IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 June 2023, accepted 5 July 2023, date of publication 10 July 2023, date of current version 13 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3293523

RESEARCH ARTICLE

An Interval Optimization Algorithm With Embedded Point Evolutionary Strategy and Its Application to Bounded Error Modeling

SHOUPING GUAN AND XINYU LI
College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
Corresponding author: Xinyu Li (xinyuli99@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 62173072.

ABSTRACT Aiming at the problems of low efficiency and difficulty in constructing acceleration devices in traditional interval optimization algorithms (IOAs), this paper constructs a valid acceleration device based on a more concise point evolutionary strategy (ES), and then proposes a novel hybrid IOA (HIOA) with no requirement on the derivative of the objective function. The HIOA first divides the initial search area into N equal parts, randomly selects multiple point individuals in each subinterval to represent their information, and performs the optimization with fewer iterations using ES for all point individuals to make them closer to the optima; then selects reliable subintervals containing more point individuals to split, and deletes unreliable subintervals without any point individuals; finally, provides a reliable upper bound to direct the pruning operation to further improve the search efficiency. Furthermore, the convergence property of the proposed algorithm is analyzed. Extensive numerical experiments on several typical test functions and the application to the bounded error parameter estimation demonstrate the superiority of HIOA by comparing it with the existing conventional algorithms, which confirms the effectiveness and applicability of the suggested algorithm.

INDEX TERMS Interval optimization algorithm (IOA), evolutionary strategy (ES), global optimization, bounded error parameter estimation.

I. INTRODUCTION

The interval optimization algorithm (IOA) is a deterministic global optimization method based on the interval analysis theory, using interval variables instead of point ones for interval calculations and combining branch-and-bound algorithm with Moore-Skelboe algorithm [1], [2], [3], [4]. The IOA can find all the global optima of a problem with a given accuracy in a limited time no matter how large the search space is, even for the multi-peak function optimization problems. The excellent performance of IOA has attracted many scholars, and a lot of research works were carried out in this field [5], [6], [7], [8], [9], [10], [11]. However, the traditional IOA based on branch-and-bound principles faces two dilemmas: the correlation and the curse of dimensionality. The first is related to the properties of interval operations and the nonlinear factors of the objective function, that is, the more obvious the correlation of the variables, the greater the difference between the result of the function calculated by the interval expansion and the actual result, which will reduce the accuracy of the calculation result. The second means that the number of intervals splitting and the calculation amount of the algorithm increase exponentially with the dimension increasing. For example, for high-dimensional optimization issues, the search efficiency depends on the objective function and the construction method of interval extension of the gradient and constraints, making the curse of dimensionality more prominent and causing problems such as low computational efficiency and high computational cost.

An effective way to combat the curse of dimensionality is to construct acceleration devices [12], which aim to ensure efficient branching and correct pruning, and further to accelerate the branching and pruning process of the interval, so as to improve the operational efficiency of IOA. At present,

The associate editor coordinating the review of this manuscript and approving it for publication was Ming Xu.

most acceleration devices delete intervals by using the monotonicity principle and convexity principle based on first or second order interval expansion through Lipschitz constant or natural interval expansion of the objective function and its derivatives. However, most accelerated methods are invalid for black-box problems whose explicit equations are not to be used or when the objective function is non-differentiable, so it is tough to construct acceleration devices.

To settle the existing problems, some researchers have combined the evolutionary algorithm with IOA [13], [14], [15], [16], [17], trying to enhance its search efficiency. The most typical one is the work of Zhang et al. [17], who combined the genetic algorithm (GA) with IOA and developed an interval GA (IGA), in which GA is considered as an acceleration device. The main idea is to apply GA to guide interval deleting and splitting. Moreover, IOA delimits the search range of GA [17]. Compared with the classical interval dichotomy, IGA has higher search efficiency and relieves the curse of dimensionality to a certain extent. Whereas IGA utilizes the single point to express an interval, which leads to limited expressed information; meanwhile, the evolution process of GA is cumbersome, and there is no pruning operation among it, so it is generally necessary to combine the monotonicity principle to improve the search efficiency. In general, the performance of IGA is not ideal, e.g., taking a long time for high-dimensional complex optimization problems. In [18] and [19], an interval particle swarm optimization (IPSO) algorithm and its improved version were proposed, which employed interval computation and PSO. Although the proposed IPSOs improve the search efficiency of high-dimensional problems, they completely abandon the branch-and-bound idea in interval dichotomy, resulting in partially losing the ability of IOA to obtain all global optima.

Aiming at the low efficiency of IGA to deal with the high-dimensional problems while avoiding the loss of global search ability like IPSO, this paper proposes a new point evolutionary strategy (ES) based hybrid IOA (HIOA) to solve the existing problems. The main process is that HIOA first divides the initial search area (expressed as an interval) into N equal divisions and randomly selects multiple points in each subinterval, which can represent the information of the interval more comprehensively and judge the superiority of the subinterval faster; then HIOA constructs an acceleration device based on a more concise point ES, avoiding the derivative requirement of the objective function; next, HIOA selects and splits the interval using the property of point individuals tending to the optimal point, deletes the unnecessary intervals by truncation selection to reduce the number of interval expansions, so that the branching and pruning operations of the interval are more accurate and efficient.

Numerical simulation experiments show that the HIOA proposed in this paper has higher search efficiency than the conventional IOA and the typical IGA. For functions with multiple global optima, HIOA can split and search the area near the optimum with fewer iterations and less running time, so it always converges to the optimum faster. For high-dimensional problems, HIOA effectively alleviates the curse of dimensionality caused by high-dimensional interval variables and dramatically reduces the calculation amount. In addition, the proposed HIOA is applied to the bounded error parameter estimation of the interval system based on the idea of interval variable optimization. Compared with the typical SIVIA (Set Inversion Via Interval Analysis) algorithm [20], it is shown that the estimation result is compelling. Therefore, this provides an effective means for the application of HIOA algorithm to a wider range of practical optimization problems, such as complex process parameter identification [21], neural network parameter optimization [22], complex industrial process set point optimization [23], communication beamforming optimization [24], [25], [26], [27], etc.

The main contributions of this paper can be summarized as follows.

1) The proposition of using multi-point information to express an interval and using the concise ES to optimize point individuals is conducive to judging the performance of the interval variables quickly.
2) As an acceleration device, ES makes IOA not need to use monotonicity test, nor does it require the derivative of the objective function, so it is suitable for optimization problems where the objective function is non-differentiable.
3) The optimization operation of interval variables is carried out by using the property of point individuals, which effectively avoids various problems faced by the operations on interval variables in the conventional IOA to mitigate the curse of dimensionality better.
4) Applying HIOA to bounded error parameter estimation of the interval system is suggested, which provides a new way to deal with the modeling of uncertain systems.

The remainder of this paper is organized as follows. Section II briefly introduces the interval calculation rules and conventional IOA. Based on the brief description of ES, a new HIOA is proposed in Section III, and the convergence of the algorithm is analyzed. In Section IV, numerical simulation experiments on several test functions are implemented and the compared performance analysis is developed. In Section V, the application of the proposed algorithm to bounded error parameter estimation is given. Section VI concludes this paper and provides some suggestions for future research.

II. PRELIMINARIES

According to the interval analysis theory [28], if there is a form such as $A = [\underline{a}, \overline{a}]$, with $\underline{a}, \overline{a} \in R$, then A is called an interval number, $\underline{a}$ and $\overline{a}$ are the lower and upper bounds of A, respectively. When $\underline{a} = \overline{a}$, A degenerates to a point value, which is called a degenerative interval, so the point can be considered as a special case of interval number. In this paper, IR denotes a set of intervals over R, IR^n denotes a set of

interval vectors on R^n, and $IR^{m\times n}$ represents the set of interval matrices over $R^{m\times n}$. Lowercase letters such as a, b for real numbers, uppercase letters such as A, B for interval numbers; bold lowercase letters such as $\mathbf{a}$, $\mathbf{b}$ for real vectors or matrices, bold uppercase letters such as $\mathbf{A}$, $\mathbf{B}$ for interval vectors or matrices.

A. INTERVAL ARITHMETIC

For intervals $A = [\underline{a}, \overline{a}]$, $B = [\underline{b}, \overline{b}]$, some concepts and operation rules commonly used are introduced as follows.

Lower bound: $\underline{a} = \inf(A)$;
Upper bound: $\overline{a} = \sup(A)$;
Midpoint: $m(A) = \frac{1}{2}[\underline{a} + \overline{a}]$;
Width: $w(A) = \overline{a} - \underline{a}$;
Absolute value: $|A| = \max(|\underline{a}|, |\overline{a}|)$;
Interval distance: $d(A, B) = \max(|\underline{a} - \underline{b}|, |\overline{a} - \overline{b}|)$;
Addition: $A + B = [\underline{a} + \underline{b}, \overline{a} + \overline{b}]$;
Subtraction: $A - B = [\underline{a} - \overline{b}, \overline{a} - \underline{b}]$;
Multiplication:

$$A \cdot B = [\min(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}), \max(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b})];$$

Division($0 \notin B$):

$$A/B = [\min(\underline{a}/\underline{b}, \underline{a}/\overline{b}, \overline{a}/\underline{b}, \overline{a}/\overline{b}), \max(\underline{a}/\underline{b}, \underline{a}/\overline{b}, \overline{a}/\underline{b}, \overline{a}/\overline{b})];$$

If A and B satisfy $A \cap B \neq \emptyset$, then the intersection and union of the interval are defined as:

$$\begin{cases} A \cap B = [\max(\underline{a}, \underline{b}), \min(\overline{a}, \overline{b})] \\ A \cup B = [\min(\underline{a}, \underline{b}), \max(\overline{a}, \overline{b})]; \end{cases}$$

Note that the containment relation $A \subseteq B$ holds if and only if $\underline{b} \leq \underline{a}$ and $\overline{a} \leq \overline{b}$.

