AIMS Mathematics, 9(4): 7917–7944.
DOI: 10.3934/math.2024385
Received: 08 January 2024
Revised: 21 February 2024
Accepted: 21 February 2024
Published: 26 February 2024

http://www.aimspress.com/journal/Math

Research article

An improved composite particle swarm optimization algorithm for solving constrained optimization problems and its engineering applications

Ying Sun[1,2,*] **and Yuelin Gao**[1,2]

[1] Ningxia Collaborative Innovation Center of Scientific Computing and Intelligent Information Processing, North Minzu University, Yinchuan 750021, China

[2] School of Mathematics and Information Sciences, North Minzu University, Yinchuan 750021, China

* **Correspondence:** Email: sunying@nun.edu.cn; Tel: +8613519200230.

Abstract: In the last few decades, the particle swarm optimization (PSO) algorithm has been demonstrated to be an effective approach for solving real-world optimization problems. To improve the effectiveness of the PSO algorithm in finding the global best solution for constrained optimization problems, we proposed an improved composite particle swarm optimization algorithm (ICPSO). Based on the optimization principles of the PSO algorithm, in the ICPSO algorithm, we constructed an evolutionary update mechanism for the personal best position population. This mechanism incorporated composite concepts, specifically the integration of the ε-constraint, differential evolution (DE) strategy, and feasibility rule. This approach could effectively balance the objective function and constraints, and could improve the ability of local exploitation and global exploration. Experiments on the CEC2006 and CEC2017 benchmark functions and real-world constraint optimization problems from the CEC2020 dataset showed that the ICPSO algorithm could effectively solve complex constrained optimization problems.

Keywords: particle swarm optimization algorithm; constrained optimization problem; DE strategy; ε-constraint; feasibility rule; engineering application
Mathematics Subject Classification: 90C26, 90C59

1. Introduction

Constrained optimization is particularly challenging because certain constraints must be satisfied while optimizing the objective function. This type of problem has a wide range of applications in many fields, such as economics, engineering, and natural sciences. The mathematical model of the

constrained optimization problem is expressed as follows [1]:

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & g_j(x) \le 0,\ j = 1, 2, \ldots, q, \\ & h_j(x) = 0,\ j = q+1, q+2, \ldots, m, \\ & x = (x_1, x_2, \ldots, x_D), \\ & L_i \le x_i \le U_i,\ i = 1, 2, \ldots, D, \end{aligned} \tag{1.1}$$

where $f(x)$ is the objective function, $g(x)$ is an inequality constraint, $h(x)$ is an equality constraint, x is a decision variable, D is the dimension of decision variable, q is the number of inequality constraints, m is the number of constraints, and L_i and U_i are the lower and upper bounds of the ith component, respectively.

For the above problem, the violation degree of any decision vector x regaiding the jth constraint can be defined as follows:

$$G_j(x) = \begin{cases} max(0, g_j(x)), & 1 \le j \le q, \\ max(0, |h_j(x)| - \delta), & q+1 \le j \le m. \end{cases} \tag{1.2}$$

Here, δ is the positive slack of the equality constraint. Therefore, the violation degree of decision variable x with respect to all constraints can be defined as follows:

$$G(x) = \sum_{j=1}^{m} G_j(x). \tag{1.3}$$

If $G(x) = 0$, then the decision variable x is a feasible solution to the constrained optimization problem. The goal of solving the constrained optimization problem is to find the feasible solution with the minimum function value.

Due to the limitations of the constraints, solving a constrained optimization problem is more difficult than solving an unconstrained optimization problem. Therefore, effectively addressing constraints becomes the top priority. At present, the commonly used constraint processing methods include the penalty function method [2], feasibility rule method [3], ε-constraint method [4], multiobjective method [5], and stochastic ranking method [6]. With the development of computer technology and optimization algorithms, intelligent optimization algorithms with a global optimization capability and objective function gradient independence are widely used to solve constrained optimization problems. Karaboga [7] described a modified artificial bee colony algorithm (mcABC) for constrained optimization problems, and for constraint handling, this algorithm uses feasibility rule, consisting of three simple heuristic rules and a probabilistic selection scheme for feasible solutions based on their fitness values and for infeasible solutions based on their violation values. Shi [8] proposed an improved fruit fly optimization algorithm (IFOA) with multi-strategy hybrid co-evolution, and the convergence accuracy of the algorithm was improved through real-time dynamic updating of the optimal information of the swarm and a local in-depth search strategy. Jia [9] proposed an improved coati optimization algorithm (ICOA) to solve constrained engineering optimization problems. Wang [10] proposed a compound differential evolution algorithm that balances the diversity and convergence, improving algorithm convergence for solving constrained optimization problems. Wang [11] proposed an improved genetic algorithm based on two-direction crossover and grouped mutation (GA-TDX)

to efficiently solve constrained optimization problems, enhancing search efficiency by leveraging directional information and maximizing the benefits of distinct mutation operators. Peng [12] designed a hybrid constraint-handling technique (HCT) based on population evolution information to balance the objective function and constraints, and proposed an evolutionary constrained optimization algorithm with HCT (ECO-HCT) to solve solve constrained optimization problems.

Among the many intelligent optimization algorithms, the PSO algorithm [13] was proposed by Drs. Eberhart and Kennedy in 1995 and is used by many researchers to solve constrained optimization problems and real-world problems [14] because of its computational simplicity and flexibility. Lu [15] proposed a double-track schema within PSO called double-track PSO (DTPSO) to address the nonlinear constrained optimization problem. When an infeasible region blocks a particle, the DTPSO algorithm creates a copy of the particle. Different search strategies are used for the original and copied particles so that the algorithm can move to feasible and infeasible regions. Liu [16] proposed a new hybrid PSO differential evolution (PSODE) algorithm to solve constrained optimization problems. When the particles in the PSODE algorithm stagnate, the DE algorithm is used to activate the particles to continue the optimization search. Wang [17] designed a hybrid multi-swarm PSO algorithm (HMPSO) to address the constrained optimization problem. This algorithm uses the parallel search operator to search for each subswarm simultaneously and the DE algorithm to improve the state of the personal best position to better expand the search range. Guo [18] used a feasibility-based rule to process constraints and used DE to optimize the personal best position with the PSO algorithm (PSO-DE) to improve the convergence efficiency of the algorithm. Venter [19] transformed all the constraints into an objective; therefore, the constrained optimization problem was transformed into an unconstrained dual-objective optimization problem, and the model was solved using a multi-objective PSO algorithm. Ang [20] proposed a constrained multi-swarm particle swarm optimization without velocity (CMPSOWV), incorporating constraint handling, dual evolution phases for robustness, and diversity maintenance schemes to overcome limitations and prevent premature convergence in solving COPs.

Although many studies have been performed to solve constrained optimization problems with the PSO algorithm, quickly optimizing the objective function value and controlling the constraint violation degree are difficult at this stage, and balancing the two has always been the main research direction. The optimization principles of the PSO algorithm depend mainly on the selection of the best position. Therefore, generating a personal best position population while considering objective functions and constraints is our focus. Hence, we propose a personal best position update method that integrates the ε-constraint, DE strategy, and feasibility rule. First, this method utilizes the ε-constraint method to give infeasible solutions located at the boundary of the feasible region the opportunity to enter the intermediate personal best position population. Second, the DE strategy is used to further guide the intermediate personal best position population towards the feasible region, forming an intermediate evolutionary population. Finally, using feasibility criteria, a stricter selection is made between the intermediate personal best position population and the intermediate evolutionary population to promote the rapid convergence of the algorithm. Compared with previous single constraint processing methods, this composite evolutionary updating method not only considers the objective function and constraints simultaneously, but also utilizes the powerful optimization capabilities of the DE strategy to select the personal best position population that can better guide the population close to the global best position, thereby ensuring the convergence speed and accuracy of the ICPSO algorithm.

2. The classical PSO algorithm

The PSO algorithm is a swarm intelligence algorithm proposed by simulating the foraging behavior of a flock of birds. In the PSO algorithm, a solution to the problem is represented as a particle in a decision space. The entire swarm is guided by the personal and global best positions and flies within the solution space to find the optimal solution. To better balance local exploitation and global exploration, this section introduces a PSO algorithm [21] with a linearly decreasing inertia weight. The specific flow is as follows:

Algorithm 1. PSO algorithm

Step 1: Assume a population size N, self-cognitive coefficient c_1 and social cognitive coefficient c_2. Let each particle be an individual in the D dimensional space, let the position vector of the ith particle be $x_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$, let the velocity vector be $v_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$, where $i = (i, 2, \cdots, N)$, and let the maximum number of iterations be T. Randomly generate N individuals to create the initial population $X(0)$, and set $t = 0$.

Step 2: Calculate or estimate the fitness of each particle in $X(t)$.

Step 3: Update the speed and position of each particle as follows:

$$v_{id}^{t+1} = \omega v_{id}^{t} + c_1 r_1 (p_{ibestd} - x_{id}^{t}) + c_2 r_2 (g_{bestd} - x_{id}^{t}), \tag{2.1}$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1}, \tag{2.2}$$

$$\omega = \omega_{max} - \frac{t(\omega_{max} - \omega_{min})}{T}, \tag{2.3}$$

where ω_{max} and ω_{min} are the maximum and minimum inertia weights, respectively.

Step 4: Update the personal best position of each particle p_{ibest} and the global best position g_{best}.

Step 5: If the termination criterion is met, then output g_{best} as the optimal solution, and terminate the calculation. Otherwise, set $t = t + 1$, and go to Step 2.

3. An improved composite particle swarm optimization algorithm for constrained optimization problems (ICPSO)

The classical PSO algorithm is good at searching for the best solution in unconstrained continuous space, but has no advantageous strategy for dealing with constraints. Therefore, in this paper, an improved composite particle swarm optimization algorithm is proposed for constrained optimization problems. The algorithm effectively combines strategies such as the ε-constraint, DE strategy, and feasibility rule to construct a novel method for updating the personal best position. This method enhances the algorithm's ability to find feasible solutions and quickly converge to the global optimal solution.

3.1. Feasibility rule

The feasibility rule is a constraint processing mechanism proposed by Deb [22] in 2000, and the specific rules are as follows: For two individuals x_i and x_j, if x_i is superior to x_j, one of the following conditions should be satisfied:

(1) When $G(x_i) = G(x_j) = 0$, $f(x_i) < f(x_j)$;

(2) When $G(x_i) \neq 0$ and $G(x_j) \neq 0$, $G(x_i) < G(x_j)$;

(3) $G(x_i) = 0$ and $G(x_j) \neq 0$.

3.2. ε-constraint

The ε-constraint method is a constraint processing mechanism proposed by Takahama and Sakai [23] in 2010. The specific rules are as follows: For two individuals x_i and x_j, if x_i is superior to x_j, one of the following conditions should be satisfied:

(1) When $G(x_i) \leq \varepsilon \bigwedge G(x_j) \leq \varepsilon$, $f(x_i) < f(x_j)$;

(2) When $G(x_i) = G(x_j)$, $f(x_i) < f(x_j)$;

(3) In other cases, $G(x_i) = 0 < G(x_j)$.

