Mathematical Biosciences and Engineering

MBE, 20(8): 14866–14898.
DOI: 10.3934/mbe.2023666
Received: 23 March 2023
Revised: 21 June 2023
Accepted: 26 June 2023
Published: 10 July 2023

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

Research article

Handling multi-objective optimization problems with a comprehensive indicator and layered particle swarm optimizer

Xianzi Zhang[1], Yanmin Liu[2,*], Jie Yang[2], Jun Liu[1] and Xiaoli Shu[1]

[1] School of Data Science and Information Engineering, Guizhou Minzu University, Guiyang 550025, China
[2] School of Mathematics, Zunyi Normal College, Zunyi 563002, China

* **Correspondence:** Email: yanmin7813@163.com.

Abstract: The multi-objective particle swarm optimization algorithm has several drawbacks, such as premature convergence, inadequate convergence, and inadequate diversity. This is particularly true for complex, high-dimensional, multi-objective problems, where it is easy to fall into a local optimum. To address these issues, this paper proposes a novel algorithm called IMOPSOCE. The innovations for the proposed algorithm mainly contain three crucial factors: 1) an external archive maintenance strategy based on the inflection point distance and distribution coefficient is designed, and the comprehensive indicator (CM) is used to remove the non-dominated solutions with poor comprehensive performance to improve the convergence of the algorithm and diversity of the swarm; 2) using the random inertia weight strategy to efficiently control the movement of particles, balance the exploration and exploitation capabilities of the swarm, and avoid excessive local and global searches; and 3) offering different flight modes for particles at different levels after each update to further enhance the optimization capacity. Finally, the algorithm is tested on 22 typical test functions and compared with 10 other algorithms, demonstrating its competitiveness and outperformance on the majority of test functions.

Keywords: distribution coefficient; distance of inflection point; multi-objective optimization; multi-objective particle swarm optimization

1. Introduction

Most real-life problems typically involve multiple, conflicting objectives that should be simultaneously considered. Solving these multi-objective optimization problems (MOPs) [1] requires optimizing multiple objectives [2,3]. If one of the objectives is optimal, it is impossible to simultaneously obtain the optimal solutions for all the other objectives, and it may even make the results of the other objectives worse. As a kind of complicated optimization problem, the research on MOPs is reflected in production scheduling [4,5], urban transportation [6], network communication [7,8], and other areas. In addition, it exists in problems like engineering design [9], data mining [10], and fund planning [11], which is significant from both a theoretical and practical standpoint. With the rapid development of the real world, MOPs confront numerous challenges, such as diversification and dynamic programming.

Optimization problems are generally divided into single-objective optimization problems and multi-objective optimization problems. Unlike the single-objective optimization problem, there is no unique global optimal solution for multi-objective optimization problems. Therefore, solving real-life MOPs faces many difficulties and challenges. Intelligent algorithms are an effective way to solve MOPs. Since the development of intelligent algorithms, many kinds of intelligent algorithms have been produced. Swarm intelligence algorithms such as the particle swarm optimization algorithm (PSO) [12], the whale optimization algorithm (WOA) [13] and the dragonfly optimization algorithm (DA) [14] are widely used to solve MOPs.

In the above intelligent algorithms, the researchers have extended PSO to multi-objective particle swarm optimization (MOPSO) [15] due to its simple structure and high efficiency. However, similar to other optimization algorithm, MOPSO has several problems, including the following: how to achieve a good balance between exploration and exploitation ability; search accuracy is insufficient; premature convergence; the algorithm is easy to fall into local optimal; and so on. In order to solve these problems, many academics have committed their time to relevant research and proposed numerous improvement strategies [16–21] to enhance the performance of MOPSO. These strategies can be divided into the following three groups: 1) parameter settings. Chen et al. [16] proposed a heuristic algorithm. The probability density function is adjusted adaptively, and the random inertia weight is generated during the search; 2) the neighborhood topology. Roshanzamir et al. [17] proposed a new hierarchical multi-swarm particle swarm optimization algorithm with different task assignments. This structure greatly improves the performance of particle swarm; and 3) hybrid strategies. Deb et al. [18,19] proposed an adaptive multi-objective optimization algorithm based on Pareto dominance. This algorithm proposed a diversity measurement strategy of "distribution entropy" which can measure the distribution of solutions more accurately. However, its main purpose is to increase diversity without considering convergence, which cannot improve its overall performance. Raquel and Naval [20] proposed an extended particle swarm optimization algorithm. The diversity of Pareto optimal solutions in external archive is maintained by increasing the crowding distance on MOPSO. Compared with the traditional algorithm, this algorithm has some improvement in diversity. However, only the crowding distance is taken as the core, and other influencing factors are not fully considered. In order to balance convergence and diversity, Liu et al. [21] used the R2 indicator and a particle swarm optimizer based on decomposition. A new speed updating method is proposed to improve the capability of exploration and exploitation.

Although the aforementioned algorithms have a certain effect on improving the comprehensive performance of the algorithm and getting rid of the local optimum, there are still some drawbacks such as the following: diversity and convergence are not taken into account at the same time; too

many parameter settings; and other influencing factors are not fully considered. Based on these problems, this paper proposes a novel algorithm with a comprehensive indicator and layered particle swarm optimizer (IMOPSOCE) to further improve the performance of MOPSO. While introducing some new strategies, IMOPSOCE retains some settings of MOPSO. Twenty-two test functions validate the effectiveness of the algorithm. The following are the primary contributions of the proposed IMOPSOCE:

1) A new strategy for external archive maintenance is proposed. In order to update and maintain the external archive, which effectively improves the convergence of the algorithm and the diversity of the swarm, a comprehensive indicator is employed to measure the comprehensive performance of the non-dominated solutions in the external archive.

2) The random inertia weight is utilized to further balance the capacity for exploration and exploitation of swarm by taking into account the global search in the early stage and the local exploration in the later stage.

3) The swarm is divided into two layers based on the levels of particles after each update, and the speed update mode of the first level of particles is altered to improve the search efficiency of the algorithm.

The rest of this paper is organized as follows. The definitions of MOPs and PSO are briefly introduced in Section 2. In Section 3, IMOPSOCE is proposed, and its improvement strategy is described in detail. Section 4 shows results for an experimental study and discussions that demonstrate the effectiveness of IMOPSOCE. Finally, Section 5 draws some conclusions based on the work done in this paper and describes future work.

2. Preliminaries

2.1. Multi-objective optimization problems

The majority of problems encountered in both practical life and scientific research belong to MOPs [1]. Different from solving single-objective optimization problems, rather than just one optimal solution, MOPs have a solution set [22] that is the Pareto optimum solution set. Multiple conflicting objective functions that minimize or maximize are present in MOPs. The formulas to minimize the MOPs can be described as follows:

$$\begin{aligned} &\text{minimize } F(x) = \left(f_1(x), f_2(x), \cdots, f_M(x)\right) \\ &\text{subject to } x = \left(x_1, x_2, \cdots, x_D\right) \in \Omega \end{aligned} \tag{1}$$

where M is the number of objective variables, $f_i()(1 \le i \le M)$ is the i-th objective function, X is the decision vector, and Ω is the D-dimensional decision space.

In MOPs, the quality of the solutions can be evaluated by Pareto dominance. The decision vector x_a strictly dominates the decision vector x_b, denoted $x_a \prec x_b$, which can be expressed as follows:

$$\forall i \in \{1,2,\cdots,N\}: f_i(x_a) \le f_i(x_b) \wedge \exists\, j \in \{1,2,\cdots,N\}: f_i(x_a) < f_i(x_b) \tag{2}$$

When a solution is not dominated by any other solution, it is considered to be a non-dominated

Pareto optimal solution, and many such solutions become a Pareto optimal solution set, that is, a non-dominated solution set.

2.2. Particle swarm optimization

Kennedy and Eberhart proposed PSO [12] in the 1990s, which is a swarm theory heuristic intelligent optimization algorithm, produced as a result of long-term observation of how birds forage. PSO simulates the foraging behavior of birds, simulates the search space to solve the problem as the flight space of birds, and abstracts each bird into a particle without mass and volume. The process of searching for the optimal solutions is regarded as the process of bird foraging, which then solves complex optimization problems.

$$v_{i,j}(t+1)=\omega(t)v_{i,j}(t)+c_1r_1\left(p_{best_{i,j}}(t)-x_{i,j}(t)\right)+c_2r_2\left(g_{best_j}(t)-x_{i,j}(t)\right) \tag{3}$$

$$x_{i,j}(t+1)=x_{i,j}(t)+v_{i,j}(t+1) \tag{4}$$

Each particle has a position and a velocity. The position of the particle is denoted by $x_i(t)=\left[x_{i,1}(t),x_{i,2}(t),\cdots,x_{i,D}(t)\right]$, and the velocity of the particle corresponding to it is $v_i(t)=\left[v_{i,1}(t),v_{i,2}(t),\cdots,v_{i,D}(t)\right]$, where $i=1,2,\cdots,N$, N is the number of particles and D is the size of the decision space. The *i*-th particle in the population updates its velocity and position according to the two formulas above. As the fundamental formula of the algorithm, Eq (3) has three parts: the first part represents the state of the previous generation of particles; the second part represents the influence of the optimal position of the individual history on particles; and the third part represents the influence of the optimal position of population on particles. In Eq (4), the current position is composed of the position of the previous generation and the current velocity where ω is the inertia weight, c_1 and c_2 are the learning factors, r_1 and r_2 are uniformly distributed random numbers in the range $[0,1]$, $v_{i,j}(t)$ and $x_{i,j}(t)$ denote the velocity and position of the *i*-th particle in the *j*-th dimension at the *t*-th iteration, respectively, $p_{best_{i,j}}(t)$ denotes the individual historical optimal position of the *i*-th particle in the *j*-th dimension, and $g_{best_j}(t)$ denotes the optimal position of the population in the *j*-th dimension.

