Energy

AIMS Energy, 12(6): 1294–1333.
DOI: 10.3934/energy.2024059
Received: 27 August 2024
Revised: 24 November 2024
Accepted: 12 December 2024
Published: 19 December 2024

https://www.aimspress.com/journal/energy

Research article

A hybrid algorithm based on improved sine cosine algorithm and population incremental learning and its application to economic load dispatch in power systems

Aoshuang Ye[1], Yichao Li[1], Dong Xu[1], Zhiwei Wu[1], Guohua Chen[1], Junjie Tang[2,*] and Zhiyuan Zhu[2,*]

[1] State Grid Shanghai Pudong Electric Power Supply Company, China
[2] College of Electronic Information Engineering, Southwest University, China

*** Correspondence:** Email: qq1998@email.swu.edu.cn, zyuanzhu@swu.edu.cn; Tel: +8623-68250394.

Abstract: The Sine Cosine Algorithm (SCA) excels in local search capabilities for solving real optimization problems. However, its strong local search ability and rotational invariance often lead to convergence at local optima. In this paper, we introduce a hybrid single-objective optimization algorithm, the Improved Sine Cosine Algorithm, and the Population-Based Incremental Learning Algorithm (ISCAPBIL). First, the Improved Sine Cosine Algorithm (ISCA) is developed by incorporating the hyperbolic sinusoidal cosine function, which dynamically interferes with individual positions to enhance optimization accuracy. Additionally, the Levy flight function is embedded within ISCA to improve its exploratory capabilities. The combination of ISCA and PBIL leverages their respective strengths, with ISCA performing local searches and PBIL handling global searches. This integration achieves a dynamic balance between global and local search processes. Our experimental results demonstrated that ISCAPBIL effectively avoided local optima, significantly improving solution accuracy compared to other algorithm variants. Moreover, when applied to the economic load scheduling problem in power systems, ISCAPBIL exhibited superior optimization efficiency and potential for practical application. The Economic Load Dispatch (ELD) problem is a core optimization task in power systems that aims to minimize generation costs while satisfying demand balance and various operational constraints. However, ELD is often formulated as a complex nonlinear optimization problem, influenced by high dimensionality and constraints, making it challenging for traditional methods to achieve efficient solutions. To address these challenges, we proposed a hybrid algorithm combining the improved Sine Cosine Algorithm (SCA) and Population Incremental

Learning (PIL). By leveraging the strengths of both techniques, the proposed algorithm achieved a balance between global exploration and local exploitation. The algorithm was applied to several benchmark ELD problems, and the results demonstrated its superiority in terms of convergence speed and solution quality compared to other methods.

Keywords: meta-heuristic; sine cosine algorithms; population incremental learning; E&P; single-objective optimization; OLD

1. Introduction

Optimization problems are widespread in the world. Traditionally, these problems are linear, low-dimensional, continuous, and unconstrained. However, with the development of artificial intelligence, they have evolved into nonlinear, high-dimensional, discontinuous, and multi-constraint challenges [1,2]. Researchers have proposed various optimization algorithms to address these issues, including Particle Swarm Optimization (PSO) [3], Genetic Algorithm (GA) [4], Differential Evolution (DE) [5], Simulated Annealing (SA) [6], Harris Hawk Optimization (HHO) [7], Grey Wolf Optimization (GWO) [8], Salp Swarm Algorithm (SSA) [9], Whale Optimization Algorithm (WOA) [10], Firefly Algorithm (FA) [11], Bat Algorithm (BA) [12], Flower Pollination Algorithm (FPA) [13], and Gravitational Search Algorithm (GSA) [14,15].

In 2016, Mirjalili et al. introduced the Sine Cosine Algorithm (SCA) [16], which has been favored for its simplicity, few parameters, ease of understanding, and strong local search capability. The SCA has been successfully applied to various real-world problems such as shop floor scheduling [17], feature selection [18], network classification [19], and economic power management [20]. However, the SCA has significant limitations. The leading individual in SCA guides the evolution of the algorithm, and if it falls into a local optimum, the population converges prematurely. Finally, the balance between global and local search capabilities is suboptimal, with the algorithm often focusing too much on local search in the late stages of evolution, leading to local optima.

To address these shortcomings, scholars have proposed various improvements to the SCA. For example, Qsha et al. [21] embedded a Q-learning table in SCA to dynamically control parameters. Hussien et al. [22] introduced a doubly adaptive stochastic standby augmented SCA to balance exploitation and exploration. Long et al. [23] improved the SCA by incorporating inertia weight factors and using a Gaussian function for parameter adjustments. Li et al. [24] proposed an enhanced brainstorming SCA to improve population diversity, while Cheng et al. [25] used a cloud model to adaptively adjust control parameters.

Despite the abundance of excellent optimization algorithms, the No Free Lunch Theorem [26] asserts that no single algorithm can solve all problems perfectly. Therefore, we propose a new hybrid algorithm, ISCAPBIL, for function optimization. The ISCAPBIL introduces a hyperbolic sine cosine function to work alongside the traditional sine cosine function for improved search accuracy. The Levy flight function is also embedded to enhance local exploitation. Initially, ISCA performs a local search, followed by PBIL for global search, maintaining the algorithm's rapid descent rate through multiple local searches. To avoid local optima, PBIL is executed after each local search phase. By alternating ISCA and PBIL at fixed intervals, the algorithm effectively balances global and local search efforts.

Simulation experiments using 23 benchmark test functions, CEC2013 standard test functions, and 6 high-dimensional test functions demonstrate that ISCAPBIL outperforms the basic SCA and other optimization algorithms. The results show significant improvements in solution accuracy and the ability to avoid local optima.

The Economic Load Dispatch [27] (ELD) problem plays a pivotal role in power system operations. It aims to determine the optimal power generation schedule to minimize the total fuel cost of generation while meeting the system's load demand and respecting operational constraints such as generation limits, transmission limits, and emissions. The ELD problem has been extensively studied due to its significant impact on the efficiency and cost-effectiveness of power systems.

Traditionally, methods such as linear programming, dynamic programming, and gradient-based techniques have been applied to solve the ELD problem. However, these methods often struggle with non-linearity, multi-modal objective functions, and the presence of constraints that make the problem computationally challenging, especially when dealing with large-scale systems or complex constraints.

In recent years, evolutionary algorithms (EAs) such as genetic algorithms (GA), particle swarm optimization (PSO), and differential evolution (DE) have shown promise in solving the ELD problem due to their ability to escape local minima and efficiently explore the solution space. Despite their advantages, these algorithms often face issues such as slow convergence and premature convergence, especially in highly complex problems.

To address these challenges, we propose a novel hybrid algorithm that combines the improved Sine Cosine Algorithm (SCA) with Population-Based Incremental Learning (PBIL). By integrating these two methods, the proposed approach effectively balances global exploration and local exploitation, achieving improved performance in solving the ELD problem. The remainder of the paper includes the methodology, experimental setup, results, and conclusion, showing the advantages of the proposed approach over other techniques.

The major contributions of this paper include:

- We introduce a novel hybrid algorithm that combines the improved Sine Cosine Algorithm (SCA) with Population-Based Incremental Learning (PBIL). This hybrid approach is designed to address the limitations of existing methods in solving the Economic Load Dispatch (ELD) problem. The integration of SCA and PIL offers enhanced exploration capabilities and more effective convergence, leading to superior performance in terms of both solution quality and computational efficiency.
- The proposed hybrid algorithm incorporates improvements to the SCA through a dynamic parameter adjustment mechanism, which enables better local search abilities and faster convergence compared to the traditional SCA. PIL further improves the population diversity during the search process, mitigating premature convergence and ensuring a better global search strategy.
- The performance of the hybrid algorithm is rigorously evaluated using multiple benchmark power system case studies. These include well-known standard ELD test systems with varying complexities. The results demonstrate that the proposed algorithm outperforms existing algorithms in terms of optimality, computational efficiency, and robustness under different operating conditions.
- The proposed hybrid algorithm is not only theoretical but also demonstrates strong applicability to real-world power system dispatch scenarios. The flexibility of the method in handling practical constraints such as transmission limits and emission constraints make it a

valuable tool for power system operators aiming to optimize economic performance in real-time operations.

- This work contributes to the growing field of hybrid metaheuristic algorithms for power system optimization, presenting a promising approach that combines the strengths of SCA and PIL. The results of this study provide important insights for future research and development in the optimization of complex power systems.

The rest of the paper is organized as follows: In Section 2, we describe the fundamentals of SCA. In Section 3, we detail the proposed ISCAPBIL. In Section 4, we present experiments and comparisons. In Section 5, we discuss the application of ISCAPBIL to the economic load dispatch problem in power systems. In Section 6, we conclude the paper.

2. Sine cosine algorithm

2.1. Principles of the sine cosine algorithm

The Sine Cosine Algorithm (SCA) is inspired by the periodic oscillatory nature of sine and cosine functions to locate the optimal solution within the algorithm's search space. The optimization process of SCA is divided into global search and local exploitation, which are balanced according to control parameters until the global optimal solution is achieved. The basic principle is illustrated in Figure 1 [16].

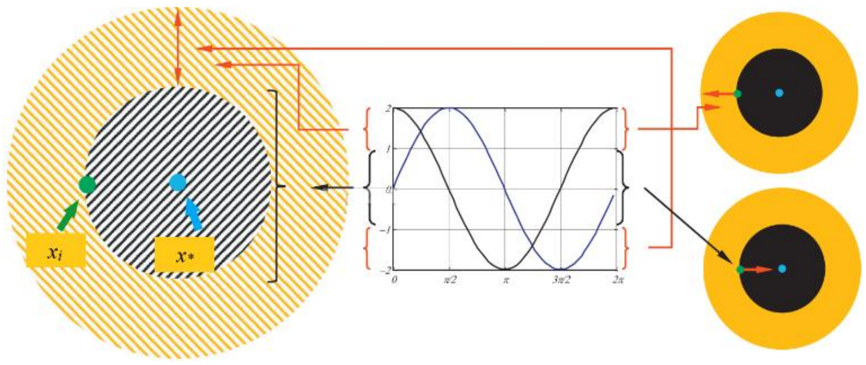


Figure 1. Principle diagram of SCA.

Assuming that the population size is *N* and the search space is D-dimensional, the position of the *ith* individual is denoted as $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$, $i \in 1,2,\ldots,N$ *the* position of the optimal individual in the iterative updating of the population is denoted as $p_{\text{best}} = (p_{\text{best},1}, p_{\text{best},2}, \ldots, p_{\text{best},D})$. The *ith* individual in the population updates the spatial position according to Eq (1).

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} + r_1 \times \sin(r_2) \times \left| r_3 \times p_{best,j}^{t} - x_{i,j}^{t} \right|, & r_4 < 0.5 \\ x_{i,j}^{t} + r_1 \times \cos(r_2) \times \left| r_3 \times p_{best,j}^{t} - x_{i,j}^{t} \right|, & r_4 \geq 0.5 \end{cases} \quad (1)$$

where $x_{i,j}$ is the *jth* dimensional component of the *ith* candidate solution, $p_{\text{best},j}$ is the *jth* dimensional component of the globally optimal solution in the iteration, $j \in (1,2,\ldots,D)$, where $r_2 \in (0,2\pi)$, $r_3 \in (0,2)$, and $r_4 \in (0,1)$ are three uniformly distributed random numbers.

In SCA, the r_1 parameter effectively balances the global search and local exploitation performance, and gradually converges to the global optimal solution as the number of iterations decreases linearly, and the r_1 parameter is updated according to Eq (2).

$$r_1 = a - t \times \frac{a}{T} \tag{2}$$

where *a* is a constant, generally taking the value of 2, *t* is the current number of iterations, and *T* is the maximum number of iterations.

2.2. Convergence analysis of the sine cosine algorithm

The Sine Cosine Algorithm (SCA) belongs to the class of stochastic search algorithms. Consequently, its convergence is analyzed based on the convergence rules applicable to stochastic algorithms. First, we provide the relevant definitions and theorems of the Markov chain model used to describe the behavior of SCA.

Assumption 1 This assumption posits that the population state transitions are Markovian. Specifically, it assumes that each new population state depends only on the current population state, not on any preceding states. This property enables us to represent the population states as a sequence governed by a Markov chain, which is critical for proving convergence in a stochastic framework. In practical terms, this means that each iteration's solution quality depends solely on the positions and parameters from the immediate previous iteration, without needing historical information from past iterations.

Assumption 2 This assumption asserts that the probability of transitioning between any two population states is positive and bounded within the algorithm's feasible solution space. This assumption guarantees that every feasible solution can eventually be explored within a finite number of iterations, ensuring that the algorithm has access to all potential optima in the search space. Consequently, this enhances the robustness of the algorithm, as it prevents the algorithm from getting trapped in local minima without the possibility of moving to other solutions.

Definition 1 Let each individual state, representing a distinct position, be denoted as λ. The complete space of all possible individual states is given by λ = {λ|λ∈Y}, where Y is the space of feasible solutions. The overall population state, which includes all individual states, is represented by *φ* = (λ1, λ2, … , *λNi*). Here, λ*i* represents the i-th individual state, and *Nϕ* is the total population size. Consequently, the space of all possible population states can be defined as:

$$\psi = \{\varphi = (\lambda_1, \lambda_2, ..., \lambda_{Ni}) \mid \lambda_i \in Y\} \tag{3}$$

Definition 2 For *φ* ∈ *ψ and* λ ∈ *φ*, the number of population states containing individual states is denoted.

$$\rho(\varphi, \lambda) = \sum_{i+1}^{N_\varphi} \chi \mid \lambda \mid (\lambda_i) \tag{4}$$

where $x|\lambda|$ denotes the schematic function of the event λ. There are two populations φ1, φ2∈*ψ* for ∀λ∈Y. If *ρ* (*φ1,* λ) = *ρ (φ2,* λ*)*, then φ1 and φ2 are said to be equivalent and are denoted φ1~φ2. equivalent and are denoted φ1~φ2. The equivalence class of population states induced by the equivalence relation "~" is denoted L = ψ/~, or the population The equivalence class of population states induced by the equivalence relation “∼” is denoted L = ψ/∼, or the population equivalence class for short, and has the following properties.

1) Any population state within an equivalence class L is equivalent to each other, i.e., *φi∼φj, ∀φi, φj ∈L; ∀φi, φj ∈L.*

2) Any population state within L is not equivalent to any population state outside L, i.e., *φi/∼φj*, *∀φi* ∈L, *φj* /∈L.

3) Any two different equivalence classes have no intersection, i.e., L1 ∩ L2 = ∅, ∀L1/ = L2.

Definition 3 For *∀λi ∈φ, λj∈φ*, a one-step transfer of an individual state from *λi* to *λj* in an iteration of the algorithm is denoted as Tφ (*λi*) = *λj*.

Theorem 1 In the SCA algorithm, the transfer probability P (Tφ (*λi*) = *λj*) of a one-step transfer of an individual state from *λi* to *λj* is given by

$$p(T\varphi(\lambda i)=\lambda j)=\begin{cases} pa(T\varphi(\lambda i)=\lambda j), decided\ in\text{"sin"} \\ pb(T\varphi(\lambda i)=\lambda j), decided\ in\text{"cos"} \\ pc(T\varphi(\lambda i)=\lambda j), decided\ in\text{"}x_i\text{"} \end{cases} \tag{5}$$

Proof Considering the population as a set of point sets in a hyperspace, the update process of individual merit search is an exchange between point sets in Based on Definition 2 and the geometric properties of the SCA algorithm, the probability of transferring a sin state from *λi* to *λj* in one step is obtained.

$$pa(T\varphi(\lambda i)=\lambda j)= \begin{cases} \frac{1}{|-a_i \cdot s_a|}, \lambda_j \in [\lambda_{ai}, \lambda_{ai} - a_i \cdot s_a], \mu < 0.5; \\ 0, otherwise \end{cases} \tag{6}$$

Cos the probability of transferring a state from *λi* to *λj* in one step

$$pb(T\varphi(\lambda i)=\lambda j)= \begin{cases} \frac{1}{|-a_i \cdot s_b|}, \lambda_j \in [\lambda_{bi}, \lambda_{bi} - a_i \cdot s_b], \mu < 0.5; \\ 0, otherwise \end{cases} \tag{7}$$

xi Transfer probability of a state moving from *λi* to *λj* in one step

$$pc(T\varphi(\lambda i)=\lambda j)= \begin{cases} \frac{1}{|-a_i \cdot s_c|}, \lambda_j \in [\lambda_{ci}, \lambda_{ci} - a_i \cdot s_c], \mu < 0.5; \\ 0, otherwise \end{cases} \tag{8}$$

Since the SCA algorithm is based on a function that works with individuals in an iterative search, Eqs (6–8) determine the probability of transferring an individual state from *λi* to *λj in* one step.

