 and the prediction errors of the models are shown in Figure 8.

Table 3. Experimental Results of Test Functions F6~F11.

Functions		PSO	GWO	SSA	ISSA	Optimal Value
F6	Best	24.2468	0	0	0	0
	Worst	84.1836	4.8221	0	0	
	Mean	53.8561	14.4021	0	0	
	STD	14.4021	1.1637	0	0	
F7	Best	2.5838	1.1546×10^{-14}	8.8818×10^{-16}	8.8818×10^{-16}	0
	Worst	6.5662	2.2204×10^{-14}	8.8818×10^{-16}	8.8818×10^{-16}	
	Mean	4.1938	1.5336×10^{-14}	8.8818×10^{-16}	8.8818×10^{-16}	
	STD	0.93679	1.8504×10^{-15}	0	0	
F8	Best	2.4427	0	0	0	0
	Worst	14.703	0.021561	0	0	
	Mean	5.8382	0.0021043	0	0	
	STD	2.7263	0.0055966	0	0	
F9	Best	0.998	0.998	0.998	0.998	1
	Worst	5.9288	10.7632	12.6705	0.998	
	Mean	1.5932	3.6837	4.6346	0.998	
	STD	1.0887	3.3394	5.2536	1.3039×10^{-16}	
F10	Best	−10.1532	−10.1531	−10.1532	−10.1532	−10.1532
	Worst	−2.6305	−4.145	−5.0552	−10.1532	
	Mean	−6.1408	−9.4457	−8.7937	−10.1532	
	STD	3.2554	1.8396	2.2929	5.8915×10^{-15}	
F11	Best	−3.8628	−3.8628	−3.8628	−3.8628	−3.8628
	Worst	−3.8549	−3.8549	−3.0898	−3.8628	
	Mean	−3.8617	−3.8612	−3.837	−3.8628	
	STD	0.002725	0.0028854	0.14113	2.6402×10^{-15}	

Table 4. Comparison results of models optimized by various algorithms.

Model	MAX	MSE	MAE
ISSA-ELM	4.1087	1.6843	0.9377
Model 1	2.1324	1.0160	0.4926
PSO-ELM	5.8202	3.3760	0.9165
GWO-ELM	6.4012	4.6723	0.9704

It can be seen from the results in Table 4 that compared with PSO-ELM and GWO-ELM, MAX, MSE and MAE of Model 1 are greatly reduced, and the accuracy of the Model 1 is greatly improved. By comprehensive comparison of the models, the overall performance of Model 1 is the best. The maximum error is 2.1324, the mean square error is 1.0160 and the average absolute error is 0.4926.

As can be seen from the Figures 6–8, the ISSA algorithm has a good performance of searching ability and jumping out of the local optimum. The Model 1 has a smaller fitness value than the other models. Compared to other models, both ISSA-ELM and Model 1 can better fit the target values than the other models, but the error distribution of the Model 1 is relatively more uniform. The MW-DLW-ELM optimized by ISSA has the fastest convergence speed, the highest accuracy and a better prediction performance than other models, which verifies the effectiveness of the ISSA algorithm.

