Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105
Volume **10**, Number **3**, August **2025**

Multi-population Phasmatodea Population Evolution Algorithm Based on Dynamic Communication

Chen Zhong[1], Cheng-Li Luo[1], Lu-Lu Liang[2], Shu-Chuan Chu[1,3]
Jia-Hong Zhu[1], Peng-Bin Luo[1], Jeng-Shyang Pan[3,4,*]

[1]School of Information Engineering, Yango University, Fuzhou 350015, China
czhong@ygu.edu.cn clluo@ygu.edu.cn 2038189627@qq.com 2329629219@qq.com

[2]College of Computer Science and Engineering, University of Science and Technology, Qingdao 266590, China
1142403625@qq.com

[3]School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing 210044, China
scchu0803@gmail.com

[4]Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan
jspan@ieee.org

*Corresponding author: Jeng-Shyang Pan

Received March 7, 2025, revised May 26, 2025, accepted June 30, 2025.

Abstract. *The Phasmatodea Population Evolution (PPE) algorithm is an efficient, easy- to- implement, and highly scalable metaheuristic algorithm. It conducts bionic experiments by simulating the evolutionary process of phasmatodea populations and obtains the optimal solution to the problem after multiple iterations of calculations. Like most metaheuristic algorithms, the PPE algorithm suffers from issues such as low convergence accuracy and susceptibility to local optima. To address these issues, this paper proposes a multi-population PPE algorithm based on dynamic communication. Firstly, during the initialization phase, the population is divided into multiple subpopulations based on the population size, and a communication period is set according to the characteristics of the dynamic communication strategy. Secondly, iteration rules are established, and each sub- population iterates independently when the communication conditions are not met. Thirdly, during the population update phase, population communication is conducted based on the dynamic communication strategy to enhance population diversity and avoid the algorithm falling into local optima. Finally, a dynamic adjustment strategy for population size is introduced into the algorithm to further improve its optimization capability. To verify the performance of the proposed algorithm, simulation experiments are conducted using the CEC 2014 test function set and two engineering design problems. The experimental results demonstrate that the proposed algorithm exhibits superior convergence performance compared to the comparison algorithms on the CEC 2014 test function set and the two engineering design problems. The results show that the algorithm has significant advantages on most functions.*
Keywords: Phasmatodea population evolution, Metaheuristic algorithm, Dynamic communication, Engineering design problem

1. **Introduction.** Optimization problems generally refer to finding the most reasonable and feasible solu- tion or a set of solutions under a given set of constraints, in order to achieve the best possible outcome. Common examples include the Traveling Salesman Problem [1], Path Planning Problem [2], and Combinatorial Optimization Problem [3]. With the improvement of computer processing capabilities, optimization problems have

received increasing attention and are widely present in various fields such as economics, engineering, logistics, and many others. In the study of optimization, methods used to solve optimization problems are commonly known as optimization methods. The correct use of optimization methods can accelerate the generation of solutions and improve their quality.

Traditional optimization methods mainly include gradient descent, Newton's method, and the least squares method [4], which can be used to solve different optimization problems. Gradient descent is suitable for solving unconstrained optimization problems, but it is prone to getting stuck in local optimal solutions, and the choice of the initial point has a significant impact on the final result. Com- pared to gradient descent, Newton's method has a faster convergence rate, but it requires second-derivative information. Therefore, it cannot be directly applied to non- smooth objective functions or problems lacking second-derivative information. The least squares method is a specific optimization method suitable for linear regression problems, but it is sensitive to outliers. When a data point deviates from other data points, this method can significantly affect the fitting result. The above algorithms have high requirements for the objective function of the problem to be solved, such as being low-dimensional, linear, and differentiable. When the objective function is nonlinear or non-differentiable, traditional methods are difficult to apply. Therefore, new optimization methods need to be explored to solve more complex optimization problems.

Metaheuristic algorithms typically employ non-deterministic search strategies, introducing random variables during the iterative process to enhance the robust- ness and adaptability of the algorithm [5]. The design process of these algorithms is often inspired by human intelligence, biological populations, and natural phenomena. Researchers construct corresponding mathematical models based on the relevant char- acteristics of these sources of inspiration and design iterative search processes based on these models [6, 7]. In this way, metaheuristic algorithms can simulate and utilize natural optimization mechanisms, providing powerful tools for solving complex problems. Compared with traditional optimization methods, metaheuristic algorithms are easier to understand, highly flexible, and can effectively solve non-differentiable and nonlinear problems. The solution process is independent of problem characteristics. Therefore, they have gradually become the primary choice for many researchers to solve various optimization problems. In recent years, there has been increasing research on metaheuristic algorithms, and different types of metaheuristic algorithms have been proposed, such as Genetic Algorithm (GA) [8], Gray Wolf Optimizer (GWO) [9], Ant Colony Optimization (ACO) [10], Quasi-affine Transformation Evolution (QUATRE) [11, 12], Dragonfly Algorithm (DA) [13], Phasmatodea Population Evolution (PPE) [14], Harmony Search (HS) [15], Moth-flame Optimization (MFO) [16], and Gradient Based Optimizer (GBO) [17].

The operational efficacy of metaheuristic algorithms is fundamentally contingent upon their capacity to maintain equilibrium between exploration and exploitation dynamics. All stochastic optimization methods remain paradoxically constrained by inherent randomness during solution space traversal, presenting persistent challenges in balancing these dual optimization imperatives. Excessive exploitation bias during iterative search processes induces premature convergence through incomplete solution space coverage, systematically increasing susceptibility to local optima entrapment. Conversely, if there is too much exploration and too little exploitation, deep exploitation of a certain area cannot be performed, resulting in neglecting the global optimal solution and reducing the convergence accuracy of the algorithm. In this case, the research directions of metaheuristic algorithms mainly involve improving existing algorithms and proposing new optimization algorithms. However, the emergence of new metaheuristic algorithms has slowed down

in recent years. This is because researchers have begun to deeply recognize that new metaheuristic algorithms perform similarly to or even worse than the original algorithms in solving complex problems. This realization has prompted researchers to focus more on deeply optimizing and improving existing algorithms [18, 19].

2. Related Work.

2.1. PPE Algorithm.

The PPE algorithm primarily simulates the evolutionary process of stick insect populations in nature. The uniqueness of this algorithm lies in its abandonment of the individual concept prevalent in most metaheuristic algorithms, instead focusing on the population as the core unit for updating and optimization. This shift enhances the autonomous decision-making capabilities of the population and makes the algorithm more flexible and adaptable when solving complex problems [20, 21, 22, 23]. Influenced by various factors, the evolutionary trends of stick insects need to consider multi-ple aspects. Based on the convergent evolution characteristics of stick insects during their evolutionary process, the stick insect population (particles) will move towards the most excellent population closest to that region, rather than the globally optimal population. During movement, the population is also influenced by previous movement trends and its own attributes. If a better position is achieved in the previous evolutionary process, it will result in path dependence, maintaining that evolutionary trend. Population size fundamentally governs evolutionary dynamics in metaheuristic systems. Suboptimal population sizes constrain evolutionary trajectory diversification, primarily manifesting as quantitative adjustments without substantive directional adaptation. Conversely, expanded populations exhibit heightened mutational propensity and spatial exploration capacity, probabilistically enhancing solution space coverage. Furthermore, inter-population competitive interactions and geographical environment factors collectively modulate evolutionary pressures through complex adaptive feedback mechanisms.

The PPE algorithm is proposed based on the population growth model and competition model, mainly consisting of four parts: initialization, position updating, evolutionary trend updating, and population size updating. The following is a detailed introduction to these four parts.