Let $f : D \subseteq R^n \to R$ be a real-valued function, if there is a mapping $F : I(D) \subseteq IR^n \to IR$, for any $\mathbf{X} = (X_1, \ldots, X_n)^n \in I(D), x_i \in X_i (i = 1, \ldots, n)$, there is $F([x_1, x_1], \ldots, [x_n, x_n]) = f(x_1, \ldots, x_n)$, i.e., $\{f(\mathbf{x}) : \mathbf{x} \in \mathbf{X}\} \subseteq F(\mathbf{X})$ holds for all $\mathbf{X} \in I(D)$, then $F(\mathbf{X})$ is said to be an interval extension of $f(\mathbf{x})$ on $\mathbf{X}$.

Obviously, the interval extension $F(\mathbf{X})$ where $\mathbf{X} \in IR^n$ is an interval-valued function from n-dimensional interval vectors to intervals, and the form is not fixed. For a given $f(\mathbf{x})$, replace the variable $\mathbf{x}$ with the interval $\mathbf{X}$ containing it, standard functions (like sin, exp, etc.) by corresponding interval extension, and real arithmetic operators by corresponding interval operators, then the natural interval extension of $f(\mathbf{x})$ is obtained, denoted as $F(\mathbf{X})$.

B. INTERVAL OPTIMIZATION ALGORITHM

Consider the following optimization problem:

$$global \min_{\mathbf{x}\in\Omega\subset R} f(\mathbf{x}) \tag{1}$$

where the objective function $f : R^n \to R$ is continuously differentiable and the feasible domain Ω is defined by $\Omega : \prod^n [l, u]$ with $l, u \in R$ and $l \leq u$.

Let $\mathbf{I}$ denote the set of all interval solution vectors of the form $\mathbf{I}^i = (I_1^i, \ldots, I_j^i, \ldots, I_D^i) \in IR^D$, where $I_j^i \in IR$ and D is the dimension of the problem, which forms $\mathbf{I} = \{\mathbf{I}^1, \ldots, \mathbf{I}^i, \ldots, \mathbf{I}^N\}$, where N is a positive integer. $\mathbf{I}^i$ represents the i-th interval vector in the set $\mathbf{I}$, and I_j^i represents the j-th interval of the i-th interval vector for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, N$. By replacing the independent variable $\mathbf{x}$ in the objective function (1) with the interval vector $\mathbf{I}^i$, the natural interval extension $F(\mathbf{I}^i)$ of $f(\mathbf{x})$ can be obtained. In what follows, $F(\mathbf{I}^i)$ is referred to as the natural interval extension of $\mathbf{I}^i$ for short. Let $U(\mathbf{I}^i)$ and $L(\mathbf{I}^i)$ denote the upper and lower bounds of $F(\mathbf{I}^i)$, respectively. Denote the natural interval extension of the gradient of $f(\mathbf{x})$ on $\mathbf{I}^i$ by $F'(\mathbf{I}^i)$ and the minimum upper bound of the natural interval extension by $\tau = \min_{i=1,\cdots,N} \{U(\mathbf{I}^i)\}$.

For the optimization problem (1), assuming that f^* is the global minimum, then f^* must satisfy:

$$\min_{i=1,\ldots N} \{L(\mathbf{I}^i)\} \leq f^* \leq \max_{i=1,\ldots N} \{U(\mathbf{I}^i)\} \tag{2}$$

The branch-and-bound idea based IOA can be simplified into several processes: branching, bounding, pruning, splitting and termination, including interval branching rules, pruning rules, and splitting rules. Different interval algorithms can be derived depending on the methods of processing these rules [29].

The processes of branching and splitting mean that when $L(\mathbf{I}^i) \leq \tau$, where τ is the minimum upper bound of natural interval extension, if the width of interval satisfies $w(F(\mathbf{I}^i)) \leq \delta$, where δ is a given precision value, then move the interval $\mathbf{I}^i$ into the solution set $\mathbf{S}$, otherwise, choose and bisect the subinterval with the smallest lower bound of F. Bounding refers to calculating the minimum upper bound of the interval after dichotomy and denoting it by τ', if $\tau' < \tau$, then update τ. Pruning means that when $L(\mathbf{I}^i) > \tau$, the interval $\mathbf{I}^i$ is removed from $\mathbf{I}$. In addition, the monotonic test is also applied to the deletion rule of the multivariate objective function, that is, when $0 \notin F'(\mathbf{I}^i)$, remove $\mathbf{I}^i$ from $\mathbf{I}$.

The steps of IOA are shown in Algorithm 1.

Algorithm 1 The Interval Optimization Algorithm

1: Initialize the working list $\mathbf{I} = \Omega$ and the solution set $\mathbf{S} = \emptyset$.
2: Calculate the minimum upper bound of the natural interval extension $\tau = U(\mathbf{I})$.
3: Let $V = \mathbf{I}^i$, $\mathbf{I} = \mathbf{I} - \mathbf{I}^i$, where $i = \{j | L(\mathbf{I}^j) \leq L(\mathbf{I}^q), 1 \leq j, q \leq N\}$.
4: Divide $\mathbf{V}$ into $\mathbf{V}^1$ and $\mathbf{V}^2$; add $\mathbf{V}^1$ and $\mathbf{V}^2$ into $\mathbf{I}$.
5: Caculate $\tau' = \min\{U(\mathbf{V}^1), U(\mathbf{V}^2)\}$. If $\tau' < \tau$, then update τ.
6: If $L(\mathbf{V}^i) > \tau$ or $0 \notin F'(\mathbf{V}^i)$, then remove $\mathbf{V}^i$ from the list $\mathbf{I}$ for $i = 1, 2$.
7: If $w(F(\mathbf{V}^i)) \leq \delta$, holds for $i = 1, 2$, then let $\mathbf{S} = \mathbf{S} \cup \mathbf{V}^i$, $\mathbf{I} = \mathbf{I} - \mathbf{V}^i$ and go to 9.
8: Go to 3.
9: End.

III. HYBRID INTERVAL OPTIMIZATION ALGORITHM

This section introduces a novel hybrid approach called Hybrid Interval Optimization Algorithm (HIOA), which combines a point ES with IOA. HIOA addresses the low efficiency of IGA in handling high-dimensional problems and avoids the loss of global search ability observed in IPSO. By incorporating ES as an acceleration mechanism, IOA eliminates the need for monotonicity tests and derivative information, making it suitable for optimizing problems with nondifferentiable objective functions. Moreover, the integration of IOA with the embedded point ES leverages multi-point information to represent intervals and employs a concise ES for optimizing individual points. This combination facilitates efficient evaluation of the interval variables' performance.

Evolutionary algorithms usually include genetic algorithms, genetic programming, ESs, and evolution programming [30]. The idea of ES is similar to that of the genetic algorithm, but there are many changes in the mode of evolution, such as selection, crossover, mutation, population control and so on.

The ES developed by German scientists Rechenberg and Schwefel in 1964 [31], is a type of evolutionary algorithm that simulates evolutionary laws of nature to solve optimization problems. The simplest form of ES which is called $(1+1)-$ES was investigated by Schwefel in 1965 [32], but the study found out that the process could get stuck in some cases. Therefore, $(\mu+1)-$ES with recombinant operators was proposed [33]. Subsequently Schwefel introduced two further versions of multimembered ES, i.e., $(\mu+\lambda)-$ ES and $(\mu,\lambda)-$ ES [34], both of which adopt three operators: recombination, mutation, and selection, which significantly improve the performance of ES. Traditional ESs based on real number coding often use a truncation selection method, making the algorithm simple, efficient and easy to implement.

An ES algorithm using $(\mu+\lambda)-$ ES combined with truncation selection operators can be described as:

1) Initialize population: randomly generate μ individuals in the feasible solution space to form the parent generation.
2) Recombination and mutation: the parent generation through recombination and mutation to generate λ offspring individuals.
3) Selection: truncation selection is used to select individuals with low fitness among parent and offspring generation to evolve to the next generation.
4) Termination: If the termination criteria are met, the algorithm ends; otherwise, go to step 2).

Here we choose the $(\mu+\lambda)-$ ES with constant mutation step size σ because its global convergence has been proven. Suppose that each individual consists of n components and is represented by $\mathbf{x}$, then the mutation operation can be described by the following formula:

$$x_i' = x_i + \sigma N(0,1) \tag{3}$$

where x_i and x_i' stand for the i-th component of individuals before and after mutation, respectively. $N(0,1)$ represents a random number that obeys standard normal distribution.

The specific operation of truncation selection in step 3) is as follows [35]:

1) Set the truncation threshold T, where $T \in [0,1]$ is the selected percentage, and the number of individuals in the population M.
2) Sort the fitness values of individuals in ascending order, and take the first $T \times M$ individuals as the new population.

A. NEW INTERVAL ALGORITHM DESCRIPTION

This section introduces the idea of the novel interval optimization algorithm with embedded point evolutionary strategy, that is, HIOA. Divide the initial search area into N equal parts and randomly generate $K(K>1)$ points from each subinterval. These K points can represent the information of the whole interval more comprehensively, then the $K \times N$ individuals are optimized according to the ES of point individuals; after cyclic G generations, the point individuals will tend to gather around the minimum point, and then count the number of point individuals in each interval. If a subinterval contains no point individuals, the interval is deleted, meanwhile, the subinterval containing the most points is marked $\mathbf{I}^*$; the minimum value of the fitness function obtained through ES will provide a reliable upper bound for guiding the pruning operation. Calculate the midpoint value and the lower bound value of natural interval expansion of all retaining subintervals, and mark the subinterval with both minimum values above as $\mathbf{I}^p$, then compare it with the previously marked subinterval $\mathbf{I}^*$, and if they are the same, divide the subinterval into five. Otherwise, divide each of the two subintervals into two. There is no monotonic test in this process, so the proposed HIOA does not require the first derivative of the objective function, and can be applied to more non-differentiable objective functions.