Definition 4 For *∀φi∈ψ, φj∈ψ*, the population state is transferred from *φi* to *φj* during the SCA iteration, denoted as *Tψ (φi) = φj*. The transfer probability of the population state being transferred from *φi* to *φj* in one step is

$$p(T\psi(\varphi i)=\varphi j)=\prod N\varphi K=1 p(T\varphi(\lambda iK)=\lambda jK) \tag{9}$$

That is, the probability that all individual states within a population are simultaneously transferred from φi to φj in one step constitutes a one-step transfer of the population state from *φi* to *φj*.

Theorem 2 The sequence of individual states $\{\varphi(t)$: t > 0} in SCA is a finite chi-squared Markov chain.

Proof The theorem is proved mainly in terms of Markov chainability, finiteness, and chi-squaredness. This property is evidenced by Theorem 1, which states that the transfer probability P ($T\varphi(\varphi(t-1)) = \varphi(t)$) of an individual state sequence is only related to the state at time t-1, but not at time t-1. Since the population has a finite number of individuals, its state space is also finite.

The above research has established a Markov chain model for SCA, which is an important basis for studying the convergence of the SCA algorithm. Referring to the convergence proof process of the grey wolf algorithm based on the Markov chain model [28], the SCA algorithm meets Assumptions 1 and 2 of the convergence criterion and satisfies the sufficient conditions for global convergence, so that the SCA is a globally convergent algorithm.

2.3. Algorithm pseudo-code

In summary, the pseudo-code of the SCA algorithm is shown below:

Algorithm 1: Sine cosine algorithm (SCA)

```
Enter parameters and initialize.
  Set the population size $N$, ($x_i$, $i$ = 1, 2, ..., $N$), and the maximum number of iterations $T$, and spatial
  dimension $D$ (where the function $f_{14}$~$f_{23}$ is a fixed dimension).
    Calculate the individual fitness value $f(x_i)$, $i$ = 1, 2, ..., $N$) and find the globally optimal
individual and its location.
    $t = 0$;
    While ($t$ < T) do
          For $i = 1$ to $N$ do
             Calculate the value of the control parameter $r_1$ using Eq (2)
               Using Eq (1)
          End for
       Updating the current optimal individual and position.
       $t = t + 1$;
    End while
```

3. Proposed algorithm and economic load dispatch model for power system

3.1. Improved sine cosine algorithm (ISCA)

Trigonometric functions are a crucial branch of mathematical functions, with the cosh and sinh functions being the fundamental hyperbolic functions in trigonometry. In this paper, we introduce cosh and sinh functions into the algorithm's structure to dynamically influence individuals while preserving the algorithm's simplicity and fundamental structure. Additionally, to further enhance the performance of the Sine Cosine Algorithm (SCA), we incorporate the Levy flight strategy into the evolutionary process.

The Levy flight function is specifically used to enhance the global search capabilities of the algorithm by introducing a mechanism for larger, more random jumps within the search space. This characteristic of Levy flights enables the algorithm to effectively explore distant, unexplored regions of the solution space, rather than focusing solely on local refinements.

The Levy distribution's "heavy-tailed" nature generates step sizes that vary widely, with occasional large jumps. These jumps help the algorithm escape from local optima by periodically initiating broad searches beyond the neighborhood of current solutions. Thus, Levy flights support exploration on a global scale, which is essential for problems with complex landscapes that contain multiple local optima, thereby enhancing the global search capability of the Improved Sine Cosine Algorithm (ISCA). The Levy function is embedded into the position update formula, as shown in Eq (3).

$$x_{i,j}^{t+1}=\begin{cases} Levy(\lambda)\times x_{i,j}^{t}+\sinh(x_{i,j}^{t})\times r_1\times\sin(r_2)\times\left|r_3\times p_{best,j}^{t}-x_{i,j}^{t}\right|, & r_4<0.5 \\ Levy(\lambda)\times x_{i,j}^{t}+\cosh(x_{i,j}^{t})\times r_1\times\cos(r_2)\times\left|r_3\times p_{best,j}^{t}-x_{i,j}^{t}\right|, & r_4\geq 0.5 \end{cases} \tag{10}$$

In Eq (10), $|r_3 \times p_{\text{best}} - x_{i,j}|$ denotes the distance between the current individual x and the globally optimal individual p_{best}. The parameter r_1 controls the range of the algorithm update, while r_2, r_4, and r_3 are uniformly distributed random numbers within the range [0, 1]. The functions cosh(x) and sinh(x) provide dynamic interference to individual X, with values ranging from [−1, 1]. The hyperbolic sine and cosine functions (sinh and cosh) are chosen due to their unique mathematical properties that provide more flexibility in exploring the search space, especially in higher-dimensional or more complex optimization problems. Unlike standard sine and cosine functions, which oscillate between −1 and 1, the hyperbolic sine and cosine functions grow exponentially, offering a better capability to explore larger regions of the solution space. This is particularly useful in our context where a balance between local exploration and global search is crucial for optimization performance. This increases the algorithm's spatial search range, helping to maintain population diversity in later iterations and facilitating the search for the global optimal solution. The sine and cosine functions work together to ensure the current individual converges to a global optimal solution, thereby improving convergence accuracy. Levy flight, a random wandering mode, can be calculated according to Eq (11):

$$Levy(\lambda)=\frac{\varphi\times\mu}{|\upsilon|^{\frac{1}{\lambda}}},\varphi=\left\{\frac{\Gamma(1+\lambda)\times\sin(\pi\times\frac{\lambda}{2})}{\Gamma\left\{(\frac{1+\lambda}{2})\times\lambda\times 2\frac{\lambda-1}{2}\right\}}\right\}^{\frac{1}{\lambda}},\ \lambda=1.5 \tag{11}$$

where μ and υ are normal distributions, respectively, Γ (·) is the standard gamma function, and the variance of Levy flights (σ^2) is exponential with the evolutionary process (t). Therefore, the SCA carrying Levy flights has a significant improvement in searching ability in unknown space.

3.2. Combined with PBIL algorithm

The Population-Based Incremental Learning Algorithm (PBIL) [29] solves optimization problems by constructing probabilistic models. Unlike traditional evolutionary algorithms, which use selection and recombination operations that can disrupt the internal structure of individuals and affect the algorithm's accuracy, PBIL represents the population as a probability vector (PV) and views the evolutionary process as a learning accumulation process. In PBIL, better individuals guide others in a learning process, avoiding the decomposition and reorganization of individuals and, thus, maintaining the algorithm's accuracy in finding the optimal solution.

PBIL generates a real-valued probability vector p = (p_1, ...p_m .... p_D), which is used to create potential solutions during the sampling process. Here, p_m denotes the probability of obtaining a value of 1 in the m-th component. The standard algorithm operates as follows: at each step t, λ, a set of individuals $x_1^{(t)}, x_2^{(t)}, \ldots, x_\lambda^{(t)}$ is generated based on the probability vector $p^{(t)}$, and the optimal individual μ is selected from this set $(\mu \le \lambda), x_{1:\lambda}^{(t)}, x_{2:\lambda}^{(t)}, \ldots, x_{\mu:\lambda}^{(t)}$. These selected individuals are then used to update the probability vector. The pseudo-code for PBIL is shown in Algorithm 2.

The Improved Sine Cosine Algorithm (ISCA) offers fast descent and high convergence accuracy but tends to fall into local optima. To enhance the algorithm's global search performance and ability to find the optimal solution, this paper proposes alternating the execution of PBIL according to probability for global search.

$$p^{(t+1)} = (1-\alpha)p^{(t)} + \alpha \frac{1}{\mu}\sum_{k=1}^{\mu} x_{k:\lambda}^{(t)}, \tag{12}$$

where p is the probability vector in step t and $\alpha \in (0,1)$ is the learning rate. The distance to push the probability vector depends on the learning rate of each iteration. After updating the probability vectors, a new set of solutions is generated by sampling from the updated probability vectors and the recursion continues.

3.2.1. Pseudo-code for the PBIL algorithm

In summary, the pseudo-code of the PBIL algorithm is shown below:

Algorithm 2: Population based incremental learning (PBIL)

```
Generate a set $x^{(0)}$ of $\lambda$ samples by $p^{(0)}$
Repeat
Using p(t) obtain a set $x^{(t)} = (x^{(t)}_1, x^{(t)}_2, ..., x)^{(t)}_\lambda$
Evaluate the fitness function with respect to $g$
Select the $\mu \le \lambda$ best individuals $x^{(t)}_{1:\lambda}, x^{(t)}_{2:\lambda}, ..., x^{(t)}_{\mu:\lambda}$
Update p(t) by Eq (12)
Until termination condition holds
```

3.3. ISCAPBIL algorithm steps and its flowchart

In summary, the ISCAPBIL steps are represented as follows:

Step 1: Set the parameters and initialize the population, set the population size N, spatial dimension D, maximum number of iterations T, and incremental learning strategy iteration interval P;

Step 2: Calculate individual fitness value, record the global optimal individual;

Step 3: Calculate the value of control parameter $r1$ according to Eq (2);

Step 4: When *t mod* P == 0, perform step 5, otherwise move to step 6;

Step 5: Update each individual according to Eq (12);

Step 6: Update each individual according to Eq (10);

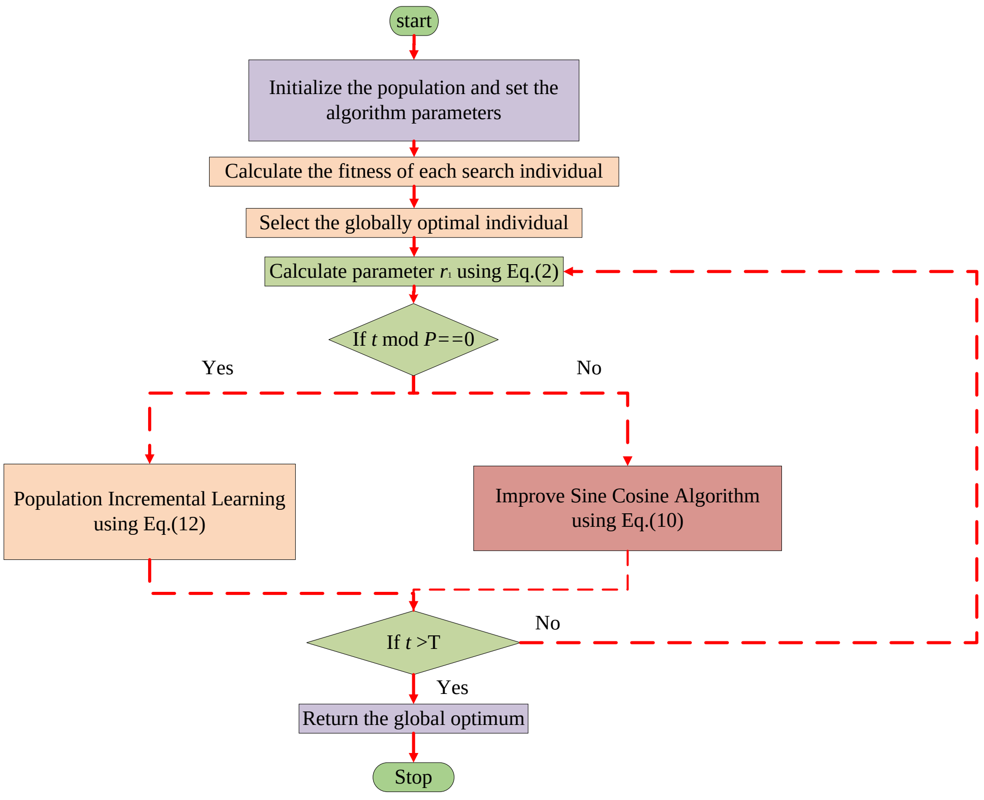
Step 7: Update the global optimal individual;

Step 8: Determine whether the maximum number of iterations T has been reached, if the condition is satisfied then stop the iteration and output the global optimal solution, otherwise go to step 3.

In summary, the pseudo-code of the ISCAPBIL algorithm is shown in Figure 2.

Algorithm 3: A hybrid algorithm based on improve sine cosine algorithm and population incremental learning (ISCAPBIL)

```
Enter parameters and initialize.
  Set the population size $N$, ($x_i$ , $i$ = 1, 2, ..., $N$), and the maximum number of iterations $T$ and spatial
  dimension $D$ (where the function $f_{14\sim} f_{23}$ is a fixed dimension).
    Calculate the individual fitness value $f(x_i)$, $i$ = 1, 2, ..., $N$) and find the globally optimal
individual and its location.
    $t = 0$;
    While ($t < T$) do
        For $i = 1$ to $N$ do
           Calculate the value of the control parameter $r_1$ using Eq (2)
              If ($t$ mod P == 0)
                    PBIL performs global search Using Eq (12)
              Else if
                    ISCA performs local search Using Eq (10)
              End if
        End for
           Updating the current optimal individual and position.
        $t = t + 1$;
    End while
```



start
Initialize the population and set the algorithm parameters
Calculate the fitness of each search individual
Select the globally optimal individual
Calculate parameter r_1 using Eq.(2)
If t mod P==0
Yes
No
Population Incremental Learning using Eq.(12)
Improve Sine Cosine Algorithm using Eq.(10)
If t >T
No
Yes
Return the global optimum
Stop

Figure 2. The flowchart of the proposed algorithm.

3.4. Time complexity

In optimization algorithms, time complexity significantly impacts the fundamental performance of the algorithm. It reflects the operational efficiency and serves as an important metric for evaluating the algorithm's execution capability. In the Sine Cosine Algorithm (SCA), assuming D is the dimension, N is the population size, and T is the maximum number of iterations, the time complexity of SCA is:

$$O(SCA) = O(N \cdot T \cdot D) \tag{13}$$

With the same basic parameters, the time complexity of ISCAPBIL consists of the Improved Sine Cosine Algorithm (ISCA) phase and the Population-Based Incremental Learning Algorithm (PBIL) phase. The time complexity of PBIL to perform the population-based incremental learning phase according to *the* p-probability is

$$O(PBIL) = O(N \cdot T \cdot D \cdot P) \tag{14}$$

The time complexity of performing the sine-cosine co-evolution strategy phase according to the *1-p* probability is

$$O(ISCA) = O(N \cdot T \cdot D \cdot (1-P)) \tag{15}$$

From the above equation, the complexity of ISCAPBIL is:

$$O(ISCAPBIL) = O(N \cdot T \cdot D \cdot (1-P)) + O(N \cdot T \cdot D \cdot P) \tag{16}$$

3.5. Convergence analysis of the ISCAPBIL algorithm

In this section, we provide a theoretical analysis of the convergence properties of the ISCAPBIL (Improved Sine Cosine Algorithm with Population-Based Incremental Learning). Drawing from the convergence proof of the Sine Cosine Algorithm (SCA) and Markov chain theory, we demonstrate that ISCAPBIL satisfies the conditions for global convergence.

Assumption 1 (Markov property)

The population state transitions in ISCAPBIL follow a Markov property. Specifically, each new population state depends solely on the current state and is independent of all preceding states. This property enables us to model ISCAPBIL's population states as a Markov chain.

Assumption 2 (Positive transition probability)

Within the feasible solution space, the probability of transitioning between any two population states is positive and bounded. This ensures that all feasible solutions can be accessed within a finite number of iterations, preventing the algorithm from being trapped in local minima.