Here, $\varepsilon = \begin{cases} \varepsilon_0(1 - \frac{t}{T}), & if\ \frac{t}{T} \leq p; \\ 0, & otherwise \end{cases}$ is a numerical value that decreases as the number of iterations increases, ε_0 is the initial threshold, usually the maximum constraint violation degree of the initial population, and $cp = -\frac{\log \varepsilon_0 + \lambda}{\log(1-p)}$, where λ is 6, and p is the degree to which the information that controls the objective function is used.

3.3. DE strategy

Although the PSO algorithm is simple and easy to implement, it is prone to becoming trapped in a local extremum during the search process. To address this shortcoming, researchers are constantly attempting to incorporate new strategies to enhance the global exploration ability. The DE strategy is one of the recognized effective strategies. PSODE algorithm [16] applied the DE strategy to the current population to help it break out of the stagnation state. However, this method uses only the DE strategy when the threshold is reached. Thus, this method has limited effectiveness. HMPSO algorithm [17] and PSO-DE algorithm [18] both used the DE strategy to find the personal best position population. However, when performing the DE operation on any personal best position, three other personal best positions are selected for mutation, and the personal best position information is not used. However, for the PSO algorithm, the personal best position stores the historical information of the individual flight and should not be completely discarded. Therefore, in this paper, the DE/rand/1 strategy is used when performing the mutation operation, and the personal best position is retained as x_{r_1}. Then, two personal best positions are randomly selected as x_{r_2} and x_{r_3}. The other operations are the same as those in the standard DE strategy. Finally, an intermediate evolutionary population with the personal best position is generated.

3.4. Best position update strategy

The optimization of the PSO algorithm mostly depends on learning the global best position and personal best positions. Therefore, the selection and update of the best position are particularly important. Moreover, an effective solution to the constrained optimization problem requires a balance between the degree of constraint violation and the objective function value. If these two indicators can be considered simultaneously when updating the best position, this could help the algorithm converge.

3.4.1. Personal best position population update with composite concepts

To update the personal best position population P_{best}, there are two constraint selections steps and one evolution step in the ICPSO algorithm. The first constraint selection step targets the new population produced after PSO evolution. To expand the search space of the algorithm and fully utilize valuable information from partially infeasible solutions, a relatively relaxed constraint violation method, the ε-constraint method, is adopted to guide the update of the population, thus forming the intermediate personal best position population; to increase the search speed of the algorithm, the DE strategy is employed to optimize and evolve the intermediate personal best position population, generating the intermediate evolutionary population. Finally, to ensure the superiority of the personal best population P_{best}, a more stringent feasibility rule with higher constraint requirements is applied to conduct a rigorous comparison and selection between the intermediate personal best position population and the intermediate evolutionary population. Clearly, the adoption of this composite updating strategy to guide the update process effectively balances the two metrics of constraint violation and objective function value, facilitating algorithm convergence. The detailed operation is shown in Figure 1.

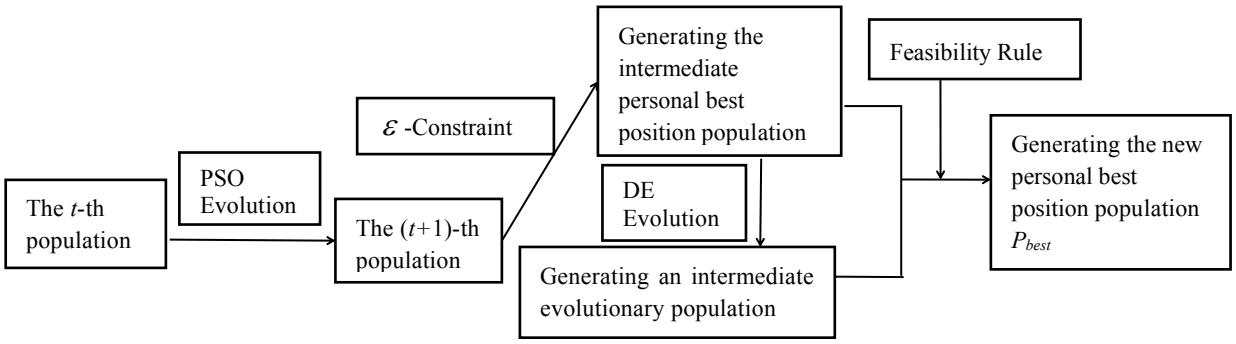


Figure 1. Constraint strategies for personal best position in different periods.

3.4.2. Global best position update

The goal of constrained optimization is to find the feasible solution with the minimum function value. To ensure that the population effectively moves to the feasible region, when updating the global best position g_{best}, the feasibility rule with stronger constraints is used to guide the generation of a new g_{best}.

3.5. ICPSO algorithm

The specific process of the ICPSO algorithm is as follows, and the flowchart of ICPSO algorithm is given in Figure 2:

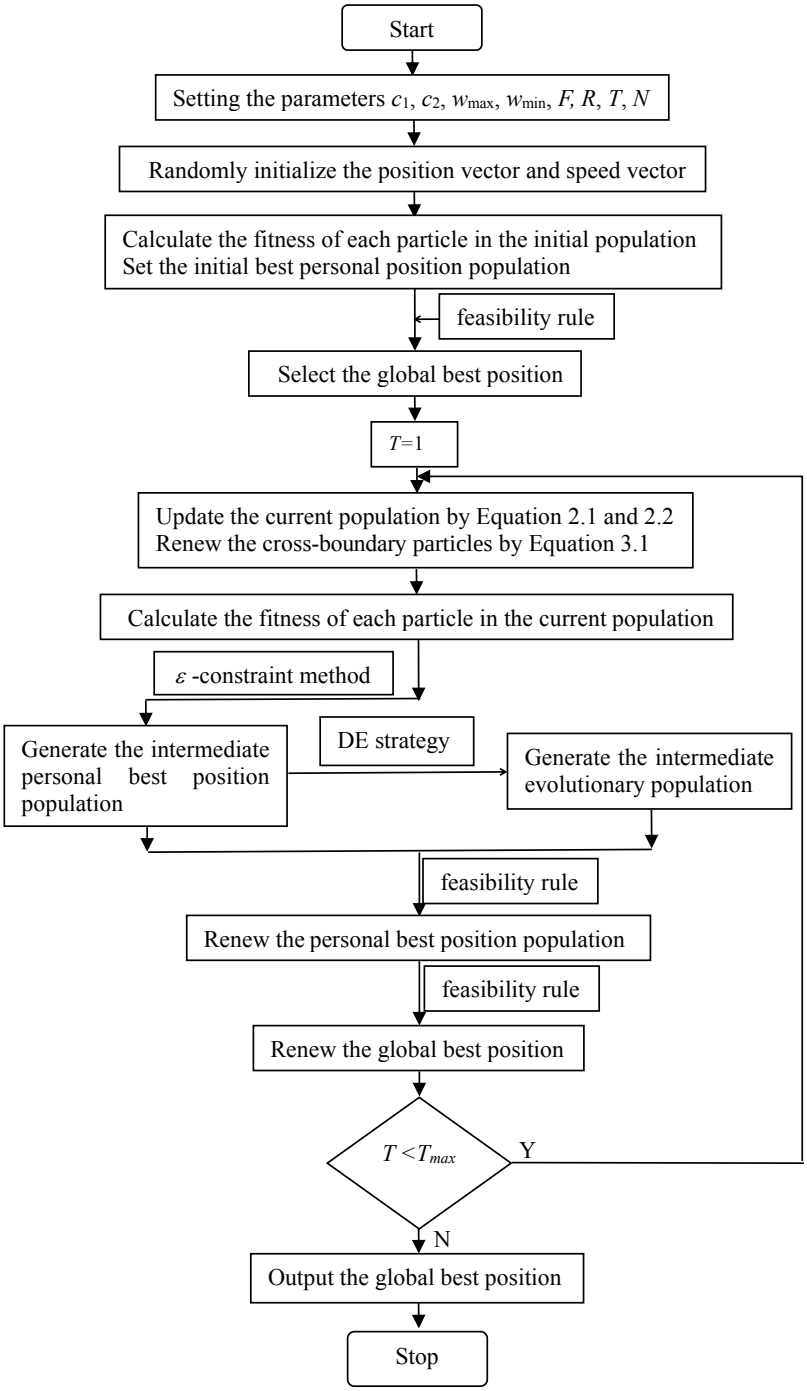


Figure 2. Flowchart of the ICPSO algorithm.

Algorithm 2. ICPSO algorithm

Step 1: Randomly initialize the position vector and speed vector and set the parameters c_1, c_2, ω_{max}, and ω_{min}, the crossover factor F and CR, the maximum number of iterations T, the population size N, and $t = 0$.

Step 2: Calculate the fitness of each particle in the initial population, set the initial best personal

position as $P_{best} = X(0)$, and use the feasibility rule to select the global best position g_{best} from the population P_{best}.

Step 3: Use speed update (Eq (2.1)) and position update (Eq (2.2)) to update the current population and form a new population, use Eq (3.1) to address the cross-boundary particles [24]:

$$x_{i,j}^{t+1} = \begin{cases} 0.5(x_{i,j}^{t} + L_j), & if\, x_{i,j}^{t} < L_j, \\ 0.5(x_{i,j}^{t} + U_j), & if\, x_{i,j}^{t} > U_j, \\ x_{i,j}^{t+1}, & otherwise. \end{cases} \tag{3.1}$$

Step 4: Calculate the fitness of each particle in the current population.

Step 5: Use the ε-constraint method and the current population to generate the intermediate personal best position population.

Step 6: Apply the DE strategy to the personal best position population to form an intermediate evolutionary population.

Step 7: Use the feasibility-based rule to compare the best personal position population and the intermediate evolutionary population to form the final P_{best}.

Step 8: Use the feasibility-based rule to select the global best position g_{best} from the population P_{best}.

Step 9: If the termination criterion is satisfied, output g_{best} as the optimal solution and terminate the calculation. Otherwise, set $t = t + 1$ and go to Step 3.

3.6. Complexity analysis

We present the computational time complexity of the ICPSO algorithm below, the main steps during one iteration for a population of size N, the dimension is D.

(1) Updating a particle: $O(N \times D)$;

(2) Generate the intermediate personal best position population: $O(N)$;

(3) Generate the intermediate evolutionary population by DE strategy: $O(N \times D)$;

(4) Renew the personal best position population: $O(N)$;

(5) Renew the global best position: $O(N)$.

Thus, the computational time complexity of the ICPSO algorithm is $O(N \times D)$, which demonstrates that this algorithm is computationally efficient.

4. Numerical experiment and result analysis

4.1. Test questions and parameter settings

To further test the performance of the ICPSO, 13 constrained optimization benchmark functions from the IEEE CEC2006 dataset and 28 constrained optimization benchmark functions from the IEEE CEC2017 dataset were selected for testing. Table 1 shows the specific characteristics of 13 functions from the IEEE CEC2006 dataset. D represents the number of decision variables, ρ represents the ratio between the feasible region and the search space estimated by stochastic simulation, LI and NI represent the numbers of linear and nonlinear inequality constraints, respectively, LE and NE represent the numbers of linear and nonlinear equality constraints, respectively, a represents the number of active constraint functions near the optimal solution, and $f(x^*)$ represents the global optimal function value,

that is, the known optimal solution of these functions. The detailed mathematical formulation are presented in the technical report of Liang [25].