The ES changes the branching, pruning, bounding, and splitting rules of the traditional interval dichotomy based on the branch-and-bound idea so that the HIOA can quickly delete intervals that obviously do not contain the global optimum, select the reliable interval to continue splitting, and provide a reliable upper bound for pruning, hence the efficiency of the proposed algorithm has been greatly improved. The flowchart of HIOA is depicted in Figure 1 and its specific steps are shown in Algorithm 2.

B. PARAMETER SELECTION OF G AND K

It can be seen from Algorithm 2 that the ES cycle is a certain algebraic optimization on point individuals, making point individuals tend to move closer to the minimum, which may be the global minimum or the local minimum, especially when the number of iterations is excessive. If the point individuals gather in the interval where the local optimum lies, the interval containing the global optimum is likely to be removed. Our goal is not to use point ES for optimization, but for pruning, in other words, delete some subintervals that do not contain the optimum, so the number of iterations should not be too large. However, if the setting value is too small,

Algorithm 2 The Hybrid Interval Optimization Algorithm

1: Divide the feasible domain Ω into $N(N > 0)$ equal parts which form the working list $\mathbf{I} = \{\mathbf{I}^1, \cdots, \mathbf{I}^i, \cdots, \mathbf{I}^N\}$, where $\mathbf{I}^i = \{\mathbf{I}_1^i, \cdots, \mathbf{I}_j^i, \cdots, \mathbf{I}_D^i\}$, and D denotes the dimension of the problem; initialize the population set $\mathbf{x} = \emptyset$ and the solution set $\mathbf{S} = \emptyset$.

2: Enter the ES cycle:

2.1 Set the number of iterations of the ES to G; randomly generate K points in each subinterval of the set $\mathbf{I}$ to form a population set $\mathbf{x}$, namely, $\mathbf{x} = \{\mathbf{x}^1, \cdots, \mathbf{x}^i, \cdots, \mathbf{x}^N\} = \{\{\mathbf{x}_1^1, \cdots, \mathbf{x}_k^1, \cdots, \mathbf{x}_K^1\}, \cdots, \{\mathbf{x}_1^N, \cdots, \mathbf{x}_k^N, \cdots, \mathbf{x}_K^N\}\}$, where $\mathbf{x}^i = (\mathbf{x}_1^i, \cdots, \mathbf{x}_k^i, \cdots \mathbf{x}_K^i)$, $\mathbf{x}_k^i = (\mathbf{x}_{1k}^i, \cdots, \mathbf{x}_{jk}^i, \cdots \mathbf{x}_{Dk}^i)$, $\mathbf{x}_{jk}^i \in \mathbf{I}_j^i$, $k = 1, \cdots, K$.

2.2 Generate new individuals through the mutation operation, which form a new population together with the pre-mutation individuals; calculate and store the fitness value $fit(\mathbf{x}_k^i)$ of each point individual; set the truncation threshold T and arrange the fitness values of the point individuals in ascending order; delete the point individuals that cannot be the optimum using the truncation selection operator and retain the excellent point individuals, denoted by $\mathbf{x} = \mathbf{x}_{rest}$.

2.3 If the iteration counter g does not reach G, then let $g := g + 1$ and go to 2.2; otherwise, go to 3.

3: Count the number of point individuals contained in each subinterval of the set $\mathbf{I}$; if the subinterval contains no point individuals of $\mathbf{x}$, delete the subinterval, at the same time, mark the subinterval containing the most point individuals of $\mathbf{x}$ as $\mathbf{I}^*$, go to 4.

4: Calculate the natural interval extension of all subintervals in the list $\mathbf{I}$, mark the subinterval with both the minimum lower bound value of the interval expansion and the function value at the midpoint of the interval as $\mathbf{I}^p$, where $p = \{i | L(\mathbf{I}^i) \leq L(\mathbf{I}^j), fit(m(\mathbf{I}^i)) \leq fit(m(\mathbf{I}^j)), 1 \leq i, j \leq N\}$, and compare it with the subinterval $\mathbf{I}^*$. If $\mathbf{I}^* = \mathbf{I}^p$, then the subinterval is divided into five; otherwise, bisect two subintervals respectively.

5: Calculate the upper bound $U(\mathbf{I}^i)$ and the lower bound $L(\mathbf{I}^i)$ of the natural interval extension of all subintervals in set $\mathbf{I}$; let $\tau = \min_{i=1,\cdots,N}\{U(\mathbf{I}^i), \min_{k=1,\cdots,K} fit(\mathbf{x}_k^i)\}$, if $L(\mathbf{I}^i) > \tau$, then let $\mathbf{I} = \mathbf{I} - \mathbf{I}^i$.

6: If $w(F(\mathbf{I}^i)) \leq \delta$, then let $\mathbf{I} = \mathbf{I} - \mathbf{I}^i$, $\mathbf{S} = \mathbf{S} \cup \mathbf{I}^i$, the algorithm stops and outputs the solution set $\mathbf{S}$; otherwise, go to 2.

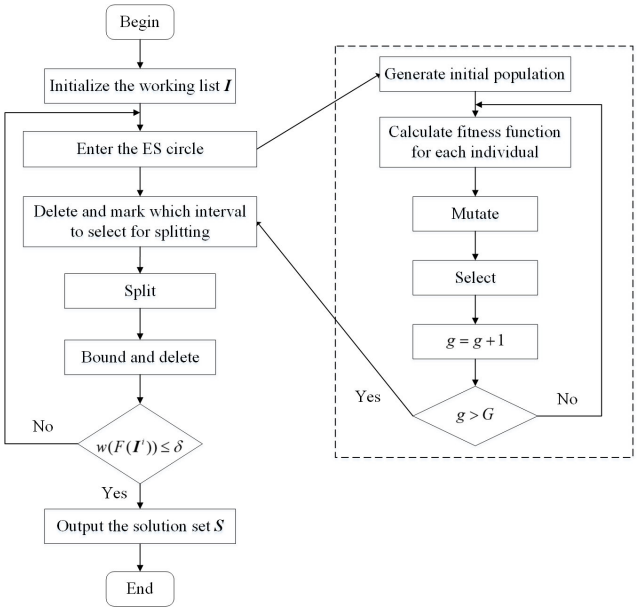

FIGURE 1. Flowchart of HIOA.

the point individuals may not have an apparent trend towards the optimum, and some unreliable subintervals cannot be eliminated. Therefore, the number of iterations G should be set appropriately.

When initializing the population, the ES does not take a single point individual but randomly takes K points in each subinterval, representing the subinterval more comprehensively so that the initial population is diversified, which is beneficial to the interval pruning operation. If the value of K is too large, the subinterval may not need so many individuals to represent its information, and at the same time, the calculation amount will increase, leading to decrease the algorithm's efficiency. On the other hand, if the value of K is too small, it may not be able to fully represent the information of the subinterval, which is not conducive to the optimization of ES. Therefore, the K value should be set reasonably.

C. CONVERGENCE ANALYSIS

The ES plays two roles in Algorithm 2: guiding branching and splitting (Step 4) as well as providing a reliable upper bound to facilitate pruning (Step 5). When the number of sampling points in ES tends to infinity, the convergence theorem of the algorithm is given below and its proof is presented in the Appendix.

Theorem 1: Assume that f is a continuous function in the feasible domain Ω, and F is an interval extension function of f with the property

$$w(F(\mathbf{I}^i)) \to 0 \quad \text{as} \quad w(\mathbf{I}^i) \to 0 \tag{4}$$

for all subintervals $\mathbf{I}^i \subseteq \mathbf{I} \subset \Omega (i = 1, 2, \ldots, N)$. Let $\mathbf{x}^*$ be the global optimum and $f^* = f(\mathbf{x}^*) \leq f(\mathbf{x})$ be the global optimum value. If $\mathbf{x}^* \in \mathbf{I}^i$, where $\mathbf{I}^i \subseteq \mathbf{I} \subset \Omega$, then

1) $f^* \in F(\mathbf{I}^i)$, that is, if an interval contains the global optimum, then its natural interval extension must also contain the global optimum value of the function.
2) when $t \to \infty$, for $\mathbf{x}^* \in \mathbf{I}^*(t) \subset \mathbf{I}(t) = \{\mathbf{I}^1(t), \cdots, \mathbf{I}^i(t), \cdots, \mathbf{I}^N(t)\}$, the sequence $F(\mathbf{I}^*(t))$ converges to the global optimum value f^*, where $\mathbf{I}^*$

denotes the interval containing the global optimum $\mathbf{x}^*$ and t represents the t-th iteration of the algorithm.

In fact, in order to make the algorithm operable and practical, we adopt a limited sampling strategy in HIOA, which may damage the global convergence to a certain extent. However, the convergence rate of HIOA to the optimum is accelerated, which greatly alleviates the curse of dimensionality of IOAs and improves the execution efficiency of the algorithm. Moreover, it can be seen from following experimental results that HIOA is able to provide good approximations to the global optimizer.

IV. EXPERIMENTS AND ANALYSIS

To verify the effectiveness of the proposed algorithm, we choose twelve common test functions described in Table 1 for experiments, where D represents the dimension of the test problem. These functions are difficult to solve, where functions 1-5 have multiple local minima and functions 10-12 have multiple global minima. It is a challenge to find the global minimum among many local minima. For example, function 1 is characterized by a nearly flat outer region, and a large hole at the centre, which makes it easy to be trapped in one of its many local minima for optimization algorithms. In addition, the global minimum of function 6 lies in a narrow, parabolic valley, which makes it difficult to converge to the minimum. Let $D = 2$, then functions 1-12 are used for low-dimensional experiments. Set the dimension D to 5, 10 and 20, and functions 1-7 are used for high-dimensional experiments.