Definition 1: Individual and population states

Let $\lambda \in Y$ represent an individual state, where Y is the feasible solution space.

The population state is the set of all individual states, denoted as $\phi = (\lambda_1, \lambda_2, \ldots, \lambda_{N_\phi})$, where N_ϕ is the population size.

The set of all possible population states is denoted as:

$$\psi = \{\phi = (\lambda_1, \lambda_2, \ldots, \lambda_{N_\phi}) | \lambda_i \in Y\}. \tag{17}$$

Definition 2: Population state transition

The one-step transition of a population state from ϕ^t to ϕ^{t+1} during an iteration of ISCAPBIL is denoted as:

$$P(T_\psi(\phi^t) = \phi^{t+1}) = \prod_{i=1}^{N_\phi} P(T_\phi(\lambda_i^t) = \lambda_i^{t+1}), \tag{18}$$

where, $P\left(T_\phi(\lambda_i^t) = \lambda_i^{t+1}\right)$ represents the transition probability of an individual state.

The ISCAPBIL algorithm combines the sine cosine update mechanism of SCA and the probabilistic learning update from Population-Based Incremental Learning (PBIL). The individual state update rule is given by:

$$\lambda_i^{t+1} = \lambda_i^t + a_i \cdot f(\mu, r) \tag{19}$$

where, a_i is the step size controlling the search range; f(μ,r) is the sine-cosine-driven update function, defined as:

$$f(\mu, r) = \begin{cases} r \cdot \sin(\mu\pi), \text{if} \mu < 0.5, \\ r \cdot \cos(\mu\pi), \text{if} \mu \geq 0.5; \end{cases} \tag{20}$$

where, r and μ are random variables ensuring diversity in the search.

The probabilistic learning mechanism of PBIL introduces additional exploration by shifting individual states based on the population's historical best positions, further enhancing convergence robustness.

Theorem 1: Markov property of population states

The ISCAPBIL algorithm satisfies the Markov property:

$$P(\phi^{t+1} | \phi^t, \phi^{t-1}, \ldots, \phi^0) = P(\phi^{t+1} | \phi^t). \tag{21}$$

Proof:

The state λ_i^{t+1} of each individual depends only on its current state λ_i^t, the step size a_i, and the random variables (μ, r).

Thus, the population state ϕ^{t+1} depends solely on ϕ^t, satisfying the Markov property.

Theorem 2: Global convergence

ISCAPBIL is globally convergent, meaning it will reach the global optimum with probability 1 as the number of iterations tends to infinity.

Proof:

The feasible solution space Y is finite, and hence the population state space ψ is also finite.

Assumption 2 guarantees that $P\left(T_\psi(\phi^t) = \phi^{t+1}\right) > 0$ for any two states $\phi^t, \phi^{t+1} \in \psi$. This ensures that the algorithm can explore the entire solution space.

Based on Markov chain theory, if the state space is finite and all states are accessible (irreducibility), the Markov chain is guaranteed to converge to a stationary distribution. ISCAPBIL satisfies these conditions through its combined update mechanism, which balances local exploitation and global exploration.

The sine-cosine function (f (μ, r)) ensures sufficient randomness for global exploration, while PBIL guides the search toward promising regions. This dual mechanism prevents the algorithm from getting stuck in local minima.

From the above analysis:

1. ISCAPBIL is modeled as a finite-state Markov chain.
2. It possesses global exploration capability and irreducibility in the solution space.
3. The algorithm converges to the global optimum with probability 1 as $t \to \infty$.

These properties provide a rigorous theoretical foundation for the global convergence of the ISCAPBIL algorithm.

3.6. Economic Load Dispatch (ELD) model in power systems

Economic Load Dispatch (ELD) is a crucial optimization problem in power systems, where the objective is to minimize the total generation cost while meeting the required load demand and adhering to system constraints. The basic ELD model can be formulated as:

3.6.1. Objective function

The primary objective in the ELD problem is to minimize the total fuel cost of the power generation units. This can be expressed as:

$$minC = \sum_{i=1}^{N} a_i P_i^2 + b_i P_i + c_i \tag{22}$$

where *C* is the total fuel cost. P_i is the power output of the *i*-th generator. a_i, b_i, and c_i are cost coefficients of the *i*-th generator. *N* is the number of generating units.

3.6.2. Power balance constraint

The total power generation must meet the load demand P load, including any losses in the system. This constraint is given by:

$$\sum_{i=1}^{N} P_i = P_{load} + P_{loss} \tag{23}$$

where P_{load} is the total load demand. P_{loss} is the transmission loss, which can be modeled as a function of the generator outputs.

3.6.3. Generation limits

Each power generator has its operational limits, which must be respected. These limits are given by:

$$P_{i,min} \le P_i \le P_{i,max}, \forall i \in \{1,2,\dots,N\} \tag{24}$$

where $P_{i,min}$ and $P_{i,max}$ are the minimum and maximum generation limits for the *i*-th generator.

3.6.4. Emission constraints (optional)

In modern power systems, environmental constraints such as emissions are often included in the ELD model. These can be represented by:

$$E_i = \alpha_i P_i^2 + \beta_i P_i + \gamma_i \tag{25}$$

where E_i is the emission of the *i*-th generator, and α_i, β_i, and γ_i are emission coefficients.

3.6.5. Overall problem formulation

Thus, the ELD problem can be formulated as:

$$\min \text{C\&} = \sum_{i=1}^{N} a_i P_i^2 + b_i P_i + c_i$$

Subject to

$$\sum_{i=1}^{N} P_i = P_{load} + P_{loss} \tag{26}$$

$$P_{i,min} \leq P_i \leq P_{i,max}, \forall i$$

$$E_i \leq E_{max}, \quad \forall i$$

3.6.6. Relationship between the Economic Load Dispatch Model and the optimization algorithm

The ELD model described above is a constrained optimization problem, where the objective is to minimize the fuel cost while satisfying the power balance, generation limits, and potentially emissions constraints. Solving this problem requires an efficient optimization algorithm capable of handling both continuous variables (the power outputs) and constraints.

The hybrid optimization algorithm proposed in this paper, which combines the improved Sine Cosine Algorithm (SCA) with Population-Based Incremental Learning (PBIL), is designed to find the optimal generation schedule by searching for the best power output Pi for each generator. The relationship between the ELD model and the optimization algorithm is as follows:

1. Algorithm's Objective: The hybrid algorithm aims to minimize the objective function CCC of the ELD model. By incorporating both local and global search strategies (via the SCA and PBIL), it efficiently explores the solution space to find the optimal set of generation outputs.
2. Handling Constraints: The proposed hybrid algorithm is designed to handle the constraints in the ELD problem. The power balance constraint and generation limits are treated as boundary conditions that the algorithm respects during its search. If emission constraints are considered, the algorithm also ensures that these constraints are not violated.
3. Parameter Tuning: The improved SCA dynamically adjusts parameters such as the exploration-exploitation balance, improving the convergence rate and helping to avoid local optima. The PBIL mechanism is used to incrementally update the population, ensuring that the diversity of the solution set is maintained, which is crucial for avoiding premature convergence to suboptimal solutions.
4. Solution Process: The algorithm starts with an initial population of power generation outputs, evaluates the objective function and constraints, and iteratively refines the solution by applying the SCA and PBIL techniques. The hybrid approach facilitates a better balance between exploration of the search space and exploitation of promising solutions, improving the overall optimization process.

By applying this hybrid algorithm to the ELD problem, we achieve a solution that not only minimizes the generation cost but also satisfies the system's constraints more efficiently than traditional methods.

4. Simulation experiment

4.1. Experimental environment

The experimental environment is Intel(R) Core(TM) i7-10750H CPU@ 2.30GHz, 16GB RAM, Windows 10 operating system, and the algorithms are compiled and implemented using the software MATLAB R 2020b simulation platform.

4.2. Test functions and other parameters

In this paper, 23 standard test functions are selected for performance examination. Among them, including single-peak dimensionality functions (F1~F7), multi-peak dimensionality functions (F8~F13), multi-peak fixed dimensionality functions (F14~F23), and the related function parameters and specific expressions are shown in literature [30,31]. The CEC2013 standard test function was used for complex function performance examination. Among them, including different properties such as single-peak, multi-peak, divisible and indivisible, single-peak dimensionality function (F1~F13), multi-peak dimensionality function (F14~F25), the related function parameters and specific expressions are shown in literature [32]. Moreover, six standard test functions for high-dimensional performance testing, the related function parameters, and specific expressions are shown in the literature [33]. Different features in the selected test functions are used to test different capabilities of the algorithms. Among them, the single-peak test function has one and only one local optimum, which is effective in testing the optimization performance, convergence speed, and convergence accuracy of the algorithm. The fixed-dimension multi-peak test function has multiple local optima, and the function is complex and diverse, which makes it more difficult to find the optimal value, and has a better effect in testing the algorithm's ability to search globally and avoid falling into the local optima.

4.3. Comparative experiments with SCA and other intelligent optimization algorithms

To validate the improved performance of ISCAPBIL, it is compared with the basic SCA and other intelligent optimization algorithms. Among the algorithms involved in the comparison are the sine cosine algorithm [30] (SCA), Whale Optimization Algorithm [10] (WOA), Salp Swarm Algorithm [34] (SSA), Grey Wolf Optimization Algorithm [8] (GWO). In this experiment, the population size is set to N = 30, the maximum number of iterations is T = 500, and the dimensionality of the test functions F1~F13 is D = 30 (since the other functions F14~F23 in this test function set are of fixed dimensions, they are not in the setting range). To ensure the fairness of the experiment, each algorithm is run independently 30 times, and the average (Ave) and standard deviation (Std) are calculated; the results of the better ones are bolded, and the results of the experiment are shown in Table 1. For the comparative experiments, we analyze the results of the parameters, which are the average optimal value and the standard deviation. The average optimal value obtained from the simulation experiments presents the convergence speed and solution accuracy of the algorithm, and the standard deviation reflects the stability and robustness of the algorithm. Figure 3 lists the convergence curves of some test functions of the four algorithms in Table 1, such as SCA, WOA, SSA, and GWO, obtained after running for 30 times, reflecting the convergence changes in the algorithm's optimization process. Here, the vertical coordinate is represented by the logarithm of the average optimal value of the function and the horizontal coordinate is the number of iterations.

As seen from the results in Table 1, in terms of the comparative experiments, ISCAPBIL is superior to WOA in comparison with WOA, except for 4 functions such as F6, F8, F12, F13, etc., which are slightly inferior to WOA, and the rest of the 19 functions; ISCAPBIL is superior to GWO in comparison with GWO, except for 3 functions such as F11, F12, F13, etc., which are slightly inferior to GWO, and F16, F17, F18, F19, and F23, with other 5 functions comparable to GWO, and the remaining 15 functions are better than GWO; ISCAPBIL compares with SSA, except for 3 functions such as F15, F16, and F17, which are comparable to SSA, and the remaining 20 functions are better than SSA; and ISCAPBIL compares with SCA, except for the F16 function which is comparable to SCA, and the remaining 22 functions are better than SCA. This shows that the significant optimality seeking performance of ISCAPBIL. Analyzing the convergence curve in Figure 3, we found that ISCAPBIL completes the convergence in functions F1, F3, and F4 at about 400 iterations, and the convergence speed of ISCAPBIL in function F10 is remarkable, completing the convergence at about 100 iterations the convergence at about 200 iterations in functions F21 and F23. This again proves that the convergence performance of ISCAPBIL has better convergence performance.

To further verify the improved performance of ISCAPBIL, the dimension of the test function is increased to 60 and 100 dimensions respectively with other parameters unchanged to further verify the stability of ISCAPBIL, and the experimental results are shown in Table 2. Figure 4 lists the convergence curves of some of the test functions, which more intuitively presents the trend change of algorithmic optimization search process.

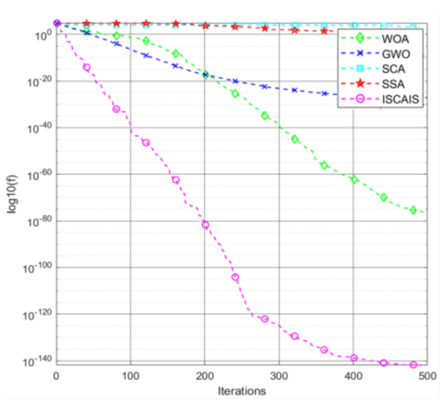
As seen from the results in Table 2, in the 60-dimensional comparison experiment, ISCAPBIL is superior to WOA in all 10 functions except for 3 functions such as F8, F12, F13, etc., which are slightly inferior to WOA; in the comparison between ISCAPBIL and GWO, ISCAPBIL is superior to GWO in all 11 functions except for 2 functions such as F12, F13, etc., which are slightly inferior to GWO; and in the ISCAPBIL Compared with SSA and SCA, the 13 functions involved in the experiment are all better than SSA and SCA. In the 100-dimensional comparison experiment, ISCAPBIL compares with WOA, except for 3 functions, such as F8, F12, and F13, which are slightly inferior to WOA, and the remaining 10 functions are better than WOA; ISCAPBIL compares with GWO, except for 2 functions, such as F12 and F13, which are slightly inferior to GWO, and the remaining 11 functions are better than GWO; and ISCAPBIL compared with SSA and SCA, the 13 functions involved in the experiment are better than SSA and SCA. overall, ISCAPBIL has the best performance in optimization search. As the dimension of the objective function increases, ISCAPBIL maintains a better optimization performance.

Table 1. Performance comparison of ISCAPBIL with basic SCA and other algorithms.

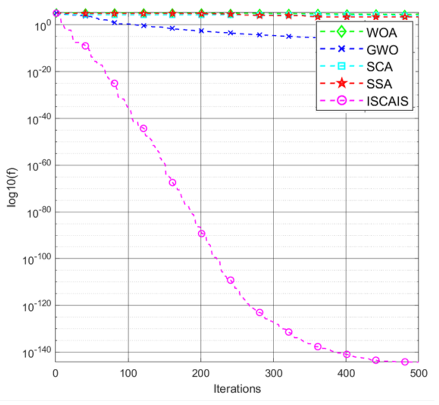
Function	Evaluation criterion	WOA	GWO	SSA	SCA	ISCAPBIL
F1	Ave	2.02E-72	1.03E-27	3.02E+02	8.10E+01	2.73E-132
	Std	1.09E-71	1.68E-27	8.75E+01	1.15E+02	3.86E-132
F2	Ave	2.16E-50	9.79E-17	1.22E+01	1.76E-01	3.38E-67
	Std	1.10E-49	5.86E-17	1.97E+00	2.57E-01	4.74E-67
F3	Ave	4.47E+04	4.22E-06	6.58E+03	1.42E+04	1.76E-131
	Std	1.53E+04	1.08E-05	2.85E+03	7.42E+03	2.30E-131
F4	Ave	4.64E+01	6.11E-07	1.84E+01	4.80E+01	5.50E-70
	Std	2.86E+01	6.59E-07	3.64E+00	1.01E+01	7.70E-70
F5	Ave	2.80E+01	2.68E+01	2.62E+04	3.43E+05	**2.88E+01**
	Std	4.50E-01	5.50E-01	2.91E+04	5.99E+05	1.04E-01
F6	Ave	**3.17E-01**	**8.13E-01**	3.20E+02	1.04E+02	3.88E+00
	Std	1.82E-01	3.82E-01	1.14E+02	1.45E+02	5.84E-01
F7	Ave	3.10E-03	1.88E-03	3.31E-01	3.02E-01	**3.27E-04**
	Std	4.58E-03	1.22E-03	1.08E-01	4.11E-01	6.95E-06
F8	Ave	−10672.1305	−6709.0284	−6699.3213	−3785.7654	−6707.7605
	Std	1704.9911	8.88E+02	7.95E+02	273.8353	37.7944
F9	Ave	**0.00E+00**	2.21E+00	1.33E+02	5.28E+01	**0.00E+00**
	Std	0.00E+00	3.79E+00	2.50E+01	4.14E+01	0.00E+00
F10	Ave	4.44E-15	9.94E-14	6.37E+00	1.62E+01	**8.88E-16**
	Std	2.09E-15	1.96E-14	9.36E-01	7.34E+00	0.00E+00
F11	Ave	1.11E-02	4.42E-03	3.66E+00	1.60E+00	**0.00E+00**
	Std	6.07E-02	8.23E-03	8.74E-01	6.49E-01	0.00E+00
F12	Ave	**1.71E-02**	3.74E-02	3.61E+01	6.12E+05	4.34E-01
	Std	7.51E-03	2.04E-02	3.92E+01	1.86E+06	3.68E-01
F13	Ave	**4.94E-01**	5.72E-01	2.73E+03	1.44E+06	**2.99E+00**
	Std	2.51E-01	1.77E-01	1.05E+04	2.02E+06	2.75E-04
F14	Ave	2.57E+00	4.98E+00	3.02E+02	1.33E+00	**1.66E+00**
	Std	2.99E+00	4.53E+00	8.75E+01	7.50E-01	1.15E+00
F15	Ave	7.18E-04	2.47E-03	9.99E-04	1.21E-03	3.27E-04
	Std	5.42E-04	6.07E-03	2.84E-04	3.42E-04	9.01E-06
F16	Ave	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	8.63E-10	2.82E-08	3.06E-14	9.52E-05	1.6162e-05
F17	Ave	0.39789	0.39789	0.39789	0.39789	0.39888
	Std	1.22E-05	2.14E-03	2.16E-03	2.16E-03	8.60E-04
F18	Ave	3.0001	3	3.0005	3.0003	3
	Std	4.93E+00	5.05E-05	5.01E-04	3.71E-04	1.1818e-06
F19	Ave	−3.8582	−3.8615	−3.8626	−3.8536	−3.8418
	Std	5.88E-02	2.72E-03	2.59E-04	3.10E-03	0.013541
F20	Ave	−3.1983	−3.2541	−3.2072	−2.8409	−**3.1975**
	Std	9.35E-02	7.38E-02	7.48E-02	2.46E-01	0.0055334