2.3. Existing MOPSOs

In addition to the MOPSOs mentioned above, some MOPSOs are designed to improve the algorithm performance, balance exploration and exploitation. Next, we briefly review some representative MOPSOs.

Hu and Yen [23] proposed a new algorithm named pccsAMOPSO. A density estimation method (PCCS) is designed, and based on this method, the distribution entropy of non-dominated solutions is

used to evaluate the approximate Pareto front uniformity obtained by the MOP optimizer. At the same time, this method can also be used for the selection of leaders and to update the external archive in MOPSO.

Han et al. [24] proposed an AMOPSO algorithm based on a hybrid framework of solution distribution entropy and population spacing (SP) information. The leader selection mechanism based on the solution distribution entropy can analyze the evolutionary trend and select suitable leaders to balance the convergence and diversity of non-dominant solutions. In addition, a flight parameter adjustment mechanism based on population spacing information is proposed to balance the global and local search abilities of particles. The above strategies can obtain a set of optimal solutions with a high diversity and achieve a balance between exploration and exploitation capabilities in the search process.

In terms of parameter setting, Tripathi et al. [25] described a time-varying multi-objective particle swarm optimization algorithm (TV-MOPSO). In a typical MOPSO algorithm, the inertia weight ω and learning factors c_1 and c_2 have a very important impact on the algorithm performance. Proper parameter settings enable the algorithm to achieve a good balance between exploration and exploitation. TVMOPSO is inherently adaptive in terms of inertia weight and acceleration coefficient. This adaptability helps the algorithm explore the search space more efficiently.

Shibata et al. [26] proposed a multi-objective discrete particle swarm optimizer (DPSO). This method introduces a hierarchical structure composed of DPSOs and a multi-objective genetic algorithm (MOGA). The hierarchical structure can reduce the computational cost of learning; therefor, the method is effective in high-dimensional problems. In addition, the diversity and accuracy of the solutions obtained are either equal to or higher than those obtained using traditional methods.

In the above MOPSOs, they propose different strategies from different aspects to get better solutions. Additionally, under their incentive, in order to improve the convergence of the algorithm and the diversity of the population, IMOPSOCE is proposed. A comprehensive indicator is used to measure the performance of the solution in the external archive to help better maintain it. In order to take the global and local search into account, a random inertia weight strategy is designed, and a hierarchical structure is proposed to divide the swarm according to the levels of particles after each update. This is described in detail in Section 3.

3. The proposed IMOPSOCE method

3.1. The comprehensive indicator application

The convergence and diversity of the algorithm have a direct impact on its overall performance. This research uses a comprehensive indicator that combines the inflection point distance (CPI) [27] and the distribution coefficient (MPI) to measure the performance of the non-dominated solutions in the external archive after each iteration. The external archive is maintained by the comprehensive indicator, while the non-dominated solutions with a bad convergence and diversity are deleted and those with a good convergence and diversity are saved in the external archive. This strategy can boost the convergence and diversity of non-dominated solutions in the external archive. The comprehensive indicator is described as follows:

$$CM = CPI + MPI \tag{5}$$

where CPI is the convergence indicator and MPI is the diversity indicator. When the external archive reaches the threshold, the proposed IMOPSOCE calculates the CM value of each non-dominated solution in the external archive and eliminates the non-dominated solutions with a poor comprehensive performance by comparing the CM values. In Eq (5), CPI can reflect the convergence of the non-dominated solutions in the external archive. The distribution of non-dominated solutions in the external archive can be measured using MPI. CPI and MPI are defined as follows:

According to the number of objective functions, the calculation of CPI is separated into the two following cases:

1) The distance between each non-dominated solution and the extreme line determined by the two extreme non-dominated solutions in the external archive is the CPI when the objective function is a bi-objective function. The formula used to calculate the CPI value in this situation is as follows:

$$CPI = \frac{\left|Ax_0 + By_0 + C\right|}{\sqrt{A^2 + B^2}} \tag{6}$$

where x_0 and y_0 are the coordinate values of each non-dominated solution in the external archive and A, B and C are real numbers that are determined by the extreme line composed of two extreme non-dominated solutions. The extreme line is denoted by the following:

$$Ax + By + C = 0 \tag{7}$$

where x and y are the coordinate values of the extreme non-dominated solutions.

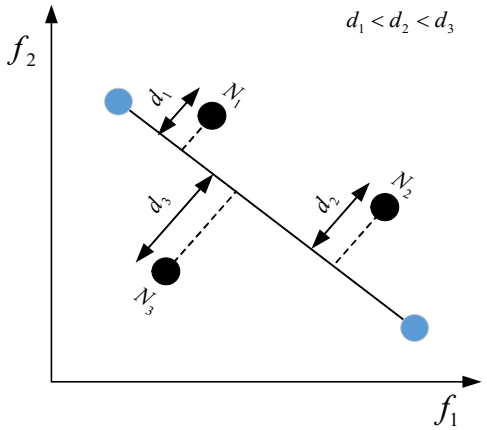


Figure 1. Schematic representation of the CPI in the bi-objective space.

2) CPI needs to calculate the distance between each non-dominated solution in the external archive and the extremal "hyperplane" if there are three or more objective functions. In this situation, the CPI is determined as follows:

$$CPI = \frac{\left|Ax_0 + By_0 + Cz_0 + D\right|}{\sqrt{A^2 + B^2 + C^2}} \tag{8}$$

The extremal "hyperplane" determines the A, B, C and D in this calculation formula. x_0, y_0 and z_0 are the coordinate values of each non-dominated solution in the external archive. The following formula is used to determine the extremal "hyperplane":

$$Ax + By + Cz + D = 0 \tag{9}$$

where x, y and z are the coordinate values of the extreme non-dominated solutions.

CPI is an indicator to measure the convergence of non-dominated solutions. The smaller the CPI values are, the better the convergence of the non-dominated solutions. In other words, the closer the non-dominated solutions are to the extreme line or extremal "hyperplane", the better the convergence of the non-dominated solutions. Using Figure 1 as an illustration, the extreme points of the non-dominated solutions are shown in blue; N_1 is the point with good convergence and N_3 has poor convergence.

Numerous brilliant scholars have suggested different superior approaches for improving population diversity. This led to the concept of crowding distance being put forth by forerunners. Based on crowding distance, Deb et al. obtained an improved population diversity [18,19]. In order to improve the population diversity, the distribution of non-dominated solutions in the objective space is measured in this paper using the distribution coefficient. The distribution coefficient is defined as follows:

when the number of objective functions is m

$$MPI = \sum_{j=1}^{m} \left(\log_2 \left(pf_{ij} \right) + \log_2 \left(pb_{ij} \right) \right) \tag{10}$$

$$pf_{ij} = \left| \frac{f_{ij}}{t_{ij}} \right| \tag{11}$$

$$pb_{ij} = \left| \frac{b_{ij}}{t_{ij}} \right| \tag{12}$$

$$t_{ij} = f_{ij} + b_{ij} \tag{13}$$

where f_{ij} is the distance from the i-th solution to its previous adjacent solution on the j-th objective function, and b_{ij} is the distance from the i-th solution to its next adjacent solution on the j-th objective function. The smaller the MPI value of a particle, the better its distribution.

Equations (10)–(13) indicate that the distribution coefficients of solutions A and B are MPI_A and MPI_B, respectively, as shown in Figure 2. $MPI_A = -4.1699$, $MPI_B = -4.2288$. Solution A clearly has a better distribution than solution B. It follows that the distribution coefficient can reflect

the distribution of non-dominated solutions in the objective space; thus, the non-dominated solutions with a better distribution in the external archive are chosen. When computing the MPI value of the non-dominated solutions, the boundary non-dominated solutions are given a minimum distribution coefficient value, so that these solutions are always selected. Equations (10)–(13) will be used to obtain the distribution coefficients of the non-dominated solutions in the middle.