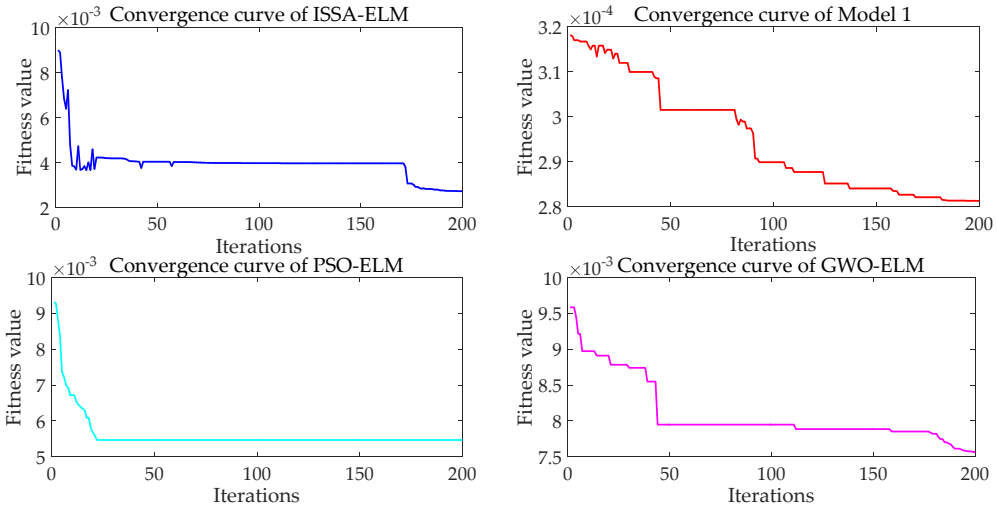

Figure 6. Comparison of convergence curves for the four algorithms.

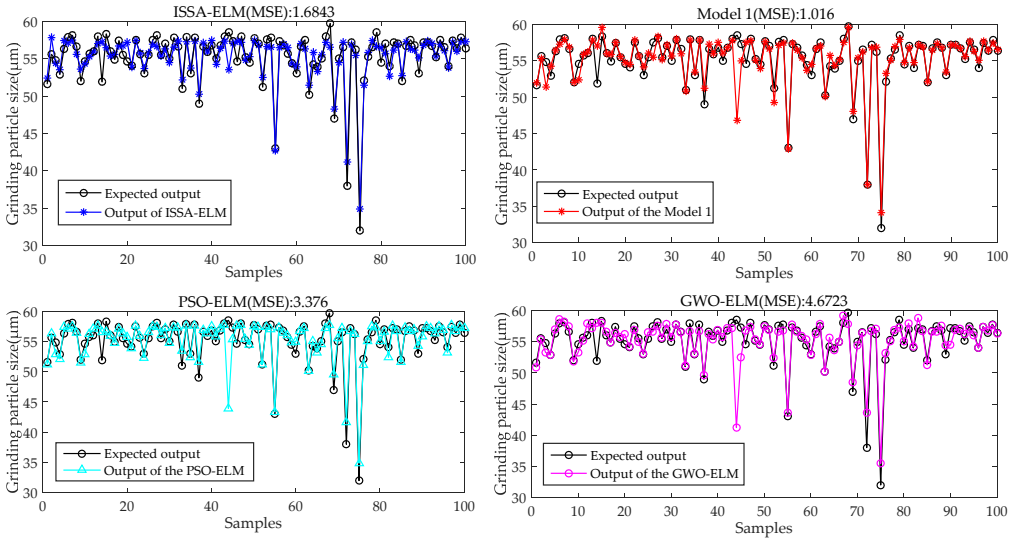

Figure 7. Prediction results of each model.

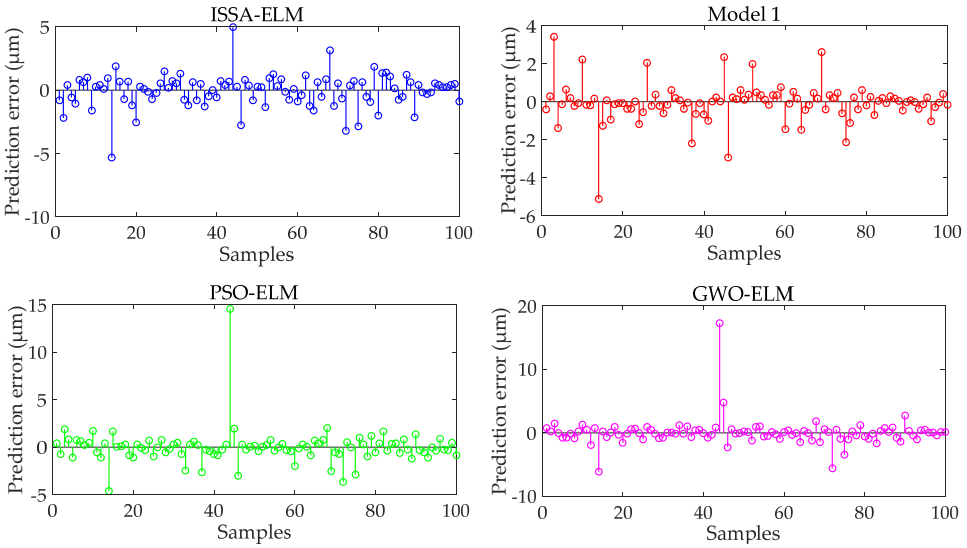

Figure 8. Error Comparison of each model.