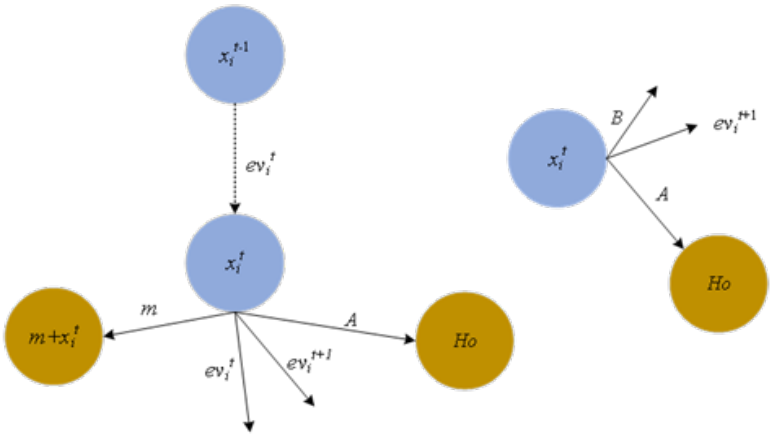


Figure 1. Two evolutionary methods of PPE algorithm

(1) Initialization: Since the algorithm treats each solution as a population, it is necessary to initialize the size of each population using the Equation (1). Additionally, the number of historical optimal solutions needs to be initialized, and the updating formula is shown in Equation (2).

$$p_i = N_p, \tag{1}$$

$$k = \lfloor \log(N_p) \rfloor + 1, \tag{2}$$

(2) Position Updating: Change the current position of the population based on its evolutionary trend, denoted as "ev". The population position update formula is Equation (3).

$$x^{t+1} = x^t + ev, \tag{3}$$

(3) Evolutionary Trend Updating: The updating of this part is divided into three modes. When a population obtains a better fitness value after moving, the updating formula for the evolutionary trend is shown in Equation (4). When the fitness value of a population is poorer after moving, it evolves according to Equation (5). When competition occurs between populations, the updating formula for the evolutionary trend is shown in Equation (6). The evolution examples for Equation (4) and Equation (5) are illustrated in Figure 1.

$$ev^{t+1} = (1 - p^{t+1})A + p^{t+1}(ev^t + m), \tag{4}$$

$$ev^{t+1} = rand \times A + s^t \times B, \tag{5}$$

$$ev^{t+1} = ev^{t+1} + \frac{f(x_i) - f(x_j)}{f(x_i)}(x_i - x_j), \tag{6}$$

(4) Population Size Updating: Population size updating can be divided into updating under competition and updating without competition. When competition is not considered among populations, the population size updating formula is shown in Equation (7). When competition occurs between populations, the population size updating formula is shown in Equation (8).

$$p^{t+1} = a^{t+1}p^t(1 - p^t), \tag{7}$$

$$p_i = p_i + a_i p_i(1 - p_i - \frac{f(x_j)}{f(x_i)}p_j), \text{ if } d(x_i, x_j) < G, \tag{8}$$

2.2. **Multi-population Strategy.** The core idea of the multi-population strategy is to use multiple populations or groups to jointly search for the optimal solution to a problem during the execution of an algorithm. Each group can be regarded as an independent search unit in the algorithm, independently exploring and developing in the search space. The multi-population approach enhances population diversity and individual communication and collabo- ration capabilities, enabling the algorithm to cover more areas when searching for the optimal solution, improving its global search capability, and thus avoiding falling into local optima [24, 25, 26]. The multi-population strategy emphasizes communication and collaboration between populations. By regularly or irregularly exchanging information, sharing excellent solutions, or performing crossover operations, different subpopulations can achieve collaborative evolution. Additionally, the dynamic adjustment of population size is often used in multi-population strategies to adapt to algorithm characteristics, increase population diversity, and further improve algorithm performance. This section will provide a detailed introduction to the multi- population strategy, starting with population communication methods and dynamic adjustment of population size.

Communication Methods: The population is divided into multiple subpopulations, each of which can evolve independently and enhance algorithm performance through internal or inter- subpopulation communication. Based on different communication methods, Lalwani et al. [24] classified communication strategies into four modes: star model, migration model, diffusion model, and hybrid model. The star model involves communication between one master subpopulation and multiple slave subpopulations, with no communication between slave subpopulations. The migration model involves unidirectional circular communication among all subpopulations, where particles from the first subpopulation migrate to the second, particles from the second migrate to the third, and so on. These two methods are easy to implement but have low communication efficiency, which may affect algorithm performance. The diffusion model involves communication between all subpopulations, where each subpopulation can communicate with any other subpopulation. This method can quickly disseminate information but takes longer than the first two methods. The hybrid model constructs a more com- plex network topology by combining the star, diffusion, and migration models. This approach can effectively avoid the drawbacks of each model but is more difficult to establish a reasonable model.

Adjusting Population Size: A reasonable population size can increase population diversity, improve the algorithm's convergence speed, and keep the algorithm's running time within a reasonable range. Dynamic adjustment of population size is a commonly used method for adjusting population size, which mainly includes increasing and decreasing the population size. The increase or decrease in population size depends on the algorithm's characteristics, and the relationship between the optimal solutions in the current iteration and the previous iteration is a commonly used judgment basis. If the solution obtained in the current iteration is better than the previous one, it may mean that the current optimal solution is still far from the global optimal solution. In this case, the population size should be appropriately increased to expand the search scope and enhance global search capability. If the optimal solutions in the last few iterations are similar, it means that the current optimal solution is close to the global optimal solution of the problem. In this case, the population size should be reduced to decrease the computational load and accelerate convergence. When adjusting the population size, the principle of "slow growth and rapid reduction" should also be followed. Slow growth ensures that the population size does not increase sharply, thereby ensuring the stability of the algorithm's search and preventing the effect from being affected by excessive changes in population size. Rapid reduction can effectively shorten the algorithm's running time and improve its efficiency.

In summary, the multi-population strategy significantly enhances the algorithm's global search capability and convergence performance by utilizing multiple subpopulations to collaborate, enhancing population diversity and individual communication and collaboration capabilities, and dynamically adjusting the population size.

3. **The Proposed Algorithm.**

3.1. **Dynamic Communication.**

The MPPE algorithm divides the entire population into n sub- populations (sub- groups), each with a size of Np/n. Before the algorithm iteration begins, each subgroup possesses the same resources, including evolutionary trends, growth rates, and search space sizes. As the iterations proceed, the degree of resource consumption varies among the subgroups, necessitating communication between them to enhance population diversity and subsequently improve the algorithm's convergence accuracy. The algorithm proposes four dynamic communication strategies to enhance its performance: the optimal communication strategy, the maximum communication strategy, the neighbor communication strategy, and the parent-child communication strategy. The

optimal communication strategy emphasizes the dissemination of optimal solutions, the maxi- mum communication strategy focuses on extensive communication between sub- groups, the neighbor communication strategy attends to local communication between sub- groups, and the parent-child communication strategy adjusts the size of subgroups. Figure 2 and Figure 3 illustrate the communication methods of these four strategies.