Table 1. Specific characteristics of 13 benchmark functions from the IEEE CEC2006 dataset.

Prob.	$f(x^*)$	D	LI	NI	LE	NE	a	$\rho(\%)$	Type of objective function
F01	-15.0000000000	13	9	0	0	0	6	0.0111%	Quadratic
F02	-0.8036191042	20	0	2	0	0	1	99.9971%	Nonlinear
F03	-1.0005001000	10	0	0	0	1	1	0.0000%	Polynomial
F04	-30665.5386717834	5	0	6	0	0	2	52.1230%	Quadratic
F05	5126.4967140071	4	2	0	0	3	3	0.0000%	Cubic
F06	-6961.8138755802	2	0	2	0	0	2	0.0066%	Cubic
F07	24.3062090681	10	3	4	0	0	6	0.0003%	Quadratic
F08	-0.0958250415	2	0	2	0	0	0	0.8560%	Nonlinear
F09	680.6300573745	7	0	4	0	0	2	0.5121%	Polynomial
F10	7049.2480205286	8	3	3	0	0	6	0.0010%	Linear
F11	0.7499000000	2	0	0	0	1	1	0.0000%	Quadratic
F12	-1.0000000000	3	0	1	0	0	0	4.7713%	Quadratic
F13	0.0539415140	5	0	0	0	3	3	0.0000%	Nonlinear

Table 2 shows the specific characteristics of 28 functions from the IEEE CEC2017 dataset. E and I represent the numbers of inequality constraints and equality constraints, respectively. The detailed mathematical formulation of all CEC2017 functions are presented in the technical report of Wu [26].

Table 2. Specific characteristics of 28 benchmark functions from the IEEE CEC2017 dataset.

Prob.	D	E	I	Type of objective function	Prob.	D	E	I	Type of objective function
C01	10	0	1	non Separable	C02	10	0	1	non Separable, rotated
C03	10	1	1	Separable	C04	10	0	2	non Separable
C05	10	0	2	non Separable	C06	10	6	0	Separable
C07	10	2	0	Separable	C08	10	2	0	Separable
C09	10	2	0	Separable	C10	10	2	0	Separable
C11	10	1	1	Separable	C12	10	0	2	Separable
C13	10	0	3	non Separable	C14	10	1	1	non Separable
C15	10	1	1	Separable	C16	10	1	1	Separable
C17	10	1	1	non Separable	C18	10	1	2	Separable
C19	10	0	2	Separable	C20	10	0	2	non Separable
C21	10	0	2	rotated	C22	10	0	3	rotated
C23	10	1	1	rotated	C24	10	1	1	rotated
C25	10	1	1	rotated	C26	10	1	1	rotated
C27	10	1	2	rotated	C28	10	0	2	rotated

The parameters of the ICPSO algorithm are set as follows: $c_1 = c_2 = 2.05$, $\omega_{max} = 1$, $\omega_{min} = 0.4$, $F = 0.7$, $CR = 1.0$, $N = 60$ and number of evaluations of the fitness function= 5×10^5 for CEC2006,

$N = 100$ and number of evaluations of the fitness function= 2×10^5 for CEC2017, the slack of the equality constraint is $\delta = 0.0001$. The algorithm was coded using MATLAB R2016a and run on a computer with an Intel(R) Core(TM) i7-6567U central processing unit (CPU) @ 3.30 GHz or 3.20 GHz and 8.00 GB of installed memory.

4.2. Results analysis

4.2.1. Comparison between ICPSO and its variant

To better illustrate the effectiveness of the various combinations of strategies proposed in this paper, two improved PSO algorithms without the DE strategy were constructed: The PSO+Eps algorithm and the PSO+Deb algorithm. The selected strategy for the constraint-guided update of the personal best position differs between the two methods, with the former using the ε-constraint method and the latter using the feasibility-based rule. Moreover, an improved algorithm containing the DE strategy PSO+DE+Deb+Deb was constructed. The difference between this algorithm and the ICPSO algorithm is that this algorithm uses the feasibility-based rule to guide the two update methods for the personal best position. To ensure the comparability of the numerical results, the parameter settings of the four algorithms are exactly the same. In this section, the Wilcoxon rank-sum test was used to compare the performances of the algorithms. The null hypothesis is that there is no significant difference in performance between the ICPSO algorithm and the other algorithms. Three symbols are used to indicate whether there is a significant difference in performance between the ICPSO algorithm and the other algorithms: (+) indicates that at the significance level of $\alpha = 0.05$, the performance of the ICPSO algorithm is significantly improved compared to that of the other algorithms (-) indicates that the performance of the ICPSO algorithm is inferior to that of the other algorithms; and (=) indicates that there is no significant difference in performance between the algorithms.

(1) Performance evaluation using CEC2006 benchmark functions

Figure 3 shows the proportion of optimal solutions that are feasible for the 13 benchmark functions after 25 independent runs of each algorithm. As long as the constraint processing mechanism is used, the PSO algorithm is more effective at solving the constrained optimization problem. Of course, for functions F04 and F10, the PSO+Deb and PSO+Eps algorithms converge to an infeasible solution; however, as long as the DE strategy is added to the algorithm, it can be guaranteed that the algorithms can converge to the feasible domain for all 13 test problems, reflecting the effectiveness of the DE strategy.

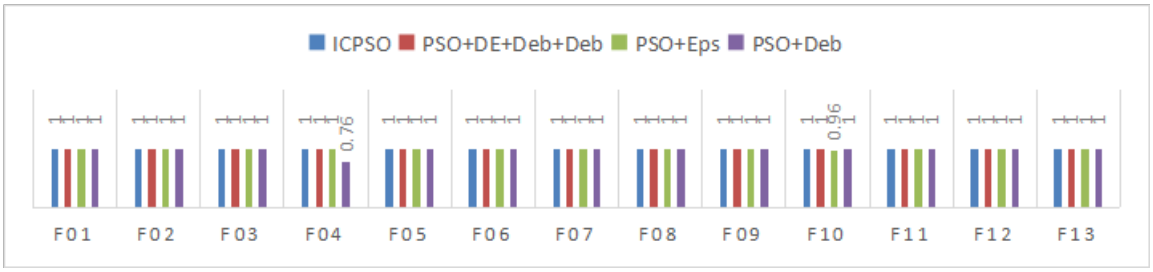


Figure 3. Proportion of feasible solutions for 13 benchmark functions from the IEEE CEC2006 dataset.

The data in Table 3 are the numerical results of the 4 algorithms after 25 independent runs of the 13 test problems. In the table, Best is the best solution, Worst is the worst solution, Median is the median value, Mean is the mean value, and Std is the standard deviation. Wil test is the Wilcoxon test result. Except for F06, F11, and F12 (on which the four algorithms obtain the same result), ICPSO outperforms the other algorithms, in particular, it has a smaller variance, indicating that ICPSO has better stability. For F01, F02, and F13, ICPSO achieves the optimal results in terms of both the mean and variance. For F03, F05, and F09, ICPSO and PSO+DE+Deb+Deb can obtain the same optimal mean value, but ICPSO appears to be more stable with a smaller variance. For F10, PSO+DE+Deb+Deb performs better than ICPSO, and PSO+Deb outperforms PSO+Eps, indicating that for a linear problem, the constraint processing mechanism guided by the feasibility-based rule is better than that of the ε-constraint method. The Wilcoxon rank-sum test shows that, except for F13, the ICPSO and PSO+DE+Deb+Deb algorithms had no significant differences in performance. However, these two algorithms are far better than the PSO+Esp and PSO+Deb algorithms, demonstrating that the introduction of the DE strategy greatly improves the performance of the algorithm.

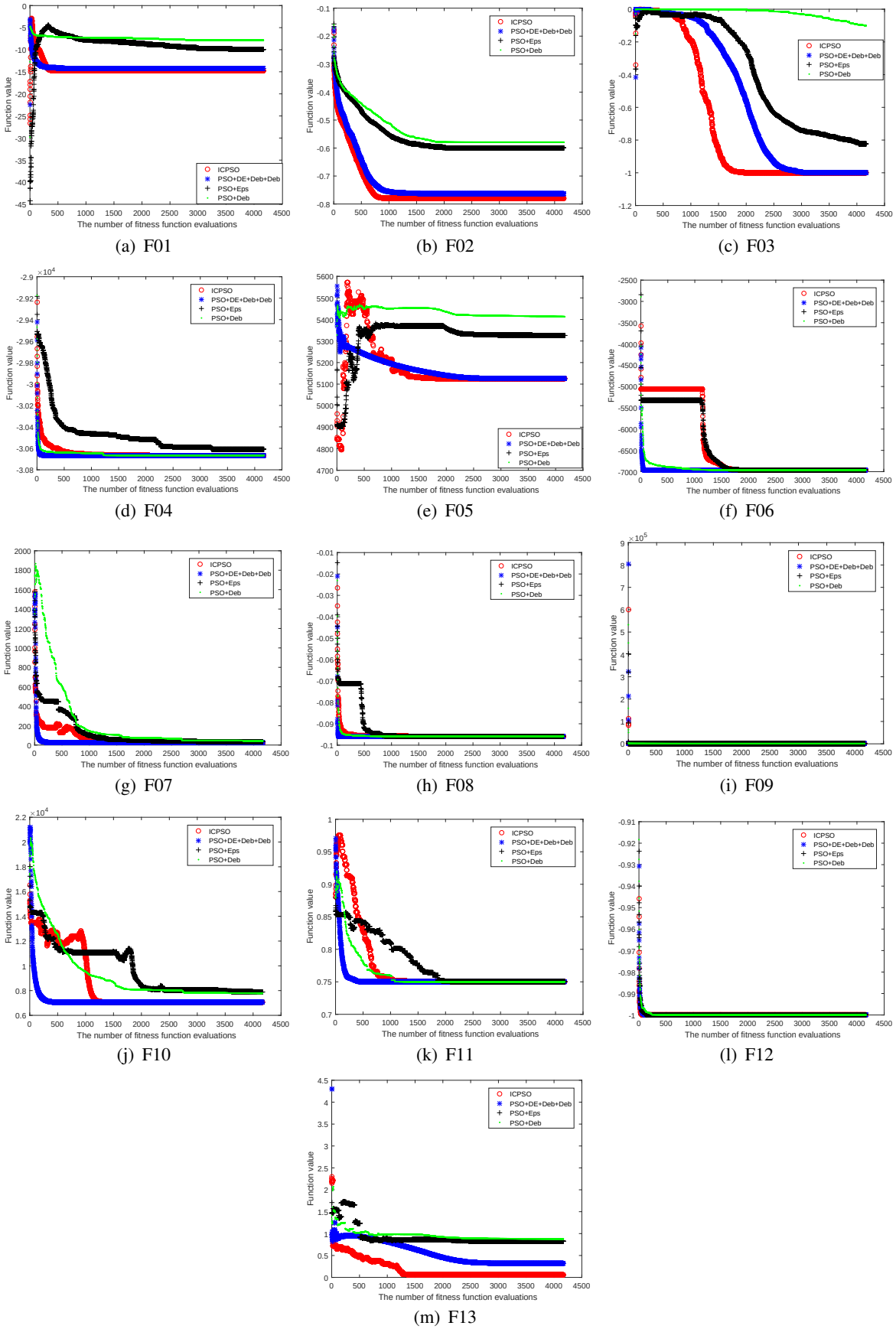
Table 3. Results of 4 improved PSO algorithms on 13 benchmark functions.