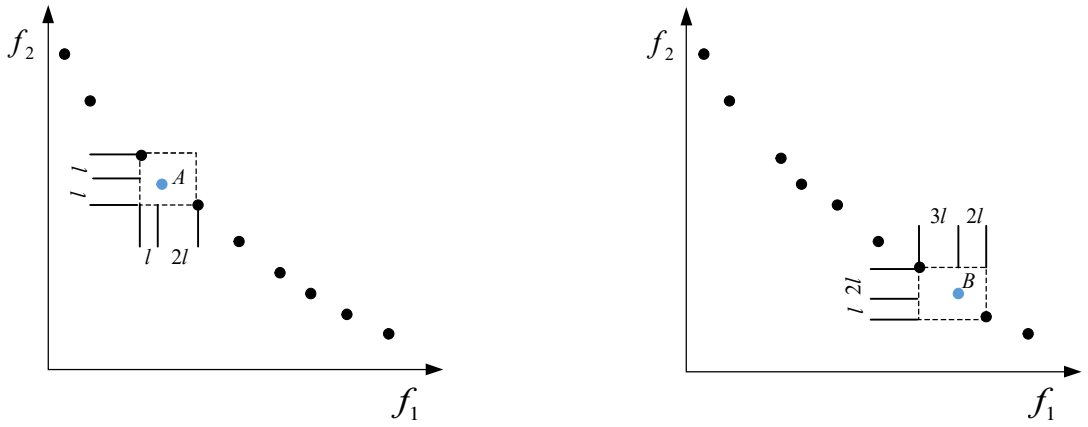


Figure 2. Schematic representation of the MPI in the bi-objective space.

The entire procedure for the external archive maintenance is listed in Algorithm 1. This strategy is mainly maintained for the external archive by using the CM indicator. First, using Eqs (6) and (8), the CPI value of the non-dominated solution is determined (line 2). Then, the MPI value of the non-dominated solution is determined using Eq (10) (line 3). Finally, Eq (5) is utilized to obtain the CM value of the non-dominated solution (line 4). The non-dominated solutions with a poor comprehensive performance are eliminated by comparing the CM values of each non-dominated solution (line 5).

Algorithm 1: Update Archive

Input: $R_{\max}$ (External archive threshold), g_{best} (Global optimal position)
Output: g_{best} (Global optimal position)
1 **While** $size(g_{best}) > R_{\max}$ **do**
2 Calculate the value of CPI via using Eqs (6) and (8)
3 Calculate the value of MPI via using Eq (10)
4 Calculate the value of CM via using Eq (5)
5 Delete the poor performance non-dominated solutions according to the value of CM
6 **End While**

3.2. The proposed random inertia weight method

The motion of the particles in the standard MOPSO follows Eqs (3) and (4). In Eq (3), the learning factors c_1 and c_2 determine the impact of the experience information of the particle and the experience information of other particles on the particle motion and reflect information exchange among the particle swarm. ω stands for the effect of the previous velocity on the current velocity, which can be used to adjust the flying speed of the particle, limit the movement range of the particle,

and balance its capacity for exploration and exploitation. Therefore, in order for the particles to search more effectively, the proper ω must be set. In conventional MOPSO, ω is typically taken as a fixed value, making it difficult to balance the global search in the early stage and the local exploration in the later stage of the particle. Therefore, we design a random inertia weight strategy to adjust ω, taking the global search and local exploration of particles into account, in order to efficiently regulate the movement of particles. Its specific formula is as follows:

$$\omega(t) = \left((1 - t / t_{\max}) * (\omega_{\max} - \omega_{\min}) + \omega_{\min}\right) * \text{rand} * \left(\exp\left(-\omega_{\min} * (\text{pi} * t / (2 * t_{\max}))^2\right)\right) \tag{14}$$

where t is the current iteration number, $t_{\max}$ is the maximum iteration number, and $\omega_{\max}$ and $\omega_{\min}$ are the maximum and minimum values of inertia weight, respectively. The proposed IMOPSOCE improves the search efficiency of the algorithm and reasonably balances the exploration and exploitation capabilities of particles when compared to the traditional MOPSO.

3.3. Bilayer velocity update with different task allocations

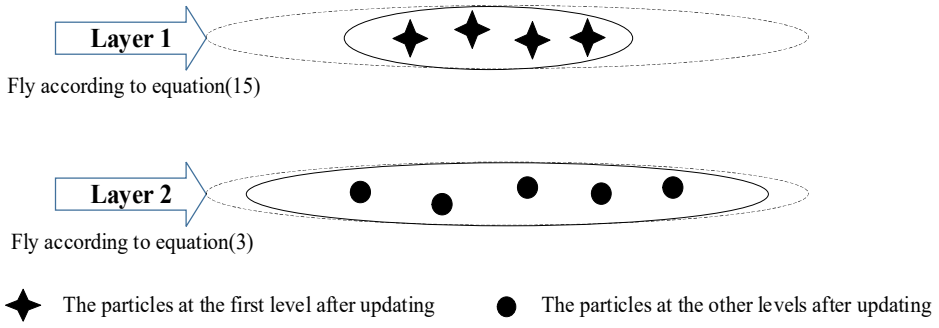


Figure 3. Schematic representation of the motion of particles.

Particles move in the classical MOPSO in accordance with Eqs (3) and (4). The IMOPSOCE proposed in this paper has two layers. After each update, the particles of the population are split into two layers based on their levels, and then the velocity update mode of the particles situated at the first level is modified. Since the particles at the first level after updating have a decent movement trend, learning from the social part may cause them to deviate from the correct movement direction. As a result, in this paper, after each update, the particles in the first level maintain their own movement trend without being influenced by the particles in the social part. Particles in the second layer continue to fly using Eq (3). In this way, some particles can explore more, diversity will be maintained, and a premature convergence will be prevented. Some particles can more effectively exploit and use appropriate information in the search process, thus improving the solutions obtained. The right balance between exploration and exploitation capabilities is created by giving particles from different layers different ways to update their speeds. The aforementioned is well illustrated in Figure 3. The particles in the first level adopt the speed update mode of Eq (15) after each update, while the particles in the other levels fly in accordance with Eq (3).

$$v_{i,j}(t+1) = \omega(t) v_{i,j}(t) + c_1 r_1 \left(p_{best_{i,j}}(t) - x_{i,j}(t)\right) \tag{15}$$

Combining the above, the pseudo-code of the particle update procedure is described in Algorithm 2. Equation (14) is used to compute the inertia weight (line 1). Next, the individual optimal position, velocity, position, etc. for the particle of the first level is found (lines 2–4). Finally, using Eqs (3), (4),

and (15), the velocity and position of each particle after updating are determined (lines 5–8).

Algorithm 2: Update Particles

Input: $\omega_{\max}$ (Inertial weight maximum value), $\omega_{\min}$ (Inertial weight minimum value), p (Particle position), vel (Particle velocity), p_{best} (Individual optimal position), g_{best} (Global optimal position)

Output: v (New speed of particle), v' (New speed of the first-level particle), $newp$ (New position of particle), $newp'$ (New position of the first-level particle)

1 Calculate the inertial weight via using Eq (14)
2 Find p_{best}' % individual optimal position of the first-level particle
3 Find vel' % velocity of the first-level particle
4 Find p' % The position of the first-level particle
5 Calculate v' via using Eq (15)
6 Calculate $newp'$ via using Eq (4)
7 Calculate v via using Eq (3)
8 Calculate $newp$ via using Eq (4)

3.4. The proposed IMOPSOCE algorithm

Algorithm 3: Framework of IMOPSOCE

Input: N (The number of particles), t (Number of current iterations), $t_{\max}$ (Maximum iteration time)

Output: NA (Pareto optimal solutions in the external archive)

1 Initialize the swarm
2 Calculate the fitness value and perform Pareto sort
3 Update P_{best} and g_{best}
4 **While** $t < t_{\max}$ **do**
5 Calculate the inertial weight via using Eq (14)
6 Update Particles **Algorithm 2**
7 Calculate the fitness values
8 Update Archive **Algorithm 1**
9 Update P_{best} and g_{best}
10 **End While**
11 **Return** NA

The proposed IMOPSOCE algorithm is presented in this section. IMOPSOCE is mainly composed of three parts. First, in IMOPSOCE, the population is split into two layers based on the levels of particles after each iteration, and different speed update modes are offered for particles in different layers. Second, in terms of the parameter setting, a random inertia weight strategy is proposed to balance the capacity for exploitation and exploration of the population. Finally, the CM indicator is used to maintain the external archive once it has reached its maximum capacity. The non-dominated solutions with a good convergence and diversity are retained in the external archive, which ensures that the non-dominated solutions in the external archive have an improved comprehensive performance. The pseudo-code of IMOPSOCE is displayed in Algorithm 3.

4. Experimental studies

In order to fully verify the performance of the proposed algorithm, three sets of benchmark functions are used in this section: ZDT, DTLZ, and UF. This section contrasts the proposed algorithm with five existing MOPSOs, including MOPSO [15], dMOPSO [28], NMPSO [29], SMPSO [30], and MPSOD [31]. In addition, we compare the proposed IMOPSOCE with five classical MOEAs, including NSGAIII [32], MOEAD [33], MOEAIGDNS [34], SPEAR [35], and VaEA [36], in order to further assess the proposed IMOPSOCE. The detailed discussions of the experimental procedure and results are as follows.