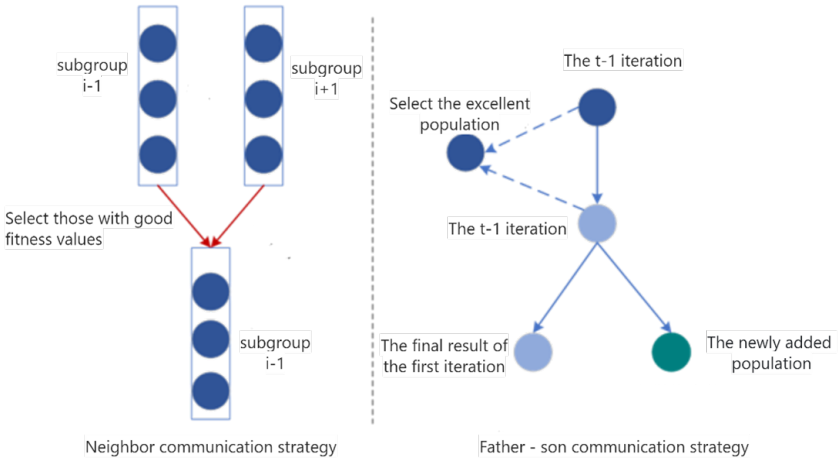


Figure 2. Example diagram of optimal communication strategy and maximum communication strategy

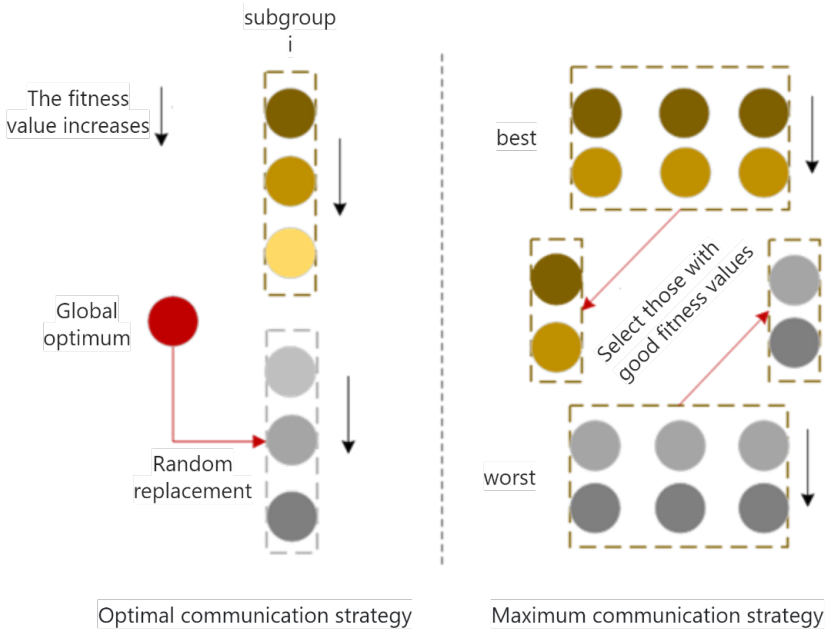


Figure 3. Example diagram of neighbor communication strategy and parent-child communication strategy

Optimal Communication Strategy: When the algorithm iterates to a multiple of the communication cycle, the currently optimal candidate solutions from all populations move into each subgroup and compete for resources within those subgroups. Since the resources of each subgroup are fixed, some candidate solutions within each subgroup will be elimi- nated. The elimination mechanism is as follows: Evaluate the fitness function of the pop- ulations in each subgroup, sort the populations in ascending order based on their fitness

values, and randomly eliminate one population from the bottom 50% of each subgroup. The currently optimal candidate solutions from all populations are also the optimal solutions of a certain subgroup. Therefore, optimal communication is an exchange between one subgroup and other subgroups, belonging to inter-group communication.

Maximum Communication Strategy: When the algorithm iterates to a multiple of the communication cycle, two better candidate solutions are randomly selected from each subgroup, and their fitness values are stored in "best". Two poorer populations are also randomly selected, and their fitness values are stored in "worst". Both "best" and "worst" are matrices with 2 rows. Subsequently, the populations in "best" and "worst" are compared in sequence. The maximum communication method is as follows: The first subgroup attempts to communicate with the remaining n-1 subgroups sequentially. When the "best" of a certain subgroup is better than the "best" of the first subgroup, the "best" of the first subgroup is replaced by the "best" of that sub-group, completing the "best" communication. When the "worst" of a certain subgroup is better than the "worst" of the first subgroup, the "worst" of the first subgroup is replaced by the "worst" of that subgroup, completing the "worst" communication. At this point, the first subgroup completes its communication and no longer compares with uncompared subgroups. The second subgroup attempts to communicate with the remaining n-2 subgroups sequentially, using the same communication method as the first subgroup. The third subgroup attempts to communicate with the remaining n-3 subgroups sequentially, and so on. The maximum communication strategy is an exchange between various subgroups, belonging to inter-group communication.

Neighbor Communication Strategy: When the algorithm iterates to a multiple of the communication cycle, each subgroup is arranged in a circular manner. The neighbors of the first subgroup are the second subgroup and the nth subgroup, the neighbors of the second subgroup are the first subgroup and the third subgroup, the neighbors of the third subgroup are the second subgroup and the fourth subgroup, and so on. Each subgroup will have local communication with its neighbors. The neighbor communication method is as follows: First, compare the k historical optimal candidate solutions of two neighbors of a certain subgroup, then select the better k candidate solutions to form a candidate solution set, and finally randomly replace some of the k historical optimal candidate solutions of the current subgroup with the candidate solution set. Neighbor communication is an exchange between a subgroup and its nearby subgroups, belonging to inter-group communication.

Parent-Child Communication Strategy: This strategy is employed during population evolution when the algorithm iterates to a multiple of the communication cycle. Different from the aforementioned strategies, the current subgroup does not communicate with other subgroups but instead communicates with its parent generation. Parent-child communication represents inter-generational communication within the population and is classified as inter- group communication. The parent-child communication method is as follows: Firstly, a matrix history is used to store the fitness values up to the (i-1)th iteration of the algorithm, and a matrix current is used to store the fitness values at the ith iteration. Then, the fitness values from the (i-1)th and ith iterations are compared, and the better-performing population is selected as the cur- rent population. Finally, the number of subgroups is dynamically adjusted based on the origin of the current population. If the current population is the offspring generation, it indicates that the current evolutionary trend is no longer applicable, and the evolutionary trend should be changed. Additionally, the number of current subgroups should be increased to enhance their search capability. If the current population is the parent generation, it suggests that the distance to the global optimal solution is relatively close. If there are many subgroups at this

time, the number of subgroups can be reduced by half. The way to increase the number of subgroups is to add 1/10 of the current subgroup count each time. If the subgroup count reaches double the initial count, the subgroup count will be quickly reduced to the initial count.

3.2. **The Execution Flow of the Algorithm.** Based on the description of the dynamic communication strategy, this section will provide a detailed explanation of the MPPE algorithm. Through performance analysis and experimental verification of the PPE algorithm, the number of subgroups n in the MPPE algorithm is set to 4, and the population size Np is set to 100. When the communication strategy conditions are not met, each subgroup operates independently without interference from others. When the communication conditions are met, information is exchanged between subgroups or within subgroups according to the communication strategy. The following is a detailed description of the execution steps of the MPPE algorithm.

- Initialize relevant parameters such as the maximum number of iterations, population size, number of subgroups, and communication cycle.
- Calculate the population size and historical optimal solutions for each subgroup.
- Initialize the positions of all populations in the search space, population growth rates, mutation parameters, and evolutionary trends.
- Update the positions and evolutionary trends of the populations, and calculate their fitness values.
- Update the positions and evolutionary trends based on the excellence of the populations before and after the update. If the updated population is better, its position becomes the current population position, and Equation (4) and Equation (7) are used to update the evolutionary trend and population size. If the population before the update is better, Equation (5) is used to update the evolutionary trend, and the updated population is still selected as the current population. If the population size is less than a certain random number at this time, Equation (7) is used to update the population size.
- Randomly select two different populations and determine if the distance between them is less than a fixed value. If it is less than the fixed value, it indicates that competition will occur between the two populations. At this time, Equation (7) and Equation (6) are used to update the population size and evolutionary trend, respectively.
- Perform dynamic communication. If the conditions are met, execute dynamic communication; otherwise, proceed to the next step.
- Determine if the iteration conditions are met. If they are met, proceed to the next step; otherwise, return to Step 4.
- Output the current population position (global optimal solution) and its fitness value, and the algorithm ends.