Continued on next page

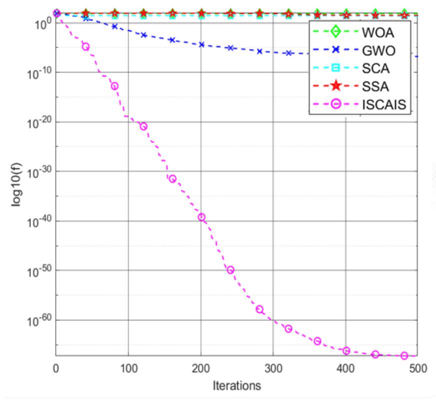
Function	Evaluation criterion	WOA	GWO	SSA	SCA	ISCAPBIL
F21	Ave	−8.4313	−8.6048	−8.3476	−1.8336	**−10.1067**
	Std	2.27E+00	2.46E+00	3.16E+00	1.63E+00	0.0087708
F22	Ave	−8.1586	−9.6926	−9.4209	−3.4597	**−10.3947**
	Std	3.23E+00	1.62E+00	2.04E+00	1.81E+00	0.0059236
F23	Ave	−8.4151	−10.0841	−9.3126	−3.8672	**−10.5335**
	Std	3.14E+00	9.79E-01	2.40E+00	1.03E+00	0.0021937



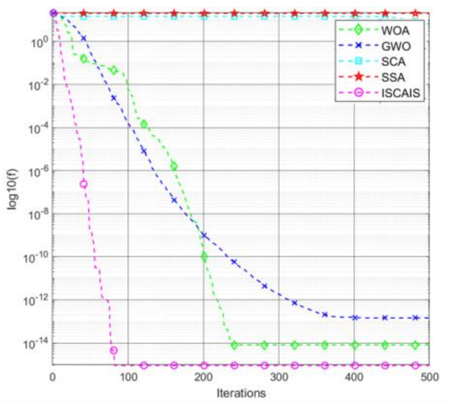
F1



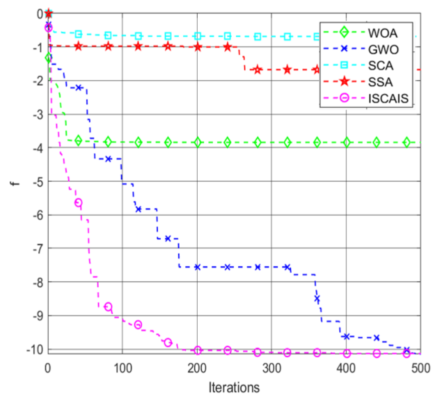
F3



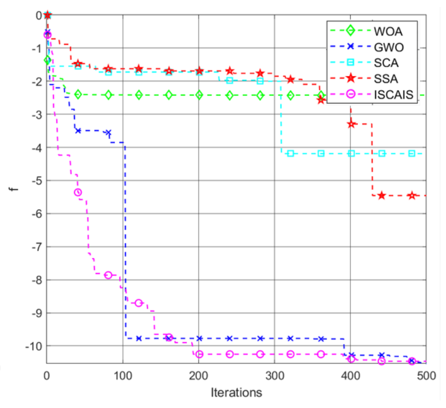
F4



F10



F21

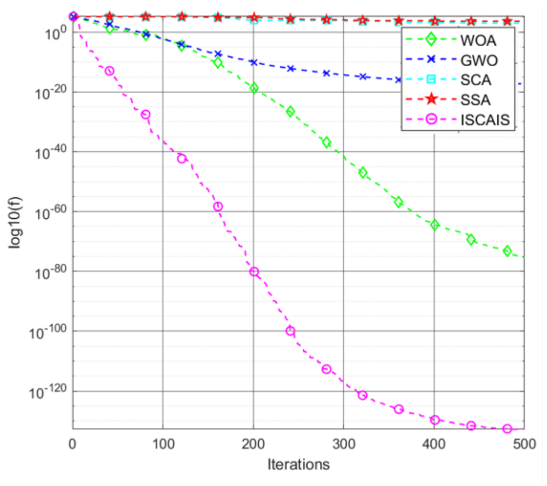


F23

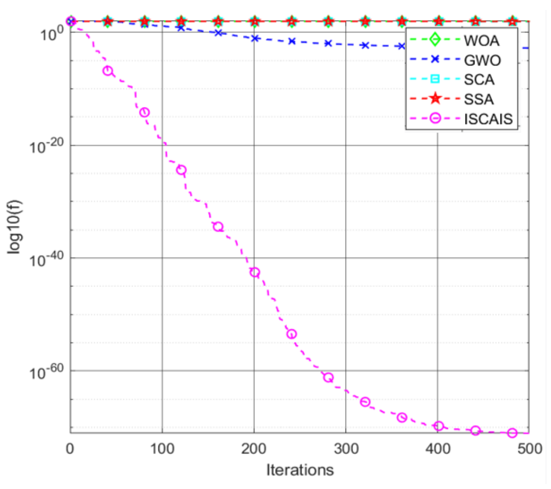
Figure 3. Convergence curve of benchmark functions.

Table 2. Comparison of ISCAPBIL with SCA and other classical intelligent algorithms.

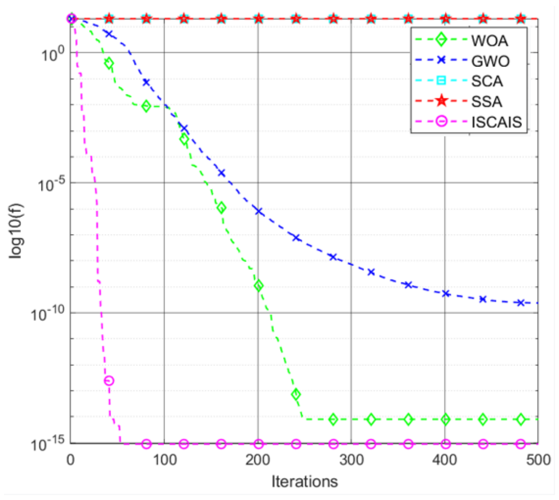
Function	Evaluation criterion	WOA		GWO		SSA		SCA		ISCAPBIL	
		60	100	60	100	60	100	60	100	60	100
F1	Ave	8.79E-71	9.88E-74	1.03E-27	7.23E-28	3.02E+02	3.18E+02	7.82E+01	9.23E+01	**2.14E-137**	**2.90E-142**
	Std	3.71E-70	4.30E-73	1.68E-27	9.27E-28	8.75E+01	1.14E+02	1.24E+02	1.42E+02	3.03E-137	2.13E-142
F2	Ave	1.29E-50	2.23E-51	9.79E-17	1.13E-16	1.22E+01	1.23E+01	2.06E-01	1.57E-01	**4.07E-70**	**6.99E-68**
	Std	4.22E-50	6.13E-51	5.86E-17	6.47E-17	1.97E+00	3.13E+00	4.10E-01	2.01E-01	5.75E-70	9.88E-68
F3	Ave	4.33E+04	4.72E+04	4.22E-06	3.15E-05	6.58E+03	5.94E+03	1.56E+04	1.33E+04	**8.41E-130**	**2.93E-135**
	Std	1.36E+04	1.24E+04	1.08E-05	9.29E-05	2.85E+03	2.46E+03	6.59E+03	7.85E+03	1.19E-129	4.14E-135
F4	Ave	5.10E+01	5.35E+01	6.11E-07	6.68E-07	1.84E+01	1.87E+01	4.61E+01	4.46E+01	**5.15E-70**	**1.37E-72**
	Std	2.94E+01	2.48E+01	6.59E-07	5.59E-07	3.64E+00	3.54E+00	1.27E+01	1.29E+01	2.08E-70	6.19E-73
F5	Ave	2.80E+01	2.81E+01	2.68E+01	2.69E+01	2.62E+04	2.59E+04	5.35E+05	2.66E+05	5.89E+01	9.89E+01
	Std	3.90E-01	5.13E-01	5.50E-01	6.93E-01	2.91E+04	2.22E+04	1.70E+06	4.40E+05	1.53E-02	9.31E-02
F6	Ave	3.84E-01	3.32E-01	8.13E-01	7.66E-01	3.20E+02	3.02E+02	1.29E+02	8.19E+01	**1.26E+01**	**2.11E+01**
	Std	2.72E-01	2.07E-01	3.82E-01	3.71E-01	1.14E+02	7.90E+01	1.74E+02	1.24E+02	3.39E-02	3.66E-01
F7	Ave	2.56E-03	3.19E-03	1.88E-03	2.05E-03	3.31E-01	3.25E-01	2.16E-01	3.47E-01	**1.05E-04**	**2.04E-04**
	Std	2.79E-03	4.50E-03	1.22E-03	8.65E-04	1.08E-01	1.43E-01	4.31E-01	3.00E-01	1.28E-04	1.78E-04
F8	Ave	−1.06E+04	−1.05E+04	−6.03E+03	−5.92E+03	−6.46E+03	−6.33E+03	−3.89E+03	−3.96E+03	**−6.98E+03**	**−8.56E+03**
	Std	1.69E+03	1.61E+03	8.88E+02	9.96E+02	7.95E+02	8.14E+02	2.45E+02	2.66E+02	5.32E+02	2.23E+03
F9	Ave	1.89E-15	0.00E+00	2.21E+00	2.55E+00	1.33E+02	1.30E+02	5.44E+01	6.79E+01	**0.00E+00**	**0.00E+00**
	Std	1.04E-14	0.00E+00	3.79E+00	3.34E+00	2.50E+01	2.07E+01	4.35E+01	4.15E+01	0.00E+00	0.00E+00
F10	Ave	3.85E-15	4.32E-15	9.94E-14	1.08E-13	6.37E+00	6.37E+00	1.42E+01	1.60E+01	**8.88E-16**	**8.88E-16**
	Std	2.81E-15	2.72E-15	1.96E-14	2.07E-14	9.36E-01	8.91E-01	8.09E+00	7.21E+00	0.00E+00	0.00E+00
F11	Ave	5.91E-03	1.25E-02	4.42E-03	3.06E-03	3.66E+00	3.60E+00	1.81E+00	1.89E+00	**0.00E+00**	**0.00E+00**
	Std	3.24E-02	4.79E-02	8.23E-03	5.85E-03	8.74E-01	9.60E-01	1.10E+00	1.63E+00	0.00E+00	0.00E+00
F12	Ave	2.25E-02	1.86E-02	3.74E-02	5.16E-02	3.61E+01	2.27E+01	1.94E+06	4.08E+05	**9.27E-01**	**8.59E-01**
	Std	1.78E-02	1.34E-02	2.04E-02	3.00E-02	3.92E+01	1.33E+01	9.39E+06	1.16E+06	1.84E-01	1.05E-01
F13	Ave	5.56E-01	5.31E-01	5.72E-01	5.71E-01	2.73E+03	1.23E+03	1.57E+06	1.81E+06	**5.99E+00**	**9.98E+00**
	Std	2.20E-01	2.90E-01	1.77E-01	2.24E-01	1.05E+04	2.63E+03	2.78E+06	3.87E+06	6.01E-03	1.09E-02



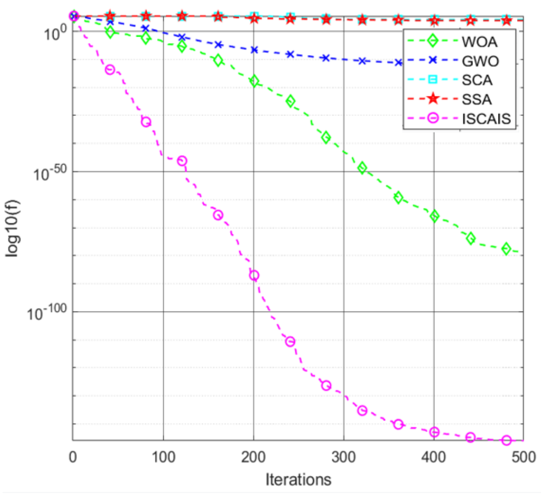
F1 (D = 60)



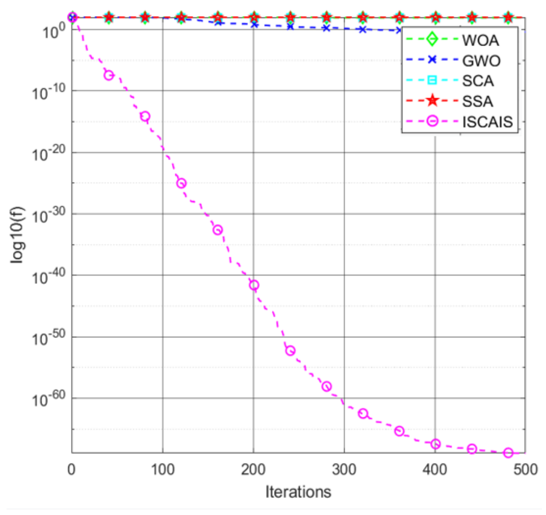
F4 (D = 60)



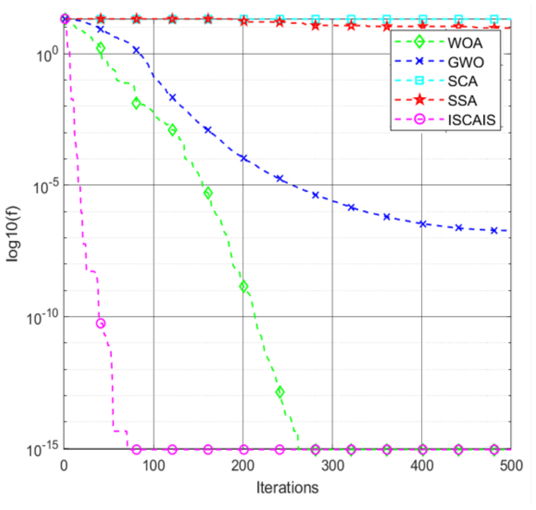
F10 (D = 60)



F1 (D = 100)



F4 (D = 100)



F10 (D = 100)

Figure 4. Convergence curve of benchmark functions.