4.1. Benchmark test functions

Table 1. Parameter settings of the test functions.

Problems	N	M	D	FEs
ZDT1–ZDT3	200	2	30	10,000
ZDT4 and ZDT6	200	2	10	10,000
DTLZ1	200	3	7	10,000
DTLZ2–DTLZ6	200	3	12	10,000
DTLZ7	200	3	22	10,000
UF1–UF7	200	2	30	10,000
UF8–UF10	200	3	30	10,000

The comprehensive performance of IMOPSOCE is verified using three different sets of benchmark functions. Specifically, to compare IMOPSOCE with the compared algorithms, five bi-objective test functions of ZDT [37], seven three-objective test functions of DTLZ [38], and seven bi-objective and three three-objective test functions of UF [39] are employed. They are not used in this paper since ZDT5 is a discrete optimization problem while DTLZ8 and DTLZ9 are two constrained optimization problems. Table 1 displays the relevant settings for these test problems where N is the number of particles, M is the number of objective functions, D is the dimension of decision variables, and FEs is the number of evaluations.

4.2. Performance indicators

In this paper, we employ two comprehensive indicators, namely inverted generational distance (IGD) [40] and hypervolume (HV) [41], to test the algorithms and verify the comprehensive performance of IMOPSOCE. Both indicators are used here to fully validate the algorithm, even though they can both detect the convergence and diversity of the algorithm.

The IGD is used to measure the distance between the true Pareto front and the set of Pareto optimal solutions obtained by the algorithm. A smaller IGD means that the set of non-dominated solutions is closer to the true Pareto front. It is calculated as follows:

$$IGD\left(PF, PF^*\right)=\frac{1}{S}\sum_{i=1}^{S}\min\left(dist\left(PF_i, PF^*\right)\right) \tag{16}$$

where PF is the Pareto front obtained by the algorithm, PF^* is a set of sampling points from the

true Pareto front, and S is the number of non-dominated solutions.

HV is achieved by measuring the hypervolume of the region consisting of the optimal set and the reference points in the objective space. HV can simultaneously evaluate the convergence and diversity of the algorithm. The larger the HV, the better the overall performance obtained by the algorithm. The reference point of HV is set to $(1.1,1.1,\cdots,1.1)$

$$HV(S) = Leb\left(\bigcup_{x \in S} [f_1(x), R_1] \times \cdots \times [f_M(x), R_M] \right) \tag{17}$$

where f_i denotes the *i*-th objective function value of S and R_i denotes the *i*-th objective function value of the reference point.

4.3. Experimental settings

Table 2. Parameter settings of all algorithms.

Algorithms	Parameter settings
MOPSO	$\omega = 0.4$
dMOPSO	$T_a = 2$, $\theta = 5$
NMPSO	$\omega \in [0.1,0.5]$, $c_1, c_2, c_3 \in [1.5,2.5]$, $p_m = 1/n$, $\eta_m = 20$
SMPSO	$\omega \in [0.1,0.5]$, $c_1, c_2 \in [1.5,2.5]$, $p_m = 1/n$, $\eta_m = 20$
MPSOD	$\omega \in [0.1,0.9]$, $c_1, c_2 \in [1.5,2.5]$, $p_c = 0.9$, $F = 0.5$, $CR = 0.5$, $p_m = 1/n$, $\eta_m = 20$
NSGAIII	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$, $\eta_m = 20$
MOEAD	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = \eta_m = 20$
MOEAIGDNS	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = \eta_m = 20$
SPEAR	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = \eta_m = 20$
VaEA	$p_c = 1.0$, $p_m = 1/n$, $\eta_c = 30$, $\eta_m = 20$
IMOPSOCE	$\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$, $c_1 = c_2 = 2$

In this paper, five representative MOPSOs and five classical MOEAs are compared to IMOPSOCE. In solving many MOPs, these algorithms have demonstrated an excellent performance. The parameter settings of the compared algorithms are compatible with the original reference literature in order to make the various different algorithms comparable to one another and to conduct a fair comparison, as indicated in Table 2.

For the compared MOPSOs, the inertia weight ω and flight parameters c_1 and c_2 are used to update velocity. p_c is crossover probability, and p_m is mutation probability. In MOEAD, MOEAIGDNS, and SPEAR, the distribution index η_c of the SBX operator is 20, and the distribution index η_m of the mutation operator is 20. In NSGAIII and VaEA, the distribution index η_c is 30 for the SBX operator, and the distribution index η_m is 20 for the mutation operator.

In IMOPSOCE, the maximum and minimum values of ω are set to 0.9 and 0.4, respectively. The learning factors c_1 and c_2 are both set to 2. In addition, in all compared algorithms, the same population and size of the archive are set for a fair comparison. Specifically, the population size is set to 200, and the number of evaluations is 10,000 for the compared algorithms. All experiments are implemented on MATLAB R2020b with an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz 2.81

GHz. The original code for the compared algorithm is provided by PlatEMO [42], and each algorithm is independently run 30 times on each test function.

4.4. Experimental results and data analysis

This part compares the characteristics of the improved algorithm, selects 22 test functions of the ZDT, DTLZ, and UF series to evaluate the performance of the proposed IMOPSOCE, and analyzes the experimental results in order to verify the comprehensive performance of IMOPSOCE.

4.4.1. Comparisons with five existing MOPSOs

Tables 3 and 4 present the means and standard deviations of the IGD indicator and HV indicator for 30 independent runs on the ZDT, DTLZ, and UF functions for the six multi-objective algorithms MOPSO, dMOPSO, NMPSO, SMPSO, MPSOD, and IMOPSOCE. The Wilcoxon rank sum test is performed at a significance level of 0.05 to show the significant differences between the test results. The symbol '+' in Tables 3 and 4 indicates that the results achieved by other MOPSOs are significantly better than the results obtained by IMOPSOCE, while '-' indicates that the results achieved by other MOPSOs are significantly worse than the results obtained by IMOPSOCE. Additionally, '≈' represents the results of comparing algorithms, and IMOPSOCE is similar. The performance of the proposed IMOPSOCE and the other five algorithms on 22 test functions can be shown by combining the data in Tables 3 and 4. Comparing the experimental results reveals that the proposed IMOPSOCE performs better overall than the other five classical algorithms. To visually show the optimization result, the optimal values in the table are bolded.

Table 3 shows the means and standard deviations of the IGD indicator of the five classical algorithms and IMOPSOCE on the 22 test functions. On the test functions ZDT1, ZDT2, ZDT3, DTLZ6, UF4, UF5, UF8, UF9, and UF10, it is immediately obvious that IMOPSOCE performs noticeably better than the five existing MOPSOs that are compared. IMOPSOCE performs well since it obtains the best IGD on 13 out of the 22 test functions. The IGD of IMOPSOCE is better than the other five algorithms, and its performance is much better than the competitive MOPSOs, which verifies its good performance. Second, NMPSO and SMPSO perform better in the remaining compared algorithms. Table 3 shows that MOPSO and dMOPSO perform poorly on 22 test functions and that their optimal IGD number is 0. According to the Wilcoxon rank sum test results presented in the second-last row, IMOPSOCE performs significantly better than MOPSO, dMOPSO, NMPSO, SMPSO, and MPSOD on 18, 17, 12, 16, and 19 out of 22 comparisons, respectively, while it performs worse than MOPSO, dMOPSO, NMPSO, SMPSO, and MPSOD on 3, 4, 7, 5, and 3 comparisons, respectively. Besides, IMOPSOCE obtains similar results to MOPSO, dMOPSO, NMPSO, SMPSO, and MPSOD on 1, 1, 3, 1, and 0 test functions, respectively. All the statistical results demonstrate the great efficacy of IMOPSOCE in solving the MOPs. On ZDT1 and UF10, IMOPSOCE performs better than NMPSO by a factor of 5 and 4, respectively. Only the proposed IMOPSOCE performs well on nearly all test functions, despite other compared algorithms performing well on some test functions. For example, NMPSO and SMPSO perform admirably on the DTLZ test functions, but poorly on the ZDT. Additionally, NMPSO and SMPSO perform well on the ZDT6 test function, but much worse on UF10 test function. IMOPSOCE performs significantly better than the other compared algorithms on UF1-UF10. IMOPSOCE shows its superior performance compared with the other five algorithms. This is mostly attributable to the adoption of the random inertia weight strategy, which effectively balances the exploitation and exploration

capacities of particles to enable the swarm to discover better non-dominated solutions. The aforementioned evidence demonstrates that, when compared to the existing MOPSOs, the proposed IMOPSOCE performs the best overall.

In addition to the IGD indicator, a crucial HV indicator is employed to further confirm the effectiveness of IMOPSOCE. The comparison results using the HV indicator are similar to those using the IGD indicator, as shown in Table 4. Performance is poorer for MOPSO, dMOPSO, and MPSOD. It is noteworthy that the proposed IMOPSOCE only has a very small gap of less than one time from the optimal result on the test functions ZDT6, DTLZ2, DTLZ5, UF1, and UF4. On the test function UF6, IMOPSOCE has a slightly worse HV value but the best overall performance. As a result, when combined with the information in Table 4, we can draw the conclusion that the IMOPSOCE performs the best among the 22 test functions in terms of the HV indicator and is very competitive when compared to the existing MOPSOs algorithms. The proposed IMOPSOCE can perform better in terms of the convergence and diversity when solving the MOPs.

