mathematics

MDPI

Article

A Multi-Strategy Marine Predator Algorithm and Its Application in Joint Regularization Semi-Supervised ELM

Wenbiao Yang [1], Kewen Xia [1,*], Tiejun Li [2,*], Min Xie [1] and Fei Song [1]

[1] School of Electronics and Information Engineering, Hebei University of Technology, Tianjin 300401, China; 201921902025@stu.hebut.edu.cn (W.Y.); 201831903025@stu.hebut.edu.cn (M.X.); 201921902005@stu.hebut.edu.cn (F.S.)

[2] School of Mechanical Engineering, Hebei University of Technology, Tianjin 300401, China

* Correspondence: kwxia@hebut.edu.cn (K.X.); 1993076@hebut.edu.cn (T.L.)

Abstract: A novel semi-supervised learning method is proposed to better utilize labeled and unlabeled samples to improve classification performance. However, there is exists the limitation that Laplace regularization in a semi-supervised extreme learning machine (SSELM) tends to lead to poor generalization ability and it ignores the role of labeled information. To solve the above problems, a Joint Regularized Semi-Supervised Extreme Learning Machine (JRSSELM) is proposed, which uses Hessian regularization instead of Laplace regularization and adds supervised information regularization. In order to solve the problem of slow convergence speed and the easy to fall into local optimum of marine predator algorithm (MPA), a multi-strategy marine predator algorithm (MSMPA) is proposed, which first uses a chaotic opposition learning strategy to generate high-quality initial population, then uses adaptive inertia weights and adaptive step control factor to improve the exploration, utilization, and convergence speed, and then uses neighborhood dimensional learning strategy to maintain population diversity. The parameters in JRSSELM are then optimized using MSMPA. The MSMPA-JRSSELM is applied to logging oil formation identification. The experimental results show that MSMPA shows obvious superiority and strong competitiveness in terms of convergence accuracy and convergence speed. Also, the classification performance of MSMPA-JRSSELM is better than other classification methods, and the practical application is remarkable.

Keywords: marine predator algorithm; learning strategy; semi-supervised extreme learning machine; oil layer identification

Citation: Yang, W.; Xia, K.; Li, T.; Xie, M.; Song, F. A Multi-Strategy Marine Predator Algorithm and Its Application in Joint Regularization Semi-Supervised ELM. *Mathematics* **2021**, *9*, 291. https://doi.org/10.3390/math9030291

Academic Editor: Javier Alcaraz
Received: 31 December 2020
Accepted: 29 January 2021
Published: 1 February 2021

1. Introduction

Swarm intelligence algorithms [1] mainly simulate the behavior of a group of animals in search of food in a cooperative manner, where each member of the group learns from his or her own experience and the experience of the whole group, and changes the direction of the prey search accordingly [2]. Swarm intelligence algorithms can effectively solve many complex and challenging optimization problems in the field of artificial intelligence [3], and are mainly applied to combinatorial optimization [4], feature selection [5], image processing [6], data mining [7], and other fields.

In recent years, new meta-heuristic algorithms have been proposed to mimic the natural behavior of birds, bees, fish, and other groups of organisms, including Particle Swarm Optimization (PSO) [8], Gray Wolf Optimization (GWO) [9], Moth Flame Optimization (MFO) [10], Seagull Optimization Algorithm (SOA) [11], Sine Cosine Algorithm (SCA) [12], Whale Optimization Algorithm (WOA) [13], Coyote Optimization Algorithm (COA) [14], Carnivorous Plant Algorithm (CPA) [15], Transient Search Algorithm (TSA) [16], and more. In 2020, Faramarzi et al. [17] proposed novel meta-heuristic algorithms that mimic marine hunting behavior, and they believe that the Lévy flight strategy and the Brownian motion strategy that balance the movement of marine organisms can effectively solve the optimization problem. Each hunting process is divided into three stages. The first stage

is the high-speed movement of the prey; the predator keeps a low-speed movement, and the second stage high-speed movement is carried out for both the prey and the predator. In the third stage, the predator moves much faster than the prey. According to the vortex formation and the fish gathering device effect, it can prevent premature convergence.

Since MPA exhibits significant superiority in solving optimization problems, it is now used to solve both continuous and engineering optimization problems, exhibiting good performance. Elaziz et al. [18] proposed a hybrid MPA and MFO algorithm (MPAMFO) to solve the problem of MPA's weak ability to avoid local optimality, replacing the local search method of MPA with MFO, effectively improving the development capability of MPA, and using it for the multi-level threshold segmentation (MLT) of images with excellent performance. Abdel et al. [19] proposed an improved marine predator algorithm (IMPA) that introduces a ranking-based diversity reduction (RDR) strategy to directly replace the position of continuously poorly adapted search agents with the best individual position, improving the performance of the MPA and speeding up the convergence. Naga et al. [20] proposed a hybrid MPA and an adaptive differential evolution algorithm based on success history, which improves the global search and local search capabilities of finding the optimal solution, and is used to solve the optimal parameter combination in the single diode model. To address the limitations of the MPA's exploratory and development capabilities, Ridha [21] combined MPA with Lambert W-functions and applied this method to solve the problem of optimizing parameter combinations in single-diode and dual-diode PV models. Yousri et al. [22] proposed an MPA-based robust photovoltaic array reconfiguration strategy for solar PV arrays that need to ensure both maximum power under weak lighting conditions and no damage under intense heat conditions. Soliman et al. [23] proposed a phototransistor (TDPV) model for the problem of photovoltaic losses in photovoltaic power plants, but it has nine parameters and cannot be solved directly. They proposed using MPA to solve the optimal parameter distribution, and finally a new high-precision TDPV model was obtained. For the optimal reactive power dispatch (ORPD) of ground efficiency due to highly stochastic wind speed and solar illumination, Ebeed et al. [24] propose an optimal combination of parameters to solve the ORPD by MPA to ensure that the system's resource waste is minimized. For the deployment and training of convolutional neural network (CNN), it is challenging to perform feature extraction quickly. Sahlol et al. [25] proposed a feature extraction method combining CNN and swarm intelligence algorithm; the combined fractional order algorithm and ocean predator algorithm (FO- MPA) is used to improve the optimization performance and convergence speed of MPA. The proposed feature extraction method not only has good performance but also reduces the computational complexity.

The classification methods that have been widely used are Random Forests (RF) [26], Support Vector Machines (SVM) [27], and Extreme Learning Machines (ELM) [28]. To address the shortcomings of the traditional missing data filling methods, Deng et al. [29] proposed an improved Random Forest filling algorithm, which combines linear interpolation, matrix combination, and matrix transformation to solve the filling problem of the large amount of missing data of electricity. To utilize the minimum number of trees for classification, Paul et al. [30] proposed an improved random forest classifier based on the number of important and unimportant features, which iteratively removes some unimportant features. To improve the protein structure prediction performance, Kalaiselvi et al. [31] introduced an improved random forest classification (WPC-IRFC) technique based on weighted Pearson correlation, which has higher accuracy and shorter time. In response to the fact that SVMs do not adequately consider the distinction between numerical and nominal attributes, Peng et al. [32] proposed a novel SVM algorithm for heterogeneous data learning, which embeds nominal attributes into the real space by minimizing the estimated generalization error. To address the lack and contamination of supervised information, Dong et al. [33] proposed a robust semi-supervised classification using a novel correlated entropy loss function and Laplace SVM (LapSVM).

Huang et al. [34] proposed a semi-supervised extreme learning machine (SSELM) based on manifold regularization and extended ELM to semi-supervised learning. On various data sets, when auxiliary unlabeled data is available, SSELM is always better than purely supervised learning algorithms such as Support Vector Machines (SVM) and ELM. To address the strong sensitivity of the classification performance of SSELM to the quality of popular graphs, She et al. [35] proposed a regularized SSELM based on balanced graphs, combining label consistency graph (LCG) and sample similarity graph (SSG), and then optimizing the weight ratio of these two graphs to obtain an optimal neighborhood graph. To address the shortcoming that SSELM cannot mine information from nonlinear data, Zhou et al. [36] proposed a semi-supervised extreme learning machine (LRR-SSELM) based on a low-rank representation, which introduces a nonlinear classifier and a low-rank representation (LRR), and the LRR can maintain the popular structure of the original data. To enhance the feature extraction and classification performance of SSELM, She et al. [37] proposed a new hierarchical semi-supervised extremal learning machine (HSSELM) that uses the HELM method for automatic feature extraction of deep structures and then uses SSELM for classification tasks. To address the problem that manifold graph are only pre-constructed before classification and not changed later, which leads to poor model performance robustness, Ma et al. [38] proposed an adaptive safety semi-supervised extreme learning machine, which allows the model to adaptively compute the safety of unlabeled samples and adaptively construct manifold graphs. To address the problem that SSELM is sensitive to outliers in labeled samples, Pei et al. [39] proposed a new robust SSELM that uses a nonconvex squared loss function to impose a constant penalty on outliers and mitigate their possible negative effects.

SSELM is based on the ELM with the addition of Laplace regularization, and mitigates the drawback that sample imbalance tends to lead to poor generalization performance by assigning weights to different classes [34]. SSELM greatly expands the practicality of the ELM algorithm, and also retains all the advantages of ELM, such as high training efficiency and the simple implementation of multi-class classification problems. Laplace regularization [40] utilizes the L2 norm of the function gradient, so when solving classification or regression problems, its minimization makes the optimal function close to the constant function, further destroying the local topology and inferred power between samples. Hessian regularization [41] utilizes the L2 norm of the Hessian generic function of a function whose minimization will make the optimal function close to a linear function, capable of adaptively changing the geodesic function with distance, with a more effective preservation of the local topology between samples as well as the ability to characterize the intrinsic local geometric properties. Hessian regularization shows superior performance over Laplace regularization in predicting data points outside the region boundaries.

Therefore, in this paper, a novel semi-supervised learning method is proposed to extract the hidden information of large amount of unlabeled data and to obtain an efficient classification method with only a small number of labeled samples for training. Since SSELM takes a Laplace regularization with poor inferential ability and does not further exploit label information, resulting in its limited classification performance and poor generalization, the Joint Regularized Semi-Supervised Extreme Learning Machine (JRSSELM) is proposed. Using Hessian regularization with better inferencing power and better local prevalence structure, a supervised information regularization term that further exploits the information of tagged samples is introduced. Since JRSSELM and SSELM maintain consistency in parameter selection, with input weights and hidden layer thresholds chosen randomly and not adjusted subsequently, this, along with the selection of a grid of hyperparameters, can lead to limited performance and poor generalization. A Multi-Strategy Marine Predator Algorithm is proposed to address the limitations of MPA development and exploration capabilities, as well as the slow convergence rate. First, to address the problem that the traditional random initialization of populations is prone to generate low-quality initial populations, the initialization of populations is based on chaotic tent mapping and opposing learning strategies to ensure the generation of high-quality initial

populations. Secondly, to balance the global extensive search in the early stage and the local fine search in the later stage, adaptive inertial weights and adaptive step control factors are adopted, which effectively enhance the exploration and exploitation capability of the algorithm. Furthermore, as the number of iterations increases, individuals in the population tend to lose their diversity and it is difficult to avoid the local optimum, so the proposed neighbor dimensional learning strategy ensures the diversity of the population in each iteration.

In order to verify the superiority of MSMPA, it is compared with 7 well-known algorithms (MPA, PSO, GWO, WOA, MFO, SOA, and SCA) on 18 classic benchmark test functions and CEC2017 competition test functions. The experimental results indicate that MSMPA shows significant competitiveness, enhances global search and local search capabilities, and accelerates the convergence speed. MSMPA is proposed to optimize the JRSSELM model with respect to the parameters mentioned in JRSSELM. JRSSELM and MSMPA-JRSSELM are applied to logging oil layer identification. The results show that JRSSELM and MSMPA-JRSSELM are significantly better than other classification method in terms of performance, especially MSMPA-JRSSELM has higher classification accuracy and stability compared to other models while ensuring not overfitting.

The rest of the paper is structured as follows. Section 2 describes the introduction of the ocean predator algorithm and its improvements. Section 3 presents and analyzes the experimental results of MSMPA. Section 4 describes the Semi-Supervised Extreme Learning Machine and its improvements. Section 5 presents and analyzes the experimental results for oil logging applications. Section 6 summarizes the work of this paper and provides an outlook for the future. This is followed by the Abbreviations section and the Appendix A section.

2. Marine Predator Algorithm and Its Improvement

2.1. Basic Marine Predator Algorithm

2.1.1. Population Location Initialization

MPA is a novel meta-heuristic algorithm that mimics marine predation by adopting a randomized localization rule for the initial population, with the following mathematical expression:

$$X_{ij} = lb + \ \mathrm{r}*(ub - lb)\ i = 0 \ldots n,\ j = 0 \ldots d \tag{1}$$

where X_{ij} represents the coordinates of the j-th dimension of the i-th population, n is the number of populations, d is the dimension, ub and lb are the upper and lower boundaries of the search space, respectively, and r is a random number between [0, 1].

Based on the location of the search agent, a matrix of prey can be constructed, as follows:

$$\text{Prey} = \begin{bmatrix} X_{1,1} & X_{1,2} & \cdots & X_{1,d} \\ X_{2,1} & X_{2,2} & \cdots & X_{2,d} \\ X_{3,1} & X_{3,2} & \cdots & X_{3,d} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ X_{n,1} & X_{n,2} & \cdots & X_{n,d} \end{bmatrix}_{n \times d} \tag{2}$$

Inspired by the law of the jungle of survival of the fittest, it is believed that the best hunters are gifted in predation, so the optimal individual is used as the ontology to replicate

$n-1$ identical predators, and these n hunters are formed into an elite matrix, as shown in the following equation:

$$\text{Elite} = \begin{bmatrix} X_{1,1}^I & X_{1,2}^I & \cdots & X_{1,d}^I \\ X_{2,1}^I & X_{2,2}^I & \cdots & X_{2,d}^I \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ X_{n,1}^I & X_{n,2}^I & \cdots & X_{n,d}^I \end{bmatrix}_{n \times d} \tag{3}$$

where X^I denotes the optimal individual vector. The search agent includes both predators and prey. As such, they are both searching for their food, and after each iteration, the elite position is updated based on adaptation.

2.1.2. Exploratory Phase of High-Speed Ratio

The standard Brownian motion [31] is a stochastic process for which the step is given by a probability function defined by the zero mean ($\mu = 0$) and unit variance ($\sigma^2 = 1$) of the normal (Gaussian) distribution. The control density function of the motion at point x is as follows:

$$f_B(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \tag{4}$$

At the beginning of an iteration, i.e., the period, where is the current iteration and the maximum number of iterations, the prey is frantically searching for food, while the predator adopts a no-movement strategy. Thus, this is the high-speed ratio case, where the prey movement trajectory proceeds as follows.

$$\begin{gathered} \overrightarrow{\text{stepsize}}_i = \overrightarrow{R}_B \otimes \left(\overrightarrow{\text{Elite}}_i - \overrightarrow{R}_B \otimes \overrightarrow{\text{Prey}}_i\right) i = 1, \ldots n \\ \overrightarrow{\text{Prey}}_i = \overrightarrow{\text{Prey}}_i + P \cdot \overrightarrow{R} \otimes \text{stepsize}_i \end{gathered} \tag{5}$$

where $\overrightarrow{R}_B$ is a vector of random numbers resulting from the normal distribution of Brownian motion, $\otimes$ is an element-by-element multiplication, P is a step control factor of constant 0.5, and $\overrightarrow{R}$ is a vector of random numbers between [0, 1]. The multiplicative $\overrightarrow{R}_B$ simulates the high-speed movement of prey.

2.1.3. Mid-Speed Ratio Transition Phase

In the middle stage, $\frac{1}{3}$Max_Iter $<$ Iter $<$ $\frac{2}{3}$Max_Iter, the predator takes the same rate of position update as the prey, and since exploration in the previous section has been going on for some time, this time transitions the behavior of half of the population to the exploitation stage. Obviously, both exploration and exploitation are equally important, and at this point, the prey is primarily responsible for exploitation, while the hunter is responsible for exploration. In this phase, then, the prey updates its position according to the Lévy movement, and the predator takes the Brownian movement.

Lévy flight is a special kind of random walking strategy in which the distribution of walking steps obeys a probability distribution of heavy-tailed features, called the Lévy distribution [42]. It can usually be approximated as a simple power function distribution $L(s) \sim |s|^{-1-\beta}$, where $0 < \beta \leq 2$, R_L is the step length, and $L(s)$ is the probability of a moving step s. In the algorithm of Mantegna et al. [43], the Lévy flight step can be defined as:

$$R_L = \frac{u}{|v|^{1/\beta}} \tag{6}$$

where u and v are random numbers that are normally distributed, i.e., $u \sim \mathrm{N}(0, \sigma_u^2)$.

For the first half of the population, the position is updated by the following formula:

$$\begin{aligned} \mathrm{stepsize}_i &= \vec{R}_L \otimes \left(\vec{\mathrm{Elite}}_i - \vec{R}_L \otimes \vec{\mathrm{Prey}}_i \right) i = 1, \ldots, n/2 \\ \vec{\mathrm{Prey}}_i &= \vec{\mathrm{Prey}}_i + P \cdot \vec{R} \otimes \vec{\mathrm{stepsize}}_i \end{aligned} \tag{7}$$

where $\vec{R}_L$ is a vector of random numbers generated by the Lévy flight step. It can be seen that the Lévy stride is very useful for development.

For the latter half of the population, the exploration behavior is still performed, and the simulated motion is updated by the following formula:

$$\begin{aligned} \vec{\mathrm{stepsize}}_i &= \vec{R}_B \otimes \left(\vec{R}_B \otimes \vec{\mathrm{Elite}}_i - \vec{\mathrm{Prey}}_i \right) i = n/2, \ldots, n \\ \vec{\mathrm{Prey}}_i &= \vec{\mathrm{Elite}}_i + P \cdot CF \otimes \overline{\mathrm{stepsize}_i} \\ \textit{where } CF &= \left(1 - \frac{\mathrm{Iter}}{\mathrm{Max\_Iter}} \right)^{\left(2 \frac{\mathrm{Iter}}{\mathrm{Max_{Iter}}}\right)} \end{aligned} \tag{8}$$

where CF is an adaptive parameter used to control the predator's moving stride.

2.1.4. Low-Speed Ratio Development Phase

The later stages, i.e., Iter $> \frac{2}{3}$Max_Iter, are also called the low ratio stages because they set the prey to move much slower than the hunter. The predator is eager to hunt prey, and the movement of the population is all exploitation, so the Lévy stride strategy is adopted and the position is updated as follows:

$$\begin{aligned} \vec{\mathrm{stepsize}}_i &= \vec{R}_L \otimes \left(\vec{R}_L \otimes \vec{\mathrm{Elite}}_i - \vec{\mathrm{Prey}}_i \right) i = 1, \ldots, n \\ \vec{\mathrm{Prey}}_i &= \vec{\mathrm{Elite}}_i + P \cdot CF \otimes \vec{\mathrm{stepsize}}_i \end{aligned} \tag{9}$$

2.1.5. Eddy Current Formation and Fish Aggregation Device Effects (FADS)

In order to circumvent the local optimal solution, Faramazi et al., taking into account that other external environmental disturbances may affect the movement of the population to a greater or lesser extent, proposed the following mechanism of position updating based on the eddy current formation and the fish aggregating device effect (FADS), which is mathematically expressed as follows:

$$\vec{\mathrm{Prey}}_i = \begin{cases} \vec{Prey}_i + CF \left[\vec{X}_{\min} + \vec{R} \otimes \left(\vec{X}_{\max} - \vec{X}_{\min} \right) \right] \otimes \vec{U} \text{ if } r \leq FADS \\ \vec{Prey}_i + [FADs(1-r) + r] \left(\vec{Prey}_{r1} - \vec{Prey}_{r2} \right) \text{ if } r > FADs \end{cases} \tag{10}$$

where FADS is specified to represent the probability of influencing the search process and is set equal to 0.2, $\vec{X}_{\max}$ is a vector consisting of the maximum value of the search boundary, $\vec{X}_{\min}$ is a vector consisting of the minimum value of the search boundary, and the subscripts $r1$ and $r2$ represent the random index of the predation matrix, which is a binary vector consisting of 0 and 1.

Because marine predators have a good memory, they are guaranteed to visit prey-rich areas after a successful hunt. Adaptation is calculated for the population after each iteration and the elite matrix is replaced if there is a better-adapted position, a process that ensures that the quality of the elite increases with the number of iterations.

2.2. Multi-Strategy Marine Predator Algorithm—MSMPA

2.2.1. Chaotic Opposition Learning Strategy

Chaotic mappings have been widely used in the optimization of intelligent algorithms due to their regularity, randomness, and traversability [44], but different chaotic mappings have a great influence on the chaotic optimization process [45]. The Tent mapping has good traversal uniformity and fast iterations and produces a uniform distribution of chaotic sequences between [0, 1]. The Tent mapping expression is as follows:

$$\lambda_{t+1} = \begin{cases} \lambda_t/\alpha, \lambda_t \in [0, \alpha) \\ (1-\lambda_t)/(1-\alpha), \lambda_t \in [\alpha, 1] \end{cases} \quad t = 0, 1, 2, \cdots, T \tag{11}$$

where λ_t is the number of chaos at the t-th iteration, T is the maximum number of iterations, and α is a customizable parameter between (0, 1). Note that when $\alpha = 0.5$, the system shows a short-period state, which is not suitable for population mapping, so $\alpha = 0.7$ is chosen in this experiment.

The resulting chaotic variable λ is used for the initial population generation according to the search boundary $[lb, ub]$, as follows:

$$X_i^j = lb + \lambda_j \times (ub - lb) \tag{12}$$

where X_i^j is the j-th dimensional coordinate of the i-th search agent, and λ_j is the coordinate of the j-th dimension of λ.

Even if the chaotic sequence is capable of generating a diverse and well-distributed population, it is undeniable that there may be better search agents on opposing sides of the search space, and the same number of opposing populations will be generated again, as follows:

$$X_{op\_i} = X_{\max} + X_{\min} - X_i \tag{13}$$

where X_{op_i} is the opposing position of the i-th agent and X_i is the position of the i-th individual.

Then, we merge them into a 2**n* population, calculate the adaptation values for each individual, and rank them in order from smallest to largest, with the top n best-adapted individuals as the initial population and the best-adapted individuals as the parent of the Elite matrix.

2.2.2. Adaptive Inertial Weights and Step Control Factors

The inertia weights are inspired by the particle swarm algorithm, which plays a decisive role in the algorithm's search ability and convergence speed. In the original MPA algorithm, the inertia weights are constant values, which constrain the algorithm's global and local search capabilities [46].

For early iterations, large inertia weights enhance the global search capability; in later iterations, the algorithm needs to perform local fine search, so small inertia weights enhance the local search capability and speed up convergence [47].

In the MPA search process, the choice of inertia weighting strategy is crucial, and thus a strategy is proposed that adaptively changes the inertia weights according to the number of iterations. In the early iterations, the inertia weights are large and decrease rapidly, which is focused on global search; in the later iterations, the inertia weights are small and decrease slowly, which is to be more detailed in the local fine search [48]. The adaptive inertia weights are expressed as follows:

$$w = a \ \cos^b(\ln(1 + e^{\frac{\text{Iter}}{\text{Max\_Iter}}}) + c \tag{14}$$

where a, b, and c are optional parameters. After the experimental analysis, a, b, and c were set to 20, 12, and 0.2, respectively.

The adaptive inertia weight curves are shown in Figure 1.