Prob.	Algorithm	Best	Worst	Median	Mean(Wil Test)	Std
F01	ICPSO	-15.0000	-13.0000	-15.0000	-14.7600	4.4000E-01
	PSO+DE+Deb+Deb	-15.0000	-12.4531	-15.0000	-14.2944(=)	1.1255E+00
	PSO+Eps	-15.0000	-3.0000	-9.7386	-9.9699(+)	9.4318E+00
	PSO+Deb	-14.9999	-2.0000	-7.0000	-7.8400(+)	1.6640E+01
F02	ICPSO	-0.8036	-0.6969	-0.7866	-0.7801	6.1414E-04
	PSO+DE+Deb+Deb	-0.7963	-0.6604	-0.7768	-0.7638(=)	1.3364E-03
	PSO+Eps	-0.7831	-0.3686	-0.6056	-0.6001(+)	1.1111E-02
	PSO+Deb	-0.6896	-0.3919	-0.6025	-0.5794(+)	7.6440E-03
F03	ICPSO	-1.0005	-1.0005	-1.0005	-1.0005	2.8144E-31
	PSO+DE+Deb+Deb	-1.0005	-1.0005	-1.0005	-1.0005(=)	6.0808E-31
	PSO+Eps	-0.9881	-0.2862	-0.9153	-0.8223(+)	4.6320E-02
	PSO+Deb	-0.5185	0.0000	-0.0624	-0.1005(+)	1.7213E-02
F04	ICPSO	-30665.5387	-30665.5387	-30665.5387	-30665.5387	1.3786E-23
	PSO+DE+Deb+Deb	-30665.5387	-30665.5387	-30665.5387	-30665.5387(=)	1.3786E-23
	PSO+Eps	-30665.5387	-30186.1587	-30665.5387	-30607.9459(+)	2.5269E+04
	PSO+Deb	-30665.5387	-30665.5387	-30665.5387	-30665.5387(=)	5.4044E-12
F05	ICPSO	5126.4967	5126.4967	5126.4967	5126.4967	7.5825E-24
	PSO+DE+Deb+Deb	5126.4967	5126.4967	5126.4967	5126.4967(=)	7.7548E-24
	PSO+Eps	5131.0320	6112.2237	5162.0209	5324.5716(+)	1.0783E+05
	PSO+Deb	5126.4967	6108.2908	5195.9861	5412.8367(+)	1.5829E+05
F06	ICPSO	-6961.8139	-6961.8139	-6961.8139	-6961.8139	0
	PSO+DE+Deb+Deb	-6961.8139	-6961.8139	-6961.8139	-6961.8139(=)	0
	PSO+Eps	-6961.8139	-6961.8139	-6961.8139	-6961.8139(=)	0
	PSO+Deb	-6961.8139	-6961.8139	-6961.8139	-6961.8139(=)	0

Continued on next page

Prob.	Algorithm	Best	Worst	Median	Mean(Wil Test)	Std
F07	ICPSO	24.3062	24.3062	24.3062	24.3062	2.7558E-28
	PSO+DE+Deb+Deb	24.3062	24.3062	24.3062	24.3062(=)	6.6264E-29
	PSO+Eps	24.6127	117.8906	26.6317	33.0362(+)	3.5326E+02
	PSO+Deb	24.5216	192.9197	27.9271	42.5169(+)	1.6092E+03
F08	ICPSO	-0.0958	-0.0958	-0.0958	-0.0958	1.2840E-34
	PSO+DE+Deb+Deb	-0.0958	-0.0958	-0.0958	-0.0958(=)	1.4444E-34
	PSO+Eps	-0.0958	-0.0958	-0.0958	-0.0958(=)	1.4444E-34
	PSO+Deb	-0.0958	-0.0958	-0.0958	-0.0958(=)	1.2037E-34
F09	ICPSO	680.6301	680.6301	680.6301	680.6301	5.9238E-26
	PSO+DE+Deb+Deb	680.6301	680.6301	680.6301	680.6301(=)	7.3778E-26
	PSO+Eps	680.6309	681.4069	680.6930	680.8072(+)	5.1920E-02
	PSO+Deb	680.6319	681.1612	680.6789	680.7082(+)	1.1588E-02
F10	ICPSO	7049.2480	7250.9673	7049.2480	7065.3856	3.1196E+03
	PSO+DE+Deb+Deb	7049.2480	7049.2480	7049.2480	1.4717E-23(=)	1.4717E-23
	PSO+Eps	7122.9001	11100.0001	7592.5522	7903.1978(+)	7.3655E+05
	PSO+Deb	7140.0571	9426.5647	7633.7621	7788.8123(+)	3.0280E+05
F11	ICPSO	0.7499	0.7499	0.7499	0.7499	1.2840E-32
	PSO+DE+Deb+Deb	0.7499	0.7499	0.7499	0.7499(=)	1.2840E-32
	PSO+Eps	0.7499	0.7499	0.7499	0.7499(=)	1.2840E-32
	PSO+Deb	0.7499	0.7499	0.7499	0.7499(=)	1.2840E-32
F12	ICPSO	-1.0000	-1.0000	-1.0000	-1.0000	0
	PSO+DE+Deb+Deb	-1.0000	-1.0000	-1.0000	-1.0000(=)	0
	PSO+Eps	-1.0000	-1.0000	-1.0000	-1.0000(=)	0
	PSO+Deb	-1.0000	-1.0000	-1.0000	-1.0000(=)	0
F13	ICPSO	0.0539	0.0539	0.0539	0.0539	5.7979E-34
	PSO+DE+Deb+Deb	0.0539	1.0000	0.4388	0.3227(+)	5.4626E-02
	PSO+Eps	0.2714	1.2208	0.9708	0.8245(+)	6.8868E-02
	PSO+Deb	0.2655	2.0483	0.9216	0.8767(+)	9.9082E-02
+/ = /−	PSO+DE+Deb+Deb	1/12/0	PSO+Eps	9/4/0	PSO+Deb	8/5/0

Figure 4 shows the mean convergence of the 13 functions for the 4 algorithms. The x-axis is the number of evaluations of the fitness function, and the y-axis is the mean of the function values for 25 runs. As shown in Figure 4, since the algorithm that uses the ε-constraint method as the constraint-guided strategy has a moderate slack effect on the infeasible solutions in the early stage, the convergence range is expanded and the convergence speed is reduced in the early stage. The results for F06, F07, F10 and F11 are shown in Figure 4. The convergence is accelerated after a short period of adjustment and is more stable for the same number of evaluations. The result for F05 in Figure 4 further shows that the ε-constraint method guides the algorithm update process. To enhance the global search ability in the early stage, some infeasible solutions with smaller objective function values are allowed to appear. However, as the number of evolutionary generations increases, the algorithm can instead converge to the optimal solution more quickly and stably.



(a) F01 (b) F02 (c) F03

(d) F04 (e) F05 (f) F06

(g) F07 (h) F08 (i) F09

(j) F10 (k) F11 (l) F12

(m) F13

Figure 4. Average convergence curve of 4 algorithms on 13 benchmark functions.

(2) Performance evaluation using CEC2017 benchmark functions

In this section, the scalability of the proposed ICPSO algorithm in addressing constrained optimization problems is further evaluated using another 28 benchmark functions that were introduced in CEC2017. These CEC2017 benchmark problems pose a greater challenge compared to those of CEC2006, as they encompass a broader array of constraints. Note that the experimental results of those benchmark functions, in which both the ICPSO and the other three algorithms cannot find feasible solutions consistently, have been excluded from the comparative analysis (C17–C19 and C26–C28).

Figure 5 shows the proportion of optimal solutions that are feasible for the 22 benchmark functions after 25 independent runs of each algorithm. For C11, except for the ICPSO algorithm with a success rate of 4%, all other algorithms did not obtain feasible solutions in 25 runs; for C06, the success rate of ICPSO is 100%, PSO+DE+DEB+DEB is 96%, and the other two variants of PSO algorithms obtained only 80% and 60%, respectively; and the PSO+Deb algorithm did not achieve a success rate of 100% for C22. It is not difficult to find that the IPSO algorithm with a composite optimization strategy can improve the success rate of PSO algorithm in solving these 22 constrained optimization problems.

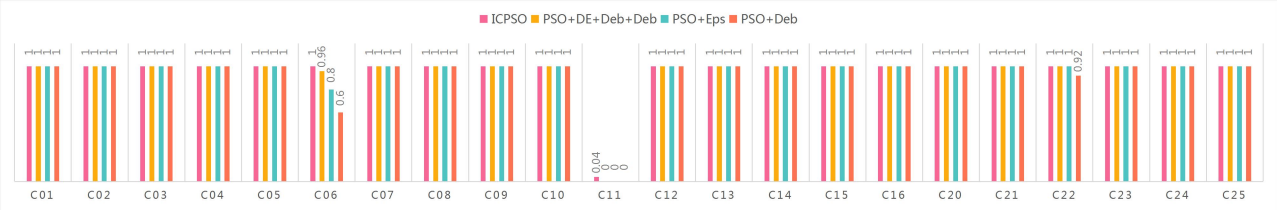


Figure 5. Proportion of feasible solutions for 22 benchmark functions from the IEEE CEC2017 dataset.

The Best, Worst, Median, Mean, and Std values obtained by the compared algorithms in solving CEC2017 benchmark functions after 25 runs are presented in Table 4. Figure 6 shows the mean convergence of the 22 functions for the 4 algorithms. For C04, C07, C13–C16, C20, C21, and C23, the ICPSO algorithm achieved the best results. As shown in Figure 6, ICPSO algorithm can stably converge on these functions. For C1–C3, C5, C8–C10, C12, and C24, both ICPSO and PSO+DE+Deb+Deb performed well, especially for C01–C03 and C05. The two algorithms stably converged to 0 in 25 calculations. For C06, combined with the success rate and Figure 6(f), it can be found that only ICPSO is much better than the other three algorithms in solving this problem. C11 is difficult to solve. Except for ICPSO, other 3 algorithms cannot obtain its feasible solution. The success rate of ICPSO is also very low. Therefore, effective solving strategies need to be set according to the characteristics of this function. The results presented in Tables 4 show that ICPSO is superior to the other 3 algorithms for solving the 22 functions; that is, the composite strategy for updating personal best position can improve the ability of the PSO algorithm to solve the constrained optimization problems. The results of the Wilcoxon test show that the performance of ICPSO is significantly better than PSO+Eps and PSO+Deb, which is equivalent to PSO+DE+Deb+Deb. However, combined with the success rates, the composite strategy can be used to solve these 22 problems.