6. Conclusions

This paper mainly studies a MW-DLW-ELM modelling technique and an improved sparrow searching algorithm. The MW-DLW-ELM combines the advantages of ELM, local modeling methods and moving window technology. In the improved sparrow searching algorithm, a dynamic weight factor is introduced for producer and scout location update to balance the local and global searching abilities and improve the convergence speed. The Gaussian distribution strategy is introduced for the follower to improve the ability of the algorithm to jump out of the local optimum. The teaching–learning-based optimization is combined with the sparrow searching algorithm, which can improve the producer's searching ability and broaden the searching scope. The performances of the proposed methods are validated by some test functions and a case study of the grinding circuit. The results show the effectiveness of the ISSA algorithm and MW-DLW-ELM. The proposed methods are expected to be applied to other complex industries in the future and have important practical significance.

Author Contributions: Data curation, H.L.; methodology, J.D.; supervision, J.D.; validation, H.L.; writing—original draft, H.L.; writing—review and editing, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that there are no conflict of interest regarding the publication of this paper.

References

1. Sbarbaro, D.; Del Villar, R. *Advanced Control and Supervision of Mineral Processing Plants*; Springer: London, UK, 2010.
2. Wang, M.H.; Yang, R.Y.; Yu, A.B. DEM investigation of energy distribution and particle breakage in tumbling ball mills. *Powder Technol.* **2012**, *223*, 83–91. [CrossRef]
3. Chai, T.Y.; Ding, J.L. Hybrid intelligent control for optimal operation of shaft furnace roasting process. *Control. Eng. Pract.* **2011**, *19*, 264–275. [CrossRef]
4. Lu, S.; Zhou, P.; Chai, T.Y.; Dai, W. Modeling and simulation of whole ball mill grinding plant for integrated control. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 1004–1019.
5. Zhou, P.; Chai, T.Y.; Wang, H. Intelligent optimal-setting control for grinding circuits of mineral processing process. *IEEE Trans. Autom. Sci. Eng.* **2009**, *6*, 730–743. [CrossRef]
6. Wei, D.H.; Craig, I.K. Grinding mill circuits-a survey of control and economic concerns. *Int. J. Miner. Process.* **2009**, *90*, 56–66. [CrossRef]
7. Zhou, P.; Lu, S.; Yuan, M.; Chai, T.Y. Survey on higher-level advanced control for grinding circuits operation. *Powder Technol.* **2016**, *288*, 324–338. [CrossRef]
8. Brics, M.; Ints, V.; Kitenbergs, G.; Cebers, A. The most energetically favorable configurations of hematite cube chains. *Phys. Rev. E* **2021**, *105*, 024605. [CrossRef]
9. Zhou, P.; Chai, T.Y.; Sun, J. Intelligence-Based Supervisory Control for Optimal Operation of a DCS-Controlled Grinding System. *IEEE Trans. Control Syst. Technol.* **2013**, *21*, 143–148. [CrossRef]
10. Mitchell, T.M. *Machine Learning*; McGrawHill: New York, NY, USA, 1997.
11. Guan, S.H.; Shen, Y.X. Power load forecasting based on PSO RBF-NN. *Transducer Microsyst. Technol.* **2021**, *40*, 128–131.
12. Cortes, C.; Vapnik, V. Support-vector networks, Mach. *Learn.* **1995**, *20*, 273–297.
13. Sannasi Chakravarthy, S.; Bharanidharan, N.; Rajaguru, H. Multi-deep CNN based experimentations for early diagnosis of breast cancer. *IETE J. Res.* **2022**, 1–16. [CrossRef]
14. Huang, G.B.; Zhu, Q.Y.; Siew, C. Extreme learning machine: Theory and applications. *Neurocomputing.* **2006**, *70*, 489–501. [CrossRef]
15. Du, Y.G.; Del Villar, R.; Thibault, J. Neural net-based softsensor for dynamic particle size estimation in grinding circuits. *Int. J. Miner. Process.* **1997**, *52*, 121–135. [CrossRef]
16. Del Villar, R.G.; Thibault, J.; Del Villar, R. Development of a softsensor for particle size monitoring. *Miner. Eng.* **1996**, *9*, 55–72. [CrossRef]
17. Sun, Z.; Wang, H.; Zhang, Z. Soft sensing of overflow particle size distributions in hydrocyclones using a combined method. *Tsinghua Sci. Technol.* **2008**, *13*, 47–53. [CrossRef]
18. Ding, J.L.; Chai, T.Y.; Cheng, W.J.; Zheng, X.P. Data-based multiple-model prediction of the production rate for hematite ore beneficiation process. *Control Eng. Practice.* **2015**, *45*, 219–229. [CrossRef]