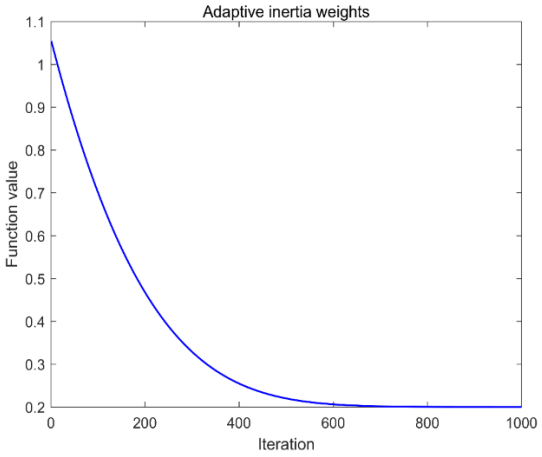

Figure 1. The plot of Adaptive Weights.

From Figure 1, it can be seen that after the introduction of adaptive inertial weights, the position of the prey will be updated by the following equation.

$$\overrightarrow{\text{Prey}}_i = w * \overrightarrow{\text{Prey}}_i + P \cdot \vec{R} \otimes \text{stepsize}_i \tag{15}$$

$$\overrightarrow{\text{Prey}}_i = w * \overrightarrow{\text{Elite}}_i + P \cdot CF \otimes \overline{\text{stepsize}}_i \tag{16}$$

Also, the step length control is particularly important for MPA, in the early hope that the step length influence is as large as possible to make the prey better traversal search space; later, a small step length influence is needed to avoid falling into the local optimum, but also local fine search, while in the original MPA, the step length control factor is constant 0.5, which will limit the performance of the algorithm. Therefore, in this paper, an adaptive step length control factor is proposed, which not only ensures the global search demand of the early traversability but also enhances the local fine search capability at the later stage. The mathematical expression is as follows:

$$P = m \ \sin(\frac{\pi}{5} \times (\frac{p}{e^{\frac{n \times \text{Iter}}{\text{Max\_Iter}}}})) + q \tag{17}$$

where m, n, p, and q are optional parameters. After experimental analysis, m, n, p, and q were set to be 1.2, 10, 2, and 0.2, respectively.

The curve of the adaptive step control factor is shown in Figure 2.

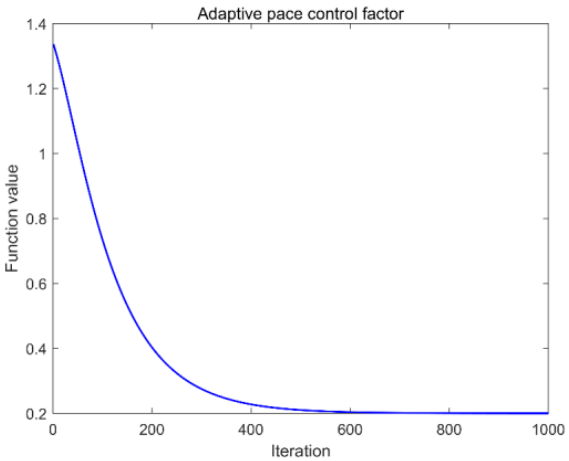

Figure 2. The plot of the adaptive step control factor.

From Figure 2, it can be seen that the early values are large and rapidly decreasing, which enhances the influence of step size, i.e., strengthening the traversability of the global search, and the values are small and slowly decreasing in order to take into account the later local fine search.

2.2.3. Neighborhood Dimensional Learning Strategy (NDL)

The population diversity plays a key role in the convergence speed and accuracy of the algorithm. The population diversity of the original MPA decreases gradually with the iteration of the algorithm, which tends to lead to the local optimum and not the global optimum when solving high-dimensional complex problems [49]. Therefore, in order to ensure that the population diversity remains rich in each iteration, a Neighborhood Dimensional Learning strategy is proposed.

First, at the end of each iteration of the original MPA, a candidate population of the same size is generated for the original population, and it is taken to relocate the new population using either optimal individual information or information about itself, as follows:

$$\overrightarrow{X_{CAND\_i}}(t) = \begin{cases} w * \vec{X}^{*}(t) + 2 \times (r - 0.5) \times (ub - lb \cdot r + lb) & p > 0.5 \\ w * \vec{X}_i(t) + 2 \times (r - 0.5) \times (ub - lb \cdot r + lb) & p < 0.5 \end{cases} \tag{18}$$

where $\vec{X}_i(t)$ is the position vector of the i-th search agent, $\overrightarrow{X_{CAND\_i}}(t+1)$ is the position vector of the i-th candidate individuals, and $\vec{X}^{*}(t)$ is the position vector of the elite individuals, and p is the random probability.

Second, based on the Euclidean distance between the current position $\vec{X}_i(t)$ and the candidate position $\overrightarrow{X_{CAND}}(t+1)$, a neighborhood radius is computed by the following equation:

$$R_i(t) = ||\vec{X}_i(t) - \overrightarrow{X_{CAND\_i}}(t+1)|| \tag{19}$$

Then, according to $R_i(t)$, the Euclidean distance which is less than the radius search agent is successively selected from the population, and these individuals are saved as neighbors of the i-th individual. The mathematical expression is as follows:

$$N_i(t) = \left\{ X_j(t) \mid D_i\left(X_i(t), X_j(t)\right) \le R_i(t), X_j(t) \in \text{population} \right\} \tag{20}$$

where $N_i(t)$ denotes the set of individual neighbors and D_i denotes performing a European distance operation.

The next step in neighborhood dimension learning is to update the dimensional coordinates of the current individual using some dimensional information from the population of neighbors and the dimensional information of an individual randomly selected from the entire population, as expressed mathematically as follows:

$$X_{NDL\_i,d}(t+1) = X_{i,d}(t) + \text{sign}(r - 0.5) \times (X_{n,d}(t) - X_{r,d}(t)) \tag{21}$$

where $X_{i,d}(t)$ represents the d-dimensional information of the i-th search agent, $X_{NDL\_i,d}(t+1)$ represents the new d-dimensional information after passing the NDL, $X_{n,d}(t)$ represents the d-dimensional information of the neighboring individuals, and $X_{r,d}(t)$ represents the d-dimensional information of the randomly selected individuals.

Finally, the fitness of the candidate population and the NDL population is calculated to select the newer individuals, as shown by the following mathematical expression.

$$X_i(t+1) = \begin{cases} X_{CAND\_i}(t+1) & \text{if } f(X_{CAND\_i}(t+1)) < f(X_{NDL\_i}(t+1)) \\ X_{NDL\_i}(t+1) & \text{if } f(X_{CAND\_i}(t+1)) < f(X_{NDL\_i}(t+1)) \end{cases} \tag{22}$$

The pseudo-code of the proposed MSMPA is shown in Algorithm 1.

Algorithm 1. Pseudo-code of the MSMPA

Input: Number of Search Agents: N, Dim, Max_Iter
Output: The optimum fitness value
Generate chaotic tent mapping sequences by the Equation (11)
Initialized populations by the Equation (12)
Generation of Opposing Populations by the Equation (13)
Selection of the first N well-adapted individuals as the first generation of the population
While Iter < Max_Iter
Calculating Adaptive Inertia Weights by the Equation (14)
Calculating Adaptive Step Control Factors by the Equation (17)
Calculate fitness values and construct an elite matrix
If Iter < Max_Iter/3
    Update prey by the Equation (15)
Else if Max_Iter/3 < Iter < 2∗Max_Iter/3
    For the first half of the populations (i = 1, . . . , $n/2$)
    Update prey by the Equation (15)
    For the other half of the populations (i = $n/2$, . . . , n)
    Update prey by the Equation (16)
Else if Iter > 2∗Max_Iter/3
Update prey by the Equation (16)
End If
Generation of candidate populations by the Equation (18)
Calculating Neighborhood Radius by the Equation (19)
Finding Neighborhood Populations by the Equation (20)
Calculation of NDL populations by the Equation (21)
Calculate fitness values to update population position by the Equation (22)
Updating Memory and Applying FADs effect and update by the Equation (10)
Calculate the fitness value of the population by the fitness function
Update the current optimum fitness value and the position of the best Search Agent
End while
Return the optimum fitness value

3. Simulation Experiments and Comparative Analysis

3.1. Experimental Environment and Algorithm Parameters

In this section, in order to verify the superiority of the proposed MSMPA in solving the large-scale optimization problem, it is shown that the proposed MSMPA can be used to solve the large-scale optimization problem. Eighteen classical benchmarking functions [50] and 30 optimization functions from the CEC2017 competition [51] were selected and tested against seven popular algorithms (MPA [17], PSO [8], GWO [9], WOA [13], MFO [10], SOA [11], and SCA [12]). In order to ensure the fairness of the comparison between evolutionary and swarm intelligence algorithms, the number of iterations is chosen to be replaced by the number of function evaluations in this paper. The population size of the experiment is 30, the maximal number of function evaluation is 30,000 with 30 independent runs on each function. The experimental environment is Windows 10 64 bit, MATLAB R2016A, Intel(R) Core CPU (i5-10210U 2.1GHz), 16G RAM. To ensure the fairness of the comparison experiments, the parameter settings in the original literature are maintained in this paper for each competing algorithm. The specific parameter settings are shown in Table 1.

Table 1. Parameter settings for each algorithm.

Algorithm	Parameter Specific Settings
PSO	$c_1 = 2, c_2 = 2$
GWO	$a = 2 * \left(1 - \frac{t}{T_{\max}}\right)$
WOA	$a = 2 * \left(1 - \frac{t}{T_{\max}}\right)$
MFO	Convergence Constant $\in [-1, -2]$, Logarithmic Spiral:0.75
SOA	$f_c = 2$, Control Parameter (A) $\in [2, 0]$
SCA	$r_1 = a - t\frac{a}{T}$
MPA	$FADs = 0.2 \quad P = 0.5$
MSMPA	$w = a\cos^b(\ln(1 + e^{\frac{\text{Iter}}{\text{Max\_Iter}}}) + c \quad P = m\sin[\frac{\pi}{5} \times (\frac{p}{e^{\frac{\text{n}\times\text{Iter}}{\text{Max\_Iter}}}})] + q \quad FADs = 0.2$

3.2. Benchmark Test Functions

The details of the 18 benchmark functions are shown in Table 2. Table 2 includes seven high-dimensional single-peak functions (F1–F7), six high-dimensional multi-peak functions (F8–F13), and five fixed-dimensional multi-peak functions (F14–F18). Since the high-dimensional single-peak function has only one peak, it is used to test the local convergence and convergence speed of the algorithm; the high-dimensional multipeak function and the fixed-dimensional multipeak function have multiple peaks and only one global optimum, so they are used to test the algorithm's global optimum search and the ability to jump out of the local optimum.

Table 2. Description of the 18 classical benchmarking functions.

Type	Function	Dim	Range	Optimum Value
Unimodal	$F_1(x) = \sum_{i=1}^{n} x_i^2$	50	$[-100, 100]$	0
Unimodal	$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	50	$[-10, 10]$	0
Unimodal	$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j\right)^2$	50	$[-100, 100]$	0
Unimodal	$F_4(x) = \max_i\{\lvert x_i \rvert, 1 \le i \le n\}$	50	$[-100, 100]$	0
Unimodal	$F_5(x) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	50	$[-30, 30]$	0
Unimodal	$F_6(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$	50	$[-100, 100]$	0
Unimodal	$F_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	50	$[-1.28, 1.28]$	0
Multimodal	$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	50	$[-500, 500]$	$-418.9829 \times$d
Multimodal	$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	50	$[-5.12, 5.12]$	0
Multimodal	$F_{10}(x) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	50	$[-32, 32]$	0
Multimodal	$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	50	$[-600, 600]$	0

Table 2. *Cont.*

Type	Function	Dim	Range	Optimum Value
Multimodal	$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right] + (y_n-1)^2\right\} + \sum_{i=1}^{n} u(x_i,10,100,4)$ $where\ y_i = 1+\frac{x_i+1}{4}\ u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i-a)^m & x_i < -a \end{cases}$	50	$[-50, 50]$	0
Multimodal	$F_{13}(x) = 0.1\left\{\begin{array}{l}\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i-1)^2\left[1+\sin^2(3\pi x_i+1)\right] + \\ (x_n-1)^2\left[1+\sin^2(2\pi x_n)\right]\end{array}\right\} + \sum_{i=1}^{n} u(x_i,5,100,4)$	50	$[-50, 50]$	0
Fixed Dimension	$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	$[-65, 65]$	1
Fixed Dimension	$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	$[-5, 5]$	0.00030
Fixed Dimension	$F_{16}(x) = -\sum_{i=1}^{5}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	−10.1532
Fixed Dimension	$F_{17}(x) = -\sum_{i=1}^{7}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	−10.4028
Fixed Dimension	$F_{18}(x) = -\sum_{i=1}^{10}\left[(X-a_i)(X-a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	−10.5363

To further test the performance of the algorithm, the CEC2017 Numerical Optimization Competition Suite functions were also selected, including 3 single-peak functions, 7 multi-peak functions, 10 mixed functions, and 10 composite functions. CEC2017 Numerical Optimization functions are shown in Table 3. The dimensionality of all functions is set to 10.

Table 3. CEC2017 numerical optimization functions.

Type	No.	Function Name	Range	Optimum Value
Unimodal	CF1	Shifted and Rotated Bent Cigar Function	$[-100, 100]$	100
Unimodal	CF2	Shifted and Rotated Sum of Different Power Function	$[-100, 100]$	200
Unimodal	CF3	Shifted and Rotated Zakharov Function	$[-100, 100]$	300
Multimodal	CF4	Shifted and Rotated Rosenbrock's Function	$[-100, 100]$	400
Multimodal	CF5	Shifted and Rotated Rastrigin's Function	$[-100, 100]$	500
Multimodal	CF6	Shifted and Rotated Expanded Scaffer's F6 Function	$[-100, 100]$	600
Multimodal	CF7	Shifted and Rotated Lunacek Bi_Rastrigin Function	$[-100, 100]$	700
Multimodal	CF8	Shifted and Rotated Non-Continuous Rastrigin's Function	$[-100, 100]$	800
Multimodal	CF9	Shifted and Rotated Levy Function	$[-100, 100]$	900
Multimodal	CF10	Shifted and Rotated Schwefel's Function	$[-100, 100]$	1000
Hybrid	CF11	Hybrid Function 1 (N = 3)	$[-100, 100]$	1100
Hybrid	CF12	Hybrid Function 2 (N = 3)	$[-100, 100]$	1200
Hybrid	CF13	Hybrid Function 3 (N = 3)	$[-100, 100]$	1300
Hybrid	CF14	Hybrid Function 4 (N = 4)	$[-100, 100]$	1400
Hybrid	CF15	Hybrid Function 5 (N = 4)	$[-100, 100]$	1500
Hybrid	CF16	Hybrid Function 6 (N = 4)	$[-100, 100]$	1600
Hybrid	CF17	Hybrid Function 6 (N = 5)	$[-100, 100]$	1700
Hybrid	CF18	Hybrid Function 6 (N = 5)	$[-100, 100]$	1800
Hybrid	CF19	Hybrid Function 6 (N = 5)	$[-100, 100]$	1900
Hybrid	CF20	Hybrid Function 6 (N = 6)	$[-100, 100]$	2000

Table 3. *Cont.*

Type	No.	Function Name	Range	Optimum Value
Composition	CF21	Composition Function 1 (N = 3)	[−100, 100]	2100
Composition	CF22	Composition Function 2 (N = 3)	[−100, 100]	2200
Composition	CF23	Composition Function 3 (N = 4)	[−100, 100]	2300
Composition	CF24	Composition Function 4 (N = 4)	[−100, 100]	2400
Composition	CF25	Composition Function 5 (N = 5)	[−100, 100]	2500
Composition	CF26	Composition Function 6 (N = 5)	[−100, 100]	2600
Composition	CF27	Composition Function 7 (N = 6)	[−100, 100]	2700
Composition	CF28	Composition Function 8 (N = 6)	[−100, 100]	2800
Composition	CF29	Composition Function 9 (N = 3)	[−100, 100]	2900
Composition	CF30	Composition Function 10 (N = 3)	[−100, 100]	3000

3.3. Experimental Results and Analysis

The convergence curve is a visual representation of the ability to evaluate the development and speed of convergence of the algorithm to find the best, and the convergence curves of MSMPA and seven other comparison algorithms on 18 benchmark functions are shown in Figure A1 in Appendix A. From Figure A1, we can see that the convergence speed of MSMPA on F1 to F4 is obviously faster than other algorithms and the end of the convergence curve is also significantly lower than other algorithms; the fastest convergence on F5 to F8, F12 and F13 is not the fastest, but the convergence accuracy of other algorithms is significantly worse than MSMPA; on F9 to F11, the convergence speed and convergence accuracy of MSMPA show significant superiority. MSMPA converges faster than the other algorithms on solid-dimensional multimodal functions, except on F15, where the convergence accuracy is poorer than that of MPA, and MSMPA exhibits significant competitiveness.

In this experiment, the average (Ave), standard deviation (Std), maximum (Max), and minimum (Min) are evaluated. The Friedman's test [52], a popular statistical test for non-parametric tests, is also used to make it easier to detect differences in the performance of individual algorithms, to compare the average performance of each method across all experimental results, and to place the results at the end of the statistical table of experimental results.

The test results of the 18 benchmark functions are shown in Table A1 in Appendix A, where the bold data are the optimal values of all the methods in the same function performance index. From Table A1, it can be seen that MSMPA performs optimally on the high-dimensional single-peak functions, especially on F1~F4, where MSMPA achieves the theoretical best value. This demonstrates the superiority of MSMPA in local search capability. In the high-dimensional multimode function, MSMPA shows obvious superiority, except for the standard deviation on F8, where SCA is the best performer, MSMPA is the best performer in other performance indexes, especially on F9 and F11, where MSMPA achieves the theoretical global optimum. On the fixed-dimensional multimode functions, except for F15, which is second to MPA, MSMPA is the best performer on all other functions, especially on F16–F18, where MSMPA is optimized to the theoretical extremes and its standard deviation is zero. MSMPA is the first place in the average rank of all functions ranked by Friedman's test.

The experimental results of CEC2017 are shown in Table A2 in Appendix A, in which the bold font in the table is the optimal value of the current function under the same evaluation index. From Table A2, it can be concluded that, in addition to the non-optimal performance on F16, F21, F22, F24, and F25, the best performance is achieved on the other functions, so the overall performance is optimal, and it is noteworthy that the theoretical extremes were achieved on CF2. The average ranking of MSMPA in the Friedman test is 1.6167, which is still the best performance, followed by MPA.

3.4. Algorithmic Stability Analysis

In order to show the stability of MSMPA and other algorithms more visually, this experiment uses a box line diagram to show the distribution of the results of each function after 30 independent tests. The numerical distribution of some of the selected functions is selected from the test functions, and the test result box diagram of the selected functions is shown in Figure A2 in Appendix A.

From Figure A2, it can be seen that MSMPA exhibits remarkable stability and superior performance over the other algorithms for all functions, as shown by the fact that the lower edge, upper edge, and median are always lower than the other algorithms for the same function.

3.5. Wilcoxon Rank Sum Test Analysis

Since there are too many random factors affecting the performance of the algorithm, a statistical test is used to compare the difference in performance between MSMPA and other algorithms. The widely used Wilcoxon's rank sum test [53] was selected, and 5% was set as the index of significance, and *p*-values were obtained from the two-rank sum analysis of MSMPA and other algorithms. If the *p*-value is greater than 5% or NaN, it means that MSMPA is not statistically significantly different on this function. Wilcoxon's rank-sum tests for MSMPA and other algorithms yielded the *p*-values shown in Table A3 in Appendix A, where the bold text in the table indicates values greater than 5% or NaN.

From Table A3, it can be concluded that on F9, the MSMPA, MPA, and WOA tests show results for NaN because on F9, all three algorithms take the theoretical optimum; both MSMPA and MPA took the theoretical optimum on F11 and CF2, with no statistically significant differences between MSMPA and GWO and WOA on F11; MSMPA was not statistically significantly different from MPA only on F14; No statistically significant differences between MSMPA and MFO on F16, F17, CF3, CF9, and CF22; No statistically significant differences between MSMPA and PSO on CF2, CF11, CF20, CF23, and CF28; MSMPA was statistically significantly different from SOA only on CF21. In general, MSMPA is statistically significantly different from other algorithms in most functions, demonstrating its superior ability to find the global optimum and jump out of the local optimum.

3.6. High-Dimensional Functional Test Analysis

MPA performs poorly when dealing with complex high-dimensional problems and tends to fall into local optimality. In order to verify the better performance of the proposed MSMPA in global search and local optimization avoidance, the dimensions of F1–F13 are set to 100, 200, and 500, respectively. It is stills run independently with the other seven algorithms at the population size of 30 and the maximum number of iterations of 1000 for 30 times, and the results are counted and used in the Friedman test. The test results for each high-dimensional function with dimensions of 100, 200, and 500 are shown in Tables A4–A6 in Appendix A, respectively.

It can be seen from Tables A4–A6 that, apart from the second better performance of MSMPA on F2 with a dimension of 500 than WOA, MSMPA is significantly more competitive than other algorithms in dealing with complex and high-dimensional problems. The final result of its Friedman test for the mean is ranked first. It is worth noting that, on F1~F4, F9 and F11, MSMPA achieved the theoretical optimal value. These just prove the stability and effectiveness of MSMPA in solving complex high-dimensional problems.

4. Semi-Supervised Extreme Learning Machines and Its Improvements

4.1. Semi-Supervised Extreme Learning Machine

The mathematical expression for the objective function of SSELM is defined by introducing a Laplacian regularization term and using information from unlabeled samples as follows:

$$\begin{array}{cc} \min\limits_{\beta \in \mathbb{R}^{n_h \times n_o}} & \frac{1}{2} \parallel \beta \parallel^2 + \frac{1}{2} \sum\limits_{i=1}^{l} C_i \parallel e_i \parallel^2 + \frac{\lambda}{2} Tr\left(F^T L F\right) \\ \text{s.t.} & h(x_i)\beta = y_i^T - e_i^T, i = 1, \ldots, l \\ & f_i = h(x_i)\beta, i = 1, \ldots, l+u \end{array} \tag{23}$$

The solution to SSELM can be obtained with the following expression:

$$\beta^* = \left(I_{n_h} + H^T C H + \lambda H^T L H\right)^{-1} H^T C \widetilde{\mathbf{Y}} \tag{24}$$

where I_{n_h} is the unit matrix of n_h.

If the number of crypto neurons is greater than the number of labeled samples, then the following solution can be adopted, with the following expression:

$$\beta^* = H^T \left(I_{l+u} + C H H^T + \lambda L H H^T\right)^{-1} C \widetilde{\mathbf{Y}} \tag{25}$$

where I_{l+u} is the unit matrix of $l+u$.

4.2. Joint Hessian and Supervised Information Regularization Semi-Supervised Extreme Learning Machine

4.2.1. Hessian Regularization

The Hessian regularization term [41] provides a simple way to establish the relationship between mappings and manifolds, which is derived from the Eells energy. The Hessian energy can also be estimated by applying a simplified form of the derivative of the second order of regular coordinates. Let $N_k(X_i)$ be the set of k hypotenuse samples at sample point Xi, and estimate the Hessian of f at Xi by computing H. The Hessian can be approximated as follows:

$$\frac{\partial^2 (X^T U)}{\partial x_r \partial x_s} \mid x_i \approx \sum_{j=1}^{k} H_{rsj}^{(i)} f\left(X_j\right) \tag{26}$$

The above equation can be solved by using linear least squares for a second-order polynomial. The ideal form of $H_{rsj}^{(i)}$. Hessian's estimate of the Frobenius paradigm is thus obtained:

$$\bigtriangledown_a \bigtriangledown_b f^2 \approx \sum_{r,s=1}^{m} \left(\sum_{\alpha=1}^{m} H_{rs\alpha}^{(i)} f_\alpha \right)^2 = \sum_{\alpha,\beta=1}^{k} f_\alpha f_\beta B_{\alpha\beta}^{(i)} \tag{27}$$

where $\boldsymbol{B}_{\alpha\beta}^{(i)} = \sum\limits_{r,s=1}^{m} H_{rs\alpha}^{(i)} H_{rs\beta}^{(i)}$ is the Hessian energy to complete all the estimates.