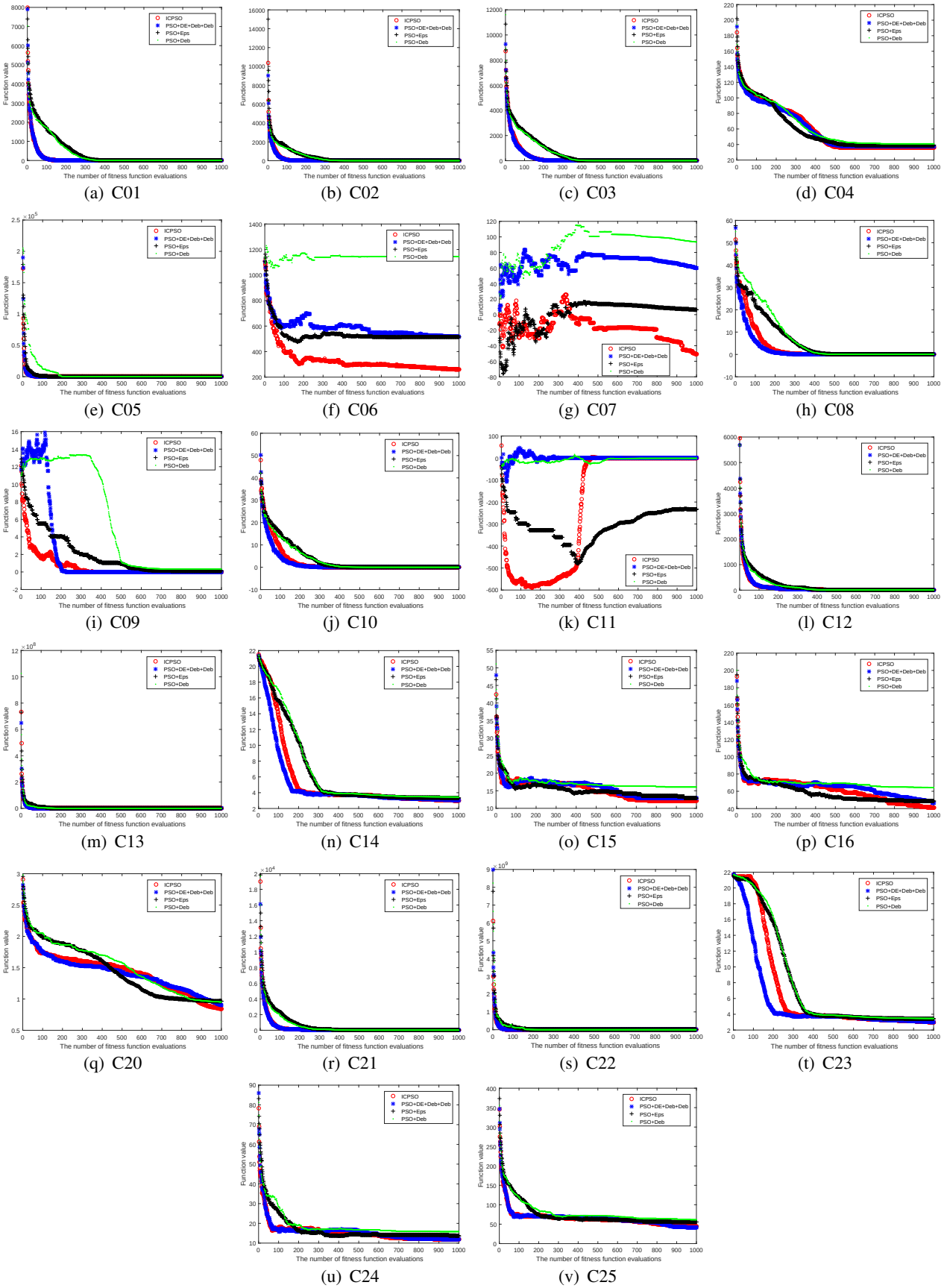


Figure 6. Average convergence curve of 4 algorithms on 22 benchmark functions.

Table 4. Results of 4 improved PSO algorithms on 22 benchmark functions.

Prob.	Algorithm	Best	Worst	Median	Mean(Wil Test)	Std
C01	ICPSO	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	PSO+DE+Deb+Deb	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00(=)	0.0000E+00
	PSO+Eps	0.0000E+00	1.1795E-12	2.8984E-19	4.8992E-14(+)	5.5547E-26
	PSO+Deb	8.4754E-27	1.4767E-14	4.9003E-18	8.5885E-16(+)	9.3175E-30
C02	ICPSO	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	PSO+DE+Deb+Deb	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00(=)	0.0000E+00
	PSO+Eps	3.3494E-23	2.8319E-16	2.1563E-19	2.7667E-17(+)	5.0451E-33
	PSO+Deb	1.8814E-24	7.8490E-15	2.3722E-18	7.1620E-16(+)	4.0864E-30
C03	ICPSO	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	PSO+DE+Deb+Deb	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00(=)	0.0000E+00
	PSO+Eps	7.3610E-23	2.3752E-12	6.1935E-18	1.2111E-13(+)	2.2544E-25
	PSO+Deb	7.9527E-24	4.5144E-11	2.4400E-19	1.8059E-12(+)	8.1520E-23
C04	ICPSO	1.6914E+01	4.5768E+01	3.6813E+01	3.5978E+01	3.4456E+01
	PSO+DE+Deb+Deb	1.5919E+01	5.5717E+01	3.7808E+01	3.7585E+01(=)	9.1817E+01
	PSO+Eps	1.8904E+01	4.7758E+01	3.7809E+01	3.8803E+01(=)	5.4940E+01
	PSO+Deb	1.7924E+01	5.5717E+01	4.2783E+01	4.0969E+01(+)	7.1367E+01
C05	ICPSO	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
	PSO+DE+Deb+Deb	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00(=)	0.0000E+00
	PSO+Eps	1.4787E-11	5.5977E+00	1.7687E+00	1.9865E+00(+)	3.2524E+00
	PSO+Deb	8.7359E-17	9.5023E+00	7.4434E-01	1.8852E+00(+)	6.8111E+00
C06	ICPSO	8.2871E+01	6.7691E+02	2.3032E+02	2.6020E+02	2.5461E+04
	PSO+DE+Deb+Deb	9.9261E+01	2.4077E+03	4.6086E+02	5.1682E+02(+)	1.9966E+05
	PSO+Eps	1.6038E+02	1.7553E+03	4.3420E+02	5.1558E+02(+)	1.3317E+05
	PSO+Deb	2.5127E+02	2.5551E+03	1.0127E+03	1.1444E+03(+)	3.3382E+05
C07	ICPSO	-2.5992E+02	1.5657E+02	-5.3364E+01	-5.0976E+01	7.8377E+03
	PSO+DE+Deb+Deb	-1.6392E+02	1.8198E+02	4.9387E+01	5.9887E+01(+)	8.1840E+03
	PSO+Eps	-2.2107E+02	2.5240E+02	-5.4356E-01	-2.1823E+00(=)	1.1646E+04
	PSO+Deb	-2.5066E+01	2.4529E+02	8.8854E+01	9.3390E+01(+)	5.6346E+03
C08	ICPSO	-1.3484E-03	-1.3484E-03	-1.3484E-03	-1.3484E-03	2.8567E-22
	PSO+DE+Deb+Deb	-1.3484E-03	-1.3484E-03	-1.3484E-03	-1.3484E-03(=)	1.3107E-21
	PSO+Eps	-1.0336E-03	1.9229E-03	1.8862E-05	2.2378E-04(+)	8.0095E-07
	PSO+Deb	-1.0562E-03	2.1660E-03	-5.2108E-04	-2.3969E-04(+)	5.2691E-07
C09	ICPSO	-4.9752E-03	-4.9752E-03	-4.9752E-03	-4.9752E-03	0.0000E+00
	PSO+DE+Deb+Deb	-4.9752E-03	-4.9752E-03	-4.9752E-03	-4.9752E-03(=)	0.0000E+00
	PSO+Eps	-4.9752E-03	1.1216E+00	-4.8661E-03	1.2915E-01(+)	9.4323E-02
	PSO+Deb	-4.9752E-03	4.7827E+00	-4.7996E-03	3.1463E-01(+)	1.0059E+00
C10	ICPSO	-5.0965E-04	-5.0965E-04	-5.0965E-04	-5.0965E-04	1.5395E-20
	PSO+DE+Deb+Deb	-5.0965E-04	-5.0965E-04	-5.0965E-04	-5.0965E-04(=)	2.9316E-22
	PSO+Eps	-3.8375E-04	7.0034E-04	-1.1079E-04	-2.8348E-05(+)	8.0706E-08
	PSO+Deb	-3.9693E-04	5.9321E-04	-1.6193E-04	-4.3811E-05(+)	7.9731E-08