19. Deris, A.M.; Zain, A.M.; Sallehuddin, R. Overview of support vector machine in modeling machining performances. In Proceedings of the 2011 International Conference on Advances in Engineering, Shanghai, China, 17–18 September 2011.
20. Tang, J.; Chai, T.Y.; Zhao, L.J.; Yu, W.; Yue, H. Soft sensor for parameters of mill load based on multi-spectral segments PLS sub-models and on-line adaptive weighted fusion algorithm. *Neurocomputing* **2012**, *78*, 38–47. [CrossRef]
21. Meng, A.B.; Zhu, Z.B.; Deng, W.S.; Ou, Z.H.; Lin, S.; Wang, C.; Xu, X.; Wang, X.L.; Yin, H.; Luo, J.Q. A novel wind power prediction approach using multivariate variational mode decomposition and multi-objective crisscross optimization based deep extreme learning machine. *Energy* **2022**, *260*, 124957–124977. [CrossRef]
22. Deepika, K.K.; Varma, P.S.; Reddy, C.R.; Sekhar, O.C.; Alsharef, M.; Alharbi, Y.; Alamri, B. Comparison of Principal-Component-Analysis-Based Extreme Learning Machine Models for Boiler Output Forecasting. *Appl. Sci.* **2022**, *12*, 7671–7686. [CrossRef]
23. Yuan, X.F.; Ge, Z.Q.; Huang, B.; Song, Z.H. A probabilistic just-in-time learning framework for soft sensor development with missing data. *IEEE Trans. Control Syst. Technol.* **2016**, *25*, 1124–1132. [CrossRef]
24. Yang, Z.; Ge, Z.Q. Rethinking the value of just-in-time learning in the era of industrial big data. *IEEE Trans. Ind. Inform.* **2021**, *18*, 976–985. [CrossRef]
25. Hu, Y.; Ma, H.H.; Shi, H.B. Enhanced batch process monitoring using just-in-time-learning based kernel partial least squares. *Chemometrics Intell. Lab. Syst.* **2013**, *123*, 15–27. [CrossRef]
26. Bai, Y.; Zhang, G.Q.; Wang, G.L.; Chen, F.F.; Bi, G.D.; Xu, D.G. Position and speed detection method based on adaptive extended moving-window linear regression for traction machine drives. *IEEE Trans. Transport. Electrific.* **2022**, *8*, 2884–2897. [CrossRef]
27. Yuan, X.F.; Ge, Z.Q.; Song, Z.H. Spatio-temporal adaptive soft sensor for nonlinear time-varying and variable drifting processes based on moving window LWPLS and time difference model. *Asia-Pac. J. Chem. Eng.* **2016**, *11*, 209–219. [CrossRef]
28. Chen, J.Y.; Gui, W.H.; Dai, J.Y.; Yuan, X.F.; Chen, N. An ensemble just-in-time learning soft-sensor model for residual lithium concentration prediction of ternary cathode materials. *J. Chemometr.* **2020**, *34*, 3225–3240. [CrossRef]
29. Dai, J.Y.; Chen, N.; Luo, B.; Gui, W.H.; Yang, C.H. Multi-scale local LSSVM based spatiotemporal modeling and optimal control for the goethite process. *Neurocomputing* **2020**, *385*, 88–99. [CrossRef]
30. Dai, J.Y.; Chen, N.; Yuan, X.F.; Gui, W.H.; Luo, L.H. Temperature prediction for roller kiln based on hybrid first-principle model and data-driven MW-DLWKPCR model. *ISA Trans.* **2020**, *98*, 403–417. [CrossRef]
31. Yuan, X.F.; Zhou, J.; Wang, Y.L.; Yang, C.H. Multi-similarity measurement driven ensemble just-in-time learning for soft sensing of industrial processes. *J. Chemometr.* **2018**, *32*, 3040–3053. [CrossRef]