Given a European clustering expression for Hessian at point $\boldsymbol{Xi}$:

$$Hess(f) = \sum_{i=1}^{n} \sum_{i=1}^{n} \left(\frac{\partial^2 f}{\partial x_r x_s} X_i \right)^2 = \sum_{i=1}^{n} \sum_{N_k(X_i)\beta \in N_k(X_i)} f_\alpha f_\beta \boldsymbol{B}_{\alpha\beta}^{(i)} \tag{28}$$

where matrix $\boldsymbol{B}$ is the sum of all data points, and $\boldsymbol{n}$ represents the number of all labeled and unlabeled data.

4.2.2. Supervisory Information Regularization

Assuming that the labels of $\boldsymbol{X_i}$ and $\boldsymbol{X_j}$ are the same, then, assuming that the coefficients $Ls\_ij = 1$ at this time, the difference between their decision functions $[f(X_i) - f(X_j)]$ is also small, then the optimization problem can be defined as $\left(f(x_i) - f\left(x_j\right)\right)^2$Ls_ij with the following expression for the elements of the coefficient matrix *LS*:

$$Ls\_ij = \begin{cases} \frac{1}{Ls}, \mathrm{X}_i \text{ and } \mathrm{X}_j \text{are in the same category} \\ 0, \text{otherwise} \end{cases} \tag{29}$$

where *Ls* is the number of combinations of all labeled consistent samples, others include the case where one of the comparative combinations exists with or without labeled samples, and *i,j* = 1, 2, ..., *l*, ..., *l* + *u*.

Assuming that the labels of X_i and X_j are different at the same time, then the product of the decision function (*f*(X_i)**f*(X_j)) is small, assuming that the coefficients $Ld\_ij = 1$. The optimization problem can be defined as $f(x_i)f\left(x_j\right)Ld\_ij$, with the following expression for the elements of the coefficient matrix *LD*:

$$Ld\_ij = \begin{cases} \frac{1}{Ld}, \mathrm{X}_i \text{ and } \mathrm{X}_j \text{are in the different category} \\ 0, \text{otherwise} \end{cases} \tag{30}$$

where *Ld* is the number of combinations of all label inconsistent samples, and the others include the cases where one of the comparative combinations has an unlabeled sample.

The mathematical expression defining the regularization term of the supervision constraint is as follows:

$$\begin{aligned} S &= \sum_{i=1}^{l+u}\sum_{j=1}^{l+u}\left(f(X_i) - f\left(X_j\right)\right)^2 Ls\_ij + \sum_{i=1}^{l+u}\sum_{j=1}^{l+u} f(X_i)f\left(X_j\right)Ld\_ij \\ &= 2F^{\mathrm{T}}(LS\_D - LS)F + 2F^{\mathrm{T}}(0.5 \times LD)F \\ &= 2F^{\mathrm{T}}(LS\_D - LS + 0.5 \times LD)F \end{aligned} \tag{31}$$

where *LS* _*D* is the diagonal matrix of the matrix *LS*.

4.2.3. Joint Regularization SSELM

In order to make full use of the direct associations of various categories contained in the label information, a regularization term of supervision information is introduced to enhance SSELM. Then, the Laplacian regularization in the original SSELM is replaced with the joint Hessian regularization and supervision information regularization term, hence, JRSSELM is proposed.

Assume that the training set consists of L labeled data and U unlabeled data, represented by $\{X_l, Y_l\} = \{X_i, Y_i\}_{i=1}^{L}$ and $X_u = \{X_i\}_{i=1}^{U}$, respectively. Among the L labeled data, there are Ls combinations with the same label, and Ld combinations with different labels. The Hessian regularization and supervision information regularization terms are introduced to replace the original Laplace regularization terms, and the objective function optimized by JRSSELM can be obtained. The mathematical expression is defined as follows:

$$\begin{gathered} \min_{\beta \in \mathbf{R}^{n_h \times n_o}} \frac{1}{2} \parallel \beta \parallel^2 + \frac{1}{2}\sum_{i=1}^{l} c_i {e_i}^2 + \frac{\lambda_1}{2}F^{\mathrm{T}}HessF + \frac{\lambda_2}{2}S \\ \text{s.t. } h(x_i)\beta = y_i^T - e_i^T,\ i = 1, \cdots, l \\ f_i = h(x_i)\beta,\ i = 1, \cdots, u + l \end{gathered} \tag{32}$$

where λ_1 and λ_2 are tradeoff parameters.

Substituting the constraints into the objective function, a new optimized objective function can be obtained, and its expression is as follows:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n, x, n_o} \frac{1}{2} \parallel \beta \parallel^2 + \frac{1}{2} \parallel C^{\frac{1}{2}}(\widetilde{\boldsymbol{Y}} - H\beta) \parallel^2 + \frac{\lambda_1}{2}Tr\left(\beta^T H^T \mathrm{Hess} H\beta\right) + \frac{\lambda_2}{2}Tr\left[\beta^T H^T (LS\_D - LS + 0.5 \times LD)H\beta\right] \tag{33}$$

where $\widetilde{\boldsymbol{Y}} \in \mathbb{R}^{(l+u)\times n_o}$ represents the training target to be reinforced, C is a $(l+u)\times(l+u)$-dimensional matrix whose first l diagonal element is $[C]_{ii} = C_i$, and all the other elements are zero.

Then the derivative of the objective function to β is expressed as follows:

$$\nabla L_{JRSSELM} = \beta + H^T C(\widetilde{\boldsymbol{Y}} - H\beta) + \lambda_1 \cdot H^T HessH\beta + \lambda_2 \cdot H^T(LS\_D - LS + 0.5 \times LD)H\beta \quad (34)$$

If the derivative value is set to zero, then the JRSSELM solution can be obtained, and its expression is as follows:

$$\beta^* = \left[I_{n_h} + H^{\mathrm{T}}CH + \lambda_1 H^{\mathrm{T}} HessH + \lambda_2 H^{\mathrm{T}}(LS\_D - LS + 0.5 \times LD)H\right]^{-1} H^{\mathrm{T}}C\widetilde{Y} \quad (35)$$

If the number of hidden layer neurons is greater than the number of labeled samples, the following solutions can be adopted, and the expression is as follows:

$$\beta^* = H^{\mathrm{T}}\left[I_{l+u} + CHH^{\mathrm{T}} + \lambda_1 HessHH^{\mathrm{T}} + \lambda_2(LS\_D - LS + 0.5 \times LD)HH^{\mathrm{T}}\right]^{-1} C\widetilde{Y} \quad (36)$$

where I_{l+u} is the unit matrix of $l+u$.

4.3. Hybrid MSMPA and Joint Regularized Semi-Supervised Extreme Learning Machine

The input weights and hidden layer deviations in the original SSELM are generated by randomization and will not be adjusted in the future. The grid selection of its hyperparameters often leads to limited performance and poor stability of the model. The values of hyperparameters are all $\{10^{-5}, 10^{-4}, \ldots, 10^4, 10^5\}$, which can be calculated by the following mathematical expressions:

$$C_0 = 10^{6-i} \quad (37)$$

$$\lambda = 10^{6-j} \quad (38)$$

where i and j are integers between [0, 11].

JRSSELM and SSELM maintain consistency in parameter settings. The input weights and hidden layer deviations are also randomly generated and will not be updated in the future. The hyperparameters are also selected for gridding, and the value range remains the same as in SSELM. Its mathematical expression is as follows:

$$\lambda_1 = 10^{6-l} \quad (39)$$

$$\lambda_2 = 10^{6-m} \quad (40)$$

where l and m are integers between [0, 11].

MSMPA is proposed to optimize the selection of various parameters in JRSSELM. To further improve the classification performance and robustness of JRSSELM, first, input weight w, hidden layer deviation b, C_0, λ_1, and λ_2 as the coordinates of the search agent. If the number of hidden layers at this time is h_n, and the number of neurons in the input layer is h_i, then each prey dimension is $[h_n(h_i+1)+3]$. It is worth noting that the first $[h_n(h_i+1)+3]$-dimensional coordinates represent the input weight and hidden layer threshold, so its value range is $[-1, 1]$, and the last three dimensions represent the position information of C_0, λ_1 and λ_2, respectively. The range is set to $[-5, 5]$. To be consistent with the value range of JRSSELM, the calculation can be performed as follows:

$$C_0 = 10^{X_i^{M-2}} \quad (41)$$

$$\lambda_1 = 10^{X_i^{M-1}} \quad (42)$$

$$\lambda_2 = 10^{X_i^{M}} \quad (43)$$

where $M = [h_n(h_i + 1) + 3]$ is the total dimension and X_i^M represents the M-dimensional coordinates of the i-th search agents.

Secondly, the selection of the fitness function is particularly important. In order to make a fair comparison with SSELM and JRSSELM, both SSELM and JRSSELM select the optimal hyperparameter combination by calculating the prediction error of the test set. Therefore, the fitness function is also selected as the test set prediction error in MSMPA, and its mathematical expression is as follows:

$$fitness = 1 - \frac{TP + TN}{TP + TN + FP + FN} \tag{44}$$

where TP stands for true positive, TN stands for true negative, FP stands for false positive, and FN stands for false negative.

The pseudo code for the MSMPA and JRSSELM mix is shown in Algorithm 2.

Algorithm 2. Pseudo-code of the MSMPA-JRSSELM

Input: L labeled samples $\{X_l, Y_l\} = \{X_i, Y_i\}_{i=1}^{L}$
U unlabeled samples, $X_u = \{X_i\}_{i=1}^{U}$
Number of Search Agents: N, Dim, Max_Iter
Output: The best mapping function of JRSSELM:f: $\mathbb{R}^{n_i} \rightarrow \mathbb{R}^{n_o}$
Step 1: Constructing Hessian regularization terms and supervisory information regularization terms via $\boldsymbol{X}_l$ and $\boldsymbol{X}_u$.
Step 2: Generate chaotic tent mapping sequences by the Equation (11)
Initialized populations by the Equation (12)
Generation of Opposing Populations by the Equation (13)
Selection of the first N well-adapted individuals as the first generation of the population
Step 3:
If $h_n \leq N$
Calculate the output weights β^* by Equation (35)
Else if
Calculate the output weights β^* by Equation (36)
Step 4:
While Iter < Max_Iter
Update all population positions by MSMPA
Repeat step 3
Calculate the fitness of each search agent by the Equation (44)
End while
Step 5: Outputs the best search agent position and optimal value
Step 6: Optimal mapping functions of JLSSELM: $f^*(x) = h(x)\beta^*$

5. Oil Logging Oil Layer Identification Applications

5.1. Design of Oil Layer Identification System

In oil logging, the oil layer identification technology is a very complex dynamic research technology, and it is also a very critical element. Many factors affect the oil layer distribution, such as reservoir thickness, oil pressure, permeability, water content, reservoir pressure, storage effective thickness, and so on. The most important thing is the accuracy of the identification, which can assist the oil layer exploration and provide effective information for the engineers to make the decision. The MSMPA-JRSSELM proposed in this paper is highly accurate in classification and can make accurate predictions of oil layer distribution on a test set using a small number of marker samples and a large number of unmarked samples. Therefore, MSMPA-JRSSELM is applied to oil logging to verify the effectiveness of this algorithm by using oil data provided by a Chinese oil field. The block diagram of the oil layer identification system is shown in Figure 3.

From Figure 3, it can be seen that there are several major steps in oil layer identification.

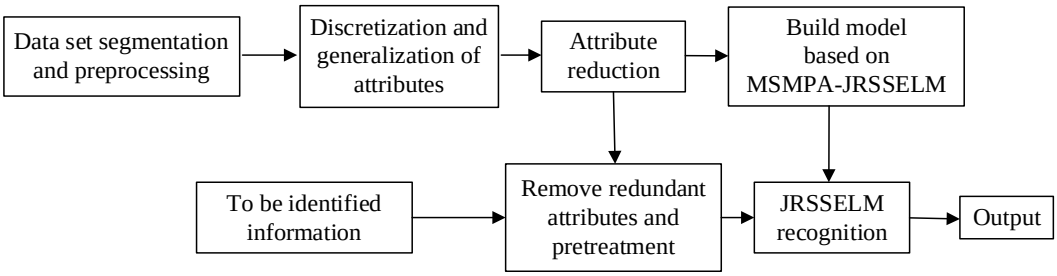

Figure 3. Block diagram of MSMPA-SSELM oil layer identification system.

Phase 1 (Data set delineation and pre-processing). The selection of the dataset should be complete and comprehensive and should be closely related to the oil layer evaluation. The data set is divided into two parts: training sample and test sample, in which a small portion of samples from the training set are selected as labeled samples.

Phase 2 (Attribute discretization and generalization). In order to approximate the attributes of the sample information, the decision attributes $D = \{d\}$,$d = \{d_i = i, i = -1, 1\}$, where -1 and 1 represent the dry layer and the oil layer, respectively. For the discretization of the conditional attributes, in order to conform to the actual geophysical characteristics, the continuous attributes are discretized by the curvilinear inflection point method [54], in which each attribute is discretized separately in turn, i.e., first, the attribute values are arranged from small to large to find the possible inflection point locations, and then the appropriate discretization points are filtered out according to the target layer range constraints.

Phase 3 (Attribute reduction of data information). Since the degree of decisiveness of each conditional attribute on the oil layer distribution is not consistent, there are more than ten conditional attributes in the logging data, but only a few conditional attributes play a decisive role, so it is necessary to eliminate other redundant attributes to avoid algorithm redundancy. In this paper, we adopt a Rough Set based on consistent coverage for attribute reduction [55].

Phase 4 (Training the MSMPA-JRSSELM classifier). In the MSMPA-JRSSELM model, the sample information is input after attribute reduction, and training is carried out using the JRSSELM model. The MSMPA is used to find better training parameters to improve the model recognition accuracy until the iterations are completed, and the optimal MSMPA-JRSSELM classification model is obtained.

Phase 5 (Test Set Prediction Task). The trained MSMPA-JRSSELM model is used to predict the formation for the entire well oil layer of the test set and output the results. To validate the validity and stability of the model, we selected data from two wells for the experimental comparative analysis.

5.2. Practical Applications

In order to verify the application effect of the semi-supervised model optimized by the improved algorithm, three logging data were selected from the database for training and testing, and they were recorded as well 1 and well 2. The data set division of these two wells is shown in Table 4. It can be seen from Table 4 that the oil layer and dry layer distribution range in the training set and test set. In addition, the attribute reduction results of the two wells are shown in Table 5. It can be seen from Table 5 that there are many redundant conditional attributes in the original data. After Rough Set attribute reduction, important attributes can be selected, which greatly simplifies the complexity of the algorithm. The value range of attributes after attribute reduction is shown in Table 6.

Table 4. Details of the data set division between the two wells.

Well	Depth (m)	Training Set		Test Set		
		Oil Layers	Dry Layers	Depth (m)	Oil Layers	Dry Layers
Well 1	3150~3330	88	247	3330~3460	115	981
Well 2	1180~1255	45	192	1230~1300	92	469

Table 5. Results of attribute reduction.

Well	Attributes
Original results (Well 1)	U,TH,K,DT,SP,WQ,LLD,LLS,CALI,GR,DEN,NPHI,PE
Reduction results (Well 1)	GR,DT,SP,LLD,LLS,DEN,K
Original results (Well 2)	AC,C2,CALI,RINC,PORT, RHOG,SW,VO,WO,PORE,VCL,VMA1, CNL,DEN,GR,RT,RI,RXO,SP,VMA6, VXO,VW,so,rnsy,rsly,rny,AC1, R2M,R025,BZSP,RA2,C1
Reduction results (Well 2)	AC,GR,RT,RXO,SP
Decision attribute	$D = \{d\}$,$d = \{d_i = i, i = -1, 1\}$ where -1, 1 represent the dry layer and oil layer, respectively.

Table 6. Range of values for each attribute after attribute reduction.

Attributes	Range of Values	
	Well 1	Well 2
GR	[6, 200]	[27, 100]
DT	[152, 462]	/
SP	$[-167, -68]$	$[-32, -6]$
LLD	$[0, 2.5 \times 10^4]$	/
LLS	[0, 3307]	/
DEN	[1, 4]	/
K	[0, 5]	/
AC	/	[54, 140]
RT	/	[2, 90]
RI	/	/
RXO	/	[1, 328]
NG	/	/

From Table 6, we can see the range of values for each of the well 1 and well 2 properties, where GR stands for natural gamma, DT stands for acoustic time difference, SP stands for natural potential, LLD stands for deep lateral resistivity, LLS stands for shallow lateral resistivity, DEN stands for compensation density, and K stands for potassium. AC stands for acoustic time difference, RT stands for in situ ground resistivity, and RXO stands for flush zone resistivity. For the simplified conditional attributes, since the units of each attribute are different and the value ranges are different, the data should be normalized first, so that the range of the sample data is between [0, 1], and then the normalized influencing factor data should be substituted into the network for training and testing to obtain the results. The formula for normalizing the sample is as follows:

$$\overline{x} = 2 \times \frac{(x - x_{\min})}{x_{\max} - x_{\min}} - 1 \tag{45}$$

where $x \in [x_{\min}, x_{\max}]$, $x_{\min}$ is the minimum value of the data sample attribute, and $x_{\max}$ is the maximum value of the data sample attribute.

After normalizing the reduced attributes, the logging curve is shown in Figure 4, where the horizontal axis represents the depth and the vertical axis represents the normalized value. It can be seen from Figure 4 that the logging curves of each condition attribute are completely different, and effective information cannot be obtained directly from the logging curves.

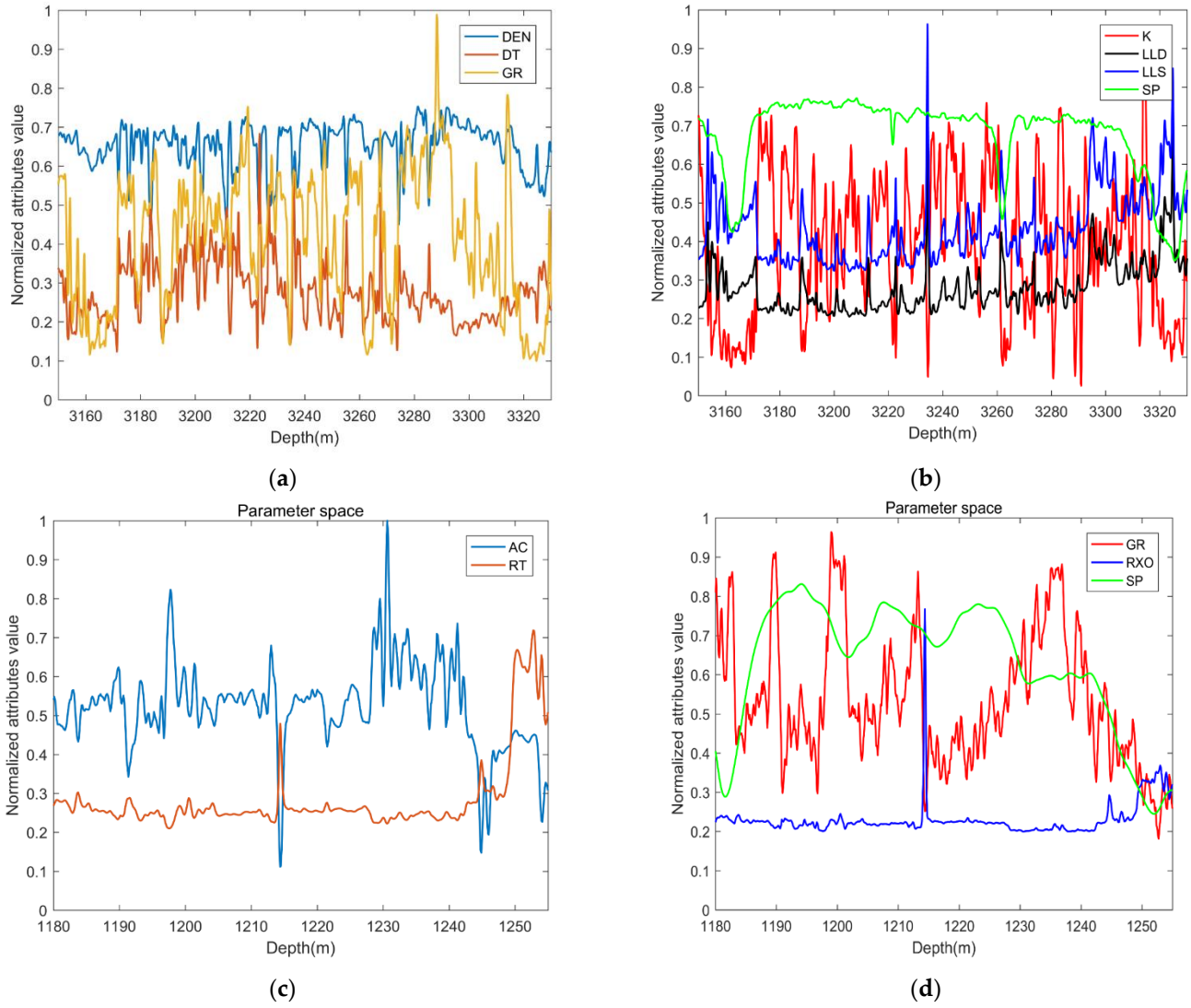

Figure 4. Normalized curves for each conditional property: (**a**,**b**) belong to well 1; (**c**,**d**) belong to well 2.

To evaluate the performance of the recognition model, in addition to the test set prediction accuracy, we defined the following performance metrics:

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(f(x)-y)^2} \tag{46}$$

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m}|f(x)-y| \tag{47}$$

where $f(x)$ and y are the predicted and desired output values, respectively.

The RMSE is a measure of the deviation between the predicted value and the true value, and the MAE is the average of the absolute errors. The smaller the RMSE and MAE, the better the performance of the algorithmic model. Therefore, RMSE is used as a criterion for evaluating the accuracy of each algorithmic model, and MAE is often used as the actual prediction and prediction error because it better reflects the actual error of the predicted value.

In order to comprehensively test the superiority of the proposed JRSSELM and MSMPA-JRSSELM models on semi-supervised learning, we divided the labeled samples into two categories of 10% and 20% in the training set and conducted comparison

experiments with the supervised learning model ELM and other semi-supervised learning models, including LapSVM [56], SSELM, and MPA-JRSSELM. They were run independently for 30 times, the number of hidden layer neurons was 100, and the maximum number of iterations was 100. Then, the average of the individual performance metrics was calculated. The classification performance of each model on well 1 is shown in Table 7. It should be noted that although the number of labeled samples has different ratios, the training set is invariant for ELM, so its ACC, MAE, and RMSE are consistent across the labeled samples.

Table 7. Performance of each model on well 1.

Proportion of Labeled Samples	Evaluation Indicators	ELM	LapSVM	SSELM	JRSSELM	MPA-JRSSELM	MSMPA-JRSSELM
10%	ACC (%)	89.9038	84.1346	90.3846	92.7885	93.4615	95.0962
	MAE	0.2019	0.3714	0.1923	0.1442	0.1308	0.0981
	RMSE	0.6355	0.7966	0.6202	0.5371	0.5114	0.4429
20%	ACC (%)	89.9038	86.4423	90.5769	93.2692	94.1346	95.3846
	MAE	0.2019	0.2712	0.1885	0.1346	0.1173	0.0923
	RMSE	0.6355	0.7364	0.6139	0.5189	0.4844	0.4641

From Table 7, it can be seen that LapSVM performs the worst, with the proposed JRSSELM improving its classification accuracy by nearly 3% over SSELM. With the increase in the number of labeled samples in the training set, the MSMPA-JRSSELM performance was further improved after MSMPA optimized selection parameters, with nearly 5% higher than SSELM and 2% higher than JRSSELM, and with the lowest MAE and RMSE, reflecting the significant competitive advantage of MSMPA-JRSSELM on well 1.