In the following simulations, divide the initial search area into fifteen equal parts, i.e., $N = 15$, randomly select ten points in each subinterval, i.e. $K = 10$, and let the number of iterations of the ES be $G = 10$, set the desired accuracy to $\delta = 1e - 6$, and calculate the truncation threshold based on $T = size(\mathbf{x})/K * 0.5$, where $size(\mathbf{x})$ denotes the total number of individuals in the population $\mathbf{x}$.

All programs are coded in Matlab language. In terms of the simulation environment, all simulation experiments are implemented on a CPU i7-2.8GHz with 64 GB of RAM and 8 cores.

A. 2-DIMENSIONAL (2-D) SIMULATION

Let $D = 2$ in functions 1-7, and perform low-dimensional numerical experiments on test functions 1-12. Three optimization algorithms of IOA, IGA and HIOA are used to compare and analyze. The final experimental results are shown in Table 2, in which Time (unit: s) indicates the average running time of the algorithms after ten independent runs. Figures 2 - 13 draw the simulation curves of twelve test functions under the comparison of three optimization algorithms, in which the x-axis is the number of iterations and the y-axis is the logarithm of the upper bound of the interval expansion of the fitness obtained by each iteration. The upper bound of fitness refers to the maximum value of the fitness interval obtained as the output of the algorithm, representing the upper limit of the fitness range. When an interval algorithm searches for the minimum value, there exists a relationship between the fitness interval values and the true minimum of the function as follows: the upper bound of the fitness interval is greater than or equal to the true minimum of the function, which is greater than or equal to the lower bound of the fitness interval. In the convergence curve, the inflection point where the upper bound of the fitness interval transitions from a rapid descent to a stable state indicates the discovery of the minimum value of the function. The number of iterations required to reach this inflection point reflects the efficiency of the algorithm. Specifically, a smaller number of iterations to reach the inflection point signifies a higher convergence rate and thus a more efficient algorithm.

We can see from Table 2 that for all test functions in the table, HIOA has shorter running time and fewer iterations compared to IGA and IOA. Especially for functions 9-12, the difference between HIOA and the other two algorithms is exceptionally high. Taking function 9 as an example, the iteration number of HIOA is less than 1/23 of IGA and 1/113 of IOA, while the running time is 4.8442 seconds shorter than that of IGA and 48.4605 seconds shorter than that of IOA, respectively. In general, the execution efficiency of HIOA is greatly improved compared to IGA and IOA in the two-dimensional case because the ES for points in HIOA plays a crucial role in guiding the subinterval splitting and deleting more unreliable subintervals. In the process of solution solving, the inner loop based on ES in HIOA makes the point individuals gather quickly near an optimum, thus retaining this subinterval for splitting, which significantly improves the algorithm efficiency.

For functions with multiple local optima, how to set the parameter G in HIOA is critical. If the value of G is too large, the iteration time of ES is too long, causing the current point individuals to gather to the local optimum, making it more difficult for the algorithm to escape from the local optimum and approach the global optimum. However, if the value of G is too small, the point individuals in the ES have not yet emerged a clear tendency to gather, but are still scattered in each subinterval, which cannot effectively guide the subinterval for splitting and pruning. Therefore, the value of G affects the performance of the algorithm. Maybe we can set G as an adaptive parameter, which is one of the future research issues.

B. HIGH-DIMENSIONAL (HIGH-D) SIMULATION

High-dimensional numerical experiments are carried out on test functions 1-7 respectively by changing the value of D, and the results are shown in Table 3.

It can be seen from Table 2 - Table 3 that HIOA has higher execution efficiency compared to IOA, and the difference is more evident as the dimension increases. In addition, HIOA has fewer iterations than IGA to achieve the same solution accuracy, although the difference is not significant. However, the running time of HIOA is shorter than that of IGA, and the gap becomes more pronounced as the value of D increases.

TABLE 1. Test function and its global optimum.

Function Name	Text Function
1. Ackley	$\min f(\mathbf{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + \exp(1), -30 \le x_i \le 30, i = 1, \cdots, D,$ $\mathbf{x}^* = (0, \cdots, 0), f(\mathbf{x}^*) = 0.$
2. Griewank	$\min f(\mathbf{x}) = 1 + \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right), -500 \le x_i \le 700, i = 1, \cdots, D, \mathbf{x}^* = (0, \cdots, 0), f(\mathbf{x}^*) = 0.$
3. Levy	$\min f(\mathbf{x}) = \sin^2(\pi y_1) + (y_D - 1)^2(1 + 10\sin^2(2\pi y_D)) + \sum_{i=1}^{D-1}[(y_i - 1)^2(1 + 10\sin^2(\pi y_i + 1))],$ $y_i = 1 + \frac{x_i - 1}{4}, -10 \le x_i \le 10, i = 1, \cdots, D, \mathbf{x}^* = (1, \cdots, 1), f(\mathbf{x}^*) = 0.$
4. Rastrigin	$\min f(\mathbf{x}) = 10D + \sum_{i=1}^{D}[x_i^2 - 10\cos(2\pi x_i)], -5.12 \le x_i \le 5.12, i = 1, \cdots, D, \mathbf{x}^* = (0, \cdots, 0), f(\mathbf{x}^*) = 0.$
5. Sphere	$\min f(\mathbf{x}) = \sum_{i=1}^{D} x_i^2, -100 \le x_i \le 100, i = 1, \cdots, D, \mathbf{x}^* = (0, \cdots, 0), f(\mathbf{x}^*) = 0.$
6. Rosenbrock	$\min f(\mathbf{x}) = \sum_{i=1}^{D-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2], -5 \le x_i \le 10, i = 1, \cdots, D, \mathbf{x}^* = (1, \cdots, 1), f(\mathbf{x}^*) = 0.$
7. Zakharov	$\min f(\mathbf{x}) = \sum_{i=1}^{D} x_i^2 + (\sum_{i=1}^{D} 0.5ix_i)^2 + (\sum_{i=1}^{D} 0.5ix_i)^4, -5 \le x_i \le 10, i = 1, \cdots, D, \mathbf{x}^* = (0, \cdots, 0), f(\mathbf{x}^*) = 0.$
8. Booth	$\min f(\mathbf{x}) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2, -10 \le x_i \le 10, i = 1, 2, \mathbf{x}^* = (1, 3), f(\mathbf{x}^*) = 0.$
9. Matyas	$\min f(\mathbf{x}) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2, -10 \le x_i \le 10, i = 1, 2, \mathbf{x}^* = (0, 0), f(\mathbf{x}^*) = 0.$
10. Branin	$\min f(\mathbf{x}) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10, -5 \le x_1 \le 10, 0 \le x_2 \le 15,$ $\mathbf{x}^* = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475), f(\mathbf{x}^*) = 0.3979.$
11. Himmelblau	$\min f(\mathbf{x}) = ({x_1}^2 + x_2 - 11)^2 + (x_1 + {x_2}^2 - 7)^2, -10 \le x_i \le 10, i = 1, 2,$ $\mathbf{x}^* = (-2.805118, 3.131312), (-3.779310, -3.283186), (3.584428, -1.848126), (3.0, 2.0), f(\mathbf{x}^*) = 0.$
12. Six-Hump Camel	$\min f(\mathbf{x}) = (4 - 2.1{x_1}^2 + \frac{{x_1}^4}{3}){x_1}^2 + x_1x_2 + (-4 + 4{x_2}^2){x_2}^2, -3 \le x_1 \le 3, -2 \le x_2 \le 2,$ $\mathbf{x}^* = (0.0898, -0.7126), (-0.0898, 0.7126), f(\mathbf{x}^*) = -1.0316.$

TABLE 2. Test function experiment results for 2-*D*.

Function	Time(s)/Iterations ($\delta = 1e-6$)		
	HIOA	IGA	IOA
1	0.1534/11	1.1413/15	0.8438/34
2	0.2420/8	0.5358/10	0.3125/21
3	0.3611/6	1.4844/12	2.0422/30
4	0.1012/8	0.3062/10	0.2813/23
5	0.0520/8	0.0850/10	0.5703/39
6	0.0753/8	0.2813/10	0.2969/19
7	0.1104/6	0.2984/14	0.4844/34
8	0.1430/13	0.9062/51	0.8656/63
9	0.0661/6	4.9103/140	48.5266/680
10	0.2830/13	7.6160/113	3.3656/165
11	0.2230/14	4.5970/83	2.3734/141
12	0.6850/23	18.1266/101	11.8141/525

In other words, as the dimension increases, the disadvantage of the long running time of IGA has gradually appeared. At this time, HIOA can achieve better results. Moreover, we can see from Figures 2 - 13 that for two-dimensional optimization problems in Table 1, HIOA can provide a linear convergence order, that is, it can achieve a good convergence effect.

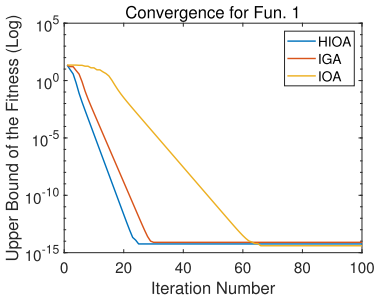

FIGURE 2. Algorithm Convergence for Fun. 1.

C. COMPARISON BASED ON DIFFERENT EMBEDDED OPTIMIZATION STRATEGIES

We have noticed that there are some state-of-the-art optimization strategies, such as LSHADE [36], SaDE [37], CMA-ES [38], etc. We also compare the performance of HIOA based on different embedded optimization strategies and the results are illustrated in Table 4 and Figures 14 - 25.