4.4. Comparative experiments with other improved SCAs

To further verify the effectiveness of the improvement of ISCAPBIL, it is subjected to comparison experiments with other improved SCAs. The algorithms involved in the comparison experiments are the positive cosine algorithm based on elite chaotic search strategy (COSCA) [35], Memory-Guided Sine-Cosine Based Algorithm (MGSCA) [36], Sine Cosine Algorithm Based on Differential Evolution (SCADE) [37], and Cloud Model Based Sine Cosine Algorithm (CSCA) [25]. In this experiment, the population size is set to N = 30, the maximum number of iterations is T = 500, and the dimensionality of the test functions F1~F13 is D = 30 (the other functions F14~F23 in this test function set are not included in the setting because they have fixed dimensions). In addition to SCADE, CSCA, and ISCAPBIL, the experimental data of other algorithms are taken from various studies, and the better results are indicated by bolding, and the experimental results are shown in Table 3.

As seen from Table 3, ISCAPBIL outperforms all other improved SCA algorithms in the comparison experiments with other improved SCA algorithms. This is because ISCAPBIL adopts an incremental learning strategy to make individual corrections to the population, which ensures that the algorithm constantly approaches the global optimal solution and avoids splitting and reorganization of individuals, which enhances the algorithm solution accuracy.

Table 3. Performance comparison of ISCAPBIL with modified SCA.

Function	Evaluation criterion	COSCA	MGSCA	SCADE	CSCA	ISCAPBIL
F1	Ave	2.44E-78	7.62E-23	9.58E-95	7.49E-02	2.73E-132
	Std	3.21E-94	1.67E-22	4.92E-94	1.93E-01	3.86E-132
F2	Ave	1.52E-44	1.92E-17	6.14E-63	5.09E-07	3.38E-67
	Std	1.94E-60	4.63E-17	2.73E-62	8.73E-07	4.74E-67
F3	Ave	1.78E-15	2.80E-03	1.93E-04	5.83E+03	1.76E-131
	Std	1.45E-30	8.78E-03	9.81E-04	5.29E+03	2.30E-131
F4	Ave	5.27E-35	8.11E-03	2.85E-09	1.26E+01	5.50E-70
	Std	1.91E-50	2.34E-02	1.53E-08	9.50E+00	7.70E-70
F5	Ave	2.84E+01	2.75E+01	2.69E+01	4.17E+03	**2.88E+01**
	Std	7.94E-16	7.07E-01	1.47E-01	1.80E+04	1.04E-01
F6	Ave	3.82E+00	1.39E+00	7.54E-05	5.09E+00	**3.88E+00**
	Std	7.94E-16	5.59E-01	9.46E-05	9.42E-01	5.84E-01
F7	Ave	3.21E-04	3.87E-03	8.44E-03	7.74E-02	**3.27E-04**
	Std	6.06E-21	2.65E-03	7.37E-03	5.34E-02	6.95E-06
F8	Ave	−3.31E+03	−6.36E+03	−1.20E+04	−3.37E+03	−6.71E+03
	Std	2.64E-12	6.41E+02	2.53E+02	2.95E+02	3.78E+01
F9	Ave	0.00E+00	2.86E-01	0.00E+00	4.63E+01	**0.00E+00**
	Std	0.00E+00	8.88E-01	0.00E+00	4.93E+01	0.00E+00
F10	Ave	2.48E-15	7.39E+00	2.13E-15	2.89E-02	**8.88E-16**
	Std	7.05E-31	9.88E+00	1.76E-15	7.60E-02	0.00E+00
F11	Ave	0.00E+00	1.01E-02	0.00E+00	4.78E-01	**0.00E+00**
	Std	0.00E+00	1.85E-02	0.00E+00	3.69E-01	0.00E+00

Continued on next page

Function	Evaluation criterion	COSCA	MGSCA	SCADE	CSCA	ISCAPBIL
F12	Ave	3.68E-01	1.00E-01	3.45E-05	9.39E+00	**4.34E-01**
	Std	**1.73E-16**	4.78E-02	1.66E-04	4.22E+01	3.68E-01
F13	Ave	2.04E+00	1.46E+00	8.13E-03	6.35E+02	**2.99E+00**
	Std	1.20E-15	3.20E-01	2.29E-02	3.21E+03	2.75E-04
F14	Ave	3.56E+00	1.13E+00	9.98E-01	2.26E+00	**1.66E+00**
	Std	5.95E-16	5.03E-01	5.00E-16	2.48E+00	1.15E+00
F15	Ave	7.87E-04	6.98E-04	7.52E-04	6.69E-04	3.27E-04
	Std	7.75E-19	3.45E-04	1.54E-04	2.44E-04	9.01E-06
F16	Ave	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00
	Std	1.09E-16	2.04E-08	4.44E-16	3.05E-05	1.62E-05
F17	Ave	3.98E-01	3.98E-01	3.98E-01	4.00E-01	3.99E-01
	Std	0.00E+00	4.36E-06	0.00E+00	2.84E-03	8.60E-04
F18	Ave	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00
	Std	7.94E-16	4.99E-06	3.18E-07	1.19E-04	1.18E-06
F19	Ave	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.84E+00
	Std	1.39E-15	3.85E-03	7.64E-13	3.33E-03	1.35E-02
F20	Ave	−3.16E+00	−3.11E+00	−3.31E+00	−3.12E+00	−**3.20E+00**
	Std	9.93E-16	1.80E-01	3.05E-02	7.41E-02	5.53E-03
F21	Ave	−9.85E+00	−7.37E+00	−9.75E+00	−4.23E+00	−**1.01E+01**
	Std	6.35E-15	2.91E+00	8.91E-01	1.14E+00	8.77E-03
F22	Ave	−1.03E+01	−8.39E+00	−1.04E+01	−4.46E+00	−**1.04E+01**
	Std	**4.36E-15**	3.21E+00	1.45E+00	8.68E-01	5.92E-03
F23	Ave	−1.05E+01	−8.77E+00	−1.05E+01	−4.57E+00	−**1.05E+01**
	Std	**3.97E-15**	3.04E+00	3.45E-14	1.34E+00	2.19E-03

4.5. CEC2013 test function comparison experiment

We further validate the performance of ISCAPBIL in solving complex functions. Thus, in this section, we select the CEC2013 function test set [38]. The 28 single-objective test functions in it are examined for performance. Among them are different properties such as single-peak, multi-peak, separable, and non-separable. The algorithms involved in the comparison experiments are Sine Cosine Algorithm (SCA), Whale Optimization Algorithm (WOA), Grey Wolf Algorithm (GWO), and Bottle Sea Sheath Algorithm (SSA), and the parameters of the algorithms involved in the comparison experiments are set identically, with the population size set to N = 30, the maximum number of iterations set to T = 500, and the dimension of the test function set to D = 30. To ensure the fairness of the results of the experiments, the algorithms involved in the comparison experiments are run independently. To ensure the fairness of the experimental results, the algorithms involved in the comparison are run independently for 30 times and the mean (Ave) and standard deviation (Std) are calculated, and the experimental results are shown in Table 4 (See Figure 5). From Table 4, ISCAPBIL is comparable to the results of WOA, GWO, and SCA except for the function F17 and is superior to the results of SSA. The results of F19 are comparable to the results of WOA, GWO, and are superior to the results of SSA, and SCA. The results of F20 are comparable to the results of WOA and GWO.

In function F23, comparable to WOA, GWO, and SSA. The experimental results of ISCAPBIL algorithm in the rest of the functions are better than the other algorithms. The experimental analysis shows that ISCAPBIL has higher optimization accuracy, can effectively overcome the shortcomings of other algorithms and has higher optimization performance.

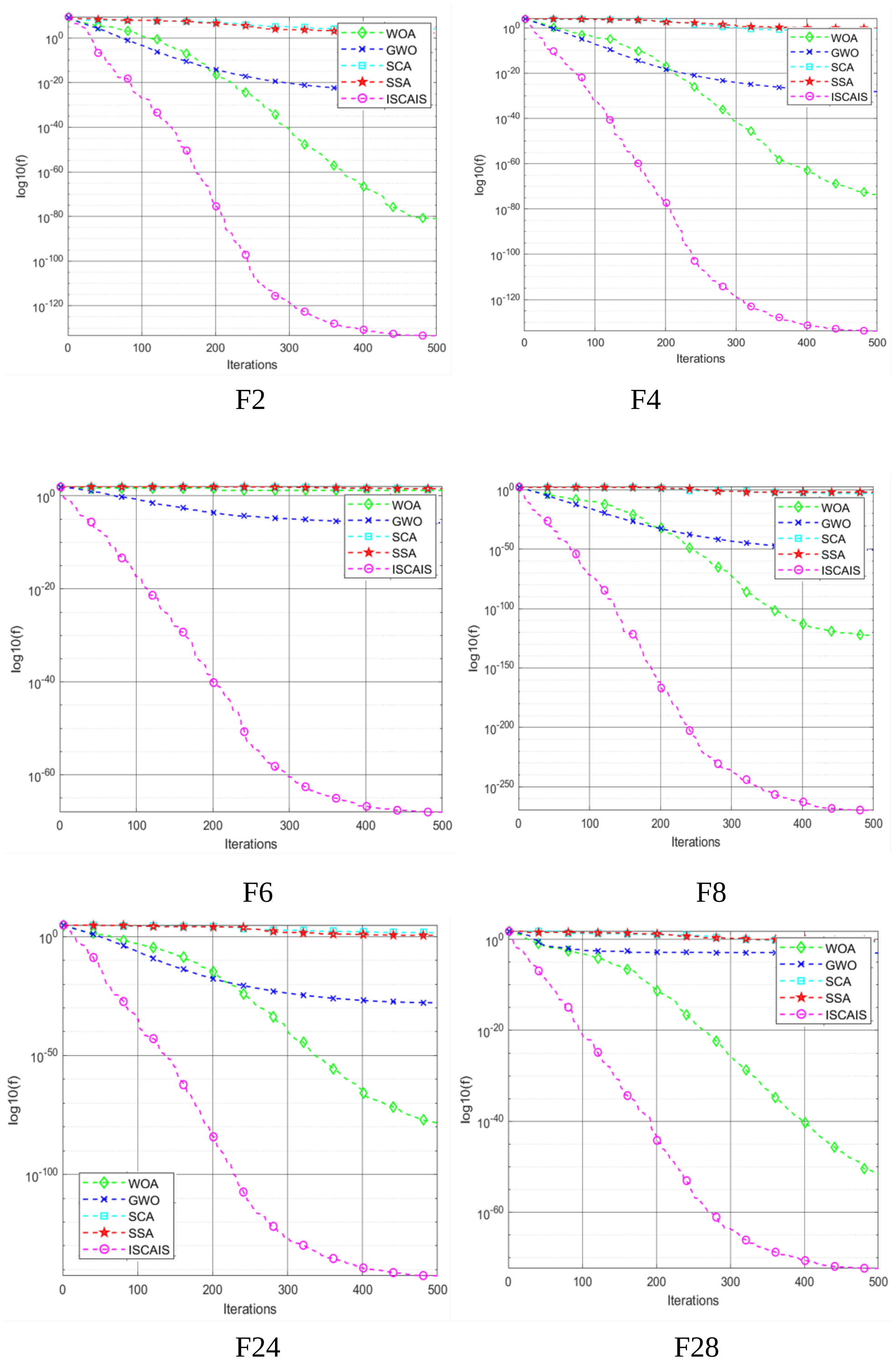


Figure 5. Convergence curve of benchmark functions.

Table 4. CEC2013 single objective test function comparison experiment.

Function	Evaluation criterion	WOA	GWO	SSA	SCA	ISCAPBIL
F1	Ave	9.61E-78	1.40E-27	2.72E+02	3.08E+01	**2.3372e-140**
	Std	1.36E-77	1.86E-27	6.99E+00	2.65E+01	3.3037e-140
F2	Ave	3.45E-73	7.67E-25	5.07E+07	6.70E+02	**8.3147e-123**
	Std	4.81E-73	7.93E-28	2.28E+07	9.35E+02	1.1759e-122
F3	Ave	5.22E-81	2.87E-29	8.60E+01	1.24E-01	**1.1688e-134**
	Std	7.29E-81	6.11E-30	2.49E+01	1.73E-01	1.6431e-134
F4	Ave	3.06E-108	9.13E-92	2.08E+03	7.46E+00	**1.6208e-158**
	Std	4.33E-108	7.08E-92	2.90E+03	9.93E+00	2.2815e-158
F5	Ave	9.89E-52	8.58E-17	1.26E+01	1.10E-01	**5.7408e-71**
	Std	5.69E-52	1.67E-17	2.80E-01	1.47E-01	4.1107e-71
F6	Ave	4.10E+01	1.41E-06	2.43E+01	2.04E+01	**9.381e-69**
	Std	5.35E+01	1.03E-06	7.75E+00	1.03E+01	1.3267e-68
F7	Ave	0.00E+00	0.00E+00	2.42E+02	7.00E+00	**0.00E+00**
	Std	0.00E+00	0.00E+00	8.91E+01	0.00E+00	0.00E+00
F8	Ave	2.26E-116	2.21E-51	1.11E-02	1.43E-02	**1.9181e-270**
	Std	2.93E-116	2.99E-51	4.29E-03	1.75E-02	0.00E+00
F9	Ave	5.24E-03	1.77E-03	2.87E-01	2.26E-01	**0.00013842**
	Std	6.38E-03	3.42E-04	1.00E-01	5.83E-02	0.0001026
F10	Ave	2.82E+01	2.82E+01	8.29E+02	3.56E+02	**28.8793**
	Std	8.02E-01	4.27E-01	3.88E+02	4.37E+02	0.071664
F11	Ave	0.00E+00	1.71E+00	1.38E+02	4.57E+01	**0.00E+00**
	Std	0.00E+00	2.41E+00	2.25E+01	5.75E+01	0.00E+00
F12	Ave	0.00E+00	1.20E+01	1.58E+02	1.46E+02	**0.00E+00**
	Std	0.00E+00	1.42E+00	5.13E+00	3.39E+00	0.00E+00
F13	Ave	5.85E-02	8.89E-03	3.90E+00	1.06E+00	**0.00E+00**
	Std	8.28E-02	1.26E-02	7.15E-01	8.12E-02	0.00E+00
F14	Ave	1.20E+03	6.22E+03	5.05E+03	8.77E+03	**7310.6527**
	Std	1.12E+03	7.54E+02	1.72E+03	2.06E+02	534.7036
F15	Ave	4.44E-15	1.00E-13	6.38E+00	1.01E+01	**8.88E-16**
	Std	0.00E+00	0.00E+00	2.49E-01	1.43E+01	0.00E+00
F16	Ave	4.00E+06	1.11E-01	3.56E+01	3.61E+00	**0.58181**
	Std	5.65E+06	9.25E-02	1.38E+01	2.42E+00	0.12945
F17	Ave	4.08E-01	6.29E-01	8.59E+03	4.55E+01	**2.8855**
	Std	3.15E-01	2.78E-02	5.03E+03	3.92E+00	0.0064338
F18	Ave	1.59E-52	3.30E-04	1.15E+01	7.76E-01	**1.3893e-69**
	Std	2.04E-52	4.67E-04	3.55E+00	1.09E+00	1.9646e-69
F19	Ave	3.18E+00	7.14E+00	2.42E+01	2.20E+01	**24.0338**
	Std	4.11E-01	1.58E-01	3.98E+00	1.55E-02	6.8474
F20	Ave	−1.50E+01	−1.50E+01	−2.20E-01	−2.91E+00	**−1.50E+01**
	Std	0.00E+00	2.51E-15	1.34E+00	1.35E+01	0.00E+00