The classification performance of each model on Well 2 is shown in Table 8. It can be seen from Table 8 that LapSVM performs the worst, and the classification performance of SSELM is inferior to ELM, while the classification accuracy of JRSSELM proposed by us is slightly higher than that of ELM, and it is more noteworthy that MSMPA-JRSSELM improves the classification accuracy by nearly 8% compared to SSELM, reaching about 98% classification accuracy, and its MAE and RMSE are also the lowest values, which is helpful for well discrimination decision making. It has engineering applications, which reflect the obvious advantages of the proposed classifier for oil layer identification in oil logging.

Table 8. Performance of each model on well 2.

Proportion of Labeled Samples	Evaluation Indicators	ELM	LapSVM	SSELM	JRSSELM	MPA-JRSSELM	MSMPA-JRSSELM
10%	ACC (%)	93.5829	88.2353	89.6613	93.7611	96.7914	97.3262
	MAE	0.1283	0.2352	0.2068	0.1248	0.0642	0.0535
	RMSE	0.5066	0.6860	0.6431	0.499554	0.3582	0.3270
20%	ACC (%)	93.5829	89.1266	90.0178	94.1177	97.8610	98.7522
	MAE	0.1283	0.2175	0.1996	0.1176	0.0428	0.0250
	RMSE	0.5066	0.6595	0.6319	0.4851	0.2925	0.2234

In order to more intuitively observe the original oil test conclusions and predicted oil layer distribution of the test set, the original and predicted oil layer distributions of the two wells are shown in Figure 5. It can be seen from Figure 5 that for Well 1, as the proportion of labeled samples increases, the oil layer prediction is significantly more accurate, and the oil layer distribution position is basically consistent with the oil test conclusion. For Well 2, whether it is a training model with 10% or 20% of the training set of labeled samples, its predictions are almost consistent with the oil test conclusions, which has application significance for auxiliary oil logging.

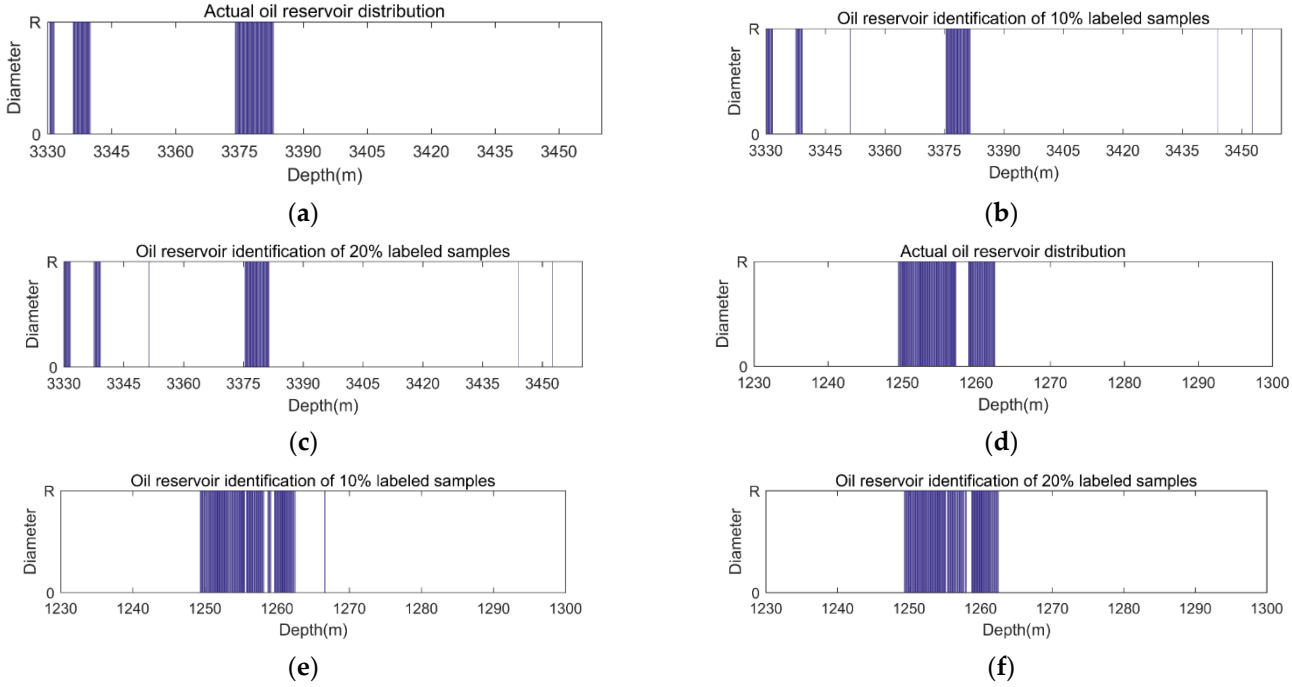

Figure 5. Original and predicted oil layer distribution for two wells, where (**a**–**c**) is the distribution for well 1, and (**d**–**f**) is the distribution for well 2.

6. Conclusions

In this paper, in order to better utilize only a small number of labeled samples and a large number of unlabeled samples to obtain better classification performance, a novel semi-supervised classification model, namely MSMPA-JRSSELM, is proposed.

First, the MSMPA is designed to address the shortcomings of the original MPA in solving global optimization problems such as slow convergence and poor ability to avoid local optima. There are 3 efficient strategies introduced in MPA. A chaotic opposition learning strategy is used to ensure high quality initial populations, adaptive inertial weights, and adaptive step control factors are adopted to enhance early global exploration and later fine-grained exploitation behavior and speed up convergence, and a neighbor dimensional learning strategy is proposed to ensure population diversity in each iteration. Among the 18 classical benchmark functions and 30 CEC2017 competition functions, MSMPA exhibits significant superiority over other algorithms, especially in solving high-dimensional complex problems, MSMPA exhibits strong global search and the ability to avoid local optimality.

Secondly, since SSELM uses Laplace regularization with weaker inferential power, Hessian regularization with stronger extrapolation power and the ability to maintain the manifold structure is used instead of Laplace regularization. Third, in response to the SSELM's inability to fully utilize the valid information embedded in the labeled samples, a supervised regularization term that assigns new coefficient weights to the given labeled information is proposed. Hessian regularization and supervised regularization are added to ELM to propose JRSSELM. finally, to further improve the classification performance of JRSSELM, MSMPA is proposed to optimize the selection of input weights, implied thresholds and hyperparameters in JRSSELM. To verify the effectiveness of JRSSELM and MSMPA-JRSSELM, they are applied to oil layer identification in logging. The experimental results show that JRSSELM and MSMPA-JRSSELM outperform SSELM and other popular classification methods in ACC, MAE and RMSE, especially MSMPA-JRSSELM shows the most excellent classification performance.

Further research and applications of MPA and SSELM are warranted, considering the combination of MPA with other meta-heuristics, the introduction of cost-sensitive

learning in SSELM, and the application of MPA and SSELM to other complex engineering problems. We consider combining SSELM with deep learning, adding self-encoder, convolution, down sampling, and other deep learning methods, which can realize automatic feature extraction and finally apply to regression problems such as short-term power load forecasting, meteorological load forecasting and wind power interval forecasting.

Author Contributions: W.Y.: Conceptualization, Methodology, Software, Writing—Original Draft, Writing—Review & Editing, Supervision. K.X.: Validation, Investigation, Project administration. T.L.: Data Curation, Resources, Funding acquisition. M.X.: Formal analysis. F.S.: Visualization. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (No.U1813222, No.42075129), Tianjin Natural Science Foundation (No.18JCYBJC16500) and Key Research and Development Project from Hebei Province(No.19210404D, No.20351802D).

Data Availability Statement: All data are available upon request from the corresponding author.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Abbreviations

The main abbreviations in this paper are shown below.

ELM	Extreme Learning Machines
SSELM	Semi-Supervised Extreme Learning Machine
JRSSELM	Joint Regularized Semi-Supervised Extreme Learning Machine
MPA	Marine Predator Algorithm
MSMPA	Multi-Strategy Marine Predator Algorithm
PSO	Particle Swarm Optimization
GWO	Gray Wolf Optimization
MFO	Moth Flame Optimization
SOA	Seagull Optimization Algorithm
SCA	Sine Cosine Algorithm
WOA	Whale Optimization Algorithm
COA	Coyote Optimization Algorithm
CPA	Carnivorous Plant Algorithm
TSA	Transient Search Algorithm
RF	Random Forests
SVM	Support Vector Machines
LapSVM	Laplace Support Vector Machines
FADS	Fish Aggregation Device Effects
NDL	Neighborhood Dimensional Learning
GR	natural gamma
DT	acoustic time difference
SP	natural potential
LLD	deep lateral resistivity
LLS	shallow lateral resistivity
DEN	compensation density
K	Potassium
AC	acoustic time difference
RT	in situ ground resistivity
RXO	flush zone resistivity

Appendix A

The experimental results in Section 3 are presented below, including statistics for each algorithm, Wilcoxon's rank sum test results, convergence curves, and box line plots.

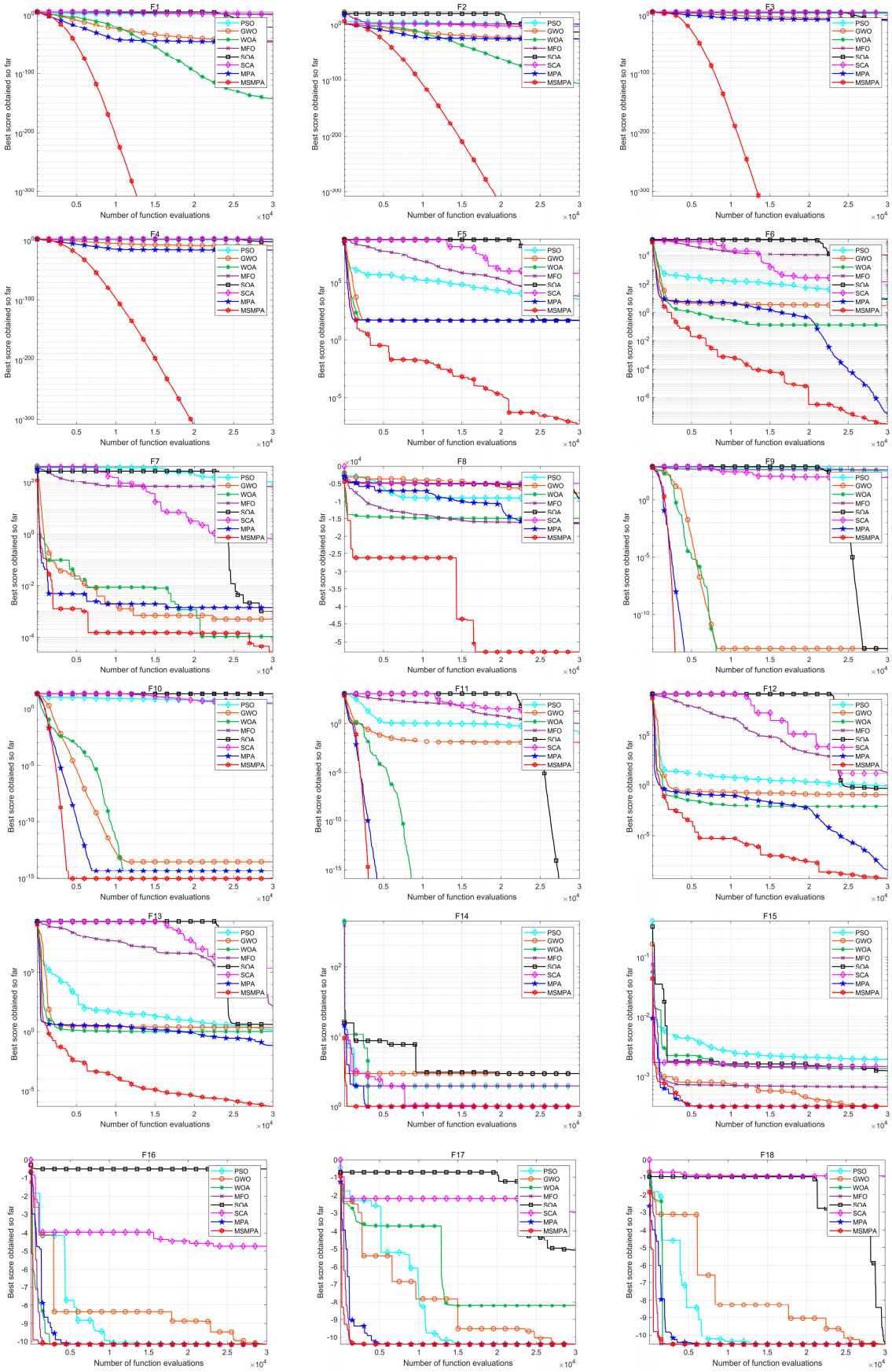

Figure A1. Convergence curves for 18 benchmark functions.

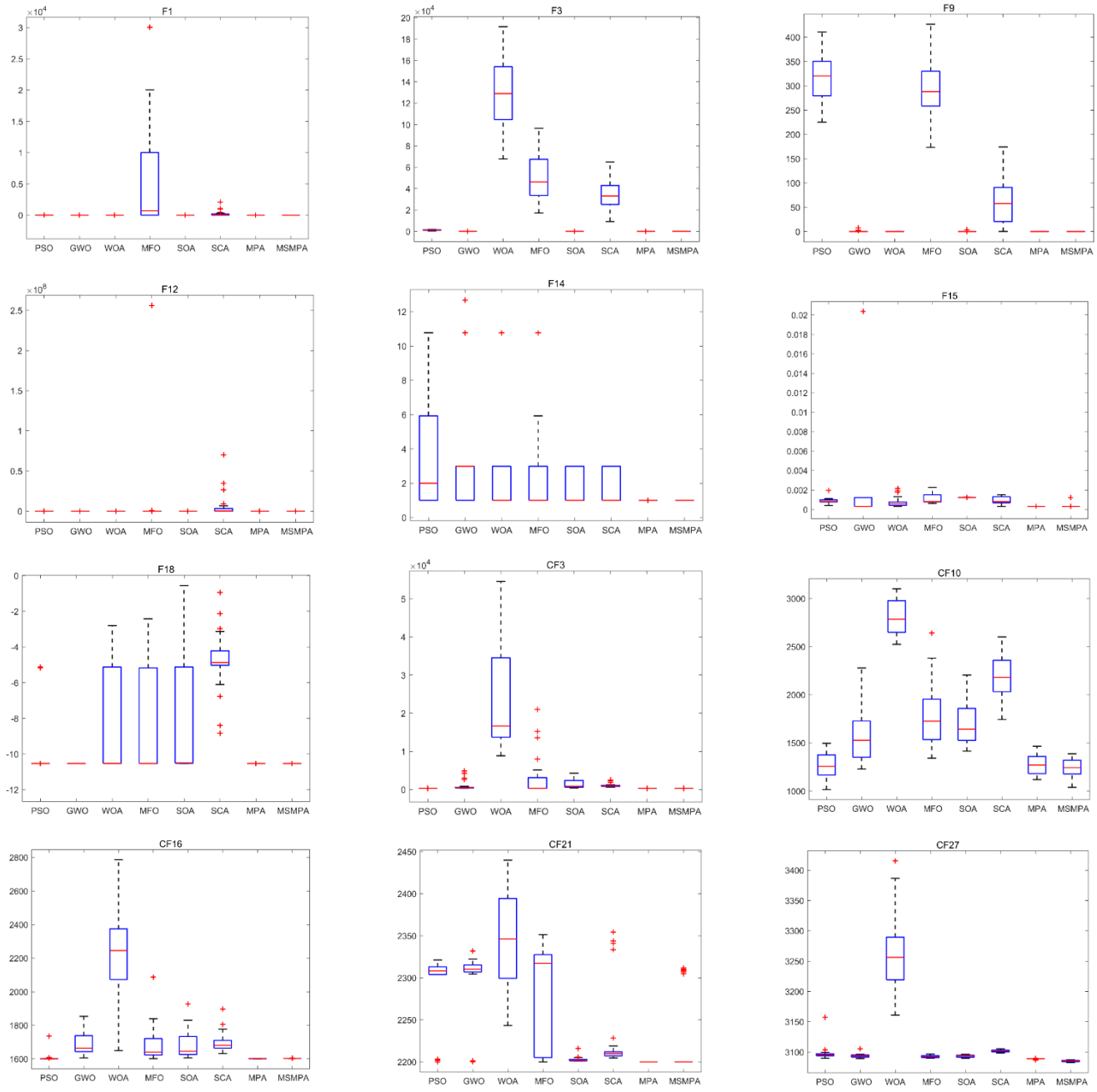

Figure A2. Box line diagram of partially selected functions.

Table A1. Test results of 18 benchmark functions.

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
F1	Ave	**0.00 × 10^{+00}**	2.69 × 10^{-46}	6.83 × 10^{+00}	6.68 × 10^{-44}	1.04 × 10^{-147}	7.07 × 10^{+03}	1.51 × 10^{-20}	2.49 × 10^{+02}
	Std	**0.00 × 10^{+00}**	5.14 × 10^{-46}	2.36 × 10^{+00}	6.83 × 10^{-44}	5.56 × 10^{-147}	8.97 × 10^{+03}	6.40 × 10^{-20}	4.54 × 10^{+02}
	Max	**0.00 × 10^{+00}**	2.51 × 10^{-45}	1.32 × 10^{+01}	2.82 × 10^{-43}	3.10 × 10^{-146}	3.00 × 10^{+04}	3.58 × 10^{-19}	2.10 × 10^{+03}
	Min	**0.00 × 10^{+00}**	8.73 × 10^{-50}	2.58 × 10^{+00}	1.17 × 10^{-45}	7.10 × 10^{-169}	5.58 × 10^{-01}	8.03 × 10^{-24}	3.35 × 10^{-01}
F2	Ave	**0.00 × 10^{+00}**	1.54 × 10^{-26}	1.07 × 10^{+01}	4.86 × 10^{-26}	1.53 × 10^{-102}	7.23 × 10^{+01}	5.22 × 10^{-14}	3.62 × 10^{-02}
	Std	**0.00 × 10^{+00}**	1.84 × 10^{-26}	1.99 × 10^{+00}	7.31 × 10^{-26}	7.40 × 10^{-102}	3.59 × 10^{+01}	4.46 × 10^{-14}	1.07 × 10^{-01}
	Max	**0.00 × 10^{+00}**	6.62 × 10^{-26}	1.43 × 10^{+01}	4.06 × 10^{-25}	4.13 × 10^{-101}	1.60 × 10^{+02}	1.68 × 10^{-13}	6.03 × 10^{-01}
	Min	**0.00 × 10^{+00}**	2.69 × 10^{-29}	6.18 × 10^{+00}	6.58 × 10^{-27}	2.77 × 10^{-113}	1.02 × 10^{+01}	2.92 × 10^{-15}	1.68 × 10^{-04}
F3	Ave	**0.00 × 10^{+00}**	1.41 × 10^{-07}	1.14 × 10^{+03}	4.79 × 10^{-05}	1.26 × 10^{+05}	5.10 × 10^{+04}	5.27 × 10^{-08}	3.40 × 10^{+04}
	Std	**0.00 × 10^{+00}**	4.73 × 10^{-07}	2.68 × 10^{+02}	2.41 × 10^{-04}	3.33 × 10^{+04}	2.18 × 10^{+04}	1.54 × 10^{-07}	1.27 × 10^{+04}
	Max	**0.00 × 10^{+00}**	1.94 × 10^{-06}	1.72 × 10^{+03}	1.35 × 10^{-03}	1.92 × 10^{+05}	9.65 × 10^{+04}	7.80 × 10^{-07}	6.49 × 10^{+04}
	Min	**0.00 × 10^{+00}**	1.15 × 10^{-15}	4.71 × 10^{+02}	9.28 × 10^{-10}	6.77 × 10^{+04}	1.72 × 10^{+04}	2.45 × 10^{-13}	9.15 × 10^{+03}