Algorithm 1 presents the pseudocode for the MPPE algorithm. Among the four communication strategies, the optimal communication strategy is the simplest, with a communication cycle set to 10; the maximum communication strategy involves communication with all subgroups except itself, has a high time complexity, and a communication cycle set to 50; the neighbor communication strategy has a single mode of communication and a communication cycle set to 30; the parent-child communication strategy may require updating the evolutionary trend, with a communication cycle set to 35. Therefore, multiple communication strategies may be used during the communication process. For example, when the iteration number is 100, the optimal communication strategy and the maximum

communication strategy are executed together, and the optimal communication strategy will always be executed when the neighbor communication strategy is executed.

3.3. **Computational Complexity and Convergence Analysis.** The time complexity analysis of the MPPE algorithm indicates that the optimal communication, maximal communication, and neighbor communication strategies each have a complexity of $O(n \cdot N_p)$ per communication cycle, where n is the number of subpopulations and N_p is the population size. Specifically, optimal communication involves the propagation and competition of global optimal solutions, maximal communication requires pairwise comparisons across subpopulations, while neighbor communication is confined to local interactions among adjacent subpopulations. The parent-child communication strategy incurs slightly higher overhead due to historical data storage (space complexity $O(n \cdot N_p)$) and dynamic adjustment of the number of subpopulations (with a worst-case time complexity of $O(n \cdot N_p \cdot \log N_p)$).

In terms of convergence, theoretical analysis demonstrates that the algorithm ensures convergence through the synergistic effects of the four strategies: optimal communication accelerates the diffusion of elite solutions, maximal communication enhances global exploration capability, neighbor communication maintains local exploitation efficiency, and parent-child communication balances the search scope by adaptively adjusting subpopulation sizes.

4. **Experiment.** In this section, to verify the effectiveness of the MPPE algorithm, it is tested on 30 benchmark functions along with the PSO [25, 27], WOA [28, 29], SSA [30], Food Digestion Algorithm (FDA) [31], BOA [32], and PPE algorithms. By comparing the average fitness values of the six algorithms, the optimization ability of the MPPE algorithm is systematically demonstrated. By comparing the convergence curves of the fitness values, the convergence performance of the MPPE algorithm is more intuitively presented. The stability of the MPPE algorithm is demonstrated through the standard deviation of the optimal fitness values. This section will describe the experimental parameter settings, optimization accuracy analysis, convergence ability analysis, and stability analysis.

4.1. **Parameter Description.** All algorithms were tested using the Windows 10 Education operating system, an Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz processor, and MATLAB R2020b as the experimental environment. The parameter settings for the MAPPE algorithm and the comparison algorithms are shown in Table 1. To ensure fairness in the experiment, the population size for all algorithms was set to 100, the number of iterations was set to 1000, and the algorithms were run independently 30 times to obtain the average value as the final result. This paper selects 30 benchmark functions from the CEC 2014 benchmark function set as test functions. Among them, F1-F3 are unimodal functions, F4-F16 are simple multimodal functions, F17-F22 are hybrid functions, and F23-F30 are composition functions.

4.2. **Optimization Accuracy Analysis.** Table 2 records the average fitness values of each algorithm after running independently 30 times. At the end of the table, the "victories" indicate the number of benchmark functions where the MPPE algorithm outperforms other algorithms. A higher victory value indicates better performance of the MPPE algorithm. The bolded values in the table represent the optimal values achieved by the algorithm under the same conditions. Benchmark functions exhibit distinct characteristics that enable systematic evaluation of optimization algorithms. Simple unimodal functions,

Algorithm 1 MPPE Algorithm

Input: The dimension D, the upper bound UB and the lower bound LB of the search space, the number of all population N_p, the number of subgroups n, the current iteration number t, and the maximum iteration number $Max\,gen$.

The value of k is calculated from N_p.
Initialize population position x, evolutionary trend ev.
Calculate the population size and the number of historical optimal solutions for each subgroup.
for $t = 1 :\ Max\,gen$ **do**
    for $i = 1 :\ n$ **do**
        Calculate the population position using Equation (3) and compute its fitness value;
        Update x, ev, and population size p.
    end for
    if $mod(t, t_1) == 0$ **then**
        Use the optimal communication strategy to search for the optimal candidate solution.
    end if
    if $mod(t, t_2) == 0$ **then**
        Use the maximum communication strategy to search for the optimal candidate solution.
    end if
    if $mod(t, t_3) == 0$ **then**
        Use the neighbor communication strategy to search for the optimal candidate solution.
    end if
    if $mod(t, t_4) == 0$ **then**
        Use the parent-child communication strategy to search for the optimal candidate solution.
    end if
end for

Output: The optimal value x_{best}.

Table 1. Parameter settings for MPPE algorithm and comparison algorithms

Algorithm	Parameter settings
PSO	$c_1 = 2, \quad c_2 = 2, \quad w = 0.8$
SSA	$\alpha \in [0, 2], \quad ST \in [0.5, 1]$
WOA	$b = 1, \quad pc \in [0, 1], \quad l \in [-2, -1]$
FDA	$a \in (0, 1], \quad R \in (0, 1], \quad S \in (0.5, 1]$
BOA	$p = 0.8, \quad c = 0.01, \quad a \in [0, 1]$
PPE	$a = 1.1, \quad c = 0.2$
MPPE	$a = 1.1, \quad c = 0.2, \quad m = 4$

characterized by a singular global optimum, primarily serve to assess fundamental convergence capabilities in non-complex search spaces. In contrast, multimodal variants introduce multiple local optima superimposed upon a single global solution, thereby testing algorithmic capacity for global exploration and local optima avoidance through effective

perturbation mechanisms. Hybrid formulations integrate unimodal and multimodal landscapes, incorporating heterogeneous peak configurations across multiple scales to evaluate the critical balance between intensive exploitation and broad exploration strategies. The most complex category, composite functions, employ weighted combinations of basic function components to simulate real-world optimization challenges through discontinuous fitness landscapes and irregular parameter interactions, requiring sophisticated adaptation mechanisms for successful navigation.

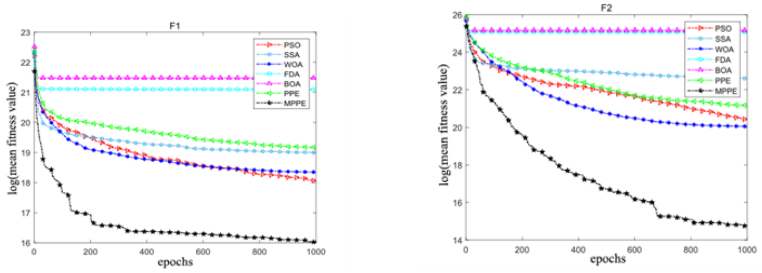
As can be seen from Table 2, the MPPE algorithm achieves excellent results on most benchmark functions. The FDA, BOA, and PPE algorithms do not obtain the optimal value on any of the benchmark functions; the PSO algorithm obtains the optimal value on 6 benchmark functions; the SSA obtains the optimal value on 4 benchmark functions; the WOA obtains the optimal value on only 1 benchmark function; and the MPPE algorithm obtains the optimal value on 19 benchmark functions, which is more than half of the total number of functions. Therefore, compared with other algorithms, the MPPE algorithm has the strongest optimization ability and the highest optimization accuracy. From Table 3, it can be seen that on the 3 unimodal functions, the MPPE algorithm achieves optimal values on 2 functions, while the PSO algorithm only achieves the optimal value on 1 function. This indicates that the MPPE algorithm performs well on unimodal functions, i.e., it performs well on simple optimization problems. On the 13 simple multimodal functions, the MPPE algorithm achieves optimal values on 6 functions, the PSO algorithm on 4 functions, the SSA on 2 functions, and the WOA only on 1 function. Although none of them exceed half of the total number of functions, the MPPE algorithm still has the most optimal values compared to other algorithms. This suggests that the MPPE algorithm's ability to escape local optima and its global search ability are better than the other six algorithms, but without a prominent advantage. On the 6 hybrid functions, the PSO algorithm only achieves the optimal value on 1 function, while the MPPE algorithm achieves optimal values on 5 functions. This indicates that the MPPE algorithm has outstanding ability to balance exploration and exploitation. On the 7 composition functions, the SSA achieves optimal values on 2 functions, while the MPPE algorithm achieves optimal values on 6 functions. These experimental outcomes conclusively validate MPPE's robustness in complex solution spaces. Collectively, these cross-category results establish MPPE's statistically superior optimization efficacy across diverse problem topologies compared to existing metaheuristic paradigms.