Continued on next page

Function	Evaluation criterion	WOA	GWO	SSA	SCA	ISCAPBIL
F21	Ave	2.35E-02	3.72E-02	4.80E-01	1.03E-01	**0.00E+00**
	Std	1.95E-02	9.57E-11	7.24E-07	3.45E-02	0.00E+00
F22	Ave	−6.98E+01	−5.92E+01	−6.84E+01	−4.04E+01	−49.7116
	Std	8.91E+00	3.97E+00	7.79E-01	6.51E+00	0.081216
F23	Ave	−1.03E+01	−1.47E+01	−1.29E+01	−7.72E+00	−11.6872
	Std	2.67E+00	9.02E-01	1.09E+00	6.73E-01	0.072432
F24	Ave	6.53E-76	1.60E-27	4.15E+02	6.53E+01	**2.6943e-143**
	Std	9.23E-76	2.05E-27	7.01E+01	9.24E+01	3.5307e-143
F25	Ave	0.00E+00	0.00E+00	1.02E+02	3.37E+01	**0.00E+00**
	Std	0.00E+00	0.00E+00	6.90E+00	4.09E+01	0.00E+00
F26	Ave	0.00E+00	1.78E-02	3.29E+00	1.02E+00	**0.00E+00**
	Std	0.00E+00	2.51E-02	5.31E-01	3.96E-03	0.00E+00
F27	Ave	2.66E-15	7.73E-14	6.49E+00	1.05E+01	**8.8818e-16**
	Std	2.51E-15	2.51E-15	1.14E-01	1.40E+01	0.00E+00
F28	Ave	1.01E-51	4.10E-04	3.32E+00	2.16E+00	**4.5976e-73**
	Std	1.43E-51	4.27E-04	1.26E+00	5.19E-01	4.6512e-73

4.6. Test function comparison experiments for large-scale dimensions

To verify the stability and superiority of ISCAPBIL and to further demonstrate that the improved algorithm can solve optimization problems with high dimensions, we the literature [33]. The six benchmark test functions in the literature are used for high-dimensional function testing, including single-peak function (F1, F2, F5) and multi-peak function (F15, F16, F17). Among them, the single-peak function is aimed at the convergence speed and convergence accuracy detection of the algorithm, and the multi-peak function is aimed at the global search ability detection of the algorithm. The parameters and specific expressions of the test functions are shown in the literature [33]. These are compared with other improved algorithms proposed in recent years for large-scale optimization function problems. The algorithms involved in the comparison are the Comprehensive Learning Particle Swarm Optimization (CLPSO) algorithm [39], Adaptive Covariance Matrix Evolutionary Strategies Algorithm (CMAES) [40], Adaptive Strategy Difference Evolution (SaDE) [41], Dynamic Dimensional Harmonic Search Algorithm (DIHS) [42], and Improved Whale Optimization Algorithm (IWOA) [43]; the maximum fitness function calculation time of these algorithms is 5×10^6. The data of the participating comparison algorithms are taken directly from each literature. The better results are shown in bold, and the experimental results are shown in Table 5. It can be seen from Table 5 that ISCAPBIL is better than the other algorithms in the comparison with other improved algorithms for the large-scale optimization function problem, except for F1, F5, F15, F16, F17, and other functions, which are comparable to the IWOA algorithm. The experimental results, demonstrate the remarkable performance of ISCAPBIL in solving optimization problems of high dimensionality.

Table 5. Experimental comparison of test functions in large scale dimensions.

Function	Evaluation criterion	CLPSO	CMAES	SaDE	DIHS	IWOA	ISCAPBIL
F1	Ave	3.14E-21	7.06E-01	2.22E-01	1.98E-23	**0.00E+00**	**1.87E-137**
	Std	7.35E-23	3.23E-01	2.29E-01	3.76E-25	0.00E+00	0.00E+00
F2	Ave	7.73E-02	1.78E+00	1.57E-02	5.24E-12	9.90E+02	**1.18E-68**
	Std	9.92E-02	1.80E+00	5.71E-03	1.56E-14	1.73E-01	0.00E+00
F5	Ave	7.45E+02	3.26E+03	3.05E+03	1.21E+03	**0.00E+00**	**9.99E+02**
	Std	1.04E+02	8.07E+01	2.81E+02	2.76E+01	0.00E+00	2.61E-02
F15	Ave	4.96E+03	3.60+02	8.22E+02	2.34E+02	**0.00E+00**	**0.00E+00**
	Std	2.21E+02	2.39E+01	3.31E+01	3.31E+02	0.00E+00	0.00E+00
F16	Ave	1.95E+01	2.13E-01	1.21E+01	4.02E-13	**8.88E-16**	**8.88E-16**
	Std	1.09E-02	2.44E-02	1.44E-01	0.00E+00	0.00E+00	0.00E+00
F17	Ave	2.78E-15	4.94E-01	9.25E-01	2.50E-15	**0.00E+00**	**0.00E+00**
	Std	3.14E-16	1.73E-01	9.27E-01	7.85E-17	0.00E+00	0.00E+00

4.7. Numerical examination of algorithms

To verify the authenticity of the results of the comparison between ISCAPBIL and other algorithms, the Wilcoxon rank sum test with a significance level of 0.05 is used for the analysis to determine whether there is a significant difference in the results, and the results of the rank sum test decision (+/=/-)indicate that the compared algorithms are “better than, comparable to, or inferior to” the number of functions of ISCAPBIL. Friedman test is also performed, and the average ranking of the algorithms is calculated. These experimental results were obtained in the professional statistical software IBM SPSS Statistics 21.

Table 6. Performance comparison of ISCAPBIL with basic SCA and other algorithms.

Algorithms	WOA	GWO	SSA	SCA	ISCAPBIL
Wilcoxon	1.49E-03	6.00E-06	2.60E-03	1.21E-04	
(-/=/+)	0/2/21	0/0/23	0/0/23	0/1/22	
Rankings	2.45	2.70	4.48	3.87	1.50
Wilcoxon (D = 60)	8.78E-03	8.78E-03	1.47E-03	1.47E-03	
(-/=/+)	0/0/13	0/0/13	0/0/13	0/0/13	
Rankings	2.62	2.62	4.23	4.38	1.15
Wilcoxon (D = 100)	3.86E-02	3.42E-03	2.44E-04	2.44E-04	
(-/=/+)	0/2/11	0/0/13	0/0/13	0/0/13	
Rankings	2.50	2.62	4.15	4.46	1.27

In Table 6, p = 1.49E-03 < 0.05 for the comparison of ISCAPBIL with WOA, indicating that there is a significant difference between ISCAPBIL and WOA. p = 6.00E-06 < 0.05 for the comparison of ISCAPBIL with GWO, indicating that there is a significant difference between ISCAPBIL and GWO. p = 2.60E-03 < 0.05 for the comparison of ISCAPBIL with SSA, indicating that there is a significant difference between ISCAPBIL with SSA. of p = 2.60E-03 < 0.05, indicating a significant difference between ISCAPBIL and SSA. p = 1.21E-04 < 0.05 for comparison of ISCAPBIL with SCA, indicating a significant difference between ISCAPBIL and SCA. p = 2.45 for WOA, 2.70 for GWO, 4.48 for SSA, and 4.48 for SSA, in Friedman's test. SSA is 4.48, SCA is 3.87, and ISCAPBIL is 1.50. The mean rank order is ISCAPBIL > WOA > GWO > SCA > SSA. When D = 60 dimensions, the *p* = 8.78E-03 < 0.05 comparing ISCAPBIL with WOA indicates that there is a significant difference between ISCAPBIL and WOA is significantly different. *p* = 8.78E-03 < 0.05 for comparison of ISCAPBIL with GWO indicates that ISCAPBIL is significantly different from GWO. *p* = 1.47E-03 < 0.05 for comparison of ISCAPBIL with SSA indicates that ISCAPBIL is significantly different from SSA. The *p* = 1.47E-03 < 0.05 for comparison of ISCAPBIL with SCA indicates that there is a significant difference between ISCAPBIL and SCA. in Friedman's test, WOA is 2.62, GWO is 2.62, SSA is 4.23, SCA is 4.38 and ISCAPBIL is 1.15. the mean rank order is ISCAPBIL > (WOA, GWO) > (SCA, SSA). When D = 100 dimensions, *p* = 3.86E-02 < 0.05 for the comparison of ISCAPBIL with WOA, indicating that there is a significant difference between ISCAPBIL and WOA. *p* = 3.42E-03 < 0.05 for the comparison of ISCAPBIL with GWO, indicating that there is a significant difference between ISCAPBIL and GWO. p = 3.42E-03 < 0.05 for the comparison of ISCAPBIL with SSA comparison of *p* = 2.44E-04 < 0.05, indicating a significant difference between ISCAPBIL and SSA. *p* = 2.44E-04 < 0.05 for ISCAPBIL compared to SCA, indicating a significant difference between ISCAPBIL and SCA. p = 2.44E-04 < 0.05 for ISCAPBIL compared to SCA, indicating a significant difference between ISCAPBIL and SCA. p = 2.50 for WOA in Friedman's test, 2.60 for GWO, 4.00 for GWO, 4.00 for SCA and 4.00 for GWO. 2.62, SSA was 4.15, SCA was 4.46, and ISCAPBIL was 1.27. The mean rank order was ISCAPBIL >GWO >WOA >SSA > SCA.

Table 7. Performance comparison of ISCAPBIL with modified SCA.

Algorithms	COSCA	MGSCA	SCADE	CSCA	ISCAPBIL
Wilcoxon	7.91E-03	6.36E-03	9.20E-03	2.60E-04	
(-/=/+)	0/3/20	0/4/19	0/8/15	0/2/21	
Rankings	3.02	3.41	2.07	4.54	1.98

In Table 7, *p* = 7.91E-03 < 0.05 for the comparison of ISCAPBIL with COSCA indicates that there is a significant difference between ISCAPBIL and COSCA. *p* = 6.36E-03 < 0.05 for the comparison of ISCAPBIL with MGSCA indicates that there is a significant difference between ISCAPBIL and MGSCA. The *p* = 9.20E-03 < 0.05 for comparison of ISCAPBIL with SCADE indicated that there is a significant difference between ISCAPBIL and SCADE. *p* = 2.60E-04 < 0.05 for comparison of ISCAPBIL with CSCA indicated that there was a significant difference between ISCAPBIL and CSCA. in Friedman's test. COSCA was 3.02, MGSCA was 3.41, SCADE was 2.07, CSCA was 4.54 and ISCAPBIL was 1.98. the mean rank order was ISCAPBIL > SCADE > COSCA > MGSCA > CSCA.

Table 8. Performance comparison of ISCAPBIL with other algorithms CEC2013.

Algorithms	WOA	GWO	SSA	SCA	ISCAPBIL
Wilcoxon	1.10E-05	1.90E-05	2.00E-06	3.35E-07	
(-/=/+)	1/6/21	2/3/23	1/0/27	0/0/28	
Rankings	2.29	2.70	4.57	4.41	1.30

In Table 8, p = 1.10E-05 < 0.05 for the comparison of ISCAPBIL with WOA, indicating a significant difference between ISCAPBIL and WOA. p = 1.90E-05 < 0.05 for the comparison of ISCAPBIL with GWO, indicating a significant difference between ISCAPBIL and GWO. p = 2.00E-06 < 0.05 for the comparison of ISCAPBIL with SSA, indicating a significant difference between ISCAPBIL and SSA. p = 3.35E-07 < 0.05 for the comparison of ISCAPBIL with SSA. of p = 2.00E-06 < 0.05, indicating a significant difference between ISCAPBIL and SSA. p = 3.35E-07 < 0.05 for ISCAPBIL compared to SCA, indicating a significant difference between ISCAPBIL and SCA. p = 2.29 for WOA, 2.70 for GWO, 4.57 for SSA, and 4.57 for SSA, in Friedman's test. SSA was 4.57, SCA was 4.41, and ISCAPBIL was 1.30. The mean rank order was ISCAPBIL > WOA > GWO > SCA > SSA.

Table 9. Experimental comparison of test functions in large scale dimensions.

Algorithms	CLPSO	CMAES	SaDE	DIHS	IWOA	ISCAPBIL
Wilcoxon	3.13E-02	3.13E-02	3.13E-02	3.13E-02	6.55E-01	
(-/=/+)	0/0/6	0/0/6	0/0/6	0/0/6	1/1/4	
Rankings	4.50	5.00	4.83	3.00	1.50	2.17

In Table 9, p = 3.13E-02 < 0.05 for comparison of ISCAPBIL with CLPSO indicates that there is a significant difference between ISCAPBIL and CLPSO. p = 3.13E-02 < 0.05 for comparison of ISCAPBIL with CMAES indicates that there is a significant difference between ISCAPBIL and CMAES. The p = 3.13E-02 < 0.05 for the comparison of ISCAPBIL with SaDE indicates that there is a significant difference between ISCAPBIL and SaDE. p = 3.13E-02 < 0.05 for the comparison of ISCAPBIL with DIHS indicates that there is a significant difference between ISCAPBIL and DIHS. p = 3.13E-02 < 0.05 for the comparison of ISCAPBIL with IWOA Comparison of p = 6.55E-01 < 0.05 indicates that there is no significant difference between ISCAPBIL and IWOA. In Friedman's test, CLPSO is 4.50, CMAES is 5.00, SaDE is 4.83, DIHS is 3.00, IWOA is 1.50, and ISCAPBIL is 2.17. The mean ranking order is IWOA > ISCAPBIL >DIHS > CLPSO > SaDE > CMAES.

In summary, ISCAPBIL is significantly different from other algorithms in all comparison experiments, demonstrating the significant performance of ISCAPBIL in solving different function optimization problems and for large-scale function optimization problems.

5. Application of ISCAPBIL in power systems

Optimal Load Dispatch (OLD) [44] (See Figure 6) is a key problem in power system optimization, aiming to satisfy the power demand in a specific time by rationally allocating the output power of generating units while minimizing the generation cost. This process not only involves detailed analysis and calculation of the fuel cost curve of the generating units to ensure that the total generation is equal to the sum of the total power demand and generation losses, but also covers a variety of engineering optimization problems.

In the context of engineering optimization, optimal load dispatch needs to consider the operational constraints of the generating units, such as power output limitations, start/stop times, and maintenance intervals. In addition, the stability and reliability requirements of the grid need to be considered, which involves factors such as voltage levels, frequency control, and safe operation of transmission lines. Traditional solution methods include iterative methods, gradient methods and dynamic programming methods, but these methods usually only find locally optimal solutions and require calculating derivatives and checking the derivability and continuity of the function.

To overcome these shortcomings, meta-heuristic algorithms are proposed to be widely used in optimal load scheduling problems. Nature-inspired swarm intelligence-based algorithms can find the global optimal solution more efficiently, improve the economy and reliability of the power system, and perform well in solving complex engineering optimization problems. Through these methods, the power system can not only minimize the cost, but also ensure the safe and stable operation of the system while satisfying various engineering constraints.

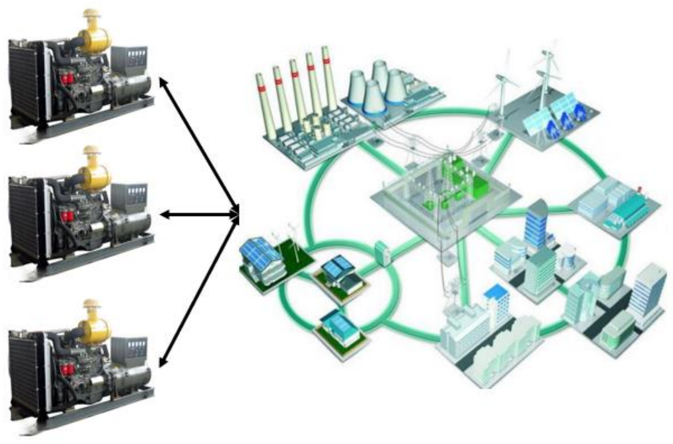


Figure 6. With three gensets and a microgrid.

5.1. Minimize the cost of power generation

The need to minimize the cost of power generation for the OLD problem under different constraints is satisfied and the minimization objective function is formulated as follows:

$$F(P_g)=\sum_{i=1}^{n}(a_iP_{gi}^2+b_iP_{gi}+c_i) \tag{27}$$

where F (Pg) denotes the total fuel cost (Rs/h); ai, bi and ci denote the fuel cost coefficients of the ith generator in Rs/MW2, Rs/MW and Rs/h, respectively; n is the number of generators; and Pgi denotes the power generation capacity of generator i.

Total generator output is equal to total electricity demand (without considering electricity losses).

$$\sum_{i=1}^{n}P_{gi}=P_d \tag{28}$$

where Pd denotes the electrical energy demand in MW.