Table A1. *Cont.*

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
F4	Ave	$\mathbf{0.00 \times 10^{+00}}$	1.34×10^{-17}	$3.27 \times 10^{+00}$	1.54×10^{-09}	$6.31 \times 10^{+01}$	$8.44 \times 10^{+01}$	2.51×10^{-01}	$5.97 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	1.04×10^{-17}	3.66×10^{-01}	1.77×10^{-09}	$2.58 \times 10^{+01}$	$4.68 \times 10^{+00}$	7.91×10^{-01}	$8.01 \times 10^{+00}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	4.37×10^{-17}	$4.44 \times 10^{+00}$	7.16×10^{-09}	$9.30 \times 10^{+01}$	$9.22 \times 10^{+01}$	$4.09 \times 10^{+00}$	$7.34 \times 10^{+01}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	2.09×10^{-18}	$2.30 \times 10^{+00}$	1.35×10^{-10}	$6.17 \times 10^{+00}$	$7.73 \times 10^{+01}$	6.95×10^{-07}	$4.50 \times 10^{+01}$
F5	Ave	$\mathbf{7.20 \times 10^{-06}}$	$4.44 \times 10^{+01}$	$3.46 \times 10^{+03}$	$4.73 \times 10^{+01}$	$4.77 \times 10^{+01}$	$1.34 \times 10^{+07}$	$4.84 \times 10^{+01}$	$6.32 \times 10^{+05}$
	Std	$\mathbf{1.10 \times 10^{-05}}$	7.72×10^{-01}	$1.57 \times 10^{+03}$	7.86×10^{-01}	5.31×10^{-01}	$3.62 \times 10^{+07}$	3.98×10^{-01}	$9.81 \times 10^{+05}$
	Max	$\mathbf{4.12 \times 10^{-05}}$	$4.77 \times 10^{+01}$	$7.11 \times 10^{+03}$	$4.86 \times 10^{+01}$	$4.86 \times 10^{+01}$	$1.60 \times 10^{+08}$	$4.88 \times 10^{+01}$	$4.37 \times 10^{+06}$
	Min	$\mathbf{3.86 \times 10^{-12}}$	$4.32 \times 10^{+01}$	$1.32 \times 10^{+03}$	$4.61 \times 10^{+01}$	$4.67 \times 10^{+01}$	$4.03 \times 10^{+02}$	$4.72 \times 10^{+01}$	$4.06 \times 10^{+03}$
F6	Ave	$\mathbf{7.49 \times 10^{-07}}$	1.13×10^{-02}	$6.38 \times 10^{+00}$	$2.40 \times 10^{+00}$	4.11×10^{-01}	$8.09 \times 10^{+03}$	$7.19 \times 10^{+00}$	$1.75 \times 10^{+02}$
	Std	$\mathbf{9.73 \times 10^{-07}}$	4.31×10^{-02}	$2.08 \times 10^{+00}$	5.60×10^{-01}	2.24×10^{-01}	$9.11 \times 10^{+03}$	5.36×10^{-01}	$3.22 \times 10^{+02}$
	Max	$\mathbf{3.87 \times 10^{-06}}$	2.16×10^{-01}	$1.25 \times 10^{+01}$	$3.25 \times 10^{+00}$	$1.04 \times 10^{+00}$	$4.02 \times 10^{+04}$	$8.13 \times 10^{+00}$	$1.22 \times 10^{+03}$
	Min	$\mathbf{1.05 \times 10^{-08}}$	3.24×10^{-08}	$3.04 \times 10^{+00}$	$1.25 \times 10^{+00}$	8.74×10^{-02}	$1.08 \times 10^{+00}$	$5.89 \times 10^{+00}$	$8.70 \times 10^{+00}$
F7	Ave	$\mathbf{2.48 \times 10^{-05}}$	6.96×10^{-04}	$1.63 \times 10^{+02}$	1.24×10^{-03}	1.50×10^{-03}	$1.56 \times 10^{+01}$	1.62×10^{-03}	9.81×10^{-01}
	Std	$\mathbf{1.95 \times 10^{-05}}$	3.03×10^{-04}	$6.65 \times 10^{+01}$	6.62×10^{-04}	1.77×10^{-03}	$1.87 \times 10^{+01}$	1.59×10^{-03}	$1.29 \times 10^{+00}$
	Max	$\mathbf{8.65 \times 10^{-05}}$	1.17×10^{-03}	$3.12 \times 10^{+02}$	2.75×10^{-03}	7.74×10^{-03}	$7.84 \times 10^{+01}$	7.56×10^{-03}	$5.10 \times 10^{+00}$
	Min	$\mathbf{8.50 \times 10^{-07}}$	1.76×10^{-04}	$4.01 \times 10^{+01}$	4.42×10^{-04}	2.61×10^{-06}	4.46×10^{-01}	2.76×10^{-04}	2.77×10^{-02}
F8	Ave	$\mathbf{-4.67 \times 10^{+04}}$	$-1.51 \times 10^{+04}$	$-9.99 \times 10^{+03}$	$-9.71 \times 10^{+03}$	$-1.86 \times 10^{+04}$	$-1.31 \times 10^{+04}$	$-7.25 \times 10^{+03}$	$-5.02 \times 10^{+03}$
	Std	$1.12 \times 10^{+04}$	$6.77 \times 10^{+02}$	$2.01 \times 10^{+03}$	$8.12 \times 10^{+02}$	$2.60 \times 10^{+03}$	$1.27 \times 10^{+03}$	$1.09 \times 10^{+03}$	$\mathbf{3.08 \times 10^{+02}}$
	Max	$\mathbf{-1.85 \times 10^{+04}}$	$-1.39 \times 10^{+04}$	$-4.30 \times 10^{+03}$	$-8.11 \times 10^{+03}$	$-1.32 \times 10^{+04}$	$-1.10 \times 10^{+04}$	$-5.82 \times 10^{+03}$	$-4.61 \times 10^{+03}$
	Min	$\mathbf{-5.45 \times 10^{+04}}$	$-1.63 \times 10^{+04}$	$-1.25 \times 10^{+04}$	$-1.12 \times 10^{+04}$	$-2.09 \times 10^{+04}$	$-1.51 \times 10^{+04}$	$-1.00 \times 10^{+04}$	$-5.68 \times 10^{+03}$
F9	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$3.18 \times 10^{+02}$	3.24×10^{-01}	$\mathbf{0.00 \times 10^{+00}}$	$2.98 \times 10^{+02}$	1.16×10^{-01}	$6.41 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$4.37 \times 10^{+01}$	$1.39 \times 10^{+00}$	$\mathbf{0.00 \times 10^{+00}}$	$6.59 \times 10^{+01}$	6.23×10^{-01}	$4.72 \times 10^{+01}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$4.11 \times 10^{+02}$	$7.48 \times 10^{+00}$	$\mathbf{0.00 \times 10^{+00}}$	$4.26 \times 10^{+02}$	$3.47 \times 10^{+00}$	$1.74 \times 10^{+02}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$2.25 \times 10^{+02}$	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.73 \times 10^{+02}$	$0.00 \times 10^{+00}$	1.32×10^{-01}
F10	Ave	$\mathbf{8.88 \times 10^{-16}}$	4.32×10^{-15}	$3.18 \times 10^{+00}$	3.32×10^{-14}	3.73×10^{-15}	$1.95 \times 10^{+01}$	$2.00 \times 10^{+01}$	$1.94 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	6.38×10^{-16}	3.72×10^{-01}	4.71×10^{-15}	2.81×10^{-15}	5.73×10^{-01}	9.30×10^{-04}	$3.73 \times 10^{+00}$
	Max	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$4.04 \times 10^{+00}$	4.35×10^{-14}	7.99×10^{-15}	$2.00 \times 10^{+01}$	$2.00 \times 10^{+01}$	$2.05 \times 10^{+01}$
	Min	$\mathbf{8.88 \times 10^{-16}}$	$\mathbf{8.88 \times 10^{-16}}$	$2.49 \times 10^{+00}$	2.22×10^{-14}	$\mathbf{8.88 \times 10^{-16}}$	$1.76 \times 10^{+01}$	$2.00 \times 10^{+01}$	9.91×10^{-02}
F11	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	1.90×10^{-01}	2.40×10^{-03}	8.03×10^{-03}	$7.33 \times 10^{+01}$	4.70×10^{-03}	$1.31 \times 10^{+00}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	7.04×10^{-02}	7.28×10^{-03}	2.42×10^{-02}	$7.82 \times 10^{+01}$	1.35×10^{-02}	7.09×10^{-01}
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	4.10×10^{-01}	2.71×10^{-02}	8.76×10^{-02}	$2.71 \times 10^{+02}$	6.16×10^{-02}	$4.82 \times 10^{+00}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	9.82×10^{-02}	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	7.64×10^{-01}	$\mathbf{0.00 \times 10^{+00}}$	6.90×10^{-01}
F12	Ave	$\mathbf{2.33 \times 10^{-09}}$	1.88×10^{-04}	1.96×10^{-01}	7.81×10^{-02}	1.15×10^{-02}	$1.71 \times 10^{+07}$	4.92×10^{-01}	$5.61 \times 10^{+06}$
	Std	$\mathbf{2.83 \times 10^{-09}}$	6.10×10^{-04}	3.31×10^{-01}	2.68×10^{-02}	7.35×10^{-03}	$6.39 \times 10^{+07}$	1.00×10^{-01}	$1.42 \times 10^{+07}$
	Max	$\mathbf{1.42 \times 10^{-08}}$	2.96×10^{-03}	$1.88 \times 10^{+00}$	1.54×10^{-01}	3.48×10^{-02}	$2.56 \times 10^{+08}$	7.93×10^{-01}	$7.01 \times 10^{+07}$
	Min	$\mathbf{2.10 \times 10^{-12}}$	1.78×10^{-09}	3.36×10^{-02}	3.38×10^{-02}	3.39×10^{-03}	$2.27 \times 10^{+00}$	3.42×10^{-01}	$5.09 \times 10^{+00}$
F13	Ave	$\mathbf{4.59 \times 10^{-08}}$	9.99×10^{-02}	$1.96 \times 10^{+00}$	$1.92 \times 10^{+00}$	6.35×10^{-01}	$4.10 \times 10^{+07}$	$3.95 \times 10^{+00}$	$3.43 \times 10^{+06}$
	Std	$\mathbf{5.09 \times 10^{-08}}$	9.80×10^{-02}	5.29×10^{-01}	3.47×10^{-01}	2.93×10^{-01}	$1.23 \times 10^{+08}$	2.44×10^{-01}	$5.41 \times 10^{+06}$
	Max	$\mathbf{2.06 \times 10^{-07}}$	3.73×10^{-01}	$2.82 \times 10^{+00}$	$2.45 \times 10^{+00}$	$1.41 \times 10^{+00}$	$4.10 \times 10^{+08}$	$4.46 \times 10^{+00}$	$2.59 \times 10^{+07}$
	Min	$\mathbf{5.46 \times 10^{-11}}$	8.88×10^{-08}	$1.14 \times 10^{+00}$	7.63×10^{-01}	1.92×10^{-01}	$2.27 \times 10^{+01}$	$3.54 \times 10^{+00}$	$6.89 \times 10^{+02}$
F14	Ave	$\mathbf{9.98 \times 10^{-01}}$	9.98×10^{-01}	$3.30 \times 10^{+00}$	$3.52 \times 10^{+00}$	$2.08 \times 10^{+00}$	$2.25 \times 10^{+00}$	$1.59 \times 10^{+00}$	$1.59 \times 10^{+00}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	5.73×10^{-17}	$2.67 \times 10^{+00}$	$3.54 \times 10^{+00}$	$2.45 \times 10^{+00}$	$2.10 \times 10^{+00}$	9.09×10^{-01}	9.08×10^{-01}
	Max	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$1.08 \times 10^{+01}$	$1.27 \times 10^{+01}$	$1.08 \times 10^{+01}$	$1.08 \times 10^{+01}$	$2.98 \times 10^{+00}$	$2.98 \times 10^{+00}$
	Min	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$	$\mathbf{9.98 \times 10^{-01}}$
F15	Ave	0.0004	**0.0003**	0.0008	0.0038	0.0007	0.001	0.0012	0.0009
	Std	0.0002	**0**	0.0003	0.0074	0.0005	0.0004	**0**	0.0004
	Max	0.0012	**0.0003**	0.0019	0.0204	0.0022	0.0023	0.0012	0.0015
	Min	**0.0003**	**0.0003**	0.0004	**0.0003**	**0.0003**	0.0006	0.0012	**0.0003**
F16	Ave	**−10.1532**	**−10.1532**	−7.7963	−9.6475	−8.6505	−8.0628	−3.5614	−2.885
	Std	**0**	**0**	2.7564	1.5157	2.5831	3.0377	4.3677	1.8174
	Max	**−10.1532**	**−10.1532**	−2.6305	−5.1003	−0.881	−2.6305	−0.3507	−0.4965
	Min	**−10.1532**	**−10.1532**	−10.1532	−10.1531	−10.153	**−10.1532**	−10.1373	−4.9475
F17	Ave	**−10.4028**	**−10.4028**	−10.2258	−10.2267	−8.5934	−8.7168	−6.0807	−4.1795
	Std	**0**	**0**	0.9541	0.9467	2.7846	3.065	4.5085	2.4846
	Max	**−10.4028**	**−10.4028**	−5.0877	−5.1284	−2.7655	−2.7659	−0.3724	−0.5224
	Min	**−10.4028**	**−10.4028**	**−10.4028**	**−10.4028**	**−10.4028**	**−10.4028**	−10.4005	−9.5476
F18	Ave	**−10.5363**	**−10.5363**	−9.8201	−10.536	−9.014	−8.2934	−8.0033	−4.7824
	Std	**0**	**0**	1.8263	0.0002	2.5492	3.225	3.938	1.5385
	Max	**−10.5363**	**−10.5363**	−5.1285	−10.5354	−2.8064	−2.4273	−0.5542	−0.9448
	Min	**−10.5363**	**−10.5363**	**−10.5363**	−10.5363	**−10.5363**	**−10.5363**	−10.5342	−8.8356
Friedman Average Rank		**1.6551**	2.4101	4.9058	4.4775	4.3225	5.3986	5.7667	7.0638
Rank		**1**	2	5	4	3	6	7	8

Table A2. CEC2017 optimization function results.

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
CF1	Ave	**$1.00 \times 10^{+02}$**	**$1.00 \times 10^{+02}$**	$2.67 \times 10^{+03}$	$2.46 \times 10^{+06}$	$1.37 \times 10^{+10}$	$4.68 \times 10^{+06}$	$1.84 \times 10^{+08}$	$6.00 \times 10^{+08}$
	Std	**1.58×10^{-05}**	6.00×10^{-03}	$2.86 \times 10^{+03}$	$6.44 \times 10^{+06}$	$5.50 \times 10^{+09}$	$1.75 \times 10^{+07}$	$1.83 \times 10^{+08}$	$2.07 \times 10^{+08}$
CF2	Ave	**$2.00 \times 10^{+02}$**	**$2.00 \times 10^{+02}$**	**$2.00 \times 10^{+02}$**	$3.68 \times 10^{+06}$	$1.57 \times 10^{+12}$	$1.04 \times 10^{+08}$	$1.29 \times 10^{+07}$	$7.48 \times 10^{+06}$
	Std	**$0.00 \times 10^{+00}$**	**$0.00 \times 10^{+00}$**	**$0.00 \times 10^{+00}$**	$1.08 \times 10^{+07}$	$5.97 \times 10^{+12}$	$4.01 \times 10^{+08}$	$1.78 \times 10^{+07}$	$1.45 \times 10^{+07}$
CF3	Ave	**$3.00 \times 10^{+02}$**	**$3.00 \times 10^{+02}$**	**$3.00 \times 10^{+02}$**	$8.85 \times 10^{+02}$	$2.36 \times 10^{+04}$	$3.13 \times 10^{+03}$	$1.47 \times 10^{+03}$	$1.06 \times 10^{+03}$
	Std	**4.01×10^{-10}**	2.83×10^{-08}	2.29×10^{-08}	$1.15 \times 10^{+03}$	$1.33 \times 10^{+04}$	$4.97 \times 10^{+03}$	$1.08 \times 10^{+03}$	$4.75 \times 10^{+02}$
CF4	Ave	**$4.00 \times 10^{+02}$**	**$4.00 \times 10^{+02}$**	$4.03 \times 10^{+02}$	$4.12 \times 10^{+02}$	$1.50 \times 10^{+03}$	$4.09 \times 10^{+02}$	$4.40 \times 10^{+02}$	$4.33 \times 10^{+02}$
	Std	1.20×10^{-07}	**8.73×10^{-08}**	$1.07 \times 10^{+00}$	$1.37 \times 10^{+01}$	$6.46 \times 10^{+02}$	$1.94 \times 10^{+01}$	$3.81 \times 10^{+01}$	$1.11 \times 10^{+01}$
CF5	Ave	**$5.06 \times 10^{+02}$**	$5.08 \times 10^{+02}$	$5.08 \times 10^{+02}$	$5.12 \times 10^{+02}$	$6.22 \times 10^{+02}$	$5.24 \times 10^{+02}$	$5.24 \times 10^{+02}$	$5.44 \times 10^{+02}$
	Std	**$2.00 \times 10^{+00}$**	$2.36 \times 10^{+00}$	$3.43 \times 10^{+00}$	$4.20 \times 10^{+00}$	$2.05 \times 10^{+01}$	$8.03 \times 10^{+00}$	$8.55 \times 10^{+00}$	$5.94 \times 10^{+00}$
CF6	Ave	**$6.00 \times 10^{+02}$**	**$6.00 \times 10^{+02}$**	**$6.00 \times 10^{+02}$**	**$6.00 \times 10^{+02}$**	$6.74 \times 10^{+02}$	**$6.00 \times 10^{+02}$**	$6.08 \times 10^{+02}$	$6.16 \times 10^{+02}$
	Std	3.28×10^{-02}	1.02×10^{-04}	**4.34×10^{-11}**	7.31×10^{-01}	$1.50 \times 10^{+01}$	9.98×10^{-01}	$4.30 \times 10^{+00}$	$3.22 \times 10^{+00}$
CF7	Ave	**$7.12 \times 10^{+02}$**	$7.19 \times 10^{+02}$	$7.17 \times 10^{+02}$	$7.26 \times 10^{+02}$	$9.65 \times 10^{+02}$	$7.35 \times 10^{+02}$	$7.53 \times 10^{+02}$	$7.70 \times 10^{+02}$
	Std	**$2.42 \times 10^{+00}$**	$2.65 \times 10^{+00}$	$4.55 \times 10^{+00}$	$9.34 \times 10^{+00}$	$1.01 \times 10^{+02}$	$1.24 \times 10^{+01}$	$1.49 \times 10^{+01}$	$7.80 \times 10^{+00}$
CF8	Ave	**$8.05 \times 10^{+02}$**	$8.06 \times 10^{+02}$	$8.08 \times 10^{+02}$	$8.10 \times 10^{+02}$	$8.92 \times 10^{+02}$	$8.26 \times 10^{+02}$	$8.24 \times 10^{+02}$	$8.37 \times 10^{+02}$
	Std	$2.27 \times 10^{+00}$	**$1.98 \times 10^{+00}$**	$3.04 \times 10^{+00}$	$4.00 \times 10^{+00}$	$1.98 \times 10^{+01}$	$1.21 \times 10^{+01}$	$6.17 \times 10^{+00}$	$5.99 \times 10^{+00}$
CF9	Ave	**$9.00 \times 10^{+02}$**	**$9.00 \times 10^{+02}$**	**$9.00 \times 10^{+02}$**	$9.04 \times 10^{+02}$	$2.74 \times 10^{+03}$	$9.38 \times 10^{+02}$	$9.84 \times 10^{+02}$	$9.72 \times 10^{+02}$
	Std	1.40×10^{-04}	2.01×10^{-08}	**6.56×10^{-14}**	$9.47 \times 10^{+00}$	$9.39 \times 10^{+02}$	$1.01 \times 10^{+02}$	$9.03 \times 10^{+01}$	$2.60 \times 10^{+01}$
CF10	Ave	**$1.19 \times 10^{+03}$**	$1.27 \times 10^{+03}$	$1.28 \times 10^{+03}$	$1.57 \times 10^{+03}$	$2.80 \times 10^{+03}$	$1.79 \times 10^{+03}$	$1.71 \times 10^{+03}$	$2.19 \times 10^{+03}$
	Std	$1.19 \times 10^{+02}$	**$9.66 \times 10^{+01}$**	$1.23 \times 10^{+02}$	$2.44 \times 10^{+02}$	$1.70 \times 10^{+02}$	$3.19 \times 10^{+02}$	$2.22 \times 10^{+02}$	$1.99 \times 10^{+02}$
CF11	Ave	**$1.10 \times 10^{+03}$**	**$1.10 \times 10^{+03}$**	**$1.10 \times 10^{+03}$**	$1.13 \times 10^{+03}$	$5.46 \times 10^{+03}$	$1.13 \times 10^{+03}$	$1.20 \times 10^{+03}$	$1.19 \times 10^{+03}$
	Std	$1.26 \times 10^{+00}$	**7.50×10^{-01}**	$2.34 \times 10^{+00}$	$1.06 \times 10^{+01}$	$5.34 \times 10^{+03}$	$3.89 \times 10^{+01}$	$7.76 \times 10^{+01}$	$6.08 \times 10^{+01}$
CF12	Ave	**$1.20 \times 10^{+03}$**	**$1.20 \times 10^{+03}$**	$1.95 \times 10^{+04}$	$4.05 \times 10^{+05}$	$4.95 \times 10^{+08}$	$9.56 \times 10^{+05}$	$2.58 \times 10^{+06}$	$8.06 \times 10^{+06}$
	Std	$1.68 \times 10^{+01}$	**$5.14 \times 10^{+00}$**	$1.91 \times 10^{+04}$	$5.72 \times 10^{+05}$	$4.98 \times 10^{+08}$	$2.16 \times 10^{+06}$	$2.43 \times 10^{+06}$	$6.55 \times 10^{+06}$
CF13	Ave	**$1.30 \times 10^{+03}$**	**$1.30 \times 10^{+03}$**	$5.92 \times 10^{+03}$	$1.07 \times 10^{+04}$	$3.07 \times 10^{+07}$	$1.17 \times 10^{+04}$	$1.67 \times 10^{+04}$	$2.28 \times 10^{+04}$
	Std	**$1.84 \times 10^{+00}$**	$1.92 \times 10^{+00}$	$6.17 \times 10^{+03}$	$6.91 \times 10^{+03}$	$5.54 \times 10^{+07}$	$1.25 \times 10^{+04}$	$1.13 \times 10^{+04}$	$1.85 \times 10^{+04}$
CF14	Ave	**$1.40 \times 10^{+03}$**	**$1.40 \times 10^{+03}$**	$1.43 \times 10^{+03}$	$2.16 \times 10^{+03}$	$5.72 \times 10^{+03}$	$2.25 \times 10^{+03}$	$1.55 \times 10^{+03}$	$1.57 \times 10^{+03}$
	Std	**$1.95 \times 10^{+00}$**	$2.07 \times 10^{+00}$	$1.10 \times 10^{+01}$	$1.40 \times 10^{+03}$	$8.46 \times 10^{+03}$	$9.25 \times 10^{+02}$	$6.39 \times 10^{+01}$	$5.21 \times 10^{+01}$
CF15	Ave	**$1.50 \times 10^{+03}$**	**$1.50 \times 10^{+03}$**	$1.53 \times 10^{+03}$	$3.10 \times 10^{+03}$	$2.09 \times 10^{+04}$	$4.43 \times 10^{+03}$	$2.26 \times 10^{+03}$	$2.09 \times 10^{+03}$
	Std	**4.36×10^{-01}**	4.79×10^{-01}	$2.45 \times 10^{+01}$	$1.90 \times 10^{+03}$	$2.58 \times 10^{+04}$	$2.65 \times 10^{+03}$	$7.50 \times 10^{+02}$	$5.56 \times 10^{+02}$
CF16	Ave	$1.61 \times 10^{+03}$	**$1.60 \times 10^{+03}$**	$1.61 \times 10^{+03}$	$1.69 \times 10^{+03}$	$2.24 \times 10^{+03}$	$1.68 \times 10^{+03}$	$1.68 \times 10^{+03}$	$1.70 \times 10^{+03}$
	Std	$2.97 \times 10^{+01}$	**4.22×10^{-01}**	$3.35 \times 10^{+01}$	$5.79 \times 10^{+01}$	$2.38 \times 10^{+02}$	$9.95 \times 10^{+01}$	$7.44 \times 10^{+01}$	$5.33 \times 10^{+01}$
CF17	Ave	**$1.71 \times 10^{+03}$**	**$1.71 \times 10^{+03}$**	**$1.71 \times 10^{+03}$**	$1.75 \times 10^{+03}$	$2.01 \times 10^{+03}$	$1.74 \times 10^{+03}$	$1.77 \times 10^{+03}$	$1.77 \times 10^{+03}$
	Std	**$6.04 \times 10^{+00}$**	$6.48 \times 10^{+00}$	$9.17 \times 10^{+00}$	$2.50 \times 10^{+01}$	$1.60 \times 10^{+02}$	$1.76 \times 10^{+01}$	$3.37 \times 10^{+01}$	$9.68 \times 10^{+00}$
CF18	Ave	**$1.80 \times 10^{+03}$**	**$1.80 \times 10^{+03}$**	$4.59 \times 10^{+03}$	$2.62 \times 10^{+04}$	$3.92 \times 10^{+07}$	$2.38 \times 10^{+04}$	$3.88 \times 10^{+04}$	$6.88 \times 10^{+04}$
	Std	$1.41 \times 10^{+00}$	**$1.17 \times 10^{+00}$**	$2.47 \times 10^{+03}$	$1.43 \times 10^{+04}$	$7.49 \times 10^{+07}$	$1.82 \times 10^{+04}$	$1.08 \times 10^{+04}$	$3.97 \times 10^{+04}$
CF19	Ave	**$1.90 \times 10^{+03}$**	**$1.90 \times 10^{+03}$**	$1.92 \times 10^{+03}$	$4.64 \times 10^{+03}$	$8.76 \times 10^{+05}$	$8.04 \times 10^{+03}$	$7.50 \times 10^{+03}$	$2.68 \times 10^{+03}$
	Std	**2.72×10^{-01}**	4.28×10^{-01}	$2.52 \times 10^{+01}$	$4.73 \times 10^{+03}$	$2.30 \times 10^{+06}$	$1.03 \times 10^{+04}$	$6.35 \times 10^{+03}$	$1.64 \times 10^{+03}$
CF20	Ave	**$2.01 \times 10^{+03}$**	**$2.01 \times 10^{+03}$**	**$2.01 \times 10^{+03}$**	$2.06 \times 10^{+03}$	$2.29 \times 10^{+03}$	$2.04 \times 10^{+03}$	$2.09 \times 10^{+03}$	$2.09 \times 10^{+03}$
	Std	**$4.48 \times 10^{+00}$**	$8.41 \times 10^{+00}$	$1.06 \times 10^{+01}$	$4.01 \times 10^{+01}$	$7.52 \times 10^{+01}$	$2.26 \times 10^{+01}$	$5.71 \times 10^{+01}$	$2.08 \times 10^{+01}$
CF21	Ave	$2.24 \times 10^{+03}$	**$2.20 \times 10^{+03}$**	$2.29 \times 10^{+03}$	$2.30 \times 10^{+03}$	$2.35 \times 10^{+03}$	$2.27 \times 10^{+03}$	**$2.20 \times 10^{+03}$**	$2.23 \times 10^{+03}$
	Std	$5.24 \times 10^{+01}$	**1.29×10^{-05}**	$4.64 \times 10^{+01}$	$3.84 \times 10^{+01}$	$5.63 \times 10^{+01}$	$6.23 \times 10^{+01}$	$2.77 \times 10^{+00}$	$4.56 \times 10^{+01}$
CF22	Ave	$2.30 \times 10^{+03}$	**$2.24 \times 10^{+03}$**	$2.30 \times 10^{+03}$	$2.31 \times 10^{+03}$	$3.32 \times 10^{+03}$	$2.30 \times 10^{+03}$	$2.45 \times 10^{+03}$	$2.35 \times 10^{+03}$
	Std	$1.84 \times 10^{+01}$	$4.78 \times 10^{+01}$	$1.51 \times 10^{+01}$	**$4.85 \times 10^{+00}$**	$4.75 \times 10^{+02}$	$2.19 \times 10^{+01}$	$3.61 \times 10^{+02}$	$1.83 \times 10^{+01}$
CF23	Ave	**$2.51 \times 10^{+03}$**	$2.58 \times 10^{+03}$	$2.61 \times 10^{+03}$	$2.61 \times 10^{+03}$	$2.73 \times 10^{+03}$	$2.63 \times 10^{+03}$	$2.63 \times 10^{+03}$	$2.65 \times 10^{+03}$
	Std	**$2.17 \times 10^{+00}$**	$8.39 \times 10^{+01}$	$4.08 \times 10^{+00}$	$6.63 \times 10^{+00}$	$2.71 \times 10^{+01}$	$8.45 \times 10^{+00}$	$6.90 \times 10^{+00}$	$6.25 \times 10^{+00}$
CF24	Ave	$2.70 \times 10^{+03}$	**$2.49 \times 10^{+03}$**	$2.72 \times 10^{+03}$	$2.75 \times 10^{+03}$	$2.90 \times 10^{+03}$	$2.76 \times 10^{+03}$	$2.75 \times 10^{+03}$	$2.76 \times 10^{+03}$
	Std	$7.88 \times 10^{+01}$	$2.49 \times 10^{+01}$	$5.98 \times 10^{+01}$	$1.03 \times 10^{+01}$	$6.38 \times 10^{+01}$	**$9.34 \times 10^{+00}$**	$9.90 \times 10^{+00}$	$6.77 \times 10^{+01}$
CF25	Ave	$2.91 \times 10^{+03}$	**$2.86 \times 10^{+03}$**	$2.92 \times 10^{+03}$	$2.93 \times 10^{+03}$	$4.01 \times 10^{+03}$	$2.93 \times 10^{+03}$	$2.94 \times 10^{+03}$	$2.95 \times 10^{+03}$
	Std	$2.00 \times 10^{+01}$	$1.01 \times 10^{+02}$	$2.34 \times 10^{+01}$	$1.89 \times 10^{+01}$	$4.44 \times 10^{+02}$	$2.80 \times 10^{+01}$	$2.61 \times 10^{+01}$	**$1.51 \times 10^{+01}$**
CF26	Ave	$2.90 \times 10^{+03}$	**$2.70 \times 10^{+03}$**	$2.93 \times 10^{+03}$	$2.93 \times 10^{+03}$	$4.47 \times 10^{+03}$	$2.98 \times 10^{+03}$	$3.06 \times 10^{+03}$	$3.06 \times 10^{+03}$
	Std	**2.44×10^{-04}**	$1.05 \times 10^{+02}$	$1.69 \times 10^{+02}$	$1.78 \times 10^{+02}$	$3.99 \times 10^{+02}$	$5.46 \times 10^{+01}$	$2.37 \times 10^{+02}$	$2.22 \times 10^{+01}$
CF27	Ave	**$3.08 \times 10^{+03}$**	$3.09 \times 10^{+03}$	$3.10 \times 10^{+03}$	$3.09 \times 10^{+03}$	$3.26 \times 10^{+03}$	$3.09 \times 10^{+03}$	$3.09 \times 10^{+03}$	$3.10 \times 10^{+03}$
	Std	$1.14 \times 10^{+00}$	**5.67×10^{-01}**	$1.15 \times 10^{+01}$	$2.90 \times 10^{+00}$	$6.07 \times 10^{+01}$	$1.95 \times 10^{+00}$	$1.91 \times 10^{+00}$	$1.90 \times 10^{+00}$
CF28	Ave	$3.11 \times 10^{+03}$	**$3.07 \times 10^{+03}$**	$3.23 \times 10^{+03}$	$3.35 \times 10^{+03}$	$3.69 \times 10^{+03}$	$3.25 \times 10^{+03}$	$3.23 \times 10^{+03}$	$3.25 \times 10^{+03}$
	Std	**$3.23 \times 10^{+01}$**	$9.00 \times 10^{+01}$	$1.59 \times 10^{+02}$	$8.68 \times 10^{+01}$	$1.15 \times 10^{+02}$	$8.01 \times 10^{+01}$	$8.22 \times 10^{+01}$	$5.64 \times 10^{+01}$
CF29	Ave	$3.14 \times 10^{+03}$	**$3.13 \times 10^{+03}$**	$3.16 \times 10^{+03}$	$3.17 \times 10^{+03}$	$3.58 \times 10^{+03}$	$3.20 \times 10^{+03}$	$3.18 \times 10^{+03}$	$3.21 \times 10^{+03}$
	Std	**$7.29 \times 10^{+00}$**	$9.20 \times 10^{+00}$	$1.90 \times 10^{+01}$	$1.98 \times 10^{+01}$	$1.34 \times 10^{+02}$	$4.05 \times 10^{+01}$	$2.57 \times 10^{+01}$	$2.17 \times 10^{+01}$
CF30	Ave	**$3.31 \times 10^{+03}$**	$3.40 \times 10^{+03}$	$6.02 \times 10^{+05}$	$5.65 \times 10^{+05}$	$1.23 \times 10^{+07}$	$5.64 \times 10^{+05}$	$7.35 \times 10^{+04}$	$6.44 \times 10^{+05}$
	Std	$3.66 \times 10^{+01}$	**$4.54 \times 10^{+00}$**	$8.68 \times 10^{+05}$	$6.42 \times 10^{+05}$	$1.03 \times 10^{+07}$	$5.53 \times 10^{+05}$	$1.25 \times 10^{+05}$	$4.64 \times 10^{+05}$
Friedman Average Rank		**1.6167**	2.4489	2.9056	4.6561	7.9300	4.7728	5.3978	6.2722
Rank		**1**	2	3	4	8	5	6	7