Table 2. Comparison of mean values of MPPE algorithm and comparison algorithms

F(x)	PSO	SSA	WOA	FDA	BOA	PPE	MPPE
F1	32058941	129626454	92643434	1447048909	2129486926	150075007	6991577.28
F2	473585.40	5058021597	510319373	77102191393	8471136521	932824998	537420.889
F3	8367.5240	40090.6186	63233.516	4397638.203	1841984.909	39694.3493	2389.40125
F4	653.80873	675.854176	702.40318	14808.7568	26595.27675	790.741023	532.299806
F5	520.91331	521.057305	520.57221	520.8679524	521.3808977	521.007561	520.001985
F6	617.91706	618.307772	637.63514	642.0278499	649.8882437	630.220236	624.757202
F7	700.43021	735.409776	703.66464	1397.392518	1765.519635	708.206539	700.236435
F8	840.39550	901.788539	992.44024	1226.776667	1311.254312	1020.73223	893.182935
F9	1040.2959	1027.37535	1173.6390	1287.748156	1495.901558	1167.63073	1049.76989
F10	2356.6200	3572.82311	5305.2444	9265.681361	10501.07164	6405.85327	2880.65394
F11	6171.0497	4930.91584	6644.9160	9592.020822	10753.34515	7041.70497	4667.90400
F12	1202.5075	1202.34469	1201.7484	1203.485273	1206.371678	1202.70607	1200.55723
F13	1300.5431	1301.04269	1300.5136	1308.083386	1310.646937	1300.58577	1300.30074
F14	1400.3882	1409.14814	1400.2648	1659.190867	1791.077011	1400.97351	1400.29226
F15	1519.4311	1927.46282	1605.3799	474581.4597	981699.4244	2222.07729	1522.92240
F16	1612.4835	1611.93821	1612.6898	1613.543735	1614.395731	1612.83181	1612.14489
F17	1697231.8	4298613.37	9188480.1	143999733.1	274382216.4	3944507.84	575609.300
F18	650485.19	19551543.6	135588.87	6247885876	9881654965	4001577.98	4718.14174
F19	1918.4040	1969.84870	1940.2895	2406.842346	2839.920696	1986.33375	1911.12442
F20	17781.402	32664.1574	67238.499	965913.2288	2748389.642	43248.8876	5056.19657
F21	606168.88	351222.172	4028267.4	106546110.7	223187046.5	1156741.02	136662.697
F22	2636.1211	2654.64243	3098.7258	38885.0724	180787.7792	2905.34099	2869.21281
F23	2617.4092	2641.90676	2657.5210	2538.484135	3768.97877	2654.78370	2500.69589
F24	2638.0749	2600.01808	2607.2156	2603.526512	2761.529937	2657.13318	2602.19881
F25	2714.2102	2707.83384	2728.2510	2700.598637	2812.034691	2726.44703	2700.01159
F26	2740.8370	2740.39790	2750.2093	2792.007424	2937.873204	2791.72783	2700.45175
F27	3354.2578	3364.70509	3808.8176	2994.152784	5176.243264	3760.37000	2942.29312
F28	4429.6357	4168.71091	5299.2330	3093.93786	12464.85103	7873.03124	3000.97328
F29	2871994.8	3014286.97	5822237.0	44009429.27	1192277005	387210.993	5099267.48
F30	14050.938	75826.9752	179021.23	2981275.498	12231016.64	74039.8020	9781.38831
Win	6	4	1	0	0	0	19

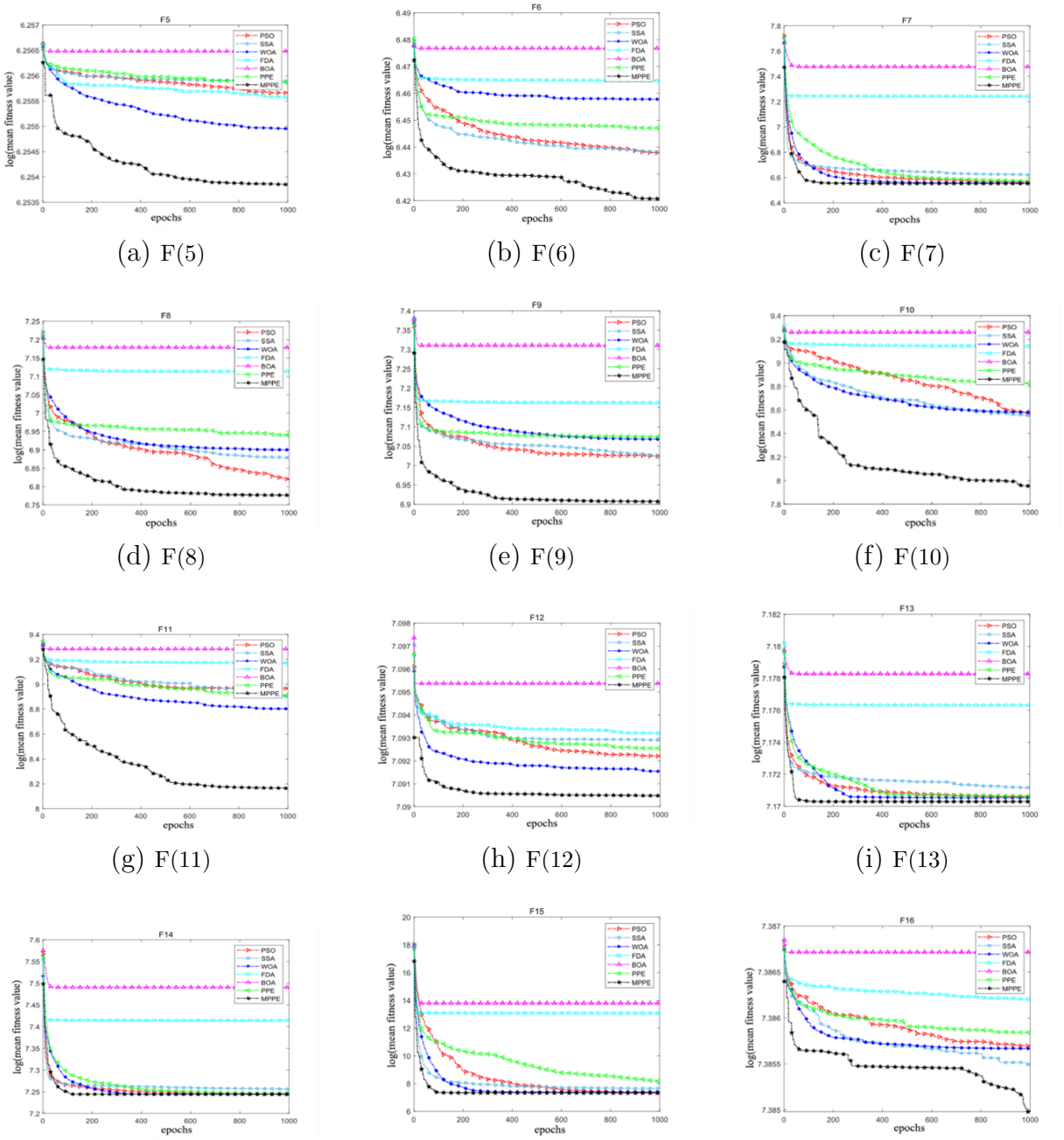
Table 3. Comparison of mean values of MPPE algorithm and comparison algorithms