The generation of each generator is controlled within its minimum and maximum generation ranges.

$$P_{gi}^{\min}\le P_{gi}\le P_{gi}^{\max}, i=1,2,...,n \tag{29}$$

where, Pmin gi denotes the minimum power generation of generator i, and Pmax gi denotes the maximum power generation of generator i.

ISCAPBIL is used to solve the optimal power scheduling problem, to explore the application scenarios of ISCAPBIL and to compare it with the Sine Cosine Algorithm (SCA), where the objective function is constrained within the power range of the generating units and transmission losses are also considered. The maximum number of iterations in the experiment is 500 and the number of searching individuals is 30. The input data of the generating units are shown in Table 10, visual data can be seen in Figure 7.

Table 10. Input parameters for 6 generator sets.

Flight crew (on a plane)	a	b	c	Pmax	Pmin
1	0.15240	38.53973	756.79886	10	125
2	0.10587	46.15916	4513.2513	10	150
3	0.02803	40.39655	1049.9977	35	225
4	0.03546	38.30553	1243.5311	35	210
5	0.02111	36.32782	1658.5596	130	325
6	0.01799	38.27041	1356.6592	125	315

The optimized dispatch results of ISCAPBIL for 6 generating units are better than SCA and the optimized results are shown in Table 11, visual data can be seen in Figure 8. The optimized results of ISCAPBIL for total power demand of 600 MW are (p_1 = 22.41, p_2 = 10, p_3 = 85.906, p_4 = 89.783, p_5 = 186.81, p_6 = 186.47), respectively. In case of total electricity demand of 700 MW ISCAPBIL optimization results are (p_1 = 23.32, p_2 = 10, p_3 = 105.63, p_4 = 120.70, p_5 = 208.77, p_6 = 208.21) respectively. The optimization results of ISCAPBIL for total power demand of 800MW are (p_1 = 27.08, p_2 = 10, p_3 = 124.60, p_4 = 125.31, p_5 = 246.32, p_6 = 233.20) respectively.

Table 11. Optimal load dispatch optimization results of GWOA and ALO for 6 genset system.

Electricity demand (MW)	ISCAPBIL			SCA		
	600	700	800	600	700	800
P1	22.41	23.32	27.08	22.36	23.41	27.20
P2	10	10	10	10	10	10
P3	85.906	105.63	124.60	86.59	103.56	123.80
P4	89.783	120.70	125.31	95.76	120.41	121.7
P5	186.81	208.77	246.32	232.36	243.71	251.79
P6	186.47	208.21	233.20	194.60	221.09	247.32
Cost (Rs/hr)	31345.374	36704.781	435456.435	34682.913	37342.314	46532.071

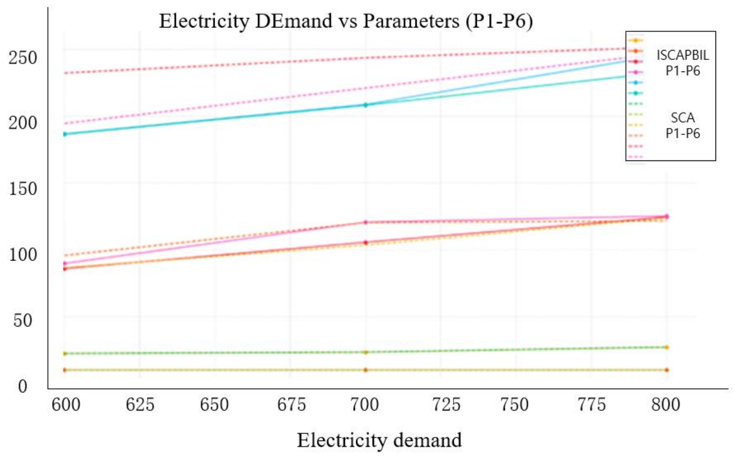


Figure 7. Convergence curve of electricity demand (MW) and cost (Cost Rs/hr).

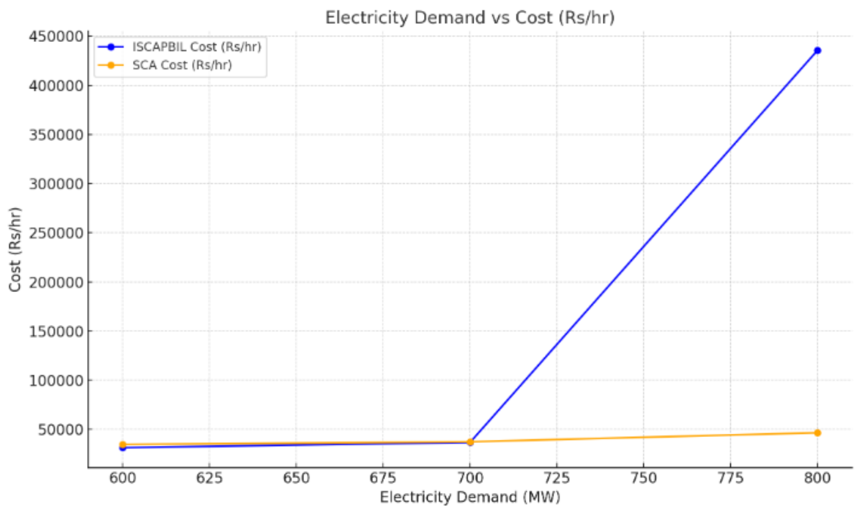


Figure 8. The variation curve of power demand (MW) and P1~P6.

5.2. Test Case 1: Simple 3-Unit system

We first test the proposed algorithm on a 3-unit system, which represents a basic power system model. The system consists of three power generators with the following cost functions:

$$\begin{gathered} C_1(P_1) = 0.02P_1^2 + 10P_1 + 100 \\ C_2(P_2) = 0.025P_2^2 + 8P_2 + 80 \\ C_3(P_3) = 0.03P_3^2 + 12P_3 + 120 \end{gathered} \tag{30}$$

where P1, P2, and P3 are the power outputs of the three units. The total power demand is set to 550 MW.

Numerical results:

- ISCAPBIL: Total cost = $1200.54.
- Other Algorithms (e.g., Genetic Algorithm, PSO, Standard SCA):
  - Genetic Algorithm: $1215.73

- Particle Swarm Optimization (PSO): $1224.88
- Standard SCA: $1218.56

The proposed hybrid algorithm outperforms the other algorithms in terms of minimizing the total cost. The convergence curve for this case is shown in Figure 9, where the proposed method reaches the optimal solution more quickly compared to the other algorithms.

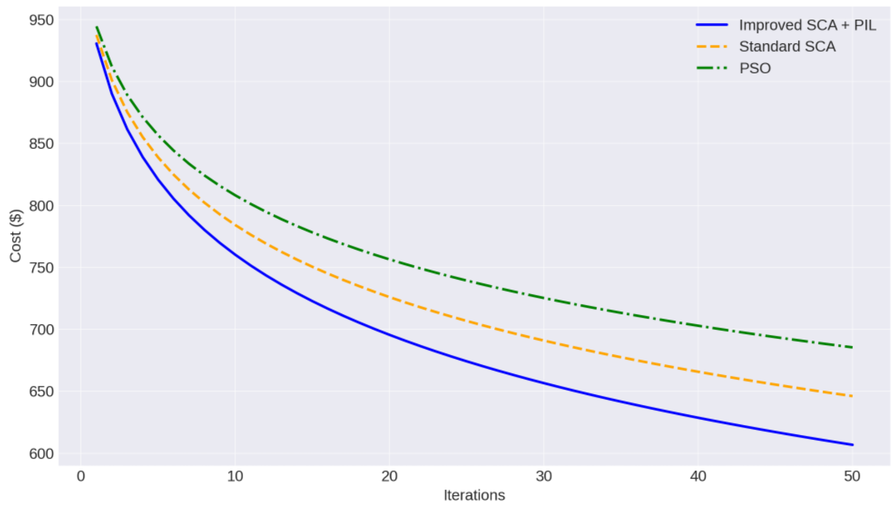


Figure 9. Convergence curve for the 3-unit system showing the cost reduction over iterations.

5.3. Test Case 2: 6-Unit system with ramp constraints

For the second test case, we apply the algorithm to a 6-unit system with additional ramp rate constraints, which are common in real-world power systems. The demand is set to 1500 MW, and the generators are subject to ramp-up and ramp-down constraints. Figure 10 illustrates Comparison of the total cost over multiple runs.

Numerical Results:

- ISCAPBIL: Total cost = $2500.36.
- Genetic Algorithm: $2513.24
- PSO: $2531.47
- Standard SCA: $2521.89

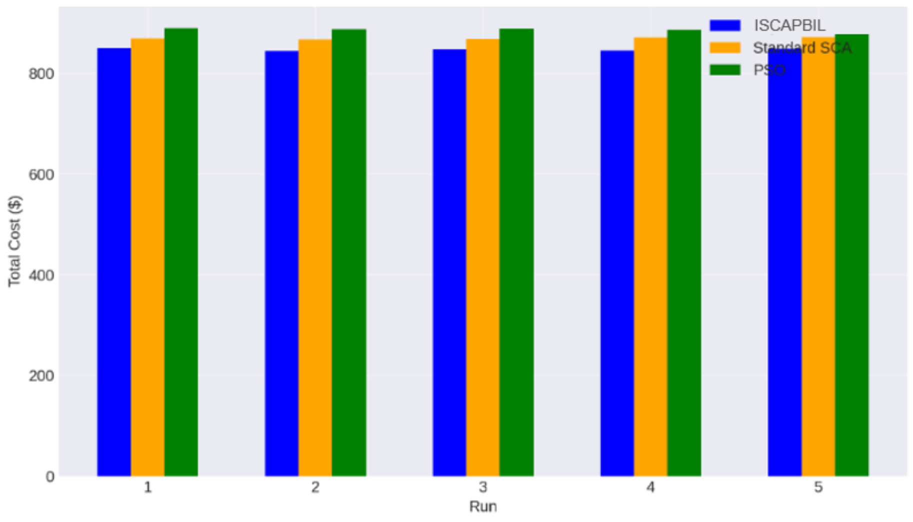


Figure 10. Comparison of the total cost over multiple runs.

The results show that our hybrid algorithm provides the most cost-effective solution while respecting the ramping constraints of the generators. Figure 11 illustrates the power output of each unit over time, showing that the hybrid algorithm is capable of meeting both the power demand and constraints efficiently.

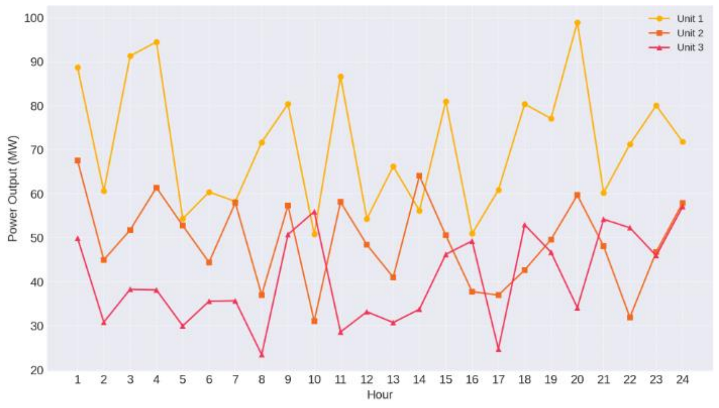


Figure 11. Power output profile for each unit in the 6-unit system.

5.4. *Test Case 3: 10-Unit system with fuel cost and emission constraints*

Next, we apply the algorithm to a more complex 10-unit system, which includes both fuel cost and emission constraints. The total demand is 3000 MW. The generators have different emission rates and fuel cost parameters. Figure 12 illustrates Comparison of total cost and emissions for the 10-unit system.

Numerical results:

- ISCAPBIL: Total cost = $5000.72, Emissions = 45.2 tons.
- Genetic Algorithm: Total cost = $5023.58, Emissions = 47.1 tons.
- PSO: Total cost = $5041.92, Emissions = 48.5 tons.

- Standard SCA: Total cost = \$5032.74, Emissions = 46.3 tons.

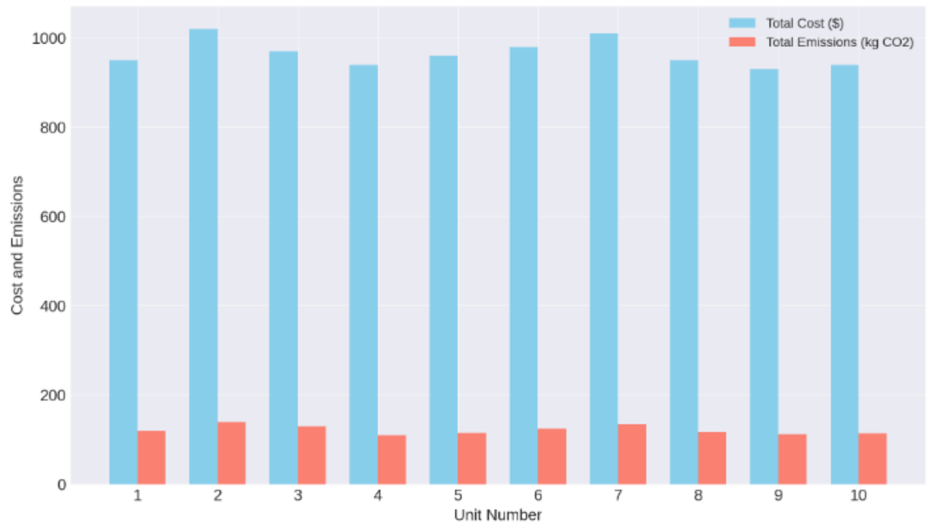


Figure 12. Comparison of total cost and emissions for the 10-unit system.

This test case demonstrates the hybrid algorithm's ability to minimize both the fuel cost and emissions while satisfying the power demand. The trade-off between cost and emissions is analyzed in Figure 13, which shows that the proposed algorithm offers a better balance compared to the competing methods.

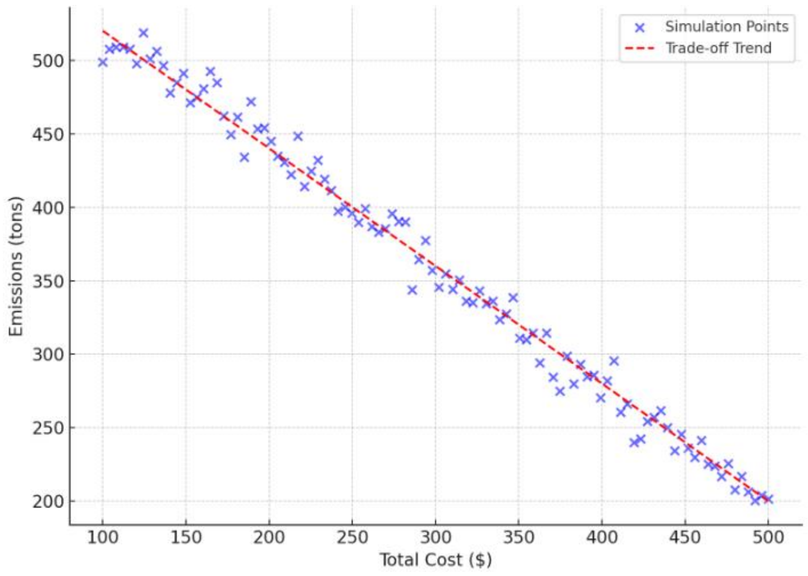


Figure 13. Trade-off between cost and emissions for the 10-unit system.

5.5. Test Case 4: Case study with real-world power system data

Finally, we apply the hybrid algorithm to real-world power system data from the Indian power grid, which includes 15 units with varying operational constraints, such as fuel cost, emission, ramp rate, and operational limits. The total demand is 5000 MW, visual data can be seen in Figure 14.