Table A3. *p*-values obtained from Wilcoxon's rank sum test for MSMPA and other algorithms.

Function		MSMPA VS.						
		MPA	**PSO**	**GWO**	**WOA**	**MFO**	**SOA**	**SCA**
F1	*p*-value	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F2	*p*-value	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F3	*p*-value	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F4	*p*-value	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F5	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F6	*p*-value	8.56×10^{-04}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F7	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	1.07×10^{-09}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F8	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	8.10×10^{-10}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F9	*p*-value	**NaN**	1.21×10^{-12}	1.14×10^{-05}	**NaN**	1.21×10^{-12}	2.79×10^{-03}	1.21×10^{-12}
F10	*p*-value	1.17×10^{-13}	1.21×10^{-12}	8.56×10^{-13}	1.97×10^{-06}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F11	*p*-value	**NaN**	1.21×10^{-12}	$\mathbf{8.15 \times 10^{-02}}$	$\mathbf{8.15 \times 10^{-02}}$	1.21×10^{-12}	5.58×10^{-03}	1.21×10^{-12}
F12	*p*-value	1.25×10^{-07}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F13	*p*-value	8.15×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F14	*p*-value	$\mathbf{1.61 \times 10^{-01}}$	4.42×10^{-08}	1.21×10^{-12}	1.21×10^{-12}	2.90×10^{-05}	1.21×10^{-12}	1.21×10^{-12}
F15	*p*-value	1.05×10^{-05}	6.59×10^{-09}	1.30×10^{-09}	2.69×10^{-09}	1.08×10^{-09}	2.73×10^{-11}	1.30×10^{-09}
F16	*p*-value	3.49×10^{-04}	1.38×10^{-09}	8.87×10^{-12}	8.87×10^{-12}	$\mathbf{2.04 \times 10^{-01}}$	8.87×10^{-12}	8.87×10^{-12}
F17	*p*-value	4.18×10^{-02}	3.03×10^{-03}	4.08×10^{-12}	4.08×10^{-12}	$\mathbf{5.20 \times 10^{-02}}$	4.08×10^{-12}	4.08×10^{-12}
F18	*p*-value	2.85×10^{-05}	1.22×10^{-09}	4.08×10^{-12}	4.08×10^{-12}	3.40×10^{-02}	4.08×10^{-12}	4.08×10^{-12}
CF1	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	2.95×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF2	*p*-value	**NaN**	**NaN**	4.57×10^{-12}	1.21×10^{-12}	5.60×10^{-11}	1.21×10^{-12}	1.21×10^{-12}
CF3	*p*-value	3.02×10^{-11}	3.50×10^{-03}	3.02×10^{-11}	3.02×10^{-11}	$\mathbf{8.77 \times 10^{-01}}$	3.02×10^{-11}	3.02×10^{-11}
CF4	*p*-value	$\mathbf{1.09 \times 10^{-01}}$	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	2.86×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF5	*p*-value	1.06×10^{-03}	5.82×10^{-03}	1.25×10^{-07}	3.02×10^{-11}	3.02×10^{-11}	4.62×10^{-10}	3.02×10^{-11}
CF6	*p*-value	3.02×10^{-11}	9.37×10^{-12}	$\mathbf{9.35 \times 10^{-01}}$	3.02×10^{-11}	3.46×10^{-04}	3.02×10^{-11}	3.02×10^{-11}
CF7	*p*-value	5.86×10^{-06}	1.87×10^{-05}	$\mathbf{5.49 \times 10^{-01}}$	3.02×10^{-11}	2.57×10^{-07}	3.02×10^{-11}	3.02×10^{-11}
CF8	*p*-value	$\mathbf{6.35 \times 10^{-02}}$	5.94×10^{-05}	1.25×10^{-07}	3.02×10^{-11}	2.15×10^{-10}	3.02×10^{-11}	3.02×10^{-11}
CF9	*p*-value	3.02×10^{-11}	1.14×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	$\mathbf{7.26 \times 10^{-02}}$	3.02×10^{-11}	3.02×10^{-11}
CF10	*p*-value	1.38×10^{-02}	2.32×10^{-02}	7.12×10^{-09}	3.02×10^{-11}	8.99×10^{-11}	4.97×10^{-11}	3.02×10^{-11}
CF11	*p*-value	2.03×10^{-07}	$\mathbf{7.96 \times 10^{-01}}$	4.50×10^{-11}	3.02×10^{-11}	7.09×10^{-08}	3.02×10^{-11}	3.02×10^{-11}
CF12	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF13	*p*-value	1.56×10^{-08}	9.92×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF14	*p*-value	7.77×10^{-09}	2.02×10^{-08}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF15	*p*-value	4.69×10^{-08}	6.70×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF16	*p*-value	4.42×10^{-06}	2.68×10^{-06}	6.72×10^{-10}	3.34×10^{-11}	3.96×10^{-08}	8.10×10^{-10}	4.20×10^{-10}
CF17	*p*-value	3.82×10^{-09}	1.04×10^{-04}	1.33×10^{-10}	3.02×10^{-11}	4.80×10^{-07}	4.08×10^{-11}	3.02×10^{-11}
CF18	*p*-value	1.60×10^{-07}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF19	*p*-value	3.02×10^{-11}	4.50×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF20	*p*-value	$\mathbf{2.90 \times 10^{-01}}$	$\mathbf{5.59 \times 10^{-01}}$	4.50×10^{-11}	3.02×10^{-11}	1.07×10^{-07}	3.02×10^{-11}	3.02×10^{-11}
CF21	*p*-value	$\mathbf{4.29 \times 10^{-01}}$	1.41×10^{-04}	2.38×10^{-07}	4.11×10^{-07}	3.99×10^{-04}	$\mathbf{1.86 \times 10^{-01}}$	4.21×10^{-02}
CF22	*p*-value	1.78×10^{-10}	1.68×10^{-04}	1.00×10^{-03}	3.02×10^{-11}	$\mathbf{1.62 \times 10^{-01}}$	6.74×10^{-06}	5.07×10^{-10}
CF23	*p*-value	$\mathbf{2.12 \times 10^{-01}}$	$\mathbf{3.63 \times 10^{-01}}$	2.15×10^{-06}	3.02×10^{-11}	3.34×10^{-11}	3.34×10^{-11}	3.02×10^{-11}
CF24	*p*-value	4.62×10^{-10}	4.11×10^{-07}	9.76×10^{-10}	1.96×10^{-10}	3.02×10^{-11}	8.89×10^{-10}	3.96×10^{-08}
CF25	*p*-value	4.50×10^{-11}	2.77×10^{-05}	7.60×10^{-07}	3.02×10^{-11}	1.39×10^{-06}	7.66×10^{-05}	4.11×10^{-07}
CF26	*p*-value	2.61×10^{-10}	6.44×10^{-09}	8.48×10^{-09}	3.02×10^{-11}	9.47×10^{-06}	3.02×10^{-11}	3.02×10^{-11}
CF27	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
CF28	*p*-value	$\mathbf{1.67 \times 10^{-01}}$	$\mathbf{3.04 \times 10^{-01}}$	9.92×10^{-11}	3.02×10^{-11}	4.59×10^{-10}	2.92×10^{-09}	3.02×10^{-11}
CF29	*p*-value	3.03×10^{-03}	1.17×10^{-05}	5.00×10^{-09}	3.02×10^{-11}	7.38×10^{-10}	5.07×10^{-10}	3.02×10^{-11}
CF30	*p*-value	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.01×10^{-11}

Table A4. Test results for each high-dimensional function in dimension 100.

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
F1	Ave	$\mathbf{0.00 \times 10^{+00}}$	3.85×10^{-43}	$9.41 \times 10^{+01}$	1.84×10^{-29}	7.06×10^{-146}	$3.17 \times 10^{+04}$	6.07×10^{-15}	$5.59 \times 10^{+03}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	3.58×10^{-43}	$1.46 \times 10^{+01}$	1.73×10^{-29}	3.80×10^{-145}	$1.25 \times 10^{+04}$	1.14×10^{-14}	$4.77 \times 10^{+03}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	1.50×10^{-42}	$1.37 \times 10^{+02}$	6.52×10^{-29}	2.12×10^{-144}	$5.46 \times 10^{+04}$	4.98×10^{-14}	$1.91 \times 10^{+04}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	2.21×10^{-44}	$7.18 \times 10^{+01}$	4.12×10^{-30}	2.81×10^{-169}	$1.43 \times 10^{+04}$	1.68×10^{-17}	$2.73 \times 10^{+01}$
F2	Ave	$\mathbf{0.00 \times 10^{+00}}$	8.27×10^{-25}	$1.10 \times 10^{+02}$	5.34×10^{-18}	3.13×10^{-101}	$1.79 \times 10^{+02}$	8.13×10^{-11}	$1.99 \times 10^{+00}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	1.44×10^{-24}	$2.24 \times 10^{+01}$	3.05×10^{-18}	1.63×10^{-100}	$5.48 \times 10^{+01}$	6.58×10^{-11}	$2.27 \times 10^{+00}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	6.14×10^{-24}	$1.71 \times 10^{+02}$	1.66×10^{-17}	9.06×10^{-100}	$3.30 \times 10^{+02}$	2.30×10^{-10}	$9.10 \times 10^{+00}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	2.27×10^{-27}	$6.46 \times 10^{+01}$	2.06×10^{-18}	1.63×10^{-110}	$1.03 \times 10^{+02}$	5.48×10^{-12}	8.84×10^{-02}
F3	Ave	$\mathbf{0.00 \times 10^{+00}}$	2.49×10^{-02}	$1.39 \times 10^{+04}$	$1.06 \times 10^{+01}$	$8.92 \times 10^{+05}$	$1.80 \times 10^{+05}$	2.11×10^{-02}	$1.92 \times 10^{+05}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	9.79×10^{-02}	$2.69 \times 10^{+03}$	$2.70 \times 10^{+01}$	$2.40 \times 10^{+05}$	$4.35 \times 10^{+04}$	8.57×10^{-02}	$4.22 \times 10^{+04}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	5.42×10^{-01}	$2.00 \times 10^{+04}$	$1.41 \times 10^{+02}$	$1.45 \times 10^{+06}$	$2.80 \times 10^{+05}$	4.77×10^{-01}	$2.93 \times 10^{+05}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	1.13×10^{-07}	$1.00 \times 10^{+04}$	2.38×10^{-02}	$4.91 \times 10^{+05}$	$1.04 \times 10^{+05}$	3.52×10^{-07}	$1.12 \times 10^{+05}$
F4	Ave	$\mathbf{0.00 \times 10^{+00}}$	5.90×10^{-16}	$1.03 \times 10^{+01}$	1.60×10^{-02}	$7.20 \times 10^{+01}$	$9.32 \times 10^{+01}$	$5.60 \times 10^{+01}$	$8.66 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	3.81×10^{-16}	$1.03 \times 10^{+00}$	7.52×10^{-02}	$2.86 \times 10^{+01}$	$1.78 \times 10^{+00}$	$2.47 \times 10^{+01}$	$3.84 \times 10^{+00}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	1.88×10^{-15}	$1.25 \times 10^{+01}$	4.21×10^{-01}	$9.62 \times 10^{+01}$	$9.60 \times 10^{+01}$	$8.51 \times 10^{+01}$	$9.34 \times 10^{+01}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	1.52×10^{-16}	$8.46 \times 10^{+00}$	3.78×10^{-05}	1.10×10^{-04}	$8.84 \times 10^{+01}$	$3.28 \times 10^{+00}$	$7.87 \times 10^{+01}$
F5	Ave	$\mathbf{2.25 \times 10^{-05}}$	$9.57 \times 10^{+01}$	$1.04 \times 10^{+05}$	$9.76 \times 10^{+01}$	$9.78 \times 10^{+01}$	$6.84 \times 10^{+07}$	$9.86 \times 10^{+01}$	$5.17 \times 10^{+07}$
	Std	$\mathbf{3.40 \times 10^{-05}}$	$1.11 \times 10^{+00}$	$3.00 \times 10^{+04}$	6.31×10^{-01}	4.12×10^{-01}	$5.92 \times 10^{+07}$	1.90×10^{-01}	$3.00 \times 10^{+07}$
	Max	$\mathbf{1.32 \times 10^{-04}}$	$9.78 \times 10^{+01}$	$2.10 \times 10^{+05}$	$9.85 \times 10^{+01}$	$9.83 \times 10^{+01}$	$2.62 \times 10^{+08}$	$9.88 \times 10^{+01}$	$1.21 \times 10^{+08}$
	Min	$\mathbf{7.28 \times 10^{-15}}$	$9.41 \times 10^{+01}$	$6.35 \times 10^{+04}$	$9.61 \times 10^{+01}$	$9.70 \times 10^{+01}$	$3.05 \times 10^{+06}$	$9.80 \times 10^{+01}$	$1.04 \times 10^{+07}$
F6	Ave	$\mathbf{1.45 \times 10^{-05}}$	9.22×10^{-01}	$1.00 \times 10^{+02}$	$9.29 \times 10^{+00}$	$2.08 \times 10^{+00}$	$3.19 \times 10^{+04}$	$1.83 \times 10^{+01}$	$6.73 \times 10^{+03}$
	Std	$\mathbf{4.13 \times 10^{-05}}$	4.35×10^{-01}	$1.87 \times 10^{+01}$	8.26×10^{-01}	7.60×10^{-01}	$1.34 \times 10^{+04}$	7.55×10^{-01}	$4.69 \times 10^{+03}$
	Max	$\mathbf{2.25 \times 10^{-04}}$	$1.82 \times 10^{+00}$	$1.40 \times 10^{+02}$	$1.09 \times 10^{+01}$	$4.49 \times 10^{+00}$	$5.75 \times 10^{+04}$	$1.97 \times 10^{+01}$	$1.78 \times 10^{+04}$
	Min	$\mathbf{1.85 \times 10^{-08}}$	1.03×10^{-01}	$7.24 \times 10^{+01}$	$7.69 \times 10^{+00}$	$1.03 \times 10^{+00}$	$5.68 \times 10^{+03}$	$1.66 \times 10^{+01}$	$1.57 \times 10^{+02}$
F7	Ave	$\mathbf{3.26 \times 10^{-05}}$	7.46×10^{-04}	$1.41 \times 10^{+03}$	2.48×10^{-03}	2.60×10^{-03}	$1.81 \times 10^{+02}$	2.95×10^{-03}	$6.74 \times 10^{+01}$
	Std	$\mathbf{2.41 \times 10^{-05}}$	3.55×10^{-04}	$2.64 \times 10^{+02}$	7.68×10^{-04}	2.09×10^{-03}	$1.27 \times 10^{+02}$	1.82×10^{-03}	$3.73 \times 10^{+01}$
	Max	$\mathbf{1.05 \times 10^{-04}}$	2.00×10^{-03}	$2.07 \times 10^{+03}$	4.45×10^{-03}	7.12×10^{-03}	$5.70 \times 10^{+02}$	7.85×10^{-03}	$1.55 \times 10^{+02}$
	Min	$\mathbf{1.16 \times 10^{-06}}$	3.16×10^{-04}	$9.92 \times 10^{+02}$	1.29×10^{-03}	7.26×10^{-05}	$4.24 \times 10^{+01}$	3.40×10^{-04}	$8.02 \times 10^{+00}$
F8	Ave	$\mathbf{-1.02 \times 10^{+05}}$	$-2.76 \times 10^{+04}$	$-2.14 \times 10^{+04}$	$-1.59 \times 10^{+04}$	$-3.77 \times 10^{+04}$	$-2.34 \times 10^{+04}$	$-1.12 \times 10^{+04}$	$-7.30 \times 10^{+03}$
	Std	$1.36 \times 10^{+04}$	$9.19 \times 10^{+02}$	$3.47 \times 10^{+03}$	$2.36 \times 10^{+03}$	$4.85 \times 10^{+03}$	$1.94 \times 10^{+03}$	$1.49 \times 10^{+03}$	$\mathbf{5.79 \times 10^{+02}}$
	Max	$\mathbf{-3.73 \times 10^{+04}}$	$-2.56 \times 10^{+04}$	$-7.61 \times 10^{+03}$	$-5.92 \times 10^{+03}$	$-2.87 \times 10^{+04}$	$-2.04 \times 10^{+04}$	$-8.71 \times 10^{+03}$	$-6.30 \times 10^{+03}$
	Min	$\mathbf{-1.09 \times 10^{+05}}$	$-2.99 \times 10^{+04}$	$-2.65 \times 10^{+04}$	$-2.01 \times 10^{+04}$	$-4.19 \times 10^{+04}$	$-2.75 \times 10^{+04}$	$-1.48 \times 10^{+04}$	$-8.72 \times 10^{+03}$
F9	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.10 \times 10^{+03}$	3.57×10^{-01}	7.58×10^{-15}	$7.61 \times 10^{+02}$	2.10×10^{-12}	$2.45 \times 10^{+02}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.06 \times 10^{+02}$	$1.00 \times 10^{+00}$	4.08×10^{-14}	$7.43 \times 10^{+01}$	8.14×10^{-12}	$9.94 \times 10^{+01}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.27 \times 10^{+03}$	$3.92 \times 10^{+00}$	2.27×10^{-13}	$9.24 \times 10^{+02}$	4.48×10^{-11}	$4.81 \times 10^{+02}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$8.67 \times 10^{+02}$	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$5.95 \times 10^{+02}$	$\mathbf{0.00 \times 10^{+00}}$	$5.70 \times 10^{+01}$
F10	Ave	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$5.45 \times 10^{+00}$	1.12×10^{-13}	3.73×10^{-15}	$1.98 \times 10^{+01}$	$2.00 \times 10^{+01}$	$1.93 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	2.94×10^{-01}	1.09×10^{-14}	2.66×10^{-15}	2.26×10^{-01}	2.26×10^{-04}	$3.81 \times 10^{+00}$
	Max	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$5.93 \times 10^{+00}$	1.47×10^{-13}	7.99×10^{-15}	$2.00 \times 10^{+01}$	$2.00 \times 10^{+01}$	$2.07 \times 10^{+01}$
	Min	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$4.71 \times 10^{+00}$	9.33×10^{-14}	$\mathbf{8.88 \times 10^{-16}}$	$1.92 \times 10^{+01}$	$2.00 \times 10^{+01}$	$5.48 \times 10^{+00}$
F11	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	7.96×10^{-01}	3.16×10^{-03}	3.78×10^{-03}	$2.98 \times 10^{+02}$	5.27×10^{-03}	$4.75 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	7.24×10^{-02}	6.69×10^{-03}	2.03×10^{-02}	$1.28 \times 10^{+02}$	1.77×10^{-02}	$3.63 \times 10^{+01}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	9.26×10^{-01}	2.59×10^{-02}	1.13×10^{-01}	$5.81 \times 10^{+02}$	8.51×10^{-02}	$1.27 \times 10^{+02}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	6.41×10^{-01}	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$6.68 \times 10^{+01}$	$\mathbf{0.00 \times 10^{+00}}$	$3.69 \times 10^{+00}$
F12	Ave	$\mathbf{3.17 \times 10^{-09}}$	9.65×10^{-03}	$5.95 \times 10^{+00}$	2.45×10^{-01}	1.50×10^{-02}	$5.35 \times 10^{+07}$	7.53×10^{-01}	$1.63 \times 10^{+08}$
	Std	$\mathbf{1.26 \times 10^{-08}}$	4.30×10^{-03}	$3.16 \times 10^{+00}$	6.61×10^{-02}	5.90×10^{-03}	$6.46 \times 10^{+07}$	6.75×10^{-02}	$9.46 \times 10^{+07}$
	Max	$\mathbf{7.02 \times 10^{-08}}$	1.98×10^{-02}	$1.53 \times 10^{+01}$	4.50×10^{-01}	2.84×10^{-02}	$2.65 \times 10^{+08}$	8.78×10^{-01}	$3.77 \times 10^{+08}$
	Min	$\mathbf{2.48 \times 10^{-15}}$	3.03×10^{-03}	$2.84 \times 10^{+00}$	1.61×10^{-01}	8.09×10^{-03}	$7.58 \times 10^{+05}$	6.21×10^{-01}	$1.73 \times 10^{+07}$
F13	Ave	$\mathbf{1.40 \times 10^{-07}}$	$7.00 \times 10^{+00}$	$1.02 \times 10^{+02}$	$6.04 \times 10^{+00}$	$1.75 \times 10^{+00}$	$2.97 \times 10^{+08}$	$8.98 \times 10^{+00}$	$3.38 \times 10^{+08}$
	Std	$\mathbf{3.46 \times 10^{-07}}$	$1.76 \times 10^{+00}$	$3.79 \times 10^{+01}$	4.62×10^{-01}	6.43×10^{-01}	$2.67 \times 10^{+08}$	2.88×10^{-01}	$2.02 \times 10^{+08}$
	Max	$\mathbf{1.44 \times 10^{-06}}$	$8.86 \times 10^{+00}$	$1.99 \times 10^{+02}$	$6.84 \times 10^{+00}$	$3.40 \times 10^{+00}$	$9.02 \times 10^{+08}$	$9.63 \times 10^{+00}$	$8.89 \times 10^{+08}$
	Min	$\mathbf{8.81 \times 10^{-14}}$	$2.32 \times 10^{+00}$	$5.92 \times 10^{+01}$	$5.25 \times 10^{+00}$	7.38×10^{-01}	$1.48 \times 10^{+07}$	$8.36 \times 10^{+00}$	$8.08 \times 10^{+07}$
Friedman Average Rank		**1.2192**	2.5654	6.0026	3.9192	3.2449	7.0949	4.9436	7.0103
Rank		**1**	2	6	4	3	8	5	7