Prob.	Algorithm	Best	Worst	Median	Mean(Wil Test)	Std
C11	ICPSO	-1.4554E+00	1.5215E+00	-6.3933E-01	-2.3206E-01	8.8931E-01
	PSO+DE+Deb+Deb	-9.8407E-01	1.4855E+00	4.5590E-01	2.7254E-01(=)	5.8345E-01
	PSO+Eps	-7.3529E+02	5.5079E+00	-3.1108E+00	-2.3229E+02(+)	8.4405E+04
	PSO+Deb	-1.9832E+01	4.0920E+00	-1.8455E+00	-1.3276E+00(=)	2.0913E+01
C12	ICPSO	-3.9879E+00	3.9879E+00	3.9879E+00	3.9879E+00	2.3666E-30
	PSO+DE+Deb+Deb	3.9879E+00	3.9879E+00	3.9879E+00	3.9879E+00(=)	1.0518E-30
	PSO+Eps	3.9879E+00	2.2842E+01	3.9879E+00	7.2027E+00(+)	3.7795E+01
	PSO+Deb	3.9879E+00	2.2790E+01	3.9879E+00	7.1918E+00(+)	3.7575E+01
C13	ICPSO	0.0000E+00	3.9866E+00	0.0000E+00	1.5946E-01	6.3571E-01
	PSO+DE+Deb+Deb	0.0000E+00	3.9866E+00	0.0000E+00	6.3785E-01(=)	2.2250E+00
	PSO+Eps	6.7949E-10	6.5227E+01	2.6291E+00	1.5059E+01(+)	6.3555E+02
	PSO+Deb	3.0976E-13	1.1698E+02	3.9871E+00	2.2894E+01(+)	1.0691E+03
C14	ICPSO	2.5960E+00	3.5734E+00	2.9585E+00	2.9945E+00	6.1596E-02
	PSO+DE+Deb+Deb	2.4024E+00	3.6975E+00	3.0641E+00	3.0485E+00(=)	1.1024E-01
	PSO+Eps	2.8896E+00	3.7035E+00	3.3127E+00	3.3353E+00(+)	3.9312E-02
	PSO+Deb	3.0536E+00	3.7656E+00	3.4716E+00	3.4532E+00(+)	2.8797E-02
C15	ICPSO	5.4977E+00	1.8064E+01	1.1781E+01	1.2158E+01	6.8429E+00
	PSO+DE+Deb+Deb	8.6393E+00	1.8064E+01	1.1781E+01	1.2912E+01(=)	6.4810E+00
	PSO+Eps	5.4977E+00	1.8064E+01	1.1781E+01	1.2912E+01(=)	8.1260E+00
	PSO+Deb	1.1781E+01	2.1206E+01	1.4922E+01	1.6053E+01(+)	1.1416E+01
C16	ICPSO	2.0420E+01	5.6549E+01	4.3982E+01	4.1155E+01	1.1700E+02
	PSO+DE+Deb+Deb	3.1416E+01	8.7964E+01	4.3982E+01	4.6433E+01(=)	1.4456E+02
	PSO+Eps	2.5133E+01	6.9115E+01	5.0265E+01	4.8757E+01(=)	9.8384E+01
	PSO+Deb	5.0265E+01	8.3252E+01	6.2832E+01	6.4088E+01(+)	9.7051E+01
C20	ICPSO	4.9902E-01	1.3840E+00	8.0014E-01	8.4173E-01	4.7821E-02
	PSO+DE+Deb+Deb	5.2126E-01	1.5701E+00	8.0599E-01	9.0244E-01(=)	7.8540E-02
	PSO+Eps	4.5423E-01	1.6113E+00	9.4499E-01	9.6239E-01(+)	6.0995E-02
	PSO+Deb	5.8430E-01	1.2462E+00	9.5933E-01	9.5106E-01(+)	3.0787E-02
C21	ICPSO	3.9879E+00	3.9879E+00	3.9879E+00	3.9879E+00	3.2951E-30
	PSO+DE+Deb+Deb	3.9879E+00	1.4603E+01	3.9879E+00	4.4125E+00(=)	4.5070E+00
	PSO+Eps	3.9879E+00	2.2850E+01	3.9879E+00	9.0290E+00(+)	6.1512E+01
	PSO+Deb	3.9879E+00	2.2793E+01	3.9880E+00	9.9733E+00(+)	5.4571E+01
C22	ICPSO	3.4625E-27	3.9866E+00	3.6837E-27	9.5678E-01	3.0196E+00
	PSO+DE+Deb+Deb	3.4625E-27	3.9866E+00	3.4625E-27	6.3785E-01(=)	2.2250E+00
	PSO+Eps	3.0195E-07	1.9152E+05	4.3250E+00	1.0107E+04(+)	1.5656E+09
	PSO+Deb	2.9269E-03	2.0452E+05	1.4042E+01	1.6059E+04(+)	3.0775E+09
C23	ICPSO	2.4447E+00	3.6105E+00	2.9945E+00	2.9792E+00	1.0704E-01
	PSO+DE+Deb+Deb	2.4089E+00	3.4898E+00	2.9703E+00	2.9962E+00(=)	5.3903E-02
	PSO+Eps	2.8923E+00	3.7480E+00	3.4481E+00	3.4188E+00(+)	4.5688E-02
	PSO+Deb	3.0375E+00	3.7435E+00	3.5176E+00	3.4698E+00(+)	3.3942E-02

Prob.	Algorithm	Best	Worst	Median	Mean(Wil Test)	Std
C24	ICPSO	8.6393E+00	1.8064E+01	1.1781E+01	1.2032E+01	5.6915E+00
	PSO+DE+Deb+Deb	5.4977E+00	1.4922E+01	1.1781E+01	1.2032E+01(=)	4.8690E+00
	PSO+Eps	8.6393E+00	1.8064E+01	1.4922E+01	1.3666E+01(+)	5.7573E+00
	PSO+Deb	1.1781E+01	2.4347E+01	1.4922E+01	1.5802E+01(+)	1.3587E+01
C25	ICPSO	2.5133E+01	6.9115E+01	4.3982E+01	4.4359E+01	1.0287E+02
	PSO+DE+Deb+Deb	2.0420E+01	6.2832E+01	4.3982E+01	4.1532E+01(=)	1.5135E+02
	PSO+Eps	3.7699E+01	7.6969E+01	5.1836E+01	5.5166E+01(+)	1.7258E+02
	PSO+Deb	4.3982E+01	8.1681E+01	6.2832E+01	6.1387E+01(+)	1.0896E+02
+/ = /−	PSO+DE+Deb+Deb	2/20/0	PSO+Eps	18/4/0	PSO+Deb	21/1/0

4.2.2. Comparison and analysis regarding algorithms in other studies

To further validate the performance of the ICPSO algorithm in solving constrained problems, the ICPSO algorithm was compared with two intelligent algorithms proposed in recent years and three improved PSO algorithms with the DE strategy for CEC2006 benchmark functions; and compared with three new intelligent algorithms for CEC2017 benchmark functions. The data of all comparison algorithms are derived from cited references, the main parameters of each algorithm were set as follows:

mcABC algorithm: The number of evaluations of the fitness function was 2.5×10^5, and the correction rate was 0.8;

IFOA algorithm: $N = 100$, $T = 5000$, Selection probability $P = 0.8$;

PSO-DE algorithm: $c_1 = c_2 = 1.5$, $\omega_{max} = 0.8$, $\omega_{min} = 0.4$, $F = 0.95$, $CR = 0.95$;

HMPSO algorithm: The number of evaluations of the fitness function was 3×10^5, $N = 60$, the size of each sub-swarm $N_s = 8$, $F = 0.7$, $CR = 1.0$, $T = 2500$;

PSODE algorithm: The number of evaluations of the fitness function was $\in [1.06\times10^4, 1.401\times10^5]$, $F \in [0.9, 1.0]$, and $CR \in [0.95, 1]$;

CMPSOWV algorithm: $N = 100$, subpopulation $N^k = 10$ with $k = 1, \cdots, 10$, acceleration coefficients $c_1 = c_2 = c_3 = 4.1/3$;

GA-TDX algorithm: $N = 100$, $T = 1000$, Penalty parameter $m = 10^10$, the proportion of the best group in the population $\beta = 0.2$, the shape parameter $\gamma = 6$;

ECO-HCT algorithm: The number of evaluations of the fitness function was 2×10^5, $N = 50$, $F = 0.8$, $CR = 0.9$, Elite replacement strategy control parameters $\omega_{min} = 0.2, \omega_{max} = 0.8$, Archive A_2 size $\alpha_1 = 40$.

Table 5 shows that the ICPSO algorithm achieves the optimal mean function values for F03–F09 and F11–F13. The standard deviations for F08 and F11 obtained by the ICPSO algorithm are inferior to those obtained by the HMPSO and PSO-DE algorithms, respectively; however, the accuracy of ICPSO is greater than 10^{-30}, which reflects the stability of this algorithm. For F01, F02, and F10, the performance of the ICPSO algorithm needs to be improved.

Table 5. Results of ICPSO and other 5 algorithms on 13 benchmark functions.

Pro.	Ind.	mcABC [7]	IFOA [8]	PSO-DE [18]	HMPSO [17]	PSODE [16]	ICPSO
F01	Mean	-15.0000	-15.0000	-15.0000	-15.0000	-15.0000	-14.7600
	Std	0	0	0	0	2.1E08	4.4000E-01
F02	Mean	-0.8017	-0.8037	-0.8036	-0.8028	-0.7567	-0.7801
	Std	3.9463E-03	2.39E-06	4.50E-03	2.6784E-03	3.3E-02	6.1414E-04
F03	Mean	-1.0039	-1.0005	-1.0000	-1.0000	-1.0005	-1.0005
	Std	3.610E-4	9.94E-16	6.20E-16	2.0501E-03	3.8E-12	2.8144E-31
F04	Mean	-30665.538	-30665.538	-30665.538	-30665.538	-30665.5387	-30665.5387
	Std	–	2.18E-11	3.71E-12	7.4260E-12	8.3E-10	1.3786E-23
F05	Mean	5221.833	5126.4954	5154.8009	5126.497	–	5126.4967
	Std	1.3474E+02	2.47E-08	1.60E+02	2.9529E-12	–	7.5825E-24
F06	Mean	-6961.814	-6961.8139	-6961.8139	-6961.8139	-6961.8139	-6961.8139
	Std	–	1.82E-12	1.90E-12	0	2.3E-09	0
F07	Mean	24.4096	24.3064	24.3064	24.3064	24.3064	24.3064
	Std	8.2581E-02	1.29E-04	1.10E-14	1.4025E-11	1.3E-06	2.7558E-28
F08	Mean	-0.0958	-0.0958	-0.0958	-0.0958	-0.0958	-0.0958
	Std	–	8.33E-17	2.80E-17	0	1.3E-12	1.2840E-34
F09	Mean	680.6301	680.6301	680.6301	680.6301	680.6301	680.6301
	Std	4.217E-03	2.02E-08	4.50E-13	2.8892E-13	4.6E-13	5.9238E-26
F10	Mean	7246.649	7 049.2607	7049.2480	7049.2480	7049.2480	7065.3856
	Std	1.8107E+02	6.40E-03	4.90E-12	1.2125E-10	3.0E-05	3.1196E+03
F11	Mean	0.75	0.75	0.7500	0.7499	0.7500	0.7499
	Std	2.69E-05	2.70E-11	0	1.1331E-16	2.5E-07	1.2840E-32
F12	Mean	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000
	Std	0	0	0	0	0	0
F13	Mean	0.9126	0.0540	–	0.2932	–	0.0539
	Std	1.3273E-01	6.98E-05	–	1.4041E-01	–	5.7979E-34

Note: Where, '–' is not involved in the literature.

In Table 6, ∇ represents a method cannot achieve at least a feasible solution on a benchmark function at the end of 25 runs. From Table 6, we observe that ICPSO is able to solve these CEC2017 benchmark functions, achieving 8 best mean values and 6 second-best mean values. This indicates promising search accuracies in solving these challenging problems. Moreover, other algorithms such as CMPSOWV, GA-TDX, and ECO-HCT are reported to be able to produce 6, 6, and 10 best mean values, respectively, in solving all CEC2017 benchmark functions. Specifically, for C11, ECO-HCT fails to produce a feasible solution.

Table 6. Results of ICPSO and other 3 algorithms on 22 benchmark functions.