As shown in Figures 14 - 25, HIOA based on three different embedded optimization strategies can linearly converge to the optimal solution, but we only need ES to guide the splitting direction and provide a reliable upper bound, instead of applying ES to find an accurate global optimum. In addition, we can see from the experimental results that HIOA based on a simple ES can converge to the optimum with fewer

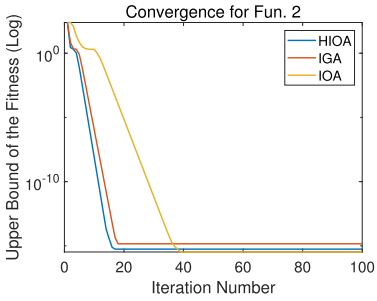

FIGURE 3. Algorithm Convergence for Fun. 2.

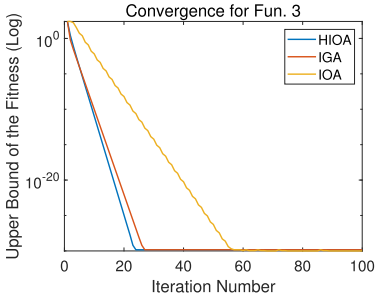

FIGURE 4. Algorithm Convergence for Fun. 3.

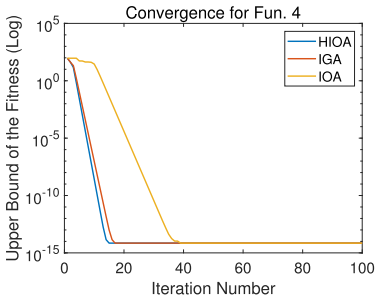

FIGURE 5. Algorithm Convergence for Fun. 4.

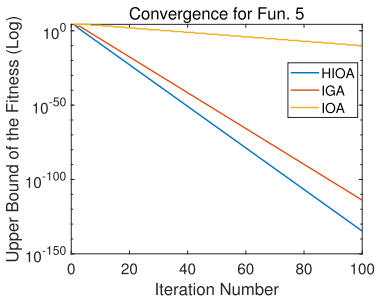

FIGURE 6. Algorithm Convergence for Fun. 5.

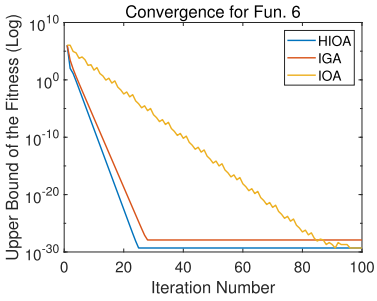

FIGURE 7. Algorithm Convergence for Fun. 6.

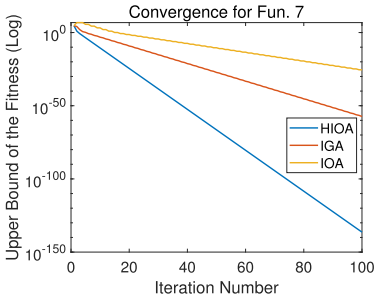

FIGURE 8. Algorithm Convergence for Fun. 7.

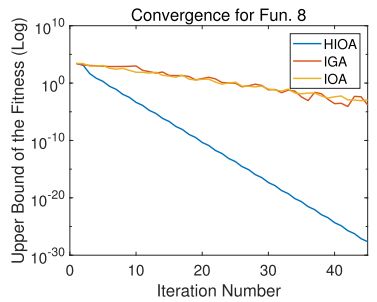

FIGURE 9. Algorithm Convergence for Fun. 8.

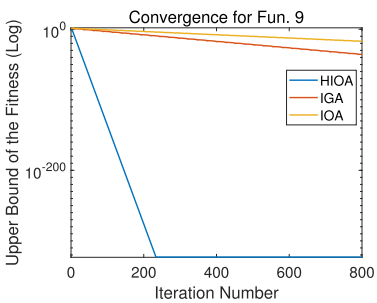

FIGURE 10. Algorithm Convergence for Fun. 9.

iterations and less running time. Therefore, using a simple point ES for HIOA is a better selection in this paper.

D. DISCUSSION

The conventional IOA and its improved one of IGA have some shortcomings. Although the IOA uses the monotonicity principle and the pruning rule to delete some subintervals that certainly do not contain the global optimum, it cannot effectively select a reliable interval for splitting, so the algorithm runs more iterations and consumes longer time. Even though IOA always converges, the rate of convergence depends on whether the interval selected to continue splitting is reliable. If the interval selected at some time is not reliable, it will lead to an overestimation of the interval. The IGA is based on IOA, using GA as an acceleration device, combined with traditional interval dichotomy. The two guide each other and gradually aggregate to the global optimum. Compared with IOA, IGA has high search efficiency, alleviates the curse of dimensionality to a certain extent, and improves the efficiency of the algorithm. Nevertheless, GA only provides a guiding direction for interval splitting and pruning, and it still

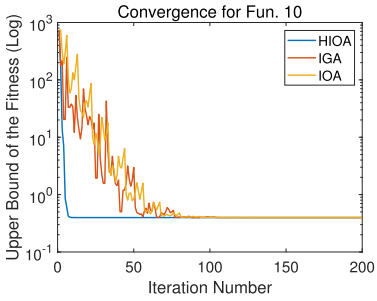

FIGURE 11. Algorithm Convergence for Fun. 10.

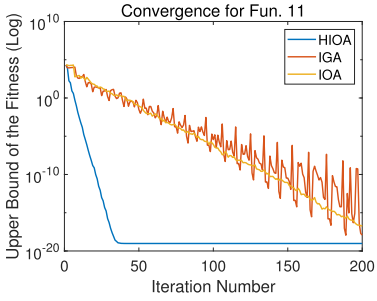

FIGURE 12. Algorithm Convergence for Fun. 11.

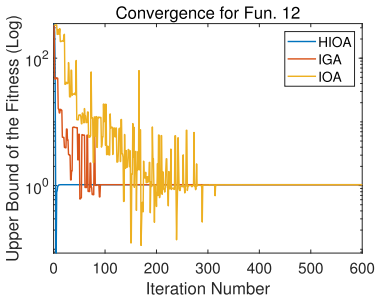

FIGURE 13. Algorithm Convergence for Fun. 12.

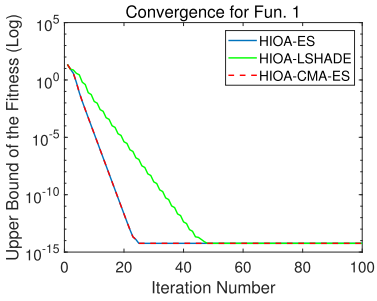

FIGURE 14. Algorithm Convergence for Fun. 1.

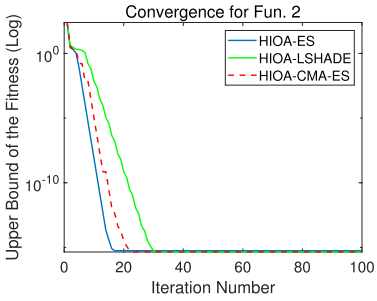

FIGURE 15. Algorithm Convergence for Fun. 2.

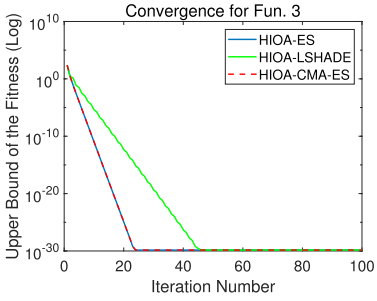

FIGURE 16. Algorithm Convergence for Fun. 3.

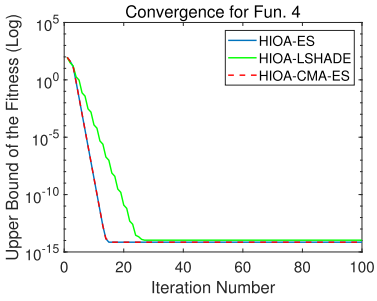

FIGURE 17. Algorithm Convergence for Fun. 4.

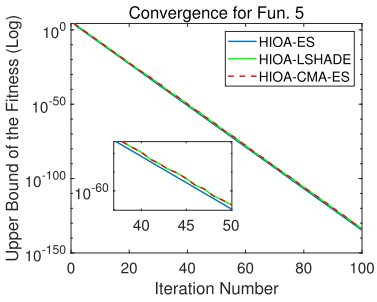

FIGURE 18. Algorithm Convergence for Fun. 5.

needs to use the monotonic test to speed up the deletion of the interval. As the dimensionality of the problem increases, the computational time to solve it is hard to deal with.

In contrast, HIOA provides a reliable upper bound to guide interval splitting and pruning and deletes invalid intervals in the ES process, thus greatly reducing the computational time. Especially when the dimensionality of the problem is high, this advantage of HIOA is more pronounced. By comparing the performance of HIOA based on different embedded optimization strategies, HIOA based on a simple point ES has shorter running time and fewer iterations under the same convergence order. In addition, HIOA does not need to rely on the monotonicity and convexity of the function, which makes the algorithm suitable for more non-differentiable optimization problems.

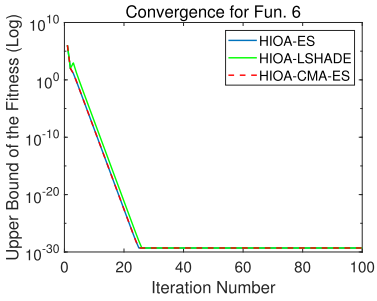

FIGURE 19. Algorithm Convergence for Fun. 6.