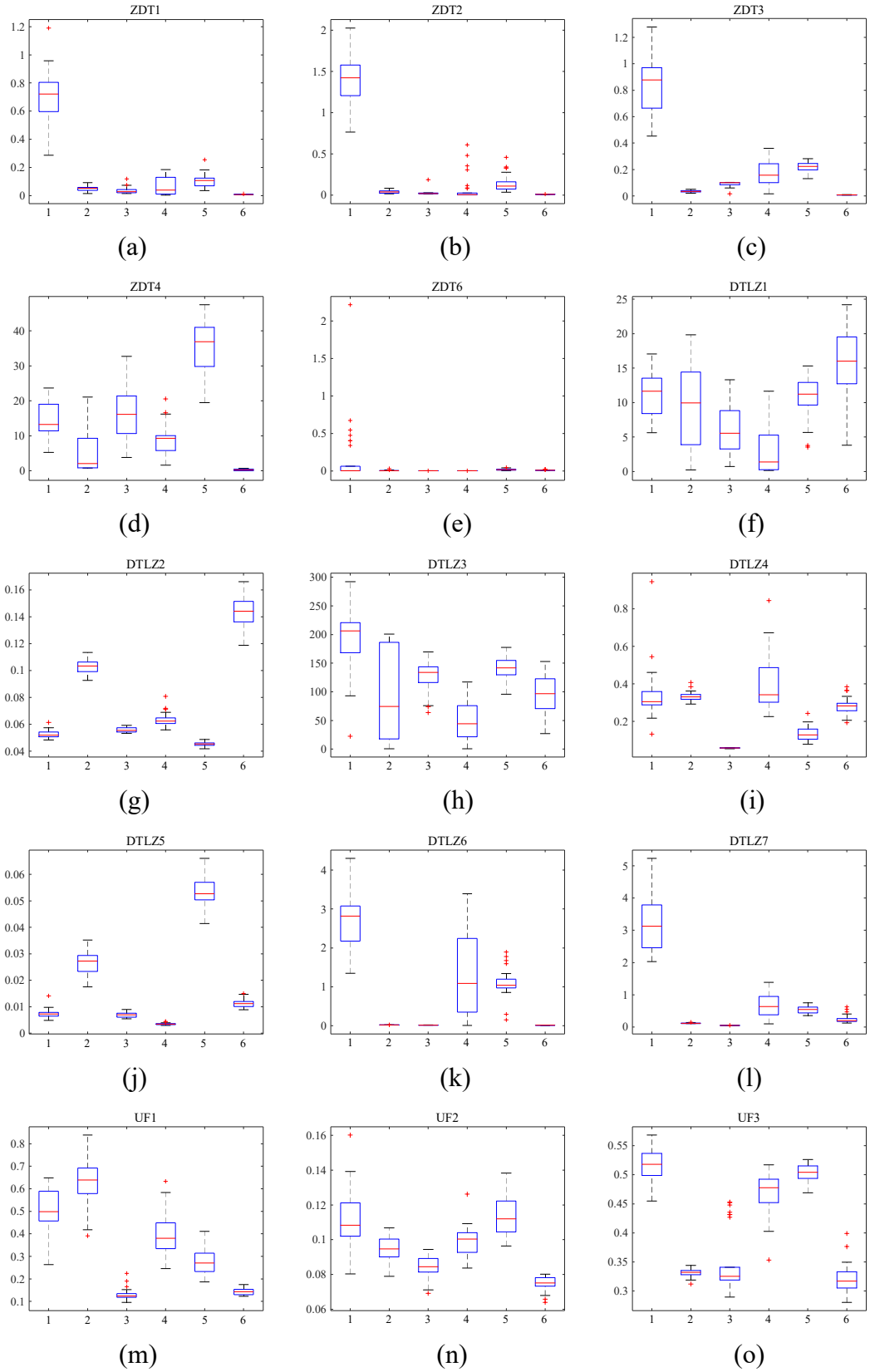
Figure 4 displays box plots of the IGD indicator for the six algorithms. It is worth noting that the lower the mean value of IGD, the shorter the box plot in the figure, indicating that the algorithm obtains better IGD values and more consistent results. When different algorithms are run independently 30 times on each test function, the box plots of the IMOPSOCE algorithm and the compared algorithms with respect to the IGD indicator are shown in Figure 4 (1, 2, 3, 4, 5, and 6 on the horizontal coordinate represent MOPSO, dMOPSO, NMPSO, SMPSO, MPSOD, and IMOPSOCE, respectively, and the vertical coordinate represents the IGD value of each algorithm). Figure 4 records the data fluctuations of the six algorithms on ZDT1-ZDT4, ZDT6, DTLZ1-DTLZ7, and UF1-UF10. It is evident that IMOPSOCE can obtain better solutions than other MOPSOs. This is in agreement with the findings of Table 3. Meanwhile, the proposed IMOPSOCE shows significant improvement on the test functions ZDT1, ZDT2, ZDT3, ZDT4, DTLZ6, UF2, UF5, UF6, UF7, and UF9. Except for the test functions DTLZ1, DTLZ2, and UF8, IMOPSOCE, NMPSO, and SMPSO perform well, while dMOPSO and MPSOD perform relatively poorly. These figures further demonstrate the superior results of IMOPSOCE and demonstrate that it outperforms the other five algorithms in terms of overall performance on the 22 test functions. It is clear from the aforementioned narration that IMOPSOCE has certain advantages and effectiveness in improving the performance of the algorithm compared with the existing MOPSOs.

Table 3. IGD values of IMOPSOCE and five MOPSOs on the test functions ZDT1-ZDT4 and ZDT6, DTLZ1-DTLZ7 and UF1-UF10.