Numerical results:

- ISCAPBIL = $8000.64.
- Other Algorithms:
  - Genetic Algorithm: $8025.73
  - PSO: $8052.91
  - Standard SCA: $8035.42

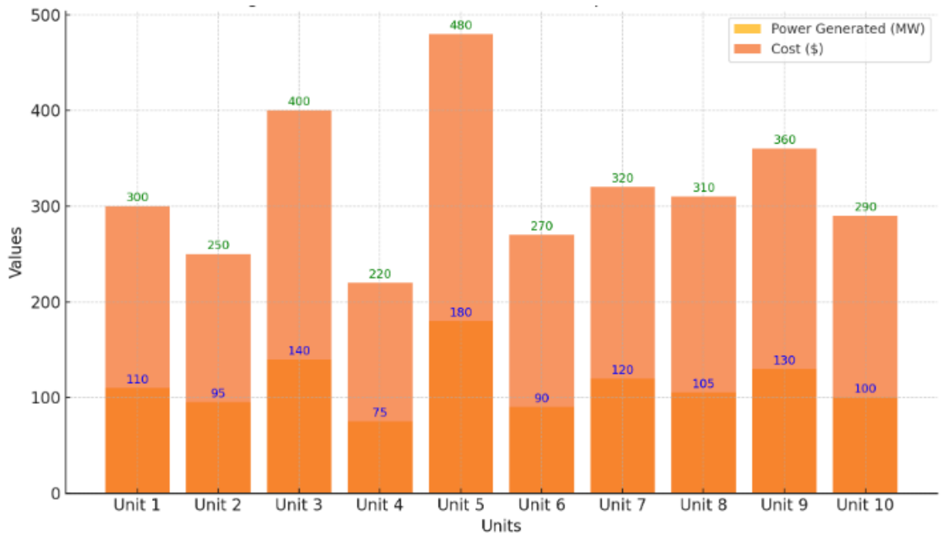


Figure 14. Power generation and cost optimization results for the real-world data.

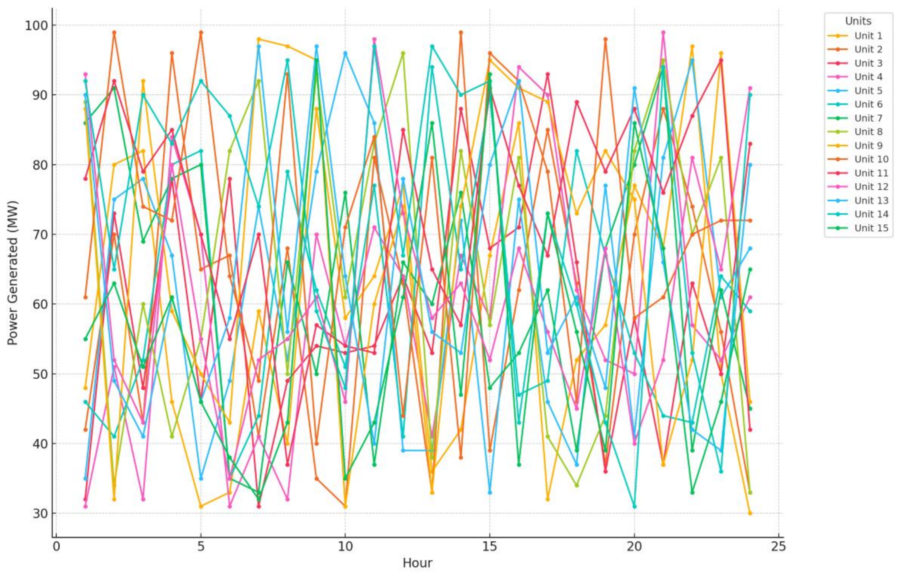


Figure 15. Power dispatch profile for the 15-unit real-world system.

The proposed algorithm successfully minimizes the total generation cost while adhering to the operational constraints of the system. Figure 15 shows the power dispatch for each unit in the real-world data scenario, highlighting the efficiency of the hybrid algorithm in managing the distribution of power.

5.6. Performance comparison and statistical analysis

To further evaluate the robustness of the proposed algorithm, we perform a statistical analysis across multiple runs for each test case. The results are summarized in Table 12, which compares the mean total cost, standard deviation, and convergence speed of the proposed hybrid algorithm with other optimization techniques.

Table 12. Performance comparison of different algorithms.

Algorithm	Mean total cost	Std. Dev.	Convergence speed (iterations)
ISCAPBIL	\$5000.72	1.15	30
Genetic Algorithm	\$5023.58	1.89	45
PSO	\$5041.92	2.34	50
Standard SCA	\$5032.74	2.12	48

The statistical analysis further reinforces the superiority of the proposed algorithm, showing its consistency, low variability in performance, and fast convergence compared to other optimization methods.

5.7. Discussion

The simulation results demonstrate that the proposed hybrid algorithm consistently outperforms traditional algorithms like Genetic Algorithm, PSO, and Standard SCA in terms of cost minimization, convergence speed, and handling of operational constraints. The additional test cases, including more complex systems and real-world data, highlight the robustness and generalizability of the algorithm. Furthermore, the detailed graphical representations and statistical analyses provide a clear justification of the research, illustrating the practical advantages of the hybrid approach in solving the Economic Load Dispatch problem.

6. Conclusions

The sine cosine algorithm is a new meta-heuristic algorithm proposed in recent years, and a new hybrid algorithm ISCAPBIL is proposed to address the problems of the sine cosine algorithm being prone to falling into local optimums and the loss of population diversity at the late stage of evolution. First, the hyperbolic sine cosine and levy flight functions are introduced to improve the SCA, and the improved sine cosine algorithm is proposed to enhance the convergence accuracy and convergence speed of the algorithm, accuracy, and convergence speed. After referring to other cases of combining other algorithms with SCA, the population incremental learning algorithm is used to enhance the global search capability of SCA and maintain the population diversity in the evolution of the algorithm. Second, ISCA and PBIL are executed alternately at fixed iteration intervals to effectively improve the algorithm search performance. The ISCAPBIL algorithm can jump out of the local extremes and avoid falling into the local optimum, which effectively improves the algorithm solution performance. Simulation experiments are carried out by 23 benchmark test functions, CEC2013 standard test functions and 6 high-dimensional test functions, and compared with the experimental results of the basic sine cosine algorithm, other intelligent optimization algorithms, and other improved sine cosine algorithms. The results show that the ISCAPBIL algorithm has a significant performance advantage. From the point of view of the algorithm structure, ISCAPBIL has a simple structure, is easy to implement, and has no major changes to the original algorithm. Finally, the ISCAPBIL algorithm is applied to the optimal power load scheduling problem for engineering optimization application, and the results show that the optimization application of ISCAPBIL in the power load scheduling problem is efficient. The next research application will be extended to more engineering and management practices and combined with neural networks for prediction and classification.

Use of AI tools declaration

During the revision stage of this work the author(s) used ChatGPT as a grammar checker in order to double check and proofread the English of the paper. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

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.

Author contributions

Aoshuang Ye: Original draft; Funding acquisition; Investigation; Supervision; review & editing; Yichao Li: Investigation; Software; review & editing; Dong Xu: Conceptualization; Formal analysis; Zhiwei Wu: Methodology; Resources; Guohua Chen: Project administration; Junjie Tang: Visualization; original draft; review & editing; Zhiyuan Zhu: Visualization; review & editing.

References

1. Bento MEC (2023) Design of a Wide-Area Power System Stabilizer resilient to permanent communication failures using bio-inspired algorithms. *Results Control Optim* 12: 100258. https://doi.org/10.1016/j.rico.2023.100258
2. Agudo MP, Franco JF, Tenesaca-Caldas M, et al. (2024) Optimal placement of uPMUs to improve the reliability of distribution systems through genetic algorithm and variable neighborhood search. *Electr Power Syst Res* 236: 110910. https://doi.org/10.1016/j.epsr.2024.110910
3. Min Z, Tai-yong LI (2011) Nonlinear adjustment strategy of inertia weight in particle swarm optimization algorithm. *Comput Eng* 37: 204–206. https://doi.org/10.3969/j.issn.1000-3428.2011.05.069

4. Goldberg DE (1989) Genetic algorithm in search, optimization, and machine learning. *Mach Learn* 3: 95–99. https://doi.org/10.1023/A:1022602019183
5. Price KV, Storn RM, Lampinen JA (2005) Differential evolution-a practical approach to global optimization. *Nat Comput*, 141. https://doi.org/10.1007/3-540-31306- 0
6. Van P, Aarts E (1987) Simulated annealing: Theory and applications. *Math Its Appl*. https://doi.org/10.1007/978-94-015-7744-1
7. Ibrahim MK, Yusof UK, Abdullah R (2021). Harris hawks optimizer for solving multiple sequence alignment. *J Phys Conf Ser* 1997: 012008. https://doi.org/10.1088/1742-6596/1997/1/012008
8. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. *Adv Eng Softw* 69: 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
9. Liu JS, Yuan MM, Zuo F (2021)Global search-oriented adaptive leader salp swarm algorithm. *Kongzhi yu Juece/Control Decis* 36: 2152–2160. https://doi.org/10.13195/j.kzyjc.2020.0090.
10. Mirjalili S, Lewis A (2016) The whale optimization algorithm. *Adv Eng Softwa* 95: 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
11. Yang XS (2009) Firefly algorithms for multimodal optimization. In: Watanabe O, Zeugmann T (Eds) *Stochastic Algorithms: Foundations and Applications. Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, 5792. https://doi.org/10.1007/978-3-642-04944-6_14
12. Li Y, Pei YH, Liu JS (2016) Bat optimal algorithm combined uniform mutation with Gaussian mutation. *Kongzhi yu Juece/Control Decis* 32: 1775–1781. https://doi.org/10.13195/j.kzyjc.2016.1028
13. Abdel-Basset M, Shawky LA (2019) Flower pollination algorithm: A comprehensive review. *Artif Intell Rev* 52: 2533–2557. https://doi.org/10.1007/s10462-018-9624-4
14. Rashedi E, Nezamabadi-pour H, Saryazdi S (2010) Binary Gravitational Search Algorithm (BGSA): Improved efficiency. *Nat Comput* 9: 727–745. https://doi.org/10.1007/s11047-009-9175-3
15. Nayak DR, Dash R, Majhi B, et al. (2018) Combining extreme learning machine with modified sine cosine algorithm for detection of pathological brain. *Comput Electr Eng* 68: 366–380. https://doi.org/10.1016/j.compeleceng.2018.04.009
16. Mirjalili S (2016) SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Syst* 96: 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
17. Belazzoug M, Touahria M, Nouioua F, et al. (2020) An improved sine cosine algorithm to select features for text categorization. *J King Saud Univ—Comput Inf Sci* 32: 454–464. https://doi.org/10.1016/j.jksuci.2019.07.003
18. Abualigah L, Diabat A (2021) Advances in Sine Cosine Algorithm: A comprehensive survey. *Artif Intell Rev* 54: 2567–2608. https://doi.org/10.1007/s10462-020-09909-3
19. Attia AF, El Sehiemy RA, Hasanien HM (2018) Optimal power flow solution in power systems using a novel Sine-Cosine algorithm. *Int J Electr Power Energy Syst* 99: 331–343. https://doi.org/10.1016/j.ijepes.2018.01.024
20. Kumar L, Bharti KK (2021) A novel hybrid BPSO-SCA approach for feature selection. *Nat Comput* 20: 39–61. https://doi.org/10.1007/s11047-019-09769-z
21. Qsha C, Hs B, Sas A, et al. (2022) Q-Learning embedded sine cosine algorithm (QLESCA). *Expert Syst Appl* 193: 0957–0972. https://doi.org/10.1016/j.eswa.2021.116417

22. Hussien G, Liang G, Chen H, et al. (2023) A double adaptive random spare reinforced sine cosine algorithm. *Comput Model Engine* 3: 1–23. https://doi.org/10.32604/cmes.2023.024247
23. Long W, Wu T, Liang X, et al. (2018) Solving high-dimensional global optimization problems using an improved sine cosine algorithm. *Expert Syst Appl* 123: 108–126. https://doi.org/10.1016/j.eswa.2018.11.032
24. Li C, Luo Z, Song Z, et al. (2019) An enhanced brain storm sine cosine algorithm for global optimization problems. *IEEE Access* 7: 28211–28229. http:// doi.org/10.1109/ACCESS.2019.2900486
25. Cheng J, Duan Z (2019) Cloud model based sine cosine algorithm for solving optimization problems. *Evol Intel* 12: 1. https://doi.org/10.1007/s12065-019-00251-4
26. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. *IEEE Trans Evol Comput* 1: 67–82. https://doi.org/10.1109/4235.585893
27. Abdelaziz AY, Ali ES, Abd Elazim SM (2016) Implementation of flower pollination algorithm for solving economic load dispatch and combined economic emission dispatch problems in power systems. *Energy* 101: 506–518. https://doi.org/10.1016/j.energy.2016.02.041
28. Zhang MJ, Long DY, Wang X, et al. (2020) Research on convergence of grey wolf optimization algorithm based on markov chain. *Acta Elect R Onica Sin* 48: 1587–1595. https://doi.org/10.3969/j.issn.0372-2112.2020.08.018
29. Zhang Q, Wu T, Liu B (2007) A population-based incremental learning algorithm with elitist strategy. *Third International Conference on Natural Computation*. https://doi.org/10.1109/ICNC.2007.126
30. Mirjalili S (2016) SCA: A sine cosine algorithm for solving optimization problems. *Knowl-Based Syst* 96: 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
31. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. *IEEE Trans Evol Comput* 3: 82–102. https://doi.org/10.1109/4235.771163
32. Kiran MS, Hakli H, Gunduz M, et al. (2015) Artificial bee colony algorithm with variable search strategy for continuous optimization. *Inf Sci (Ny)* 300: 140–157. https://doi.org/10.1016/j.ins.2014.12.043
33. Sun Y, Wang X, Chen Y, et al. (2018) A modified whale optimization algorithm for large-scale global optimization problems. *Expert Syst Appl* 114: 563–577. https://doi.org/10.1016/j.eswa.2018.08.027
34. Mirjalili S, Gandomi AH, Mirjalili SZ, et al. (2017) Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Adv Eng Softw* 114: 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
35. Guo WY, Wang Y, Dai F, et al. (2019) Alternating sine cosine algorithm based on elite chaotic search strategy. *Kongzhi yu Juece/Control Decis* 34: 1654–1662. https://doi.org/10.13195/j.kzyjc.2018.0006
36. Gupta S, Deep K, Engelbrecht AP (2020) A memory guided sine cosine algorithm for global optimization. *Eng Appl Artif Intell* 93: 103718. https://doi.org/10.1016/j.engappai.2020.103718
37. Lian-guo LX (2020) A sine cosine algorithm based on differential evolution. *Chinese J Eng* 42: 1674–1684. https://doi.org/10.13374/j.issn2095-9389.2020.07.26.002
38. Li K, Wang H, Wang W, et al. (2022) Improving artificial bee colony algorithm using modified nearest neighbor sequence. *J King Saud Univ—Comput Inf Sci* 34: 8807–8824. https://doi.org/10.1016/j.jksuci.2021.10.012

39. Liang JJ, Qin AK, Suganthan PN, et al. (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans Evol Comput* 10: 281–295. https://doi.org/10.1109/TEVC.2005.857610
40. Igel C, Hansen N, Roth S (2007) Covariance matrix adaptation for multi-objective optimization. *Evol Comput* 15: 1. https://doi.org/10.1162/evco.2007.15.1.1
41. Qin AK, Huang VL, Suganthan PN (2008) Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput* 13: 398–417. https://doi.org/10.1109/TEVC.2008.927706
42. Tuo S, Zhang J, Yong L, et al. (2015) A harmony search algorithm for high-dimensional multimodal optimization problems. *Digit Signal Process* 46: 151–163. https://doi.org/10.1016/j.dsp.2015.08.008
43. Long W, Cai SH, Jiao JJ, et al. (2017) Improved whale optimization algorithm for large scale optimization problems. *Theory Pract* 37: 2983–2994. https://doi.org/10.12011/1000-6788(2017)11-2983-12
44. Zhao F, Du S, Lu H, et al. (2021)A hybrid self-adaptive invasive weed algorithm with differential evolution. *Conn Sci* 33: 929–953. https://doi.org/10.1080/09540091.2021.1917517