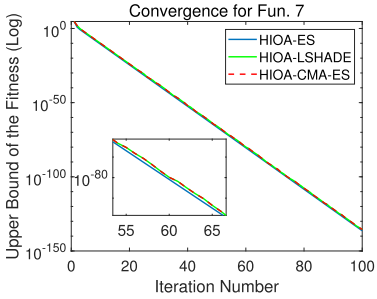

FIGURE 20. Algorithm Convergence for Fun. 7.

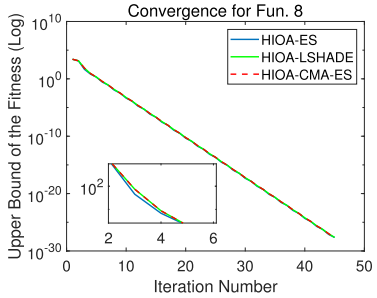

FIGURE 21. Algorithm Convergence for Fun. 8.

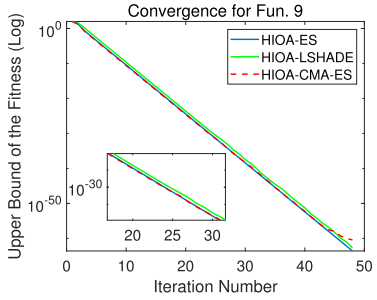

FIGURE 22. Algorithm Convergence for Fun. 9.

V. APPLICATION OF HIOA IN BOUNDED ERROR PARAMETER ESTIMATION

In most practical control systems, there are more or less uncertain factors existed, so it is usually impossible to establish an accurate mathematical model for a control system. In order to describe an actual control system more realistically, the uncertainties are introduced to its mathematical

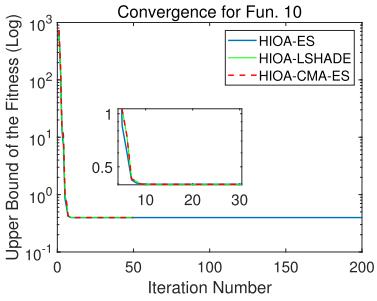

FIGURE 23. Algorithm Convergence for Fun. 10.

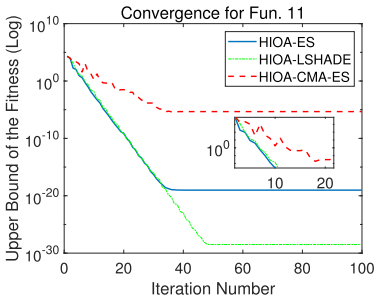

FIGURE 24. Algorithm Convergence for Fun. 11.

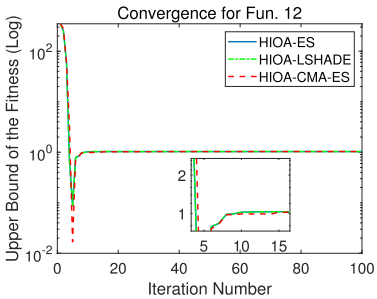

FIGURE 25. Algorithm Convergence for Fun. 12.

model, which can be expressed as interval numbers. A system with interval parameters to describe the uncertainties is called an interval system.

A. PROBLEM OF BOUNDED ERROR PARAMETER ESTIMATION

The problem studied in this paper is to estimate the interval parameters of a nonlinear interval system under the condition of unknown-but-bounded (UBB) errors [39], [40], [41], which can be described as follows.

For the nonlinear system $\mathbf{y}(t) = f(\mathbf{x}, \mathbf{p}) + \mathbf{e}(t)$, where $\mathbf{x} \in R^m$ and $\mathbf{y}(t) \in R^m$ denote the actual system input and output, respectively, $\hat{\mathbf{y}}(\mathbf{p}, t) = f(\mathbf{x}, \mathbf{p})$ denotes the prediction output by the system model, $\mathbf{p} \in R^n$ is the unknown parameters, and $\mathbf{e}(t)$ is the estimated error, then the relations of them can be expressed by

$$\mathbf{y}(t) = \hat{\mathbf{y}}(\mathbf{p}, t) + \mathbf{e}(t) \tag{5}$$

Suppose $\underline{\mathbf{e}}(t)$ and $\bar{\mathbf{e}}(t)$ are the lower and upper bounds of the estimated error, respectively, then $\mathbf{p}$ is said to be feasible if and only if $\mathbf{e}(t) \in E = \{\mathbf{e}(t) | \underline{\mathbf{e}}(t) \leq \mathbf{e}(t) \leq \bar{\mathbf{e}}(t)\}$. The purpose of bounded error parameter estimation is to determine the

TABLE 3. Test function experiment results for high *D*.

Function	*D*	Time(s)/Iterations ($\delta = 1e-6$)		
		HIOA	IGA	IOA
1	5	0.1715/12	3.8306/14	31.4375/141
	10	0.1874/12	12.7355/15	211.0459/281
	20	0.2199/12	50.2107/15	1755.8594/560
2	5	0.4779/8	0.9279/9	9.1875/100
	10	0.7536/8	2.6680/10	55.6672/200
	20	2.9175/9	7.6430/10	395.9271/402
3	5	0.3865/6	1.5687/7	7.9734/74
	10	0.7253/6	3.6279/8	32.1828/150
	20	1.6083/7	7.5483/8	135.6625/310
4	5	0.2839/8	0.7641/10	3.3281/94
	10	0.5739/8	1.5854/10	13.3594/190
	20	0.9937/8	3.2569/10	59.6719/391
5	5	0.0406/8	0.1080/10	2.5047/101
	10	0.0489/8	0.1380/10	7.3922/196
	20	0.0772/9	0.2180/11	28.3578/399
6	5	0.0894/8	0.4066/10	3.8438/127
	10	0.1154/8	0.7069/10	14.7344/252
	20	0.1460/8	1.4284/10	62.2969/502
7	5	0.1879/7	0.7270/8	5.7984/84
	10	0.3499/8	2.3413/9	34.1969/184
	20	0.7018/9	8.6768/10	265.601565/402

TABLE 4. Comparison of HIOA based on different optimization strategies for 2-*D*.

Function	Time(s)/Iterations ($\delta = 1e-6$)		
	HIOA(ES)	HIOA(CMA-ES)	HIOA(LSHADE)
1	0.1534/11	1.8947/11	2.1899/21
2	0.2420/8	0.3799/8	1.2960/15
3	0.3611/6	1.7593/6	1.7827/11
4	0.1012/8	1.1480/8	1.0392/13
5	0.0520/8	1.4480/8	0.1499/8
6	0.0753/8	0.1656/8	0.3820/9
7	0.1104/6	0.9166/7	0.1951/7
8	0.1430/13	0.2880/13	0.3980/13
9	0.0661/6	0.8229/7	0.1492/7
10	0.2830/13	0.6790/26	0.5520/14
11	0.2230/14	0.3860/22	0.5090/15
12	0.6850/23	6.5910/90	1.021/23

feasible parameter set P, i.e.,

$$P = \{\mathbf{p} \in R^n | y(t_i) - \hat{y}(\mathbf{p}, t_i) \in [\underline{e}(t_i), \bar{e}(t_i)], i = 1, \ldots, k\} \tag{6}$$

where $y(t_i)$ and $\hat{y}(\mathbf{p}, t_i)$ represent the corresponding values at the moment t_i (total k time), respectively.

The SIVIA algorithm, proposed by L. Jaulin and E. Walter [20], is a common method to obtain the set P, which is compatible with the UBB errors as (6). However, in this paper, we will adopt another approach to determine the set P. The main idea is as follows: 1) transform the interval parameter estimation problem into an optimization one with a certain interval objective function; 2) use an interval algorithm to optimize this interval objective function to obtain the set P. In the context, a glutamate bacterial growth model is taken as an example, where the proposed HIOA is used to estimate its interval parameters, i.e., the set P. The simulation results demonstrate the advantages of the proposed method compared with the SIVIA.

B. BACTERIAL GROWTH MODEL

The bacteria will grow and reproduce according to the natural laws after they are put in a fermenter. During the entire fermentation process, if there is no invasion of miscellaneous bacteria and bacteriophages, as well as large-scale bacterial migration inside and outside the tank, then the growth and reproduction process of the bacteria can be described by the Verhulst model [42]:

$$\begin{aligned} &\frac{d\hat{y}(t)}{dt} = r\hat{y}(t)(1 - \frac{\hat{y}(t)}{k}), \\ &\hat{y}(0) = \hat{y}_0 \end{aligned} \tag{7}$$

where $\hat{y}(t)$ is the momentary number of the bacterial at time t, $\hat{y}_0$ is the number of bacteria at the initial moment, r is the growth rate, and k is the habitat's carrying capacity, quantified as the number of bacteria that it can support.

When bacteria are added to a new medium, they require time to adapt to the new environment before dividing. Hence, a lag time term denoted by $\bar{\tau}$ is introduced, and the modified Verhulst model is as follows:

$$\begin{aligned} &\frac{d\hat{y}(t)}{dt} = r\hat{y}(t)(1 - \frac{\hat{y}(t)}{k}), \\ &\hat{y}(t) = \hat{y}_0, \quad 0 \le t \le \bar{\tau} \end{aligned} \tag{8}$$

The solution to the differential equation (8) is as (9), which is the growth model of the bacteria in the fermenter.

$$\hat{y}(t) = \frac{k}{1 + e^{a - rt}} \tag{9}$$

where a is a constant with respect to the initial condition $\hat{y}_0$.

For the nonlinear system model (9), k, a, and r are the parameters to be estimated. For a fermentation process, total twenty samples of the actual observed data $y(t)(t = 2, 3, \ldots, 21$, unit:hour) are obtained and shown in Table 5, each one is expressed as $y_i(i = 1, 2, \ldots, n, n = 20)$. Similarly, the corresponding prediction data $\hat{y}(t)$ is expressed as samples $\hat{y}_i(i = 1, 2, \ldots, n, n = 20)$.