Problem	MOPSO	dMOPSO	NMPSO	SMPSO	MPSOD	IMOPSOCE
ZDT1	7.0211e-1-(1.81e-1)	5.0137e-2-(1.89e-2)	3.4869e-2-(2.18e-2)	6.7559e-2-(6.62e-2)	1.0674e-1-(4.49e-2)	**7.4580e-3 (1.13e-3)**
ZDT2	1.4189e+0-(3.31e-1)	3.7406e-2-(1.70e-2)	2.4724e-2-(3.09e-2)	7.2003e-2≈(1.55e-1)	1.4470e-1-(1.05e-1)	**8.5043e-3 (1.40e-3)**
ZDT3	8.4763e-1-(2.13e-1)	3.4836e-2-(6.64e-3)	9.2071e-2-(1.79e-2)	1.6969e-1-(9.53e-2)	2.1877e-1-(3.61e-2)	**7.7408e-3 (1.15e-3)**
ZDT4	1.4592e+1-(5.39e+0)	5.4598e+0-(6.43e+0)	1.6690e+1-(7.66e+0)	9.0668e+0-(4.25e+0)	3.5681e+1-(7.17e+0)	**2.7971e-1 (2.29e-1)**
ZDT6	1.7468e-1≈(4.32e-1)	5.6438e-3+(6.07e-3)	2.2404e-3+(1.90e-4)	**1.9430e-3+ (9.80e-5)**	1.8824e-2-(1.03e-2)	1.0595e-2 (3.76e-3)
DTLZ1	1.1190e+1+(3.10e+0)	9.6588e+0+(6.15e+0)	5.9159e+0+(3.51e+0)	**3.2280e+0+ (3.57e+0)**	1.0907e+1+(2.92e+0)	1.5513e+1 (5.12e+0)
DTLZ2	5.2526e-2+(2.95e-3)	1.0314e-1+(5.70e-3)	5.5851e-2+(1.85e-3)	6.3330e-2+(4.98e-3)	**4.5371e-2+ (1.56e-3)**	1.4368e-1 (1.17e-2)
DTLZ3	1.9302e+2-(5.55e+1)	9.9189e+1≈(7.79e+1)	1.2628e+2-(2.63e+1)	**5.0028e+1+ (3.46e+1)**	1.3952e+2-(2.02e+1)	9.4523e+1 (3.49e+1)
DTLZ4	3.4110e-1-(1.36e-1)	3.3172e-1-(2.48e-2)	**5.7062e-2+ (1.60e-3)**	3.9519e-1-(1.40e-1)	1.3259e-1+(3.83e-2)	2.7893e-1 (4.56e-2)
DTLZ5	7.4875e-3+(1.63e-3)	2.6307e-2-(4.44e-3)	6.9371e-3+(9.66e-4)	**3.4514e-3+ (3.56e-4)**	5.3972e-2-(5.53e-3)	1.1122e-2 (1.53e-3)
DTLZ6	2.7753e+0-(7.33e-1)	2.0642e-2-(3.26e-4)	1.2579e-2-(1.91e-3)	1.3720e+0-(1.05e+0)	1.0854e+0-(3.51e-1)	**1.0169e-2 (1.79e-3)**
DTLZ7	3.2173e+0-(9.29e-1)	1.1497e-1+(1.09e-2)	**4.6442e-2+ (1.82e-3)**	6.7488e-1-(3.48e-1)	5.4007e-1-(1.07e-1)	2.3897e-1 (1.21e-1)
UF1	5.0187e-1-(1.04e-1)	6.2479e-1-(9.81e-2)	**1.3046e-1+ (2.56e-2)**	4.0364e-1-(8.99e-2)	2.7516e-1-(5.50e-2)	1.4230e-1 (1.39e-2)
UF2	1.1058e-1-(1.64e-2)	9.4457e-2-(6.86e-3)	8.4078e-2-(6.33e-3)	9.9407e-2-(8.59e-3)	1.1393e-1-(1.07e-2)	**7.4755e-2 (4.15e-3)**
UF3	5.1669e-1-(2.85e-2)	3.3089e-1-(7.06e-3)	3.4965e-1-(5.17e-2)	4.6952e-1-(3.51e-2)	5.0281e-1-(1.39e-2)	**3.2163e-1 (2.43e-2)**
UF4	1.1075e-1-(1.43e-2)	1.3658e-1-(6.61e-3)	6.2902e-2≈(8.05e-3)	1.0863e-1-(9.35e-3)	9.7480e-2-(2.85e-3)	**6.2371e-2 (7.02e-3)**
UF5	3.2536e+0-(3.12e-1)	3.2928e+0-(3.12e-1)	1.7124e+0≈(3.30e-1)	2.8025e+0-(6.67e-1)	2.7346e+0-(2.66e-1)	**1.5851e+0 (1.68e-1)**
UF6	2.5561e+0-(4.90e-1)	2.2119e+0-(5.62e-1)	**6.4992e-1≈ (1.33e-1)**	1.3667e+0-(4.85e-1)	1.4099e+0-(2.57e-1)	6.7761e-1 (8.19e-2)
UF7	6.1441e-1-(1.07e-1)	3.8236e-1-(7.33e-2)	2.2779e-1-(1.93e-1)	3.5257e-1-(1.20e-1)	2.2707e-1-(5.46e-2)	**1.0721e-1 (1.10e-2)**
UF8	4.0833e-1-(3.23e-2)	3.5044e-1-(3.68e-2)	4.6655e-1-(9.42e-2)	3.8784e-1-(3.85e-2)	5.4451e-1-(3.97e-2)	**3.1105e-1 (8.65e-2)**
UF9	5.3785e-1-(4.18e-2)	5.9435e-1-(3.94e-2)	4.5878e-1-(4.77e-2)	5.6715e-1-(4.60e-2)	6.5006e-1-(4.57e-2)	**1.3122e-1 (1.25e-2)**
UF10	2.3147e+0-(3.19e-1)	9.4530e-1-(1.24e-3)	1.4957e+0-(3.32e-1)	2.8189e+0-(4.83e-1)	4.1294e+0-(3.67e-1)	**4.2001e-1 (9.46e-2)**
+ / - / ≈	3/18/1	4/17/1	7/12/3	5/16/1	3/19/0	-
Best/all	0/22	0/22	4/22	4/22	1/22	13/22

Table 4. HV values of IMOPSOCE and five MOPSOs on the test functions ZDT1-ZDT4 and ZDT6, DTLZ1-DTLZ7 and UF1-UF10.

Problem	MOPSO	dMOPSO	NMPSO	SMPSO	MPSOD	IMOPSOCE
ZDT1	9.2160e-2-(9.00e-2)	6.6118e-1-(2.16e-2)	6.8424e-1-(2.41e-2)	6.3545e-1-(8.30e-2)	5.6884e-1-(5.86e-2)	**7.1622e-1 (1.31e-3)**
ZDT2	0.0000e+0-(0.00e+0)	3.8867e-1-(2.62e-2)	4.2996e-1-(3.14e-2)	3.8413e-1≈(1.16e-1)	2.7548e-1-(9.50e-2)	**4.4006e-1 (1.76e-3)**
ZDT3	7.7767e-2-(8.58e-2)	5.9695e-1≈(1.25e-2)	5.7099e-1-(7.07e-3)	5.2579e-1-(7.42e-2)	4.4626e-1-(3.73e-2)	**5.9870e-1 (3.73e-4)**
ZDT4	0.0000e+0-(0.00e+0)	3.9766e-2-(5.70e-2)	0.0000e+0-(0.00e+0)	0.0000e+0-(0.00e+0)	0.0000e+0-(0.00e+0)	**5.0728e-1 (1.61e-1)**
ZDT6	3.1423e-1≈(1.30e-1)	3.8610e-1+(6.80e-3)	3.8982e-1+(1.60e-4)	**3.9000e-1+ (1.08e-4)**	3.7374e-1-(9.84e-3)	3.8190e-1(3.20e-3)
DTLZ1	0.0000e+0≈(0.00e+0)	9.8649e-3≈(5.40e-2)	0.0000e+0≈(0.00e+0)	**1.2189e-1+ (1.61e-1)**	0.0000e+0≈(0.00e+0)	0.0000e+0(0.00e+0)
DTLZ2	5.3450e-1+(6.34e-3)	4.4006e-1+(1.13e-2)	**5.6947e-1+ (9.91e-4)**	5.0876e-1+(1.02e-2)	5.4982e-1+(3.26e-3)	3.9933e-1(1.82e-2)
DTLZ3	0.0000e+0≈(0.00e+0)	3.4244e-3≈(1.88e-2)	0.0000e+0≈(0.00e+0)	**3.5422e-3≈ (1.94e-2)**	0.0000e+0≈(0.00e+0)	0.0000e+0(0.00e+0)
DTLZ4	3.2980e-1-(7.19e-2)	2.8026e-1-(4.41e-2)	**5.6894e-1+ (1.03e-3)**	3.1611e-1-(9.53e-2)	4.2814e-1+(4.88e-2)	3.8637e-1(3.76e-2)
DTLZ5	1.9433e-1≈(3.04e-3)	1.7053e-1-(7.99e-3)	1.9805e-1+(5.53e-4)	**2.0000e-1+ (2.56e-4)**	1.4229e-1-(8.87e-3)	1.9343e-1(2.04e-3)
DTLZ6	0.0000e+0-(0.00e+0)	1.8982e-1-(2.57e-4)	1.9805e-1≈(4.60e-4)	4.3321e-2-(8.04e-2)	8.5754e-3-(3.32e-2)	**1.9827e-1 (7.60e-4)**
DTLZ7	3.7338e-5-(9.96e-5)	2.4411e-1+(4.47e-3)	**2.8144e-1+ (7.47e-4)**	1.1621e-1≈(7.15e-2)	6.6672e-2-(3.21e-2)	1.0280e-1(2.29e-2)
UF1	1.7708e-1-(7.82e-2)	7.9527e-2-(5.74e-2)	**5.2511e-1+ (4.00e-2)**	2.4429e-1-(7.44e-2)	3.5195e-1-(6.05e-2)	5.0140e-1(2.34e-2)
UF2	5.9919e-1-(1.21e-2)	6.1685e-1-(5.85e-3)	6.1854e-1-(6.17e-3)	6.0938e-1-(9.05e-3)	5.7836e-1-(1.09e-2)	**6.3620e-1 (4.11e-3)**
UF3	1.5673e-1-(1.65e-2)	3.0588e-1-(8.25e-3)	2.9064e-1-(4.41e-2)	1.9546e-1-(3.11e-2)	1.7059e-1-(1.24e-2)	**3.2391e-1 (1.96e-2)**
UF4	2.9248e-1-(1.65e-2)	2.5374e-1-(8.01e-3)	**3.6044e-1≈ (1.15e-2)**	2.9542e-1-(1.10e-2)	3.0969e-1-(3.63e-3)	3.5712e-1(8.97e-3)
UF5	0.0000e+0≈(0.00e+0)	0.0000e+0≈(0.00e+0)	0.0000e+0≈(0.00e+0)	0.0000e+0≈(0.00e+0)	0.0000e+0≈(0.00e+0)	0.0000e+0(0.00e+0)
UF6	0.0000e+0-(0.00e+0)	0.0000e+0-(0.00e+0)	**2.3919e-2+ (3.04e-2)**	0.0000e+0-(0.00e+0)	0.0000e+0-(0.00e+0)	6.1517e-3(9.61e-3)
UF7	4.9807e-2-(3.74e-2)	1.7201e-1-(5.79e-2)	3.6077e-1-(1.24e-1)	2.0239e-1-(8.51e-2)	2.8265e-1-(5.97e-2)	**4.2467e-1 (1.43e-2)**
UF8	1.7705e-1-(3.09e-2)	2.6619e-1-(1.93e-2)	2.9081e-1-(4.41e-2)	1.6892e-1-(4.14e-2)	5.2525e-2-(1.66e-2)	**3.3260e-1 (2.37e-2)**
UF9	2.2338e-1-(3.78e-2)	2.2908e-1-(1.90e-2)	3.1397e-1-(4.32e-2)	2.0128e-1-(3.77e-2)	1.1838e-1-(3.41e-2)	**6.2740e-1 (1.61e-2)**
UF10	0.0000e+0-(0.00e+0)	9.0892e-2-(3.65e-5)	3.0066e-6-(1.65e-5)	0.0000e+0-(0.00e+0)	0.0000e+0-(0.00e+0)	**2.2771e-1 (6.40e-2)**
+ / - / ≈	1/16/5	3/15/4	7/10/5	4/14/4	2/17/3	-
Best/all	0/22	0/22	6/22	4/22	0/22	11/22