Function Types	PSO	SSA	WOA	FDA	BOA	PPE	MPPE	Total
Unimodal Function	1	0	0	0	0	0	2	3
Simple Multimodal Function	4	2	1	0	0	0	6	13
Hybrid Function	1	0	0	0	0	0	5	6
Composition Function	0	2	0	0	0	0	6	8



(a) F(1) (b) F(2)

Figure 4. The convergence curve of MPPE algorithm and comparison algorithms



(a) F(5) (b) F(6) (c) F(7)

(d) F(8) (e) F(9) (f) F(10)

(g) F(11) (h) F(12) (i) F(13)

(j) F(14) (k) F(15) (l) F(16)

Figure 5. The convergence curve of MPPE algorithm and comparison algorithms on simple multimodal functions

4.3. **Convergence Analysis.** Convergence performance is one of the important indicators for evaluating the performance of optimization algorithms, as it reflects the speed and stability of the algorithm in finding the optimal solution during the iterative process. To further evaluate the convergence performance of the MPPE algorithm, this section compares and analyzes the convergence curves of the MPPE algorithm with those of the comparison algorithms. Figure 4 to Figure 7 present the comparison charts of convergence curves between the MPPE algorithm and six comparison algorithms on unimodal functions, simple multimodal functions, hybrid functions, and composition functions. Additionally, since the MPPE algorithm performs exceptionally well on the unimodal function

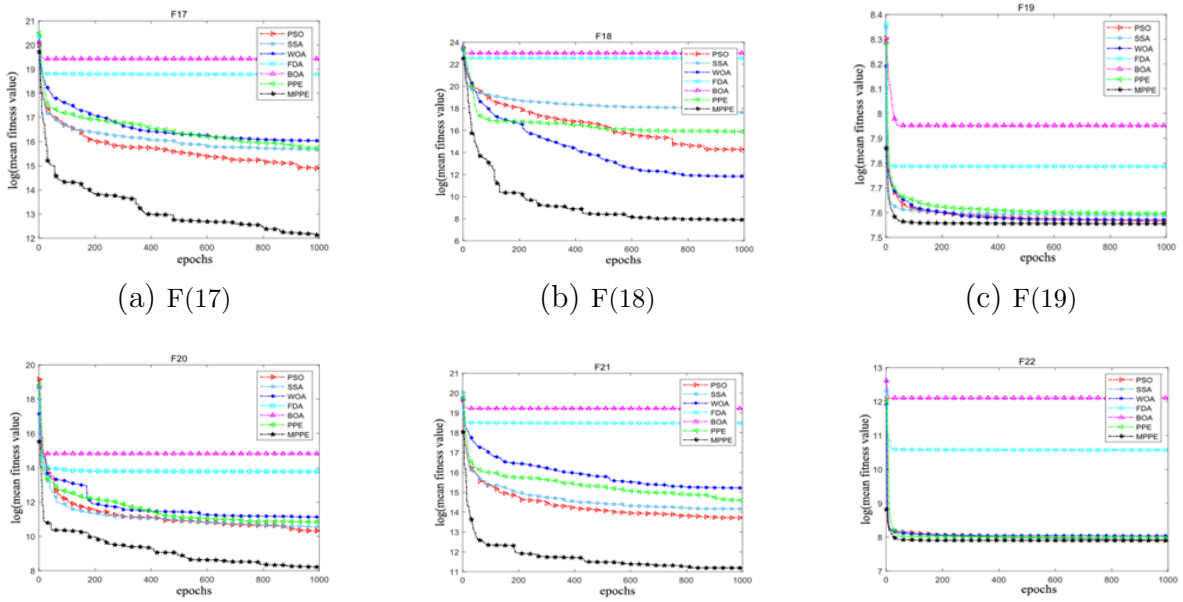
F3 and the simple multimodal function F4, this section does not provide convergence curve charts comparing the MPPE algorithm with the comparison algorithms on F3 and F4. As evident from the convergence curves in Figure 4, the MPPE algorithm outperforms other algorithms in terms of convergence accuracy across all unimodal functions, while its convergence speed is on par with the other six comparative algorithms. Specifically, the fitness value of the MPPE algorithm consistently surpasses those of other algorithms. On F1, MPPE exhibits a faster initial convergence rate than others but gradually slows down after 200 iterations. On F2, MPPE sustains its convergence throughout, especially in the early iterations, far outpacing other algorithms. In summary, the proposed MPPE algorithm demonstrates robust performance on unimodal functions, indicating its superior optimization ability over the six benchmark algorithms for simple problems.

Regarding Figure 5, the MPPE algorithm displays comparable performance to PPE, PSO, WOA, and SSA on F7, F13, and F14, achieving a certain level of accuracy early in the iterations before gradually converging at a slower pace in later stages. However, MPPE exhibits distinct advantages on other simple multimodal functions. On F5 and F11, MPPE (assuming this refers to a variant or specific application of MPPE) shows remarkable convergence capability in the initial iterations, rapidly reaching a high level of accuracy. On F6 and F16, MPPE repeatedly converges swiftly to a certain precision and maintains a superior convergence accuracy throughout, demonstrating a clear edge over other algorithms. In contrast, other algorithms lag behind in convergence speed and fail to match MPPE's performance. On F9 and F12, MPPE achieves a fast initial convergence rate and consistently maintains a high level of accuracy. For F8, while all algorithms converge slowly in later iterations, PSO's convergence accelerates, yet its accuracy remains inferior to MPPE's. In conclusion, MPPE excels on most simple multimodal functions, showcasing its formidable ability to escape local optima and significantly surpassing other algorithms in terms of efficiency and accuracy in solving multimodal optimization problems.

Analysis of experimental data and convergence curves reveals that the MPPE (Modified Parallel Particle Swarm Optimization) algorithm exhibits significantly faster convergence speed compared to the PPE (traditional Parallel Particle Swarm Optimization). In terms of search space characteristics, since both PPE and MPPE are based on the same problem modeling and encoding rules, their search spaces largely overlap in dimensionality, boundary ranges, and feasible region distribution. This high degree of similarity makes a comparative analysis based on search space size, shape, and other structural dimensions less meaningful. Therefore, it is more appropriate to focus on differentiated performance metrics such as convergence efficiency and global optimization capability.

5. **Algorithm for Engineering Design Problem.** To rigorously validate the MPPE algorithm's engineering applicability, this investigation employs two constrained mechanical design benchmarks: tension/compression spring optimization and pressure vessel configuration. These canonical engineering problems evaluate critical algorithmic competencies in constrained search space navigation and precision convergence under nonlinear constraints. MPPE's hybrid optimization architecture, integrating adaptive exploration operators and intensified local search, demonstrates accelerated solution discovery capabilities essential for practical engineering requirements.