Assume that the error between $y(t)$ and $\hat{y}(t)$ is:

$$e(t) = \begin{cases} y(t) - \hat{y}(t) \in [-0.1, 0.1] & t = 2, \ldots, 7 \\ y(t) - \hat{y}(t) \in [-0.05, 0.05] & t = 8, \ldots, 21 \end{cases} \tag{10}$$

TABLE 5. Actual observed data.

t	$y(t)$	t	$y(t)$	t	$y(t)$
2	0.32	9	0.78	16	0.90
3	0.35	10	0.82	17	0.90
4	0.36	11	0.85	18	0.90
5	0.40	12	0.86	19	0.90
6	0.58	13	0.87	20	0.90
7	0.64	14	0.87	21	0.90
8	0.74	15	0.89		

TABLE 6. Parameter estimation results.

Parameter	Estimated Interval	Midpoint of Interval
K	[0.9045, 0.9048]	0.90465
A	[1.7678, 1.7687]	1.76825
R	[0.3913, 0.3916]	0.391453

The priori feasible set of the parameters k, a, and r to be estimated is $[P] = [0, 6] \times [0, 6] \times [0, 5]$.

C. ESTIMATED RESULTS

The least square algorithm is used to define the objective function as the form of sum of squared deviations, as follows:

$$\min j(t) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{11}$$

In the problem of UBB parameter estimation, the actual observed $y(t)$ is not a specific point value, but is an interval composed of the actual value plus a bounded error, namely, $Y(t) = [y(t) - \bar{e}(t), y(t) - \underline{e}(t)]$; meantime, the prediction output $\hat{y}(t)$ is the natural interval extension of function (9), denoted by $\hat{Y}(t)$ with interval parameters K, A, R, where $k \in K$, $a \in A$, $r \in R$. Then the objective function (11) becomes the following (12) according to the rules of interval arithmetic.

$$\min J(t) = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \tag{12}$$

The root mean square error (RMSE) is used to be an evaluation indicator to measure the estimated accuracy, which is defined as:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2} \tag{13}$$

Set the width precision of the interval parameter to 0.001, use the proposed HIOA to solve the UBB parameter estimation problem of the bacterial growth model (9), we can obtain the parameter estimation results shown in Table 6. Then take the estimated interval parameters back to the model (9) to calculate the prediction data $\hat{Y}(t)$, combined with the actual data $Y(t)$, the RMSE can be calculated as [0, 0.3684].

Similarly, the SIVIA algorithm is used to estimate the parameter sets of k, a, and r with the same actual data in Table 5, and the estimation results are shown in Table 7 together with the estimation ones of HIOA. Compared with SIVIA, the width of each parameter interval obtained by HIOA is smaller when both meet the accuracy requirements, and the estimated error (here denoted as $\hat{e}(t) = y(t) - \hat{y}(t)$) is completely contained in the assumed error $e(t)$ shown in (10), thereby ensuring that the estimated parameter is within the bounded error range and is closer to the actual value.

From the perspective of RMSE, it is evident that the RMSE value of HIOA is smaller, demonstrating that the fitting degree is higher when the parameters are brought back to the original model (9), while the RMSE value of the SIVIA is somewhat larger and the fitting degree is slightly worse. The simulation results indicate that HIOA can be used to the interval parameter estimation for a nonlinear system with a satisfactory accuracy.

TABLE 7. Comparison of SIVIA and HIOA.

Algorithm		SIVIA	HIOA
Parameter	K	[0.9140,0.9200]	[0.9045, 0.9048]
	A	[1.7519,1.7579]	[1.7678, 1.7687]
	R	[0.3759,0.3809]	[0.3913, 0.3916]
RMSE		[0, 0.4025]	[0, 0.3684]

VI. CONCLUSION

This paper improves the performance of the traditional IOA by combining with the ES as an acceleration device, to form a novel HIOA. The ES not only helps to delete some invalid intervals, but also provides a reliable upper bound to guide the splitting direction so that the reliable interval containing the global optimum continues to split. In addition, since HIOA does not require the monotonicity of the objective function, it is more suitable for black-box and non-differentiable optimization problems. The solution processes of twelve typical test functions show that the HIOA has higher search efficiency than IOA and IGA; meanwhile, the compared result with SIVIA indicates that the HIOA can be used to the interval parameter estimation for an interval system with a satisfactory performance.

It can be seen from the HIOA that it still has some limitations and needs further research, such as the selection of the number of individuals K in the subinterval, the setting of the number of cycles G in the ES, the selection of the truncation threshold T and so on, which will directly affect the algorithm performance. So how to set these parameters reasonably to deal with different types of functions is the focus of our future work.

APPENDIX

This appendix gives the proof of Theorem 1 in the Section C of Section III.

Proof:

1) Divide the feasible domain Ω into N subintervals which form the set $\mathbf{I}(t) = \{\mathbf{I}^1(t), \cdots, \mathbf{I}^i(t), \cdots, \mathbf{I}^N(t)\}$. Assume the global optimum is $\mathbf{x}^*$, then $\mathbf{x}^*$ must belongs to a certain interval. Without loss of generality, suppose that $\mathbf{x}^* \in \mathbf{I}^i(t)$.
Since the $(\mu + \lambda)-$ ES we choose in this paper converges to the global optimum in probability, the small cycle of the ES in Algorithm 2 will always make the point individuals move closer to the reliable subinterval that may contain the global optimum. In contrast, the subintervals not containing the global optimum will contain very few even no individuals of the population in the ES. Therefore, the subinterval $\mathbf{I}^*$ containing the most points will contain the global optimum $\mathbf{x}^*$. Divide the subinterval $\mathbf{I}^i(t)$ containing the most points into $\mathbf{I}^i_1(t+1)$ and $\mathbf{I}^i_2(t+1)$, then $\mathbf{x}^* \in \mathbf{I}^i_1(t+1)$ or $\mathbf{x}^* \in \mathbf{I}^i_2(t+1)$ holds. It is assumed that $\mathbf{x}^* \in \mathbf{I}^i_1(t+1) \subset \mathbf{I}^i(t)$ with $1 \le i \le N$, and continue to split this interval into two new subintervals $\mathbf{I}^i_{11}(t+2)$ and $\mathbf{I}^i_{12}(t+2)$.
Assume $\mathbf{x}^* \in \mathbf{I}^i_{11}(t+2) \subset \mathbf{I}^i_1(t+1)$ and continue to split this interval, keep doing this to get nested intervals $\{\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)\}(n = 1, 2, \ldots)$. Obviously, according to the nested intervals theorem, there must be:

$$\mathbf{x}^* = \lim_{n\to\infty} \bar{\mathbf{I}}^i_{\underbrace{1\cdots1}_{n}}(t+n) = \lim_{n\to\infty} \underline{\mathbf{I}}^i_{\underbrace{1\cdots1}_{n}}(t+n) \tag{14}$$

where $\bar{\mathbf{I}}^i_{\underbrace{1\cdots1}_{n}}(t+n)$ is the upper bound of $\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)$ and $\underline{\mathbf{I}}^i_{\underbrace{1\cdots1}_{n}}(t+n)$ is the lower bound of $\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)$, i.e.,

$$\lim_{n\to\infty} \mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n) = \mathbf{x}^* \tag{15}$$

Since $f(\mathbf{x})$ is continuous, then

$$\begin{aligned} f(\mathbf{x}^*) &\in f(\lim_{n\to\infty} \mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)) \\ &= \lim_{n\to\infty} f(\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)) \\ &\subset \lim_{n\to\infty} F(\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)) \end{aligned} \tag{16}$$

where $F(\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n))$ is the natural interval extension of $\mathbf{I}^i_{\underbrace{1\cdots1}_{n}}(t+n)$.

2) According to the literature [43], $w(\mathbf{I}^*(t)) \to 0$ as $t \to \infty$. And because of $\mathbf{x}^* \in \mathbf{I}^*(t)$, $\mathbf{I}^*(t) \to \mathbf{x}^*$ holds. By (4), also $w(F(\mathbf{I}^*(t))) \to 0$ as $t \to \infty$. Since $f^* \in F(\mathbf{I}^*(t))$, there is $F(\mathbf{I}^*(t)) \to f^*$.

Complete the proof.

REFERENCES

[1] E. R. Hansen, "Global optimization using interval analysis: The one-dimensional case," *J. Optim. Theory Appl.*, vol. 29, no. 3, pp. 331–344, Nov. 1979, doi: 10.1007/BF00933139.

[2] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*. New York, NY, USA: Springer, 1996, doi: 10.1007/978-1-4757-2495-0.

[3] F. Messine, "Deterministic global optimization using interval constraint propagation techniques," *RAIRO Oper. Res.*, vol. 38, no. 4, pp. 277–293, Dec. 2004, doi: 10.1051/ro:2004026.

[4] H. Ratschek and J. Rokne, "Interval global optimization," in *Encyclopedia of Optimization* C. Floudas, P. Pardalos, Eds. Boston, MA, USA: Springer, 2008, doi: 10.1007/978-0-387-74759-0_306.

[5] T. Wei and F. Li, "An adaptive bivariate decomposition method for interval optimization problems with multiple uncertain parameters," *Eng. Comput.*, vol. 76, no. 3, pp. 416–419, Jan. 2022, doi: 10.1007/s00366-021-01589-z.

[6] E. Carrizosa and F. Messine, "An interval branch and bound method for global robust optimization," *J. Global Optim.*, vol. 80, pp. 507–522, Mar. 2021, doi: 10.1007/s10898-021-01010-5.