Table A5. Test results for each high-dimensional function in dimension 200.

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
F1	Ave	$\mathbf{0.00 \times 10^{+00}}$	3.65×10^{-41}	$6.66 \times 10^{+02}$	4.56×10^{-20}	2.02×10^{-145}	$1.85 \times 10^{+05}$	7.43×10^{-12}	$3.09 \times 10^{+04}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	5.44×10^{-41}	$7.06 \times 10^{+01}$	3.74×10^{-20}	1.09×10^{-144}	$2.68 \times 10^{+04}$	9.09×10^{-12}	$1.20 \times 10^{+04}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	2.13×10^{-40}	$7.89 \times 10^{+02}$	1.78×10^{-19}	6.05×10^{-144}	$2.51 \times 10^{+05}$	3.52×10^{-11}	$4.97 \times 10^{+04}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	5.40×10^{-43}	$5.44 \times 10^{+02}$	6.38×10^{-21}	3.67×10^{-167}	$1.28 \times 10^{+05}$	1.61×10^{-13}	$8.45 \times 10^{+03}$
F2	Ave	$\mathbf{0.00 \times 10^{+00}}$	6.55×10^{-24}	$1.66 \times 10^{+21}$	1.41×10^{-12}	1.40×10^{-101}	$5.62 \times 10^{+02}$	6.40×10^{-09}	$1.61 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	8.57×10^{-24}	$8.94 \times 10^{+21}$	4.75×10^{-13}	7.52×10^{-101}	$5.37 \times 10^{+01}$	5.07×10^{-09}	$1.46 \times 10^{+01}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	3.26×10^{-23}	$4.98 \times 10^{+22}$	2.44×10^{-12}	4.19×10^{-100}	$6.90 \times 10^{+02}$	1.88×10^{-08}	$7.48 \times 10^{+01}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	8.37×10^{-27}	$5.42 \times 10^{+02}$	5.81×10^{-13}	9.51×10^{-116}	$4.59 \times 10^{+02}$	7.53×10^{-10}	7.98×10^{-01}
F3	Ave	$\mathbf{0.00 \times 10^{+00}}$	$1.82 \times 10^{+01}$	$8.16 \times 10^{+04}$	$3.22 \times 10^{+03}$	$4.56 \times 10^{+06}$	$7.31 \times 10^{+05}$	$1.74 \times 10^{+02}$	$8.38 \times 10^{+05}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$4.51 \times 10^{+01}$	$2.07 \times 10^{+04}$	$2.46 \times 10^{+03}$	$1.42 \times 10^{+06}$	$1.37 \times 10^{+05}$	$3.79 \times 10^{+02}$	$1.51 \times 10^{+05}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$2.50 \times 10^{+02}$	$1.62 \times 10^{+05}$	$1.10 \times 10^{+04}$	$7.63 \times 10^{+06}$	$1.02 \times 10^{+06}$	$1.77 \times 10^{+03}$	$1.17 \times 10^{+06}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	1.83×10^{-05}	$5.26 \times 10^{+04}$	$1.67 \times 10^{+02}$	$1.70 \times 10^{+06}$	$4.24 \times 10^{+05}$	8.97×10^{-03}	$5.52 \times 10^{+05}$
F4	Ave	$\mathbf{0.00 \times 10^{+00}}$	1.01×10^{-14}	$1.95 \times 10^{+01}$	$9.86 \times 10^{+00}$	$7.72 \times 10^{+01}$	$9.70 \times 10^{+01}$	$9.20 \times 10^{+01}$	$9.51 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	4.83×10^{-15}	$1.53 \times 10^{+00}$	$5.31 \times 10^{+00}$	$2.46 \times 10^{+01}$	7.95×10^{-01}	$3.87 \times 10^{+00}$	$1.34 \times 10^{+00}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	2.33×10^{-14}	$2.41 \times 10^{+01}$	$2.33 \times 10^{+01}$	$9.83 \times 10^{+01}$	$9.87 \times 10^{+01}$	$9.81 \times 10^{+01}$	$9.69 \times 10^{+01}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	3.70×10^{-15}	$1.70 \times 10^{+01}$	$1.70 \times 10^{+00}$	$2.14 \times 10^{+01}$	$9.53 \times 10^{+01}$	$7.82 \times 10^{+01}$	$9.18 \times 10^{+01}$
F5	Ave	$\mathbf{1.02 \times 10^{-04}}$	$1.96 \times 10^{+02}$	$1.66 \times 10^{+06}$	$1.98 \times 10^{+02}$	$1.97 \times 10^{+02}$	$6.50 \times 10^{+08}$	$1.99 \times 10^{+02}$	$3.48 \times 10^{+08}$
	Std	$\mathbf{2.67 \times 10^{-04}}$	9.30×10^{-01}	$2.85 \times 10^{+05}$	6.41×10^{-01}	2.29×10^{-01}	$1.09 \times 10^{+08}$	1.44×10^{-01}	$1.06 \times 10^{+08}$
	Max	$\mathbf{1.42 \times 10^{-03}}$	$1.97 \times 10^{+02}$	$2.15 \times 10^{+06}$	$1.98 \times 10^{+02}$	$1.98 \times 10^{+02}$	$9.02 \times 10^{+08}$	$1.99 \times 10^{+02}$	$6.70 \times 10^{+08}$
	Min	$\mathbf{1.70 \times 10^{-13}}$	$1.94 \times 10^{+02}$	$1.14 \times 10^{+06}$	$1.96 \times 10^{+02}$	$1.97 \times 10^{+02}$	$4.56 \times 10^{+08}$	$1.98 \times 10^{+02}$	$1.85 \times 10^{+08}$
F6	Ave	$\mathbf{1.26 \times 10^{-05}}$	$8.10 \times 10^{+00}$	$6.94 \times 10^{+02}$	$2.79 \times 10^{+01}$	$5.90 \times 10^{+00}$	$1.76 \times 10^{+05}$	$4.22 \times 10^{+01}$	$3.33 \times 10^{+04}$
	Std	$\mathbf{2.31 \times 10^{-05}}$	$1.04 \times 10^{+00}$	$9.50 \times 10^{+01}$	$1.05 \times 10^{+00}$	$1.60 \times 10^{+00}$	$2.03 \times 10^{+04}$	8.24×10^{-01}	$1.29 \times 10^{+04}$
	Max	$\mathbf{1.13 \times 10^{-04}}$	$1.13 \times 10^{+01}$	$9.09 \times 10^{+02}$	$3.03 \times 10^{+01}$	$9.21 \times 10^{+00}$	$2.08 \times 10^{+05}$	$4.44 \times 10^{+01}$	$6.01 \times 10^{+04}$
	Min	$\mathbf{3.84 \times 10^{-08}}$	$6.64 \times 10^{+00}$	$4.85 \times 10^{+02}$	$2.56 \times 10^{+01}$	$2.94 \times 10^{+00}$	$1.40 \times 10^{+05}$	$4.08 \times 10^{+01}$	$5.36 \times 10^{+03}$
F7	Ave	$\mathbf{2.76 \times 10^{-05}}$	8.44×10^{-04}	$7.55 \times 10^{+03}$	4.41×10^{-03}	1.83×10^{-03}	$1.85 \times 10^{+03}$	4.48×10^{-03}	$9.36 \times 10^{+02}$
	Std	$\mathbf{2.47 \times 10^{-05}}$	2.82×10^{-04}	$9.21 \times 10^{+02}$	1.57×10^{-03}	2.02×10^{-03}	$4.40 \times 10^{+02}$	2.51×10^{-03}	$3.30 \times 10^{+02}$
	Max	$\mathbf{8.58 \times 10^{-05}}$	1.42×10^{-03}	$9.37 \times 10^{+03}$	7.72×10^{-03}	5.60×10^{-03}	$2.87 \times 10^{+03}$	1.09×10^{-02}	$1.77 \times 10^{+03}$
	Min	$\mathbf{3.62 \times 10^{-07}}$	3.35×10^{-04}	$5.62 \times 10^{+03}$	1.58×10^{-03}	3.43×10^{-05}	$1.02 \times 10^{+03}$	7.65×10^{-04}	$3.59 \times 10^{+02}$
F8	Ave	$\mathbf{-2.09 \times 10^{+05}}$	$-4.89 \times 10^{+04}$	$-3.92 \times 10^{+04}$	$-2.95 \times 10^{+04}$	$-7.18 \times 10^{+04}$	$-3.97 \times 10^{+04}$	$-1.55 \times 10^{+04}$	$-1.04 \times 10^{+04}$
	Std	$2.57 \times 10^{+04}$	$1.67 \times 10^{+03}$	$8.71 \times 10^{+03}$	$2.16 \times 10^{+03}$	$1.08 \times 10^{+04}$	$3.44 \times 10^{+03}$	$2.77 \times 10^{+03}$	$\mathbf{7.08 \times 10^{+02}}$
	Max	$\mathbf{-8.16 \times 10^{+04}}$	$-4.50 \times 10^{+04}$	$-8.93 \times 10^{+03}$	$-2.52 \times 10^{+04}$	$-5.03 \times 10^{+04}$	$-3.38 \times 10^{+04}$	$-1.21 \times 10^{+04}$	$-8.57 \times 10^{+03}$
	Min	$\mathbf{-2.18 \times 10^{+05}}$	$-5.28 \times 10^{+04}$	$-5.31 \times 10^{+04}$	$-3.35 \times 10^{+04}$	$-8.38 \times 10^{+04}$	$-4.85 \times 10^{+04}$	$-2.32 \times 10^{+04}$	$-1.19 \times 10^{+04}$
F9	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$2.71 \times 10^{+03}$	$1.48 \times 10^{+00}$	1.52×10^{-14}	$1.96 \times 10^{+03}$	$2.06 \times 10^{+00}$	$4.89 \times 10^{+02}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.72 \times 10^{+02}$	$3.30 \times 10^{+00}$	8.16×10^{-14}	$9.12 \times 10^{+01}$	$5.77 \times 10^{+00}$	$2.06 \times 10^{+02}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$3.03 \times 10^{+03}$	$1.47 \times 10^{+01}$	4.55×10^{-13}	$2.13 \times 10^{+03}$	$2.69 \times 10^{+01}$	$9.71 \times 10^{+02}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$2.40 \times 10^{+03}$	2.27×10^{-13}	$\mathbf{0.00 \times 10^{+00}}$	$1.74 \times 10^{+03}$	4.55×10^{-13}	$3.80 \times 10^{+01}$
F10	Ave	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$8.26 \times 10^{+00}$	1.35×10^{-11}	4.09×10^{-15}	$2.00 \times 10^{+01}$	$2.00 \times 10^{+01}$	$1.85 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	3.02×10^{-01}	3.57×10^{-12}	2.12×10^{-15}	1.56×10^{-02}	1.73×10^{-04}	$4.14 \times 10^{+00}$
	Max	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$9.01 \times 10^{+00}$	2.03×10^{-11}	7.99×10^{-15}	$2.00 \times 10^{+01}$	$2.00 \times 10^{+01}$	$2.07 \times 10^{+01}$
	Min	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$7.69 \times 10^{+00}$	6.78×10^{-12}	$\mathbf{8.88 \times 10^{-16}}$	$1.99 \times 10^{+01}$	$2.00 \times 10^{+01}$	$8.37 \times 10^{+00}$
F11	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.19 \times 10^{+00}$	4.17×10^{-04}	$\mathbf{0.00 \times 10^{+00}}$	$1.67 \times 10^{+03}$	5.75×10^{-03}	$2.94 \times 10^{+02}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	2.46×10^{-02}	2.24×10^{-03}	$\mathbf{0.00 \times 10^{+00}}$	$2.06 \times 10^{+02}$	1.54×10^{-02}	$1.67 \times 10^{+02}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.26 \times 10^{+00}$	1.25×10^{-02}	$\mathbf{0.00 \times 10^{+00}}$	$2.03 \times 10^{+03}$	6.44×10^{-02}	$8.22 \times 10^{+02}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$1.15 \times 10^{+00}$	1.11×10^{-16}	$\mathbf{0.00 \times 10^{+00}}$	$1.35 \times 10^{+03}$	8.15×10^{-14}	$6.68 \times 10^{+01}$
F12	Ave	$\mathbf{4.26 \times 10^{-10}}$	4.80×10^{-02}	$1.35 \times 10^{+02}$	4.84×10^{-01}	2.44×10^{-02}	$1.28 \times 10^{+09}$	8.91×10^{-01}	$9.91 \times 10^{+08}$
	Std	$\mathbf{1.71 \times 10^{-09}}$	6.59×10^{-03}	$1.35 \times 10^{+02}$	5.06×10^{-02}	9.13×10^{-03}	$3.11 \times 10^{+08}$	3.28×10^{-02}	$3.66 \times 10^{+08}$
	Max	$\mathbf{9.41 \times 10^{-09}}$	6.18×10^{-02}	$6.02 \times 10^{+02}$	5.71×10^{-01}	5.21×10^{-02}	$2.14 \times 10^{+09}$	9.56×10^{-01}	$1.82 \times 10^{+09}$
	Min	$\mathbf{1.31 \times 10^{-19}}$	3.57×10^{-02}	$3.67 \times 10^{+01}$	3.88×10^{-01}	9.73×10^{-03}	$8.47 \times 10^{+08}$	8.10×10^{-01}	$3.88 \times 10^{+08}$
F13	Ave	$\mathbf{1.35 \times 10^{-07}}$	$1.80 \times 10^{+01}$	$2.17 \times 10^{+04}$	$1.61 \times 10^{+01}$	$4.29 \times 10^{+00}$	$2.48 \times 10^{+09}$	$1.90 \times 10^{+01}$	$1.56 \times 10^{+09}$
	Std	$\mathbf{4.51 \times 10^{-07}}$	4.03×10^{-01}	$9.83 \times 10^{+03}$	4.78×10^{-01}	$1.33 \times 10^{+00}$	$5.53 \times 10^{+08}$	2.17×10^{-01}	$5.53 \times 10^{+08}$
	Max	$\mathbf{2.39 \times 10^{-06}}$	$1.87 \times 10^{+01}$	$4.99 \times 10^{+04}$	$1.70 \times 10^{+01}$	$7.36 \times 10^{+00}$	$3.76 \times 10^{+09}$	$1.97 \times 10^{+01}$	$2.81 \times 10^{+09}$
	Min	$\mathbf{2.74 \times 10^{-20}}$	$1.71 \times 10^{+01}$	$8.75 \times 10^{+03}$	$1.51 \times 10^{+01}$	$1.48 \times 10^{+00}$	$1.47 \times 10^{+09}$	$1.85 \times 10^{+01}$	$6.37 \times 10^{+08}$
Friedman Average Rank		**1.1615**	2.6487	6.0256	4.0372	2.9410	7.2051	5.1038	6.8769
Rank		**1**	2	6	4	3	8	5	7

Table A6. Test results for each high-dimensional function with dimension 500.