Continued on next page

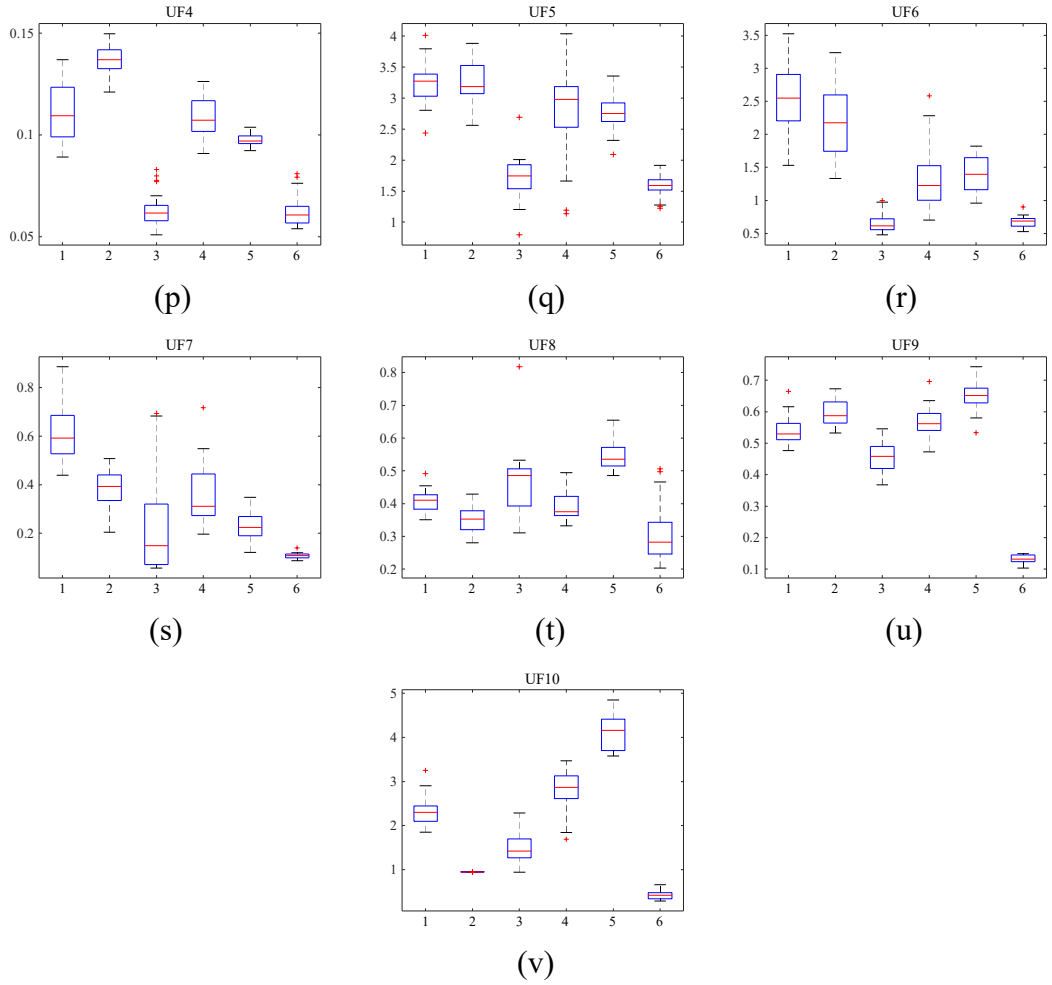


Figure 4. Box statistical plots of IGD on the 22 test functions.

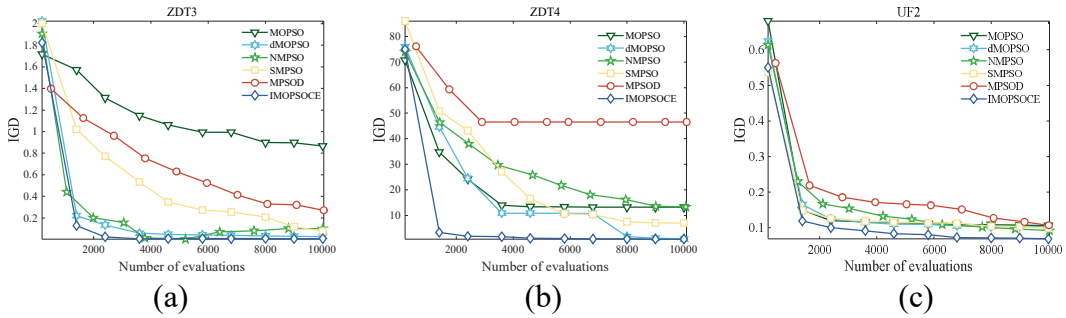


Figure 5. IGD convergence trajectories of IMOPSOCE and five MOPSOs on ZDT3, ZDT4 and UF2.

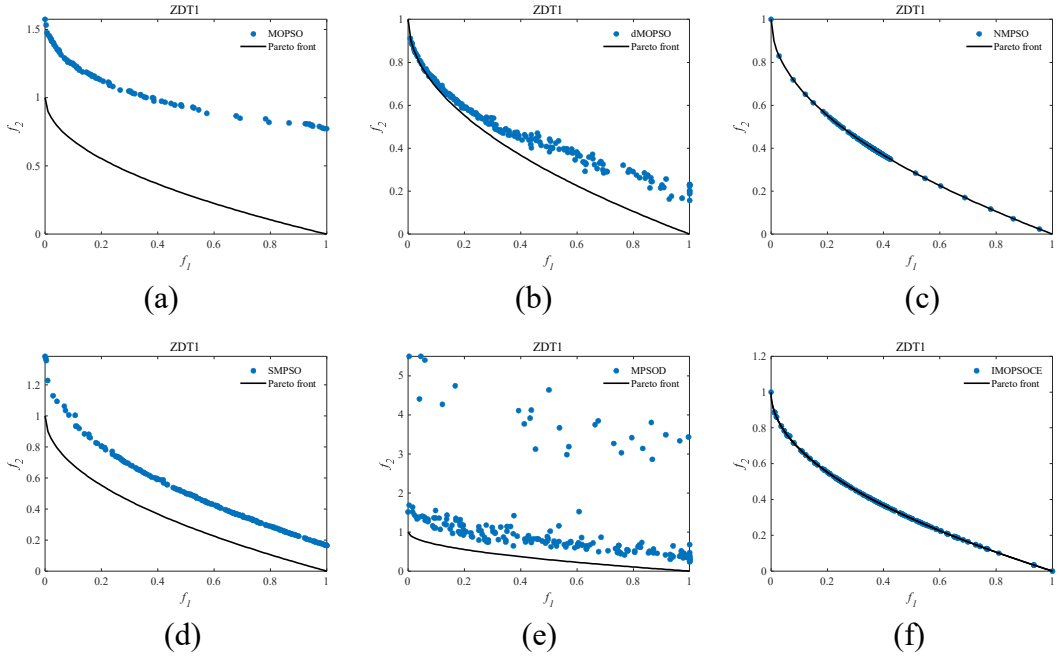


Figure 6. Approximate Pareto front of IMOPSOCE and MOPSOs on the bi-objective test function ZDT1. (a) MOPSO, (b) dMOPSO, (c) NMPSO, (d) SMPSO, (e) MPSOD and (f) IMOPSOCE.