32. Pan, B.; Jin, H.P.; Wang, L.; Qian, B.; Chen, X.G.; Huang, S.; Li, J.G. Just-in-time learning based soft sensor with variable selection and weighting optimized by evolutionary optimization for quality prediction of nonlinear processes. *Chem. Eng. Res. Des.* **2019**, *144*, 285–299. [CrossRef]
33. Ding, Y.Y.; Wang, Y.Q.; Zhou, D.H. Mortality prediction for ICU patients combining just-in-time learning and extreme learning machine. *Neurocomputing* **2018**, *281*, 12–19. [CrossRef]
34. Peng, X.; Tang, Y.; Du, W.L.; Qian, F. Online performance monitoring and modeling paradigm based on just-in-time learning and extreme learning machine for a non-Gaussian chemical process. *Ind. Eng. Chem. Res.* **2017**, *56*, 6671–6684. [CrossRef]
35. Xue, J.K.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control Eng.* **2020**, *8*, 22–34. [CrossRef]
36. Oliva, D.; Aziz, M.A.E.; Hassanien, A.E. Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. *Appl. Energy* **2017**, *200*, 141–154. [CrossRef]
37. Hezagy, A.E.; Makhlouf, M.A.; El-tawel, S.G. Improved salp swarm algorithm for feature selection. *J. King Saud Univ.-Comput. Inf. Sci.* **2020**, *32*, 335–344.
38. Wang, X.W.; Wang, W.; Wang, Y. An adaptive bat algorithm. In Proceedings of the 9th International Conference on Intelligent Computing Theories and Technology, Nanning, China, 28–31 July 2013.
39. Li, J.; Luo, Y.K.; Wang, C.; Zeng, Z.G. Simplified particle swarm algorithm based on nonlinear decrease extreme disturbance and Cauchy mutation. *Int. J. Parallel Emerg. Distrib. Syst.* **2020**, *35*, 236–245. [CrossRef]
40. Wang, W.H.; Xu, L.; Chau, K.W.; Xu, D.M. Yin-Yang firefly algorithm based on dimensionally Cauchy mutation. *Expert Syst. Appl.* **2020**, *150*, 113216–113232. [CrossRef]
41. Xu, Y.T.; Chen, H.L.; Luo, J.; Zhang, Q.; Jiao, S.; Zhang, X.Q. Enhanced Moth- flame optimizer with mutation strategy for global optimization. *Inf. Sci.* **2019**, *492*, 181–203. [CrossRef]
42. Pappula, L.; Ghosh, D. Synthesis of linear aperiodic array using Cauchy mutated cat swarm optimization. *AEU-Int. J. Electron. Commun.* **2017**, *72*, 52–64. [CrossRef]
43. Farah, A.; Belazi, A.; Alqunun, K.; Almalaq, A.; Alshammari, B.M.; Ben Hamida, M.B.; Abbassi, R. A New Design Method for optimal parameters setting of PSSs and SVC damping controllers to alleviate power system stability problem. *Energies* **2021**, *14*, 7312–7337. [CrossRef]
44. Dong, J.; Dou, Z.H.; Si, S.Q.; Wang, Z.C.; Liu, L.X. Optimization of capacity configuration of wind–solar–diesel–storage using improved sparrow search algorithm. *J. Electr. Eng. Technol.* **2022**, *17*, 1–14. [CrossRef]
45. Ding, C.; Ding, Q.C.; Wang, Z.Y.; Zhou, Y.Y. Fault diagnosis of oil-immersed transformers based on the improved sparrow search algorithm optimised support vector machine. *IET Electr. Power Appl.* **2022**, *16*, 985–995. [CrossRef]

46. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995.
47. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]