Function	Criteria	MSMPA	MPA	PSO	GWO	WOA	MFO	SOA	SCA
F1	Ave	$\mathbf{0.00 \times 10^{+00}}$	4.83×10^{-39}	$7.34 \times 10^{+03}$	1.47×10^{-12}	2.28×10^{-145}	$9.71 \times 10^{+05}$	5.54×10^{-09}	$1.45 \times 10^{+05}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	5.89×10^{-39}	$4.01 \times 10^{+02}$	7.71×10^{-13}	1.22×10^{-144}	$3.73 \times 10^{+04}$	5.53×10^{-09}	$4.32 \times 10^{+04}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	2.68×10^{-38}	$7.94 \times 10^{+03}$	3.35×10^{-12}	6.82×10^{-144}	$1.04 \times 10^{+06}$	1.81×10^{-08}	$2.37 \times 10^{+05}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	9.09×10^{-41}	$6.32 \times 10^{+03}$	4.23×10^{-13}	3.36×10^{-164}	$9.02 \times 10^{+05}$	7.71×10^{-11}	$4.46 \times 10^{+04}$
F2	Ave	4.63×10^{-36}	3.83×10^{-12}	$2.47 \times 10^{+134}$	5.89×10^{-08}	$\mathbf{4.95 \times 10^{-99}}$	$2.24 \times 10^{+03}$	2.05×10^{-07}	$7.28 \times 10^{+01}$
	Std	1.57×10^{-35}	2.06×10^{-11}	$1.33 \times 10^{+135}$	1.37×10^{-08}	$\mathbf{2.66 \times 10^{-98}}$	$9.70 \times 10^{+01}$	1.80×10^{-07}	$4.68 \times 10^{+01}$
	Max	7.61×10^{-35}	1.15×10^{-10}	$7.42 \times 10^{+135}$	9.51×10^{-08}	$\mathbf{1.48 \times 10^{-97}}$	$2.37 \times 10^{+03}$	8.30×10^{-07}	$2.15 \times 10^{+02}$
	Min	6.78×10^{-66}	7.25×10^{-25}	$4.29 \times 10^{+28}$	3.55×10^{-08}	$\mathbf{7.77 \times 10^{-113}}$	$2.00 \times 10^{+03}$	3.58×10^{-08}	$1.47 \times 10^{+01}$
F3	Ave	$\mathbf{0.00 \times 10^{+00}}$	$1.63 \times 10^{+03}$	$5.50 \times 10^{+05}$	$1.31 \times 10^{+05}$	$2.87 \times 10^{+07}$	$3.97 \times 10^{+06}$	$3.07 \times 10^{+04}$	$5.83 \times 10^{+06}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$2.12 \times 10^{+03}$	$1.15 \times 10^{+05}$	$4.40 \times 10^{+04}$	$9.45 \times 10^{+06}$	$6.08 \times 10^{+05}$	$5.00 \times 10^{+04}$	$1.09 \times 10^{+06}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$7.46 \times 10^{+03}$	$9.02 \times 10^{+05}$	$2.27 \times 10^{+05}$	$5.00 \times 10^{+07}$	$5.65 \times 10^{+06}$	$2.34 \times 10^{+05}$	$8.20 \times 10^{+06}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$1.85 \times 10^{+01}$	$3.77 \times 10^{+05}$	$5.61 \times 10^{+04}$	$1.40 \times 10^{+07}$	$3.03 \times 10^{+06}$	$9.77 \times 10^{+00}$	$2.78 \times 10^{+06}$
F4	Ave	$\mathbf{0.00 \times 10^{+00}}$	5.34×10^{-13}	$2.79 \times 10^{+01}$	$5.74 \times 10^{+01}$	$8.08 \times 10^{+01}$	$9.89 \times 10^{+01}$	$9.80 \times 10^{+01}$	$9.88 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	4.83×10^{-13}	$1.04 \times 10^{+00}$	$5.82 \times 10^{+00}$	$1.57 \times 10^{+01}$	3.50×10^{-01}	7.60×10^{-01}	3.71×10^{-01}
	Max	$\mathbf{0.00 \times 10^{+00}}$	2.35×10^{-12}	$2.99 \times 10^{+01}$	$6.81 \times 10^{+01}$	$9.84 \times 10^{+01}$	$9.95 \times 10^{+01}$	$9.93 \times 10^{+01}$	$9.93 \times 10^{+01}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	6.38×10^{-14}	$2.59 \times 10^{+01}$	$4.41 \times 10^{+01}$	$3.26 \times 10^{+01}$	$9.81 \times 10^{+01}$	$9.53 \times 10^{+01}$	$9.79 \times 10^{+01}$
F5	Ave	$\mathbf{2.22 \times 10^{-04}}$	$4.96 \times 10^{+02}$	$5.16 \times 10^{+07}$	$4.98 \times 10^{+02}$	$4.96 \times 10^{+02}$	$4.01 \times 10^{+09}$	$4.99 \times 10^{+02}$	$1.48 \times 10^{+09}$
	Std	$\mathbf{8.38 \times 10^{-04}}$	4.67×10^{-01}	$5.90 \times 10^{+06}$	2.36×10^{-01}	3.08×10^{-01}	$2.39 \times 10^{+08}$	5.87×10^{-02}	$2.87 \times 10^{+08}$
	Max	$\mathbf{4.62 \times 10^{-03}}$	$4.96 \times 10^{+02}$	$6.31 \times 10^{+07}$	$4.98 \times 10^{+02}$	$4.97 \times 10^{+02}$	$4.32 \times 10^{+09}$	$4.99 \times 10^{+02}$	$2.00 \times 10^{+09}$
	Min	$\mathbf{1.16 \times 10^{-21}}$	$4.94 \times 10^{+02}$	$3.78 \times 10^{+07}$	$4.97 \times 10^{+02}$	$4.95 \times 10^{+02}$	$3.48 \times 10^{+09}$	$4.98 \times 10^{+02}$	$9.03 \times 10^{+08}$
F6	Ave	$\mathbf{1.77 \times 10^{-05}}$	$5.21 \times 10^{+01}$	$7.37 \times 10^{+03}$	$9.29 \times 10^{+01}$	$2.04 \times 10^{+01}$	$9.54 \times 10^{+05}$	$1.16 \times 10^{+02}$	$1.70 \times 10^{+05}$
	Std	$\mathbf{3.55 \times 10^{-05}}$	$1.74 \times 10^{+00}$	$4.36 \times 10^{+02}$	$1.72 \times 10^{+00}$	$7.52 \times 10^{+00}$	$3.83 \times 10^{+04}$	8.53×10^{-01}	$6.92 \times 10^{+04}$
	Max	$\mathbf{1.93 \times 10^{-04}}$	$5.63 \times 10^{+01}$	$8.27 \times 10^{+03}$	$9.58 \times 10^{+01}$	$4.01 \times 10^{+01}$	$1.07 \times 10^{+06}$	$1.17 \times 10^{+02}$	$3.91 \times 10^{+05}$
	Min	$\mathbf{1.32 \times 10^{-10}}$	$4.92 \times 10^{+01}$	$6.30 \times 10^{+03}$	$8.98 \times 10^{+01}$	$1.01 \times 10^{+01}$	$8.82 \times 10^{+05}$	$1.14 \times 10^{+02}$	$5.59 \times 10^{+04}$
F7	Ave	$\mathbf{3.04 \times 10^{-05}}$	1.30×10^{-03}	$5.68 \times 10^{+04}$	1.25×10^{-02}	2.16×10^{-03}	$3.08 \times 10^{+04}$	9.87×10^{-03}	$1.13 \times 10^{+04}$
	Std	$\mathbf{2.37 \times 10^{-05}}$	7.40×10^{-04}	$2.14 \times 10^{+03}$	4.40×10^{-03}	2.16×10^{-03}	$1.95 \times 10^{+03}$	6.62×10^{-03}	$2.83 \times 10^{+03}$
	Max	$\mathbf{1.06 \times 10^{-04}}$	2.77×10^{-03}	$6.12 \times 10^{+04}$	2.35×10^{-02}	1.17×10^{-02}	$3.40 \times 10^{+04}$	3.47×10^{-02}	$1.65 \times 10^{+04}$
	Min	$\mathbf{2.04 \times 10^{-06}}$	1.28×10^{-04}	$5.19 \times 10^{+04}$	6.37×10^{-03}	2.64×10^{-05}	$2.72 \times 10^{+04}$	2.55×10^{-03}	$5.74 \times 10^{+03}$
F8	Ave	$\mathbf{-5.35 \times 10^{+05}}$	$-9.78 \times 10^{+04}$	$-9.25 \times 10^{+04}$	$-5.96 \times 10^{+04}$	$-1.92 \times 10^{+05}$	$-7.43 \times 10^{+04}$	$-2.65 \times 10^{+04}$	$-1.59 \times 10^{+04}$
	Std	$1.93 \times 10^{+04}$	$2.46 \times 10^{+03}$	$2.53 \times 10^{+04}$	$9.63 \times 10^{+03}$	$2.29 \times 10^{+04}$	$6.21 \times 10^{+03}$	$5.83 \times 10^{+03}$	$\mathbf{1.01 \times 10^{+03}}$
	Max	$\mathbf{-4.49 \times 10^{+05}}$	$-9.20 \times 10^{+04}$	$-1.36 \times 10^{+04}$	$-1.34 \times 10^{+04}$	$-1.45 \times 10^{+05}$	$-6.32 \times 10^{+04}$	$-1.89 \times 10^{+04}$	$-1.41 \times 10^{+04}$
	Min	$\mathbf{-5.45 \times 10^{+05}}$	$-1.05 \times 10^{+05}$	$-1.15 \times 10^{+05}$	$-7.17 \times 10^{+04}$	$-2.09 \times 10^{+05}$	$-8.84 \times 10^{+04}$	$-4.58 \times 10^{+04}$	$-1.90 \times 10^{+04}$
F9	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$7.90 \times 10^{+03}$	$5.58 \times 10^{+00}$	$\mathbf{0.00 \times 10^{+00}}$	$6.47 \times 10^{+03}$	4.53×10^{-01}	$1.24 \times 10^{+03}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$2.80 \times 10^{+02}$	$6.52 \times 10^{+00}$	$\mathbf{0.00 \times 10^{+00}}$	$1.62 \times 10^{+02}$	$2.03 \times 10^{+00}$	$5.14 \times 10^{+02}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$8.34 \times 10^{+03}$	$2.38 \times 10^{+01}$	$\mathbf{0.00 \times 10^{+00}}$	$6.85 \times 10^{+03}$	$1.11 \times 10^{+01}$	$2.37 \times 10^{+03}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$6.99 \times 10^{+03}$	7.46×10^{-11}	$\mathbf{0.00 \times 10^{+00}}$	$6.21 \times 10^{+03}$	2.73×10^{-12}	$3.61 \times 10^{+02}$
F10	Ave	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$1.29 \times 10^{+01}$	5.84×10^{-08}	4.09×10^{-15}	$2.01 \times 10^{+01}$	$2.00 \times 10^{+01}$	$1.89 \times 10^{+01}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	1.93×10^{-01}	1.60×10^{-08}	2.49×10^{-15}	1.35×10^{-01}	6.35×10^{-05}	$3.77 \times 10^{+00}$
	Max	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$1.32 \times 10^{+01}$	9.69×10^{-08}	7.99×10^{-15}	$2.04 \times 10^{+01}$	$2.00 \times 10^{+01}$	$2.08 \times 10^{+01}$
	Min	$\mathbf{8.88 \times 10^{-16}}$	4.44×10^{-15}	$1.26 \times 10^{+01}$	3.53×10^{-08}	$\mathbf{8.88 \times 10^{-16}}$	$2.00 \times 10^{+01}$	$2.00 \times 10^{+01}$	$8.93 \times 10^{+00}$
F11	Ave	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$3.47 \times 10^{+00}$	1.22×10^{-03}	$\mathbf{0.00 \times 10^{+00}}$	$8.69 \times 10^{+03}$	1.69×10^{-03}	$1.55 \times 10^{+03}$
	Std	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	1.24×10^{-01}	4.84×10^{-03}	$\mathbf{0.00 \times 10^{+00}}$	$4.05 \times 10^{+02}$	6.37×10^{-03}	$5.95 \times 10^{+02}$
	Max	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$3.75 \times 10^{+00}$	2.45×10^{-02}	$\mathbf{0.00 \times 10^{+00}}$	$9.32 \times 10^{+03}$	2.87×10^{-02}	$3.26 \times 10^{+03}$
	Min	$\mathbf{0.00 \times 10^{+00}}$	$\mathbf{0.00 \times 10^{+00}}$	$3.27 \times 10^{+00}$	7.19×10^{-14}	$\mathbf{0.00 \times 10^{+00}}$	$7.66 \times 10^{+03}$	3.62×10^{-12}	$5.10 \times 10^{+02}$
F12	Ave	$\mathbf{2.97 \times 10^{-11}}$	2.01×10^{-01}	$9.21 \times 10^{+05}$	7.52×10^{-01}	4.25×10^{-02}	$9.39 \times 10^{+09}$	$1.02 \times 10^{+00}$	$3.98 \times 10^{+09}$
	Std	$\mathbf{1.57 \times 10^{-10}}$	1.83×10^{-02}	$3.10 \times 10^{+05}$	3.43×10^{-02}	1.51×10^{-02}	$8.34 \times 10^{+08}$	2.24×10^{-02}	$1.01 \times 10^{+09}$
	Max	$\mathbf{8.76 \times 10^{-10}}$	2.42×10^{-01}	$1.57 \times 10^{+06}$	8.25×10^{-01}	7.47×10^{-02}	$1.09 \times 10^{+10}$	$1.07 \times 10^{+00}$	$5.90 \times 10^{+09}$
	Min	$\mathbf{5.25 \times 10^{-22}}$	1.69×10^{-01}	$4.69 \times 10^{+05}$	6.86×10^{-01}	1.85×10^{-02}	$7.38 \times 10^{+09}$	9.78×10^{-01}	$1.75 \times 10^{+09}$
F13	Ave	$\mathbf{3.83 \times 10^{-13}}$	$4.75 \times 10^{+01}$	$8.79 \times 10^{+06}$	$4.60 \times 10^{+01}$	$1.06 \times 10^{+01}$	$1.76 \times 10^{+10}$	$4.95 \times 10^{+01}$	$6.93 \times 10^{+09}$
	Std	$\mathbf{1.53 \times 10^{-12}}$	5.31×10^{-01}	$1.67 \times 10^{+06}$	5.50×10^{-01}	$3.28 \times 10^{+00}$	$1.39 \times 10^{+09}$	5.33×10^{-01}	$1.57 \times 10^{+09}$
	Max	$\mathbf{8.49 \times 10^{-12}}$	$4.84 \times 10^{+01}$	$1.17 \times 10^{+07}$	$4.69 \times 10^{+01}$	$1.82 \times 10^{+01}$	$2.00 \times 10^{+10}$	$5.05 \times 10^{+01}$	$9.33 \times 10^{+09}$
	Min	$\mathbf{1.55 \times 10^{-22}}$	$4.64 \times 10^{+01}$	$5.43 \times 10^{+06}$	$4.51 \times 10^{+01}$	$5.31 \times 10^{+00}$	$1.41 \times 10^{+10}$	$4.87 \times 10^{+01}$	$3.15 \times 10^{+09}$
Friedman Average Rank		**1.2251**	2.6936	5.9231	4.2103	2.7205	7.2692	5.0333	6.8949
Rank		**1**	2	6	4	3	8	5	7

References

1. Chakraborty, A.; Kar, A.K. Swarm intelligence: A review of algorithms. In *Modeling and Optimization in Science and Technologies*; Springer: Berlin/Heidelberg, Germany, 2017.
2. Wei, C.-L.; Wang, G.-G. Hybrid Annealing Krill Herd and Quantum-Behaved Particle Swarm Optimization. *Mathematics* **2020**, *8*, 1403. [CrossRef]
3. Blum, C.; Li, X. Swarm Intelligence in Optimization. In *Swarm Intelligence*; Springer: Berlin/Heidelberg, Germany, 2008.
4. Fister, I.; Yang, X.S.; Brest, J.; Fister, D. A Brief Review of Nature-Inspired Algorithms for Optimization. *arXiv* **2013**, arXiv:1307.4186.
5. Brezočnik, L.; Fister, I.; Podgorelec, V. Swarm Intelligence Algorithms for Feature Selection: A Review. *Appl. Sci.* **2018**, *8*, 1521. [CrossRef]
6. Omran, M.G.H. Particle Swarm Optimization Methods for Pattern Recognition and Image Processing. Ph.D. Thesis, University of Pretoria, Pretoria, South Africa, 2004.
7. Martens, D.; Baesens, B.; Fawcett, T. Editorial survey: Swarm intelligence for data mining. *Mach. Learn.* **2011**, *82*, 1–42. [CrossRef]
8. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995.
9. Alejo-Reyes, A.; Cuevas, E.; Rodríguez, A.; Mendoza, A.; Olivares-Benitez, E. An Improved Grey Wolf Optimizer for a Supplier Selection and Order Quantity Allocation Problem. *Mathematics* **2020**, *8*, 1457. [CrossRef]
10. Mirjalili, S. Moth-Flame Optimization Algorithm: A Novel Nature-Inspired Heuristic Paradigm. *Knowl.-Based Syst.* **2015**, *89*. [CrossRef]
11. Dhiman, G.; Kumar, V. Seagull Optimization Algorithm: Theory and Its Applications for Large-Scale Industrial Engineering Problems. *Knowl.-Based Syst.* **2019**, *165*. [CrossRef]
12. Mirjalili, S. SCA: A Sine Cosine Algorithm for Solving Optimization Problems. *Knowl.-Based Syst.* **2016**, *96*. [CrossRef]
13. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*. [CrossRef]
14. Pierezan, J.; Dos Santos Coelho, L. Coyote Optimization Algorithm: A New Metaheuristic for Global Optimization Problems. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation, Rio de Janeiro, Brazil, 8–13 July 2018.
15. Meng, O.K.; Pauline, O.; Kiong, S.C. A Carnivorous Plant Algorithm for Solving Global Optimization Problems. *Appl. Soft Comput.* **2021**, *98*. [CrossRef]
16. Qais, M.H.; Hasanien, H.M.; Alghuwainem, S. Transient Search Optimization: A New Meta-Heuristic Optimization Algorithm. *Appl. Intell.* **2020**, *50*. [CrossRef]
17. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A Nature-Inspired Metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
18. Elaziz, M.A.; Ewees, A.A.; Yousri, D.; Alwerfali, H.S.N.; Awad, Q.A.; Lu, S.; Al-Qaness, M.A.A. An Improved Marine Predators Algorithm with Fuzzy Entropy for Multi-Level Thresholding: Real World Example of COVID-19 CT Image Segmentation. *IEEE Access* **2020**, *8*, 125306–125330. [CrossRef]
19. Abdel-Basset, M.; Mohamed, R.; Elhoseny, M.; Chakrabortty, R.K.; Ryan, M. A Hybrid COVID-19 Detection Model Using an Improved Marine Predators Algorithm and a Ranking-Based Diversity Reduction Strategy. *IEEE Access* **2020**, *8*, 79521–79540. [CrossRef]
20. Naga, J.; Naraharisetti, L.; Devarapalli, R. Environmental Effects Parameter Extraction of Solar Photovoltaic Module by Using a Novel Hybrid Marine Predators—Success History Based Adaptive Differential Evolution Algorithm. *Energy Sources Part A Recovery Util. Environ. Eff.* **2020**, *1*, 1–23. [CrossRef]
21. Ridha, H.M. Parameters Extraction of Single and Double Diodes Photovoltaic Models Using Marine Predators Algorithm and Lambert W Function. *Sol. Energy* **2020**, *209*, 674–693. [CrossRef]
22. Yousri, D.; Babu, T.S.; Beshr, E.; Eteiba, M.B.; Allam, D. A Robust Strategy Based on Marine Predators Algorithm for Large Scale Photovoltaic Array Reconfiguration to Mitigate the Partial Shading Effect on the Performance of PV System. *IEEE Access* **2020**, *4*. [CrossRef]
23. Soliman, M.A.; Hasanien, H.M.; Alkuhayli, A. Marine Predators Algorithm for Parameters Identification of Triple-Diode Photovoltaic Models. *IEEE Access* **2020**, *8*, 155832–155842. [CrossRef]
24. Ebeed, M.; Alhejji, A.; Kamel, S.; Jurado, F. Solving the Optimal Reactive Power Dispatch Using Marine Predators Algorithm Considering the Uncertainties in Load and Wind-Solar Generation Systems. *Energies* **2020**, *13*, 4316. [CrossRef]
25. Sahlol, A.T.; Yousri, D.; Ewees, A.A.; Al-qaness, M.A.A.; Damasevicius, R.; Elaziz, M.A. COVID-19 Image Classification Using Deep Features and Fractional-Order Marine Predators Algorithm. *Sci. Rep.* **2020**, *10*, 1–15. [CrossRef]
26. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
27. Çevik, A.; Kurtoğlu, A.E.; Bilgehan, M.; Gülşan, M.E.; Albegmprli, H.M. Support Vector Machines in Structural Engineering: A Review. *J. Civ. Eng. Manag.* **2015**, *21*, 261–281. [CrossRef]
28. Cambria, E.; Huang, G.-B.; Kasun, L.L.C.; Zhou, H.; Vong, C.M.; Lin, J.; Yin, J.; Cai, Z.; Liu, Q.; Li, K.; et al. Extreme Learning Machines [Trends & Controversies]. *IEEE Intell. Syst.* **2013**, *28*, 30–59. [CrossRef]
29. Deng, W.; Guo, Y.; Liu, J.; Li, Y.; Liu, D.; Zhu, L. A Missing Power Data Filling Method Based on Improved Random Forest Algorithm. *Chin. J. Electr. Eng.* **2019**, *5*, 33–39. [CrossRef]
30. Paul, A.; Mukherjee, D.P.; Das, P.; Gangopadhyay, A.; Chintha, A.R.; Kundu, S. Improved Random Forest for Classification. *IEEE Trans. Image Process.* **2018**, *27*, 4012–4024. [CrossRef]

31. Kalaiselvi, B.; Thangamani, M. An Efficient Pearson Correlation Based Improved Random Forest Classification for Protein Structure Prediction Techniques. *Measurement* **2020**, *162*, 107885. [CrossRef]
32. Peng, S.; Hu, Q.; Chen, Y.; Dang, J. Improved Support Vector Machine Algorithm for Heterogeneous Data. *Pattern Recognit.* **2015**, *48*, 2072–2083. [CrossRef]
33. Dong, H.; Yang, L.; Wang, X. Robust Semi-Supervised Support Vector Machines with Laplace Kernel-Induced Correntropy Loss Functions. *Appl. Intell.* **2021**, *51*, 819–833. [CrossRef]
34. Huang, G.; Song, S.; Gupta, J.N.D.; Wu, C. Semi-Supervised and Unsupervised Extreme Learning Machines. *IEEE Trans. Cybern.* **2014**, *44*. [CrossRef]
35. She, Q.; Zou, J.; Meng, M.; Fan, Y.; Luo, Z. Balanced Graph-Based Regularized Semi-Supervised Extreme Learning Machine for EEG Classification. *Int. J. Mach. Learn. Cybern.* **2020**, *1*. [CrossRef]
36. Zhou, W.; Qiao, S.; Yi, Y.; Han, N.; Chen, Y.; Lei, G. Automatic Optic Disc Detection Using Low-Rank Representation Based Semi-Supervised Extreme Learning Machine. *Int. J. Mach. Learn. Cybern.* **2020**, *11*, 55–69. [CrossRef]
37. She, Q.; Hu, B.; Luo, Z.; Nguyen, T.; Zhang, Y. A Hierarchical Semi-Supervised Extreme Learning Machine Method for EEG Recognition. *Med. Biol. Eng. Comput.* **2019**, *57*, 147–157. [CrossRef] [PubMed]
38. Ma, J.; Yuan, C. Adaptive Safe Semi-Supervised Extreme Machine Learning. *IEEE Access* **2019**, *7*, 76176–76184. [CrossRef]
39. Pei, H.; Wang, K.; Lin, Q.; Zhong, P. Robust Semi-Supervised Extreme Learning Machine. *Knowl.-Based Syst.* **2018**, *159*, 203–220. [CrossRef]
40. Tuan, N.H.; Trong, D.D.; Quan, P.H. A Note on a Cauchy Problem for the Laplace Equation: Regularization and Error Estimates. *Appl. Math. Comput.* **2010**, *217*, 2913–2922. [CrossRef]
41. Tao, D.; Jin, L.; Liu, W.; Li, X. Hessian Regularized Support Vector Machines for Mobile Image Annotation on the Cloud. *IEEE Trans. Multimed.* **2013**, *15*. [CrossRef]
42. Mörters, P.; Peres, Y.; Schramm, O.; Werner, W. *Brownian Motion*; Cambridge University Press: Cambridge, UK, 2010.
43. Reynolds, A.M.; Rhodes, C.J. The Lévy Flight Paradigm: Random Search Patterns and Mechanisms. *Ecology* **2009**, *90*. [CrossRef]
44. Sun, Y.; Gao, Y.; Shi, X. Chaotic Multi-Objective Particle Swarm Optimization Algorithm Incorporating Clone Immunity. *Mathematics* **2019**, *7*, 146. [CrossRef]
45. Yang, X.S.; Deb, S. Cuckoo Search via Lévy Flights. In Proceedings of the 2009 World Congress on Nature and Biologically Inspired Computing, Coimbatore, India, 9–11 December 2009.
46. Li, C.; Luo, G.; Qin, K.; Li, C. An Image Encryption Scheme Based on Chaotic Tent Map. *Nonlinear Dyn.* **2017**, *87*. [CrossRef]
47. Park, T.S.; Lee, J.H.; Choi, B. Optimization for Artificial Neural Network with Adaptive Inertial Weight of Particle Swarm Optimization. In Proceedings of the 2009 8th IEEE International Conference on Cognitive Informatics, Kowloon, Hong Kong, 15–17 June 2009.
48. Li, M.; Chen, H.; Wang, X.; Zhong, N.; Lu, S. An Improved Particle Swarm Optimization Algorithm with Adaptive Inertia Weights. *Int. J. Inf. Technol. Decis. Mak.* **2019**, *18*. [CrossRef]
49. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S. An Improved Grey Wolf Optimizer for Solving Engineering Problems. *Expert Syst. Appl.* **2021**, *166*. [CrossRef]
50. Jamil, M.; Yang, X.S. A Literature Survey of Benchmark Functions for Global Optimisation Problems. *Int. J. Math. Model. Numer. Optim.* **2013**, *4*. [CrossRef]
51. Kommadath, R.; Kotecha, P. Teaching Learning Based Optimization with Focused Learning and Its Performance on CEC2017 Functions. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation, San Sebastian, Spain, 5–8 June 2017.
52. Chen, H.; Yang, C.; Heidari, A.A.; Zhao, X. An Efficient Double Adaptive Random Spare Reinforced Whale Optimization Algorithm. *Expert Syst. Appl.* **2020**, *154*. [CrossRef]
53. Derrac, J.; García, S.; Molina, D.; Herrera, F. A Practical Tutorial on the Use of Nonparametric Statistical Tests as a Methodology for Comparing Evolutionary and Swarm Intelligence Algorithms. *Swarm Evol. Comput.* **2011**, *1*. [CrossRef]
54. He, Z.; Xia, K.; Niu, W.; Aslam, N.; Hou, J. Semisupervised SVM Based on Cuckoo Search Algorithm and Its Application. *Math. Probl. Eng.* **2018**, *2018*. [CrossRef]
55. Bai, J.; Xia, K.; Lin, Y.; Wu, P. Attribute Reduction Based on Consistent Covering Rough Set and Its Application. *Complexity* **2017**, *2017*. [CrossRef]
56. Gómez-Chova, L.; Camps-Valls, G.; Muñoz-Mari, J.; Calpe, J. Semisupervised Image Classification with Laplacian Support Vector Machines. *IEEE Geosci. Remote Sens. Lett.* **2008**, *5*. [CrossRef]