Pro.	Ind.	CMPSOWV [20]	GA-TDX [11]	ECO-HCT [12]	ICPSO
C01	Mean	1.77E-02	2.00E-17	0.00E+00	0.00E+00
	Std	1.60E-02	5.23E-17	0.00E+00	0.00E+00
C02	Mean	3.39E+00	2.31E-16	0.00E+00	0.00E+00
	Std	8.22E-01	1.10E-15	0.00E+00	0.00E+00
C03	Mean	1.20E-02	1.67E+04	0.00E+00	0.00E+00
	Std	1.41E-02	1.20E+04	0.00E+00	0.00E+00
C04	Mean	1.37E+01	2.81E+01	3.87E+01	3.60E+01
	Std	1.06E-01	5.84E+00	2.20E+01	3.45E+01
C05	Mean	4.70E-01	4.80E-01	0.00E+00	0.00E+00
	Std	3.02E-01	1.32E+00	0.00E+00	0.00E+00
C06	Mean	7.61E+01	9.06E+01	1.79E+02	2.60E+02
	Std	2.50E+00	4.94E+01	7.68E+01	2.55E+04
C07	Mean	-7.02E+01	9.77E+01	-4.24E+01	-5.10E+01
	Std	3.72E+01	5.91E+01	8.80E+01	7.78E+03
C08	Mean	-1.21E-03	1.07E+02	-1.35E-03	1.35E-03
	Std	2.28E-04	2.42E-06	2.03E-14	2.86E-22
C09	Mean	-4.47E-03	1.86E+0	5.98E-03	-4.98E-03
	Std	7.70E-04	3.19E-01	3.86E-02	0.00E+00
C10	Mean	-5.09E-04	4.34E-07	-5.10E-04	-5.10E-04
	Std	9.81E-06	6.98E-07	3.17E-09	1.54E-20
C11	Mean	-3.37E+01	3.07E+00	∇	-2.32E-01
	Std	9.63E+01	2.27E+00	∇	8.89E-01
C12	Mean	4.44E+00	2.58E-07	6.73E+00	3.99E+00
	Std	5.41E-01	2.17E-07	5.80E+00	2.37E-30
C13	Mean	9.19E+00	7.99E-01	1.59E-01	1.59E-01
	Std	1.43E-01	1.63E+00	7.97E-01	6.36E-01
C14	Mean	1.13E+00	6.37E-01	2.38E+00	2.99E+00
	Std	1.34E-02	3.31E-01	1.40E-02	6.16E-02
C15	Mean	1.05E+01	4.67E+00	1.74E+01	1.22E+01
	Std	4.76E+00	2.57E+00	2.40E+00	6.84E+00
C16	Mean	6.28E+01	1.28E+01	1.25E+01	4.12E+01
	Std	9.29E+00	6.87E+00	5.97E+00	1.17E+02
C20	Mean	8.01E-01	2.35E-01	1.28E+00	8.42E-01
	Std	1.28E-01	1.07E-01	2.18E-01	4.78E-02
C21	Mean	7.87E+00	4.25E-01	7.31E+00	3.99E+00
	Std	2.29E+00	2.12E+00	6.82E+00	3.30E-30
C22	Mean	9.41E+01	2.29E+10	1.58E-24	9.57E-01
	Std	8.07E+01	9.31E+10	3.08E-24	3.02E+00

Continued on next page

Pro.	Ind.	CMPSOWV [20]	GA-TDX [11]	ECO-HCT [12]	ICPSO
C23	Mean	2.82E+00	7.19E-01	2.38E+00	2.98E+00
	Std	4.51E-01	2.57E-01	1.25E-02	1.07E-01
C24	Mean	5.68E+00	8.04E+00	7.38E+00	1.20E+01
	Std	4.66E-02	2.04E+00	4.16E+00	5.69E+00
C25	Mean	7.10E+01	3.63E+03	1.64E+01	4.44E+01
	Std	8.98E+00	4.96E+03	7.62E+00	1.03E+02

4.3. Engineering applications

To further verify the performance of ICPSO, four real-world application examples are used [27]. The ICPSO algorithm is tested against three of latest evolutionary algorithms: the self-adaptive spherical search algorithm (SASS) [28], L-SHADE for constrained optimization with Lévy flight (COLSHADE) [29], and modified covariance matrix adaptation evolution strategy (sCMAgES) [30].

4.3.1. Pressure vessel design

The goal of this problem is to optimize the cost of vessel welding, material, and formation. This problem has four constraints that need to be satisfied, and four variables are used to calculate the objective function: Shell thickness, head thickness, inner radius, and vessel length (excluding the head). The current theoretical optimal solution is 5885.3327736. The mathematical model of this problem is as follows:

$$\begin{aligned}
\min \quad & f(x) = 1.7781 z_2 x_3^2 + 0.6224 z_1 x_3 x_4 + 3.1661 z_1^2 x_4 + 19.84 z_1^2 x_4 \\
s.t. \quad & g_1(x) = 0.00954 x_3 - z_2 \le 0, \\
& g_2(x) = 0.0193 x_3 - z_1 \le 0, \\
& g_3(x) = x_4 - 240 \le 0, \\
& g_4(x) = -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \le 0, \\
& z_1 = 0.0625 x_1, \\
& z_2 = 0.0625 x_2, \\
& 10 \le x_3, x_4 \le 200, \\
& 1 \le x_1, x_2 \le 99 (integer).
\end{aligned} \tag{4.1}$$

Table 7. Results of each algorithm in pressure vessel design.

Prob.	ICPSO	SASS	sCMAgES	COLSHADH
Best	6059.7143	6059.7143	6059.7143	6059.7143
Worst	6059.7143	6059.7143	6370.7797	6090.5262
Median	6059.7143	6059.7143	6063.7632	6059.7143
Mean	6059.7143	6059.7143	6088.6007	6062.1793
Std	1.7868E-17	3.7130E-12	6.6365E+01	8.359059
Feasible rate	100	100	100	100

Table 7 lists the calculation results of the four algorithms for the pressure vessel design problem. The overall results have certain deviations from the optimal solution. However, the ICPSO and SASS algorithms perform slightly better than the other two algorithms, and the ICPSO algorithm outperforms the SASS algorithm in terms of standard deviation by 5 orders of magnitude, indicating that the ICPSO algorithm is more stable.

4.3.2. Weight minimization of a reducer

This problem involves the design of a small aircraft engine reducer. The current theoretical optimal solution is 2994.4244658. The mathematical model of this problem is as follows:

$$
\begin{aligned}
\min \quad & f(x) = 0.7854x_2^2x_1(14.9334x_3 - 43.0934 + 3.3333x_3^2) + 0.7854(x_5x_7^2 + x_4x_6^2) \\
& \qquad -1.508(x_7^2 + x_6^2) + 7.447(x_7^3 + x_6^3) \\
s.t. \quad & g_1(x) = -x_1x_2^2x_3 + 27 \le 0, \\
& g_2(x) = -x_1x_2^2x_3^2 + 397.5 \le 0, \\
& g_3(x) = -x_2x_6^4x_3x_4^{-3} + 1.93 \le 0, \\
& g_4(x) = -x_2x_7^4x_3x_5^{-3} + 1.93 \le 0, \\
& g_5(x) = 10x_6^{-3}\sqrt{16.91 \times 10^6 + (745x_4x_2^{-1}x_3^{-1})^2} - 1100 \le 0, \\
& g_6(x) = 10x_7^{-3}\sqrt{157.5 \times 10^6 + (745x_5x_2^{-1}x_3^{-1})^2} - 850 \le 0, \\
& g_7(x) = x_2x_3 - 40 \le 0, \\
& g_8(x) = -x_1x_2^{-1} + 5 \le 0, \\
& g_9(x) = x_1x_2^{-1} - 12 \le 0, \\
& g_{10}(x) = 1.5x_6 - x_4 + 1.9 \le 0, \\
& g_{11}(x) = 1.1x_7 - x_5 + 1.9 \le 0, \\
& 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 2.6 \le x_1 \le 3.6, \\
& 5 \le x_7 \le 5.5, 7.3 \le x_4, x_5 \le 8.3, 2.9 \le x_6 \le 3.9.
\end{aligned} \tag{4.2}
$$

Table 8 lists the calculation results of the weight minimization problem for the reducer. For this problem, all the algorithms obtain similar results. The stability of ICPSO, SASS, and COLSHADE is similar, and all reach 10^{-13}, indicating that the ICPSO algorithm is stable and effective at solving this type of problem.

Table 8. Results of each algorithm in pressure vessel design.

Prob.	ICPSO	SASS	sCMAgES	COLSHADH
Best	2994.4245	2994.4245	2994.4245	2994.4245
Worst	2994.4245	2994.4245	2994.4245	2994.4245
Median	2994.4245	2994.4245	2994.4245	2994.4245
Mean	2994.4245	2994.4245	2994.4244	2994.4245
Std	8.6529E-13	4.6412E-13	2.7723E-12	4.5475E-13
Feasible rate	100	100	100	100

4.3.3. Design of tension/compression springs (Case 1)

The goal of this problem is to optimize the weight of a tension or compression spring. This problem contains four constraints and uses three variables to calculate the weight: The wire diameter, average coil diameter, and number of active coils. The current theoretical optimal solution is 0.012665232788. The mathematical model of this problem is as follows:

$$\begin{aligned} \min \quad & f(x) = x_1^2 x_2 (2 + x_3) \\ s.t. \quad & g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0, \\ & g_2(x) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leq 0, \\ & g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, \\ & g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0, \\ & 0.05 \leq x_1 \leq 2.00, \\ & 0.25 \leq x_2 \leq 1.30, \\ & 2.00 \leq x_3 \leq 15.00. \end{aligned} \tag{4.3}$$

Table 9 lists the calculation results for the tension/compression spring design problem (Case 1). For this problem, all the algorithms obtain similar results. Among them, SASS is the most stable, followed by ICPSO, with a standard deviation of 10^{-24}, which is far lower than those of sCMAgES and COLSHADE, demonstrating the stability of the ICPSO algorithm in solving this problem.

Table 9. Results of each algorithm in pressure vessel design.

Prob.	ICPSO	SASS	sCMAgES	COLSHADH
Best	1.2665E-02	1.2665E-02	1.2665E-02	1.2665E-02
Worst	1.2665E-02	1.2665E-02	1.2686E-02	1.2665E-02
Median	1.2665E-02	1.2665E-02	1.2666E-02	1.2665E-02
Mean	1.2665E-02	1.2665E-02	1.2668E-02	1.2665E-02
Std	6.3188E-24	0	4.6358E-06	1.0625E-07
Feasible rate	100	100	100	100

4.3.4. Design of tension/compression springs (Case 2)

The goal of this problem is to optimize the volume of steel wire needed to construct a helical compression spring. There are three design variables in this problem, namely, the outer diameter (continuous variable), the number of spring coils (integer variable), and the spring wire diameter (discrete variable). This problem contains eight nonlinear inequality constraints, and the current theoretical optimal solution is 2.6138840583. The mathematical model of this problem is as follows:

$$
\begin{aligned}
\min \quad & f(x) = \frac{\pi^2 x_2 x_3^2 (x_1+2)}{4} \\
s.t. \quad & g_1(x) = \frac{8000 G_f x_2}{\pi x_3^3} - 189000 \le 0, \\
& g_2(x) = l_f - 14 \le 0, \\
& g_3(x) = 0.2 - x_3 \le 0, \\
& g_4(x) = x_2 - 3 \le 0, \\
& g_5(x) = 3 - \frac{x_2}{x_3} \le 0, \\
& g_6(x) = \sigma_p - 6 \le 0, \\
& g_7(x) = \sigma_p + \frac{700}{K} + 1.05(x_1+2)x_3 - l_f \le 0, \\
& g_8(x) = 1.25 - \frac{700}{K} \le 0, \\
& G_f = \frac{4\frac{x_2}{x_3}-1}{4\frac{x_2}{x_3}-4} + \frac{0.615 x_3}{x_2}, \\
& K = \frac{11.5\times 10^6 x_3^4}{8 x_1 x_2^3}, \sigma_p = \frac{300}{K}, \\
& l_f = \frac{100}{K} + 1.05(x_1+2)x_3, \\
& 1 \le x_1(integer) \le 70, \\
& 0.6 \le x_2(contimuous) \le 3, \\
& x_3(discreate) \in \{0.009, 0.0095, 0.0104, 0.0118, 0.0128, 0.0132, 0.014, 0.015, \\
& 0.0162, 0.0173, 0.018, 0.020, 0.023, 0.025, 0.028, 0.032, 0.035, 0.041, 0.047, 0.054, \\
& 0.063, 0.072, 0.080, 0.092, 0.0105, 0.120, 0.135, 0.148, 0.162, 0.177, 0.192, 0.207, \\
& 0.225, 0.263, 0.283, 0.307, 0.0331, 0.362, 0.394, 0.4375, 0.500\}.
\end{aligned}
\tag{4.4}
$$

Table 10 lists the calculation results for the tension/compression spring design problem (Case 2). For this problem, except for the sCMAgES algorithm, the algorithms all obtain similar results. Among them, the ICPSO algorithm is the most stable, with a standard deviation of 10^{-31}, which is far lower than those of the other algorithms, demonstrating the high stability of the ICPSO algorithm in solving this problem.