A comparative framework is established against five established metaheuristics (PPE, BOA, WOA, FDA, MFO) under standardized experimental conditions: 30 population members, 1000 iterations per trial, and 30 independent runs. Table 4 data reveals MPPE's dominance across all evaluation metrics, achieving the minimum objective value



(a) F(17) (b) F(18) (c) F(19)

(d) F(20) (e) F(21) (f) F(22)

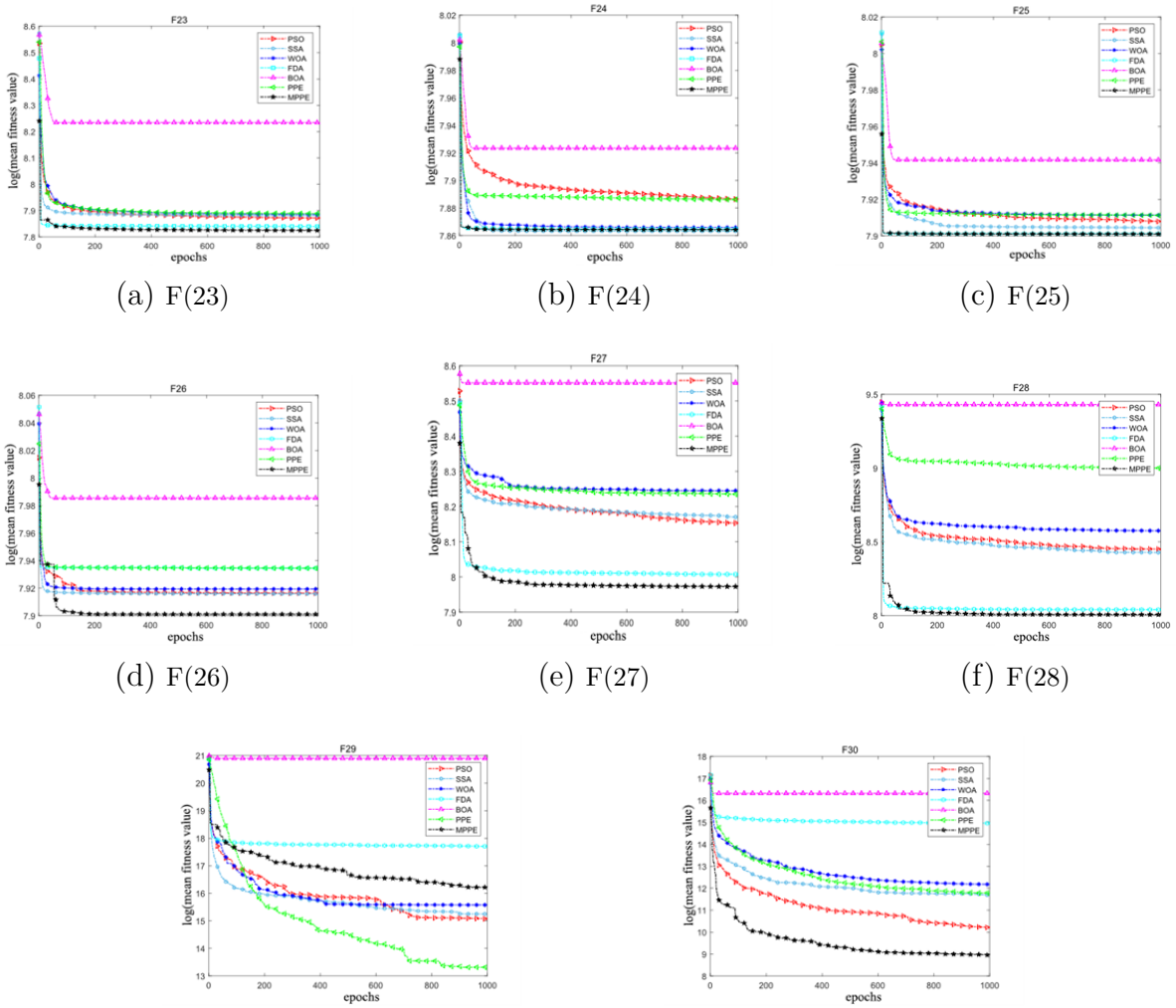
Figure 6. The convergence curve of MPPE algorithm and comparison algorithms on unimodal functions

(0.012669) within merely 30 iterations - the closest approximation to theoretical optimum. The algorithm's triple superiority in solution quality (optimal/average values) and reliability (standard deviation) confirms its engineering optimization efficacy, particularly in spring design applications requiring strict dimensional tolerances.

As indicated by the bolded values in Table 5, the MPPE algorithm performs the best and achieves the optimal results. According to the optimal values, FDA attains an optimal value of 6061.672576, which is the closest to the optimal value of the objective function. However, the difference between the optimal values of the MPPE algorithm and the MFO algorithm is not significant, suggesting that their single- run optimization capabilities are similar. In terms of the average values, the MPPE algorithm surpasses all other comparison algorithms, achieving a decisive victory. As for the standard deviation, the MPPE algorithm's value of 852.175656 also indicates that it is the best-performing algorithm. This demonstrates that the MPPE algorithm not only has strong optimization capabilities but also exhibits excellent stability. In summary, the MPPE algorithm performs well in solving the pressure vessel design problem.

Table 4. Comparison results of SAPPE algorithm and comparison algorithms on tension/compression spring design problem

	PPE	BOA	WOA	FDA	MFO	MPPE
x1	0.0582121	0.1610643	0.0595724	0.1651645	0.0543948	0.05430313
x2	0.6350807	0.6943183	0.5948487	0.6804075	0.5086246	0.5077573
x3	5.7257256	9.6991398	8.4703319	8.4483281	7.5607232	5.7061820
Best	0.0127609	0.0427829	0.0127714	0.0127057	0.0127284	0.01266931
Mean	0.0130843	0.2210255	0.0126956	0.02504013	0.0131843	0.0126873
Std	0.0037440	0.1565897	0.0030322	0.02027173	0.0076100	0.0033278



(a) F(23) (b) F(24) (c) F(25)

(d) F(26) (e) F(27) (f) F(28)

(g) F(29) (h) F(30)

Figure 7. The convergence curve of MPPE algorithm and comparison algorithms on unimodal functions

Table 5. Comparison results of SAPPE algorithm and comparison algorithms on pressure vessel design problem

	PPE	BOA	WOA	FDA	MFO	MPPE
x1	0.0582121	0.1610643	0.0595724	0.1651645	0.0543948	0.05430313
x2	0.6350807	0.6943183	0.5948487	0.6804075	0.5086246	0.5077573
x3	5.7257256	9.6991398	8.4703319	8.4483281	7.5607232	5.7061820
Best	0.0127609	0.0427829	0.0127714	0.0127057	0.0127284	0.01266931
Mean	0.0130843	0.2210255	0.0126956	0.02504013	0.0131843	0.0126873
Std	0.0037440	0.1565897	0.0030322	0.02027173	0.0076100	0.0033278

6. **Conclusion and Future Work.** This study addresses the inherent limitations of the Phasmatodea Population Evolution (PPE) algorithm, notably its local entrapment susceptibility and suboptimal convergence precision, through the development of a Multi-Swarm Phasmatodea Population Evolution framework with Dynamic Inter-Swarm Communication (MPPE). The proposed architecture implements parallel subpopulation evolution

through phased independent evolution cycles, enhanced by adaptive population resizing mechanisms during intergenerational communication. This strategic diversification approach effectively mitigates premature convergence while maintaining solution quality. Algorithmic validation through CEC 2014 benchmark analysis demonstrates MPPE's superior convergence characteristics, evidenced by accelerated optimization velocity and enhanced precision across multimodal landscapes. Comparative stability metrics, quantified through minimized standard deviation values, confirm the framework's operational reliability. Engineering validation studies on constrained optimization problems reveal MPPE's practical efficacy: 1) For spring design optimization, the algorithm achieves dominant performance across all evaluation metrics (optimal value, mean solution quality, and deviation control); 2) In pressure vessel design scenarios, it exhibits superior average solution precision and variance reduction compared to six established metaheuristics. The synthesized experimental evidence establishes MPPE as a robust optimization paradigm combining accelerated convergence kinetics, solution precision, and operational stability, demonstrating significant potential for complex engineering applications requiring reliable optimization performance.