[7] L. Wang, G. Yang, Z. Li, and F. Xu, "An efficient nonlinear interval uncertain optimization method using Legendre polynomial chaos expansion," *Appl. Soft Comput.*, vol. 108, Sep. 2021, Art. no. 107454, doi: 10.1016/j.asoc.2021.107454.

[8] B. G.-Tóth, L. G. Casado, E. M. T. Hendrix, and F. Messine, "On new methods to construct lower bounds in simplicial branch and bound based on interval arithmetic," *J. Global Optim.*, vol. 80, no. 4, pp. 779–804, Jul. 2021, doi: 10.1007/s10898-021-01053-8.

[9] F. Ge, K. Li, and Y. Han, "Solving interval many-objective optimization problems by combination of NSGA-III and a local fruit fly optimization algorithm," *Appl. Soft Comput.*, vol. 114, Jan. 2022, Art. no. 108096, doi: 10.1016/j.asoc.2021.108096.

[10] Z. Zhang, M. Zhao, H. Wang, Z. Cui, and W. Zhang, "An efficient interval many-objective evolutionary algorithm for cloud task scheduling problem under uncertainty," *Inf. Sci.*, vol. 583, pp. 56–72, Jan. 2022, doi: 10.1016/j.ins.2021.11.027.

[11] W. Du, W. Song, Y. Tang, Y. Jin, and F. Qian, "Searching for robustness intervals in evolutionary robust optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 1, pp. 58–72, Feb. 2022, doi: 10.1109/TEVC.2021.3092343.

[12] T. Csendes, "Numerical experiences with a new generalized subinterval selection criterion for interval global optimization," *Reliab. Comput.*, vol. 9, no. 2, pp. 109–125, Apr. 2003, doi: 10.1023/A:1023086201037.

[13] G. Shou-ping, H. Yu-huan, P. Xiu-yuan, and L. Chuang, "A new evaluation strategy-based interval optimization algorithm and its simulation analysis," in *Proc. 29th Chin. Control Decis. Conf. (CCDC)*, Chongqing, China, May 2017, pp. 887–890, doi: 10.1109/CCDC.2017.7978645.

[14] D. Gong, J. Sun, and Z. Miao, "A set-based genetic algorithm for interval many-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 47–60, Feb. 2018, doi: 10.1109/TEVC.2016.2634625.

[15] J. Yi, J. Bai, H. He, W. Zhou, and L. Yao, "A multifactorial evolutionary algorithm for multitasking under interval uncertainties," *IEEE Trans. Evol. Comput.*, vol. 24, no. 5, pp. 908–922, Oct. 2020, doi: 10.1109/TEVC.2020.2975381.

[16] J. Sun, Z. Miao, D. Gong, X. Zeng, J. Li, and G. Wang, "Interval multiobjective optimization with memetic algorithms," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3444–3457, Aug. 2020, doi: 10.1109/TCYB.2019.2908485.

[17] X. Zhang and S. Liu, "A new interval-genetic algorithm," in *Proc. 3rd Int. Conf. Natural Comput. (ICNC)*, vol. 4. Washington, DC, USA, Aug. 2007, pp. 193–197, doi: 10.1109/ICNC.2007.95.

[18] S. Guan and S. Fang, "New interval particle swarm optimization algorithm," *J. Northeastern Univ.*, vol. 33, no. 10, p. 1381, Oct. 2012.

[19] S. Guan and Y. Zhang, "An improved interval PSO algorithm with dynamic shrinking," in *Proc. Chin. Control Decis. Conf. (CCDC)*, May 2016, pp. 408–412, doi: 10.1109/CCDC.2016.7531019.

[20] L. Jaulin and E. Walter, "Set inversion via interval analysis for nonlinear bounded-error estimation," *Automatica*, vol. 29, no. 4, pp. 1053–1064, Jul. 1993, doi: 10.1016/0005-1098(93)90106-4.

[21] J. M. Bravo, T. Alamo, and E. F. Camacho, "Bounded error identification of systems with time-varying parameters," *IEEE Trans. Autom. Control*, vol. 51, no. 7, pp. 1144–1150, Jul. 2006, doi: 10.1109/TAC.2006.878750.

[22] S. Guan and X. Yu, "Bounded error modeling using interval neural networks with parameter optimization," *Neurocomputing*, vol. 502, pp. 84–97, Sep. 2022, doi: 10.1016/j.neucom.2022.06.093.

[23] G. Guo and D. Li, "PMP-based set-point optimization and sliding-mode control of vehicular platoons," *IEEE Trans. Computat. Social Syst.*, vol. 5, no. 2, pp. 553–562, Jun. 2018, doi: 10.1109/TCSS.2018.2829626.
[24] Z. Lin, M. Lin, B. Champagne, W. Zhu, and N. Al-Dhahir, "Secrecy-energy efficient hybrid beamforming for satellite-terrestrial integrated networks," *IEEE Trans. Commun.*, vol. 69, no. 9, pp. 6345–6360, Sep. 2021, doi: 10.1109/TCOMM.2021.3088898.
[25] Z. Lin, K. An, H. Niu, Y. Hu, S. Chatzinotas, G. Zheng, and J. Wang, "SLNR-based secure energy efficient beamforming in multibeam satellite systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 59, no. 2, pp. 2085–2088, Apr. 2023, doi: 10.1109/TAES.2022.3190238.
[26] Z. Lin, H. Niu, K. An, Y. Wang, G. Zheng, S. Chatzinotas, and Y. Hu, "Refracting RIS-aided hybrid satellite-terrestrial relay networks: Joint beamforming design and optimization," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 4, pp. 3717–3724, Aug. 2022, doi: 10.1109/TAES.2022.3155711.
[27] Z. Lin, M. Lin, T. de Cola, J. Wang, W. Zhu, and J. Cheng, "Supporting IoT with rate-splitting multiple access in satellite and aerial-integrated networks," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11123–11134, Jul. 2021, doi: 10.1109/JIOT.2021.3051603.
[28] R. E. Moore, *Interval Analysis*. Upper Saddle River, NJ, USA: Prentice-Hall, 1966.
[29] M. C. Markót, J. Fernández, L. G. Casado, and T. Csendes, "New interval methods for constrained global optimization," *Math. Program.*, vol. 106, no. 2, pp. 287–318, Apr. 2006, doi: 10.1007/s10107-005-0607-2.
[30] T. Back, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 3–17, Apr. 1997, doi: 10.1109/4235.585888.
[31] H. G. Beyer and H.-P. Schwefel, "Evolution strategies—A comprehensive introduction," *Nat. Comput.*, vol. 1, no. 1, pp. 3–52, Mar. 2002, doi: 10.1023/A:1015059928466.
[32] H.-P. Schwefel, "Cybernetic evolution as experimental research strategy in fluid mechanics," Diploma thesis, Dept. Eng., Tech. Univ. Berlin, Germany, Jan. 1965.
[33] I. Rechenberg, *Evolution Strategy: Optimization of Technical Systems According to the Principles of Biological Evolution*. Stuttgart, Germany: Frommann-Holzboog Verlag, 1973.
[34] H.-P. Schwefel, "Evolutionary strategy and numerical optimization," Ph.D. dissertation, Dept. Process Eng., Tech. Univ. Berlin, Germany, Jan. 1975.
[35] B. Morsky and C. T. Bauch, "The impact of truncation selection and diffusion on cooperation in spatial games," *J. Theor. Biol.*, vol. 466, pp. 64–83, Apr. 2019, doi: 10.1016/j.jtbi.2019.01.023.
[36] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 1658–1665, doi: 10.1109/CEC.2014.6900380.
[37] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2005, pp. 1785–1791, doi: 10.1109/CEC.2005.1554904.
[38] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, Mar. 2003, doi: 10.1162/106365603321828970.
[39] C.-Y. Gau and M. A. Stadtherr, "Reliable nonlinear parameter estimation using interval analysis: Error-in-variable approach," *Comput. Chem. Eng.*, vol. 24, nos. 2–7, pp. 631–637, Jul. 2000, doi: 10.1016/S0098-1354(00)00363-X.
[40] L. Jaulin and E. Walter, "Nonlinear bounded-error parameter estimation using interval computation," in *Granular Computing*, vol. 70, W. Pedrycz, Ed. Berlin, Germany: Physica-Verlag, 2001, pp. 58–71.
[41] L. Jaulin, "Nonlinear bounded-error state estimation of continuous-time systems," *Automatica*, vol. 38, no. 6, pp. 1079–1082, Jun. 2002, doi: 10.1016/S0005-1098(01)00284-9.
[42] M. Peleg, M. G. Corradini, and M. D. Normand, "The logistic (Verhulst) model for sigmoid microbial growth curves revisited," *Food Res. Int.*, vol. 40, no. 7, pp. 808–818, Aug. 2007, doi: 10.1016/j.foodres.2007.01.012.
[43] H. Ratschek, "Inclusion functions and global optimization," *Math. Program.*, vol. 33, no. 3, pp. 300–317, Dec. 1985, doi: 10.1007/BF01584379.

SHOUPING GUAN received the B.S. and M.S. degrees in automatic control from the Harbin Shipbuilding Engineering Institute, Harbin, China, in 1989 and 1992, respectively, and the Ph.D. degree in control engineering from Northeastern University, Shenyang, China, in 1995. He is currently a Professor with the College of Information Science and Engineering, Northeastern University. His current research interests include intelligent control, precision measurement and control, and modeling and optimization of industrial process.

XINYU LI received the B.S. degree in information and computing sciences from Shanghai University, Shanghai, China, in 2021. She is currently pursuing the master's degree in control science and engineering with Northeastern University, Shenyang, China. Her current research interest includes interval optimization algorithm and its application.