Table 10. Results of each algorithm in pressure vessel design.

Prob.	ICPSO	SASS	sCMAgES	COLSHADH
Best	2.6586	2.6586	2.8523	2.6586
Worst	2.6586	2.6586	6.7397	2.6995
Median	2.6586	2.6586	3.8786	2.6586
Mean	2.6586	2.6586	4.2369	2.6618
Std	2.05433E-31	4.5325E-16	1.0566E+00	1.1105E-02
Feasible rate	100	100	100	100

5. Conclusions

Our results present the improved composite particle swarm optimization algorithm algorithm (ICPSO) for solving constrained optimization problems. We develop a new method for updating the personal best position population, which is a composite strategy consisting of two

constrained selections steps and one evolution step. The method utilizes the relaxation mechanism of the ε-constraint to absorb some "excellent" infeasible solutions into the intermediate personal best position population. To accelerate convergence, the DE strategy is used to optimize this population and generate the intermediate evolutionary population. Finally, using a feasibility rule method with strict constraint control, we select individuals with better constraint violation degrees from the two intermediate populations to enter the personal best position population, thereby guiding the population to efficiently converge towards the global optimal position. To verify the effectiveness of the composite strategy in the ICPSO algorithm, we construct three comparative algorithms that use partial strategies: PSO+EPS, PSO+Deb, and PSO+DE+Deb+Deb. The numerical results indicate that the composite strategy in the ICPSO algorithm is effective. When one or more components are missing, the convergence performance of the algorithm is significantly reduced. Finally, comparing the performance of the ICPSO algorithm to that of other algorithms demonstrated the effectiveness and stability of the proposed algorithm for solving the 13 functions from CEC2006, 22 functions from CEC2017, and 4 real-world constraint optimization problems.

In the future, we will mostly focus on two areas. First, more effective mechanisms based on ICPSO will be considered to handle more complex constrained optimization problems, such as large-scale constrained optimization problems [31] and multi-objective constrained optimization problems [32]. Second, more effective and targeted operators will be designed to form an operator pool, enabling the ICPSO algorithm to solve more real-world problems.

Use of AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was supported by the Natural Science Foundation of NingXia Hui Autonomous Region (No. 2021AAC03185), the Research Startup Foundation of North Minzu University (No. 2020KYQD23), the First-Class Disciplines Foundation of NingXia (No. NXYLXK2017B09) and the Major Project of North Minzu University (No. 2019MS003).

Conflict of interest

The authors declare no conflicts of interest.

References

1. W. W. Jia, T. W. Huang, S. T. Qin, A collective neurodynamic penalty approach to nonconvex distributed constrained optimization, *Neural Netw.*, **171** (2024), 145–158. https://doi.org/10.1016/j.neunet.2023.12.011
2. R. Y. Xu, J. Y. Tian, J. F. Li, X. P. Zhai, Trajectory planning of rail inspection robot based on an improved penalty function simulated annealing particle swarm algorithm, *Int. J. Control Autom. Syst.*, **21** (2023), 3368–3381. https://doi.org/10.1007/s12555-022-0163-z

3. Y. Wang, B. C. Wang, H. X. Li, G. G. Yen, Incorporating objective function information into the feasibility rule for constrained evolutionary optimization, *IEEE Trans. Cybern.*, **46** (2016), 2938–2952. https://doi.org/10.1109/TCYB.2015.2493239
4. W. C. Wang, W. C. Tian, K. Chau, H. F. Zang, M. W. Ma, Z. K. Feng, et al., Multi-reservoir flood control operation using improved bald eagle search algorithm with ε-constraint method, *Water*, **15** (2023), 1–24. https://doi.org/10.3390/w15040692
5. Y. Wang, Z. X. Cai, G. Q. Guo, Y. R. Zhou, Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems, *IEEE Trans. Syst. Man Cybern. B*, **37** (2007), 560–575. https://doi.org/10.1109/tsmcb.2006.886164
6. M. A. Jan, M. Sagheer, H. U. Khan, M. I. Uddin, R. A. Khanum, M. Mahmoud, et al., Hybrid stochastic ranking for constrained optimization, *IEEE Access*, **8** (2020), 227270–227287. https://doi.org/10.1109/ACCESS.2020.3044439
7. D. Karaboga, B. Akay, A modified artificial bee colony (ABC) algorithm for constrained optimization problems, *Appl. Soft Comput.*, **11** (2011), 3021–3031. https://doi.org/10.1016/j.asoc.2010.12.001
8. J. P. Shi, P. S. Li, G. P. Liu, P. Liu, Improved fruit fly optimization algorithm for solving constrained problems and engineering applications (Chinese), *Control Decis.*, **36** (2021), 314–324. https://doi.org/10.13195/j.kzyjc.2019.0557
9. H. M. Jia, S. Z. Shi, D. Wu, H. H. Rao, J. R. Zhang, L. Abualigah, Improve coati optimization algorithm for solving constrained engineering optimization problems, *J. Comput. Des. Eng.*, **10** (2023), 2223–2250. https://doi.org/10.1093/jcde/qwad095
10. B. C. Wang, H. X. Li, J. P. Li, Y. Wang, Composite differential evolution for constrained evolutionary optimization, *IEEE Trans. Syst. Man Cybern. Syst.*, **49** (2019), 1482–1495. https://doi.org/10.1109/TSMC.2018.2807785
11. F. L. Wang, G. Xu, M. Wang, An improved genetic algorithm for constrained optimization problems, *IEEE Access*, **11** (2023), 10032–10044. https://doi.org/10.1109/ACCESS.2023.3240467
12. H. Peng, Z. Z. Xu, J. Y. Qian, X. G. Dong, W. Li, Z. J. Wu, Evolutionary constrained optimization with hybrid constraint-handling technique, *Expert Syst. Appl.*, **211** (2023), 118660. https://doi.org/10.1016/j.eswa.2022.118660
13. J. Kennedy, R. Eberhart, Particle swarm optimization, In: *Proceedings of ICNN'95-International Conference on Neural Networks*, **4** (1995), 1942–1948. https://doi.org/10.1109/ICNN.1995.488968
14. Karishma, H. Kumar, A new hybrid particle swarm optimizationalgorithm for optimal tasks scheduling in distributed computing system, *Intell. Syst. Appl.*, **18** (2023), 200219. https://doi.org/10.1016/j.iswa.2023.200219
15. H. C. Lu, H. Y. Tseng, S. W. Lin, Double-track particle swarm optimizer for nonlinear constrained optimization problems, *Inform. Sci.*, **662** (2023), 587–628. https://doi.org/10.1016/j.ins.2022.11.164

16. H. Liu, Z. X. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, *Appl. Soft Comput.*, **10** (2010), 629–640. https://doi.org/10.1016/j.asoc.2009.08.031
17. Y. Wang, Z. X. Cai, A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems, *Front. Comput. Sci. China*, **3** (2009), 38–52. https://doi.org/10.1007/s11704-009-0010-x
18. E. Y. Guo, Y. L. Gao, C. Y. Hu, J. J. Zhang, A hybrid PSO-DE intelligent algorithm for solving constrained optimization problems based on feasibility rules, *Mathematics*, **11** (2023), 1–34. https://doi.org/10.3390/math11030522
19. G. Venter, R. T. Haftka, Constrained particle swarm optimization using a bi-objective formulation, *Struct. Multidiscip. Optim.*, **40** (2010), 65–76. https://doi.org/10.1007/s00158-009-0380-6
20. K. M. Ang, W. H. Lim, N. A. M. Isa, S. S. Tiang, C. H. Wong, A constrained multi-swarm particle swarm optimization without velocity for constrained optimization problems, *Expert Syst. Appl.*, **140** (2020), 112882. https://doi.org/10.1016/j.eswa.2019.112882
21. Y. Shi, R. Eberhart, A modified particle swarm optimizer, In: *1998 IEEE International Conference on Evolutionary Computation Proceedings*, 1998, 69–73. https://doi.org/10.1109/ICEC.1998.699146
22. K. Deb, An efficient constraint handling method for genetic algorithms, *Comput. Method. Appl. Mech. Eng.*, **186** (2000), 311–338. https://doi.org/10.1016/S0045-7825(99)00389-8
23. T. Takahama, S. Sakai, Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation, *IEEE Congress on Evolutionary Computation*, Spain: Barcelona, 2010, 1–9. https://doi.org/10.1109/CEC.2010.5586484
24. Z. Liu, Z. Y. Li, P. Zhu, W. Chen, A parallel boundary search particle swarm optimization algorithm for constrained optimization problems, *Struct. Multidiscip. Optim.*, **58** (2018), 1505–1522. https://doi.org/10.1007/s00158-018-1978-3
25. J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. A. C. Coello, et al., *Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization*, Technical Report, Singapore: Nangyang Technological University, 2006, 1–24.
26. G. Wu, R. Mallipeddi, P. Suganthan, *Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization*, Technical Report, Singapore: Nangyang Technological University, 2017, 1–18.
27. A. Kumar, G. H. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, S. Das, A test-suite of non-convex constrained optimization problems from the real-world and some baseline results, *Swarm Evol. Comput.*, **56** (2020), 100693. https://doi.org/10.1016/j.swevo.2020.100693
28. A. Kumar, S. Das, I. Zelinka, A self-adaptive spherical search algorithm for real-world constrained optimization problems, In: *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, 13–14. https://doi.org/10.1145/3377929.3398186
29. J. Gurrola-Ramos, A. Hernàndez-Aguirre, O. Dalmau-Cedeño, COLSHADE for real-world single-objective constrained optimization problems, *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, 1–8. https://doi.org/10.1109/CEC48606.2020.9185583

30. A. Kumar, S. Das, I. Zelinka, A modified covariance matrix adaptation evolution strategy for real-world constrained optimization problems, In: *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, 11–12. https://doi.org/10.1145/3377929.3398185

31. C. He, L. H. Li, Y. Tian, X. Y. Zhang, R. Cheng, Y. C. Jin, et al., Accelerating large-scale multiobjective optimization via problem reformulation, *IEEE Trans. Evol. Comput.*, **23** (2019), 949–961. https://doi.org/10.1109/TEVC.2019.2896002

32. S. C. Liu, N. Lu, W. J. Hong, C. Qian, K. Tang, Effective and imperceptible adversarial textual attack via multi-objectivization, 2021, arXiv: 2111.01528.

AIMS Press