Acknowledgment. This work·was,supported by·the following projects:

[1] The "Coding" Technology of Special Materials Based on the "One Thing, One Code" Technology in Fuzhou, Fujian Province, China (Grant no. 2023FZZD0112).

[2] The Research on Student Personalized Learning, Behavior Characteristics and Intervention Based on Deep Learning in Fuzhou, Fujian Province, China (Grant no. FJJKBK24-080).

[3] The Fujian Provincial Department of Education Innovation Team Project for Universities (Industrialization Specialization), Fujian Province, China (Grant no. Min-Jiao-Ke [2020] 12).

[4] The "Coding" Assignment Process of Special Materials Based on the "One Thing, One Code" Technology in Fuzhou, Fujian Province, China (Grant no. 2023-ZD-009).

REFERENCES

[1] P. Stodola and J. Nohel, "Adaptive ant colony optimization with node clustering for the multidepot vehicle routing problem," IEEE Transactions on Evolutionary Computation, vol. 27, no. 6, pp. 1866–1880, 2022.

[2] A.-E. Taha and N. AbuAli, "Route planning considerations for autonomous vehicles," IEEE Communications Magazine, vol. 56, no. 10, pp. 78–84, 2018.

[3] A. Turky, N. R. Sabar, S. Dunstall, and A. Song, "Hyper-heuristic local search for combinatorial optimisation problems," Knowledge-Based Systems, vol. 205, Art. no. 106264, 2020.

[4] M. Jelali, "Estimation of valve stiction in control loops using separable least-squares and global search algorithms," Journal of Process Control, vol. 18, no. 7–8, pp. 632–642, 2008.

[5] W. Wong and C. I. Ming, "A review on metaheuristic algorithms: recent trends, benchmarking and applications," in 2019 7th International Conference on Smart Computing & Communications (ICSCC). IEEE, 2019, pp. 1–5.

[6] E.-G. Talbi, Metaheuristics: From Design to Implementation. John Wiley & Sons, 2009.

[7] S. P. Adam, S.-A. N. Alexandropoulos, P. M. Pardalos, and M. N. Vrahatis, "No free lunch theorem: A review," in Approximation and Optimization: Algorithms, Complexity and Applications. Springer, 2019, pp. 57–82.

[8] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," SIAM Journal on Computing, vol. 2, no. 2, pp. 88–105, 1973.

[9] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," Advances in Engineering Software, vol. 69, pp. 46–61, 2014.

[10] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28–39, 2007.

[11] Z. Meng, Y. Chen, X. Li, C. Yang, and Y. Zhong, "Enhancing QUasi-Affine TRansformation Evolution (QUATRE) with adaptation scheme on numerical optimization," Knowledge-Based Systems, vol. 197, Art. no. 105908, 2020.
[12] Y.-F. Peng, S.-C. Chu, R.-Y. Wang, J. Zhao, and J.-S. Pan, "Gannet Optimization Algorithm Enhanced by Quasi-Affine Transformation Algorithm and Its Application in 3D Coverage of Wireless Sensor Network," Journal of Network Intelligence, vol. 9, no. 1, pp. 69–87, 2024.
[13] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," Neural Computing and Applications, vol. 27, pp. 1053–1073, 2016.
[14] P.-C. Song, S.-C. Chu, J.-S. Pan, and H. Yang, "Phasmatodea population evolution algorithm and its application in length-changeable incremental extreme learning machine," in 2020 2nd International Conference on Industrial Artificial Intelligence (IAI). IEEE, 2020, pp. 1–5.
[15] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," Simulation, vol. 76, no. 2, pp. 60–68, 2001.
[16] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," Knowledge-Based Systems, vol. 89, pp. 228–249, 2015.
[17] I. Ahmadianfar, O. Bozorg-Haddad, and X. Chu, "Gradient-based optimizer: A new metaheuristic optimization algorithm," Information Sciences, vol. 540, pp. 131–159, 2020.
[18] T.-Y. Wu, H. Li, and C.-M. Chen, "A Spectral Convolutional Neural Network Model Based on Adaptive Fick's Law for Hyperspectral Image Classification," Computers, Materials & Continua, vol. 79, no. 1, 2024.
[19] T.-Y. Wu, A. Shao, and J.-S. Pan, "CTOA: Toward a chaotic-based tumbleweed optimization algorithm," Mathematics, vol. 11, no. 10, Art. no. 2339, 2023.
[20] P.-C. Song, S.-C. Chu, J.-S. Pan, and H. Yang, "Simplified Phasmatodea population evolution algorithm for optimization," Complex and Intelligent Systems, vol. 8, no. 4, pp. 2749–2767, 2022.
[21] D. Xiang, H. Lin, J. Ouyang, and D. Huang, "Combined improved A* and greedy algorithm for path planning of multi-objective mobile robot," Scientific Reports, vol. 12, no. 1, Art. no. 13273, 2022.
[22] A. Sitenda, P.-C. Song, S.-C. Chu, and S.-H. Chen, "Differential Evolution Strategy with Chebyshev Chaos Based Mutation," Information Hiding and Multimedia Signal Processing, vol. 14, no. 4, pp. 172–183, 2024.
[23] T.-Y. Wu, H. Li, and S.-C. Chu, "CPPE: An improved phasmatodea population evolution algorithm with chaotic maps," Mathematics, vol. 11, no. 9, Art. no. 1977, 2023.
[24] S. Lalwani, H. Sharma, S. C. Satapathy, K. Deep, and J. C. Bansal, "A survey on parallel particle swarm optimization algorithms," Arabian Journal for Science and Engineering, vol. 44, pp. 2899–2923, 2019.
[25] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: an overview," Soft Computing, vol. 22, no. 2, pp. 387–408, 2018.
[26] J.-S. Pan, F. McInnes, and M. Jack, "Application of parallel genetic algorithm and property of multiple global optima to VQ codevector index assignment for noisy channels," Electronics Letters, vol. 32, no. 4, pp. 296–297, 1996.
[27] T.-G. Ngo, H. Nguyen-Cong, T.-T.-T. Nguyen, and T.-K. Dao, "Optimal Parameter-Feature Selection Using Binary PSO for Enhanced Classification Performance," J. Inf. Hiding Multim. Signal Process, vol. 14, no. 4, pp. 172–183, 2023.
[28] S. Mirjalili and A. Lewis, "The whale optimization algorithm," Advances in Engineering Software, vol. 95, pp. 51–67, 2016.
[29] D.-T. Pham and D.-T.-T. Hoang, "An Improved Whale Optimization Algorithm for Optimal Multithreshold Image Segmentation," J. Inf. Hiding Multim. Signal Process, vol. 14, no. 2, pp. 1–11, 2023.
[30] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," Systems Science and Control Engineering, vol. 8, no. 1, pp. 22–34, 2020.
[31] S.-C. Chu, Z.-Y. Shao, N. Zhong, G.-G. Liu, and J.-S. Pan, "An Enhanced Food Digestion Algorithm for Mobile Sensor Localization," Sensors, vol. 23, no. 17, Art. no. 7508, 2023.
[32] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," Soft Computing, vol. 23, pp. 715–734, 2019.