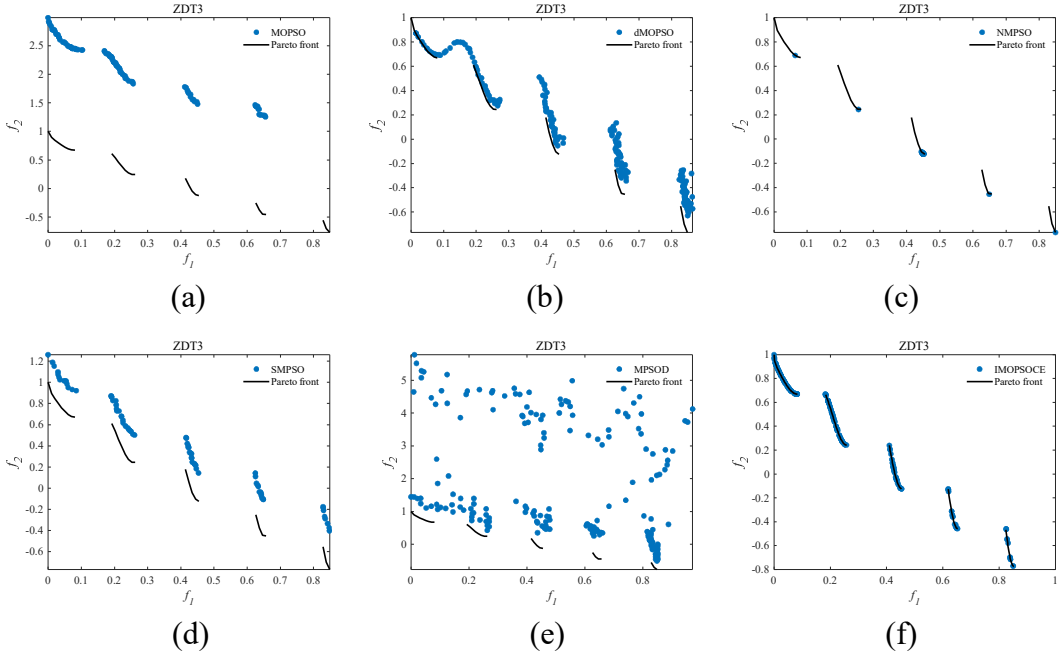


Figure 7. Approximate Pareto front of IMOPSOCE and MOPSOs on the bi-objective test function ZDT3. (a) MOPSO, (b) dMOPSO, (c) NMPSO, (d) SMPSO, (e) MPSOD and (f) IMOPSOCE.

As shown in Figure 5, the IGD convergence trajectories of the six algorithms for the test functions ZDT3, ZDT4, and UF2 are plotted. IGD values are recorded 10 times in a certain time interval for each algorithm. The figure shows that, in comparison to the other five algorithms,

IMOPSOCE better balances the exploitation and exploration capabilities, leading to a better convergence of the algorithm. The proposed IMOPSOCE has a good convergence because its proposed strategy can successfully prevent the population from falling into the local optimum in order to find solutions with better convergence.

Figures 6 and 7 plot the approximate Pareto front obtained on the bi-objective test functions ZDT1 and ZDT3 to visualize the optimization result. Figure 6 shows that NMPSO and IMOPSOCE can cover the true Pareto front very well, mainly because of their good convergence, while the other four compared algorithms have a poor coverage of the true Pareto front. Meanwhile, the approximate Pareto fronts of the other five compared algorithms cannot be uniformly distributed, as obtained by IMOPSOCE. Figure 7 shows that on ZDT3, except for IMOPSOCE, which can cover the true Pareto front, the other five compared algorithms have difficulty approaching the true Pareto front. Additionally, MOPSO, SMPSO, and MPSOD cannot converge to the true Pareto front for ZDT3 with an unconnected PF, which is consistent with the conclusion in Table 3. NMPSO can only find a small number of Pareto optimal solutions for the ZDT3 function on the Pareto front. This can intuitively show that, when compared to the other five algorithms, IMOPSOCE has the best convergence effect. This is mainly because IMOPSOCE uses an external archive maintenance strategy based on the inflection point distance and the distribution coefficient to guarantee that the non-dominated solutions in the external archive have better comprehensive performance. IMOPSOCE takes advantage of the hierarchical idea to provide different flight modes for particles at different layers, which improves the search capabilities of the algorithm and makes it have better convergence.

4.4.2. Comparisons with five existing MOEAs

Tables 5 and 6 show the means and standard deviations of the IGD indicator and HV indicator for 30 independent runs on the ZDT, DTLZ, and UF functions for NSGAIII, MOEAD, MOEAIGDNS, SPEAR, VaEA, and the proposed IMOPSOCE, respectively. Additionally, the experiments adopt the Wilcoxon rank sum test to obtain statistically sound conclusions at the 0.05 significance level. Combining the data in Tables 5 and 6, the proposed IMOPSOCE still performs better than the other five classical algorithms and still outperforms MOEAs on these test functions. To visually show the optimization result, the optimal values in the table are bolded.

Table 5 shows the means and standard deviations of the IGD indicator of the five classical algorithms and IMOPSOCE on the 22 test functions. IMOPSOCE performs the best compared to the five compared algorithms on ZDT1-ZDT4, ZDT6, DTLZ6, UF3-4, and UF7-10. According to the Wilcoxon rank sum test results shown in the second-last row of Table 5, the IMOPSOCE has significantly better IGD values than NSGAIII, MOEAD, MOEAIGDNS, SPEAR, and VaEA on 12, 16, 12, 12, and 11 test functions, respectively. The performance of IMOPSOCE is the most competitive, even though its advantage compared to these five MOEAs is not particularly outstanding. For instance, IMOPSOCE outperforms VaEA on ZDT1 by a factor of 10 and SPEAR on DTLZ6 by a factor of 17. IMOPSOCE outperforms the compared MOEAs and exhibits the best performance on the ZDT and UF test functions. Due to the somewhat higher IGD values of IMOPSOCE, its performance for the three-objective DTLZ test functions is less than optimal. This indicates that the classical algorithms are better suited to solving the three-objective series of problems. When it comes to solving the three-objective DTLZ test functions, MOEAD demonstrates its superiority. The findings of the comparisons also demonstrate that the proposed IMOPSOCE has a good overall performance. IMOPSOCE achieves the 12 statistically best results on the 22 test

functions, according to the results in the last row of Table 5. To sum up the foregoing content, the proposed IMOPSOCE ranks first among the aforementioned compared algorithms due to the use of the external archive maintenance strategy, the comprehensive indicator combined with inflection point distance and distribution coefficient, and focusing on the convergence and diversity of non-dominated solutions at the same time; thus, the non-dominated solutions in the external archive have a better performance.

A similar conclusion that IMOPSOCE has superior performance can also be obtained from Table 6. The proposed IMOPSOCE achieves more than half of the best HV values on the 22 test functions, as shown in Table 6. IMOPSOCE achieves the best results on ZDT1-ZDT4, ZDT6, DTLZ6, UF2-UF4, and UF7-UF10. There are 0 optimal HV values for SPEAR, 1 optimal HV value each for NSGAIII and MOEAIGDNS, 3 optimal HV values each for MOEAD and VaEA, and 13 optimal HV values for the proposed IMOPSOCE. SPEAR performs relatively poorly compared to the other algorithms on the 22 test functions. IMOPSOCE outperforms NSGAIII, MOEAD, MOEAIGDNS, SPEAR, and VaEA on 13, 14, 13, 14, and 13 test functions, respectively. From the findings, it is evident that IMOPSOCE performs well on both the bi-objective and three-objective test functions. In general, compared with the existing MOEAs, the proposed IMOPSOCE achieves the best overall performance on the test functions, especially on the bi-objective test functions. The flight mode of the particles at the first layer is changed after each update due to the hierarchical update strategy of particle velocity, and the particles at the first layer continue their original movement trend. The convergence of the algorithm is effectively improved.

Figure 8 presents box plots of the IGD indicator for the six algorithms to illustrate the distribution of the data. When different algorithms are independently run 30 times on each test function, the box plots of the IMOPSOCE algorithm and compared algorithms with respect to the IGD indicator are shown in Figure 8 (1, 2, 3, 4, 5, and 6 on the horizontal coordinate represent NSGAIII, MOEAD, MOEAIGDNS, SPEAR, VaEA, and IMOPSOCE, respectively, and the vertical coordinate represents the IGD value of each algorithm). The data fluctuations of the six algorithms on ZDT1-ZDT4, ZDT6, DTLZ1-DTLZ7, and UF1-UF10 are shown in Figure 8. The above contents demonstrates unequivocally that, when compared to other MOEAs, IMOPSOCE can achieve the best Pareto optimal solutions. The results match those in Table 5. On the test functions ZDT1, ZDT2, ZDT3, ZDT6, UF7, and UF9, IMOPSOCE outperforms the other five algorithms by a wide margin. There is no discernible difference between NSGAIII, MOEAD, MOEAIGDNS, SPEAR, and VaEA on the test functions of DTLZ, with the exception of DTLZ4 and DTLZ6. IMOPSOCE performs better than the other five algorithms overall. As may be observed from the aforementioned experimental results, IMOPSOCE is very competitive in solving MOPs compared with the existing MOEAs.

The IGD convergence trajectories of the six algorithms for the test functions ZDT3, ZDT4, and UF2 are plotted in Figure 9. IGD values are recorded 10 times in a certain time interval for each algorithm. The figure shows that IMOPSOCE converges more quickly than the other five algorithms because we utilize an external archive maintenance strategy that combines the inflection point distance and distribution coefficient, resulting in the non-dominated solutions in the external archive having the best performance.

Figures 10 and 11 plot the Pareto fronts of NSGAIII, MOEAD, MOEAIGDNS, SPEAR, VaEA, and IMOPSOCE on the bi-objective test functions ZDT1 and ZDT2. Only IMOPSOCE can fit the true Pareto front in Figure 10, whereas the other five compared algorithms barely do. The primary reason is that IMOPSOCE makes particle searches more effective by using a random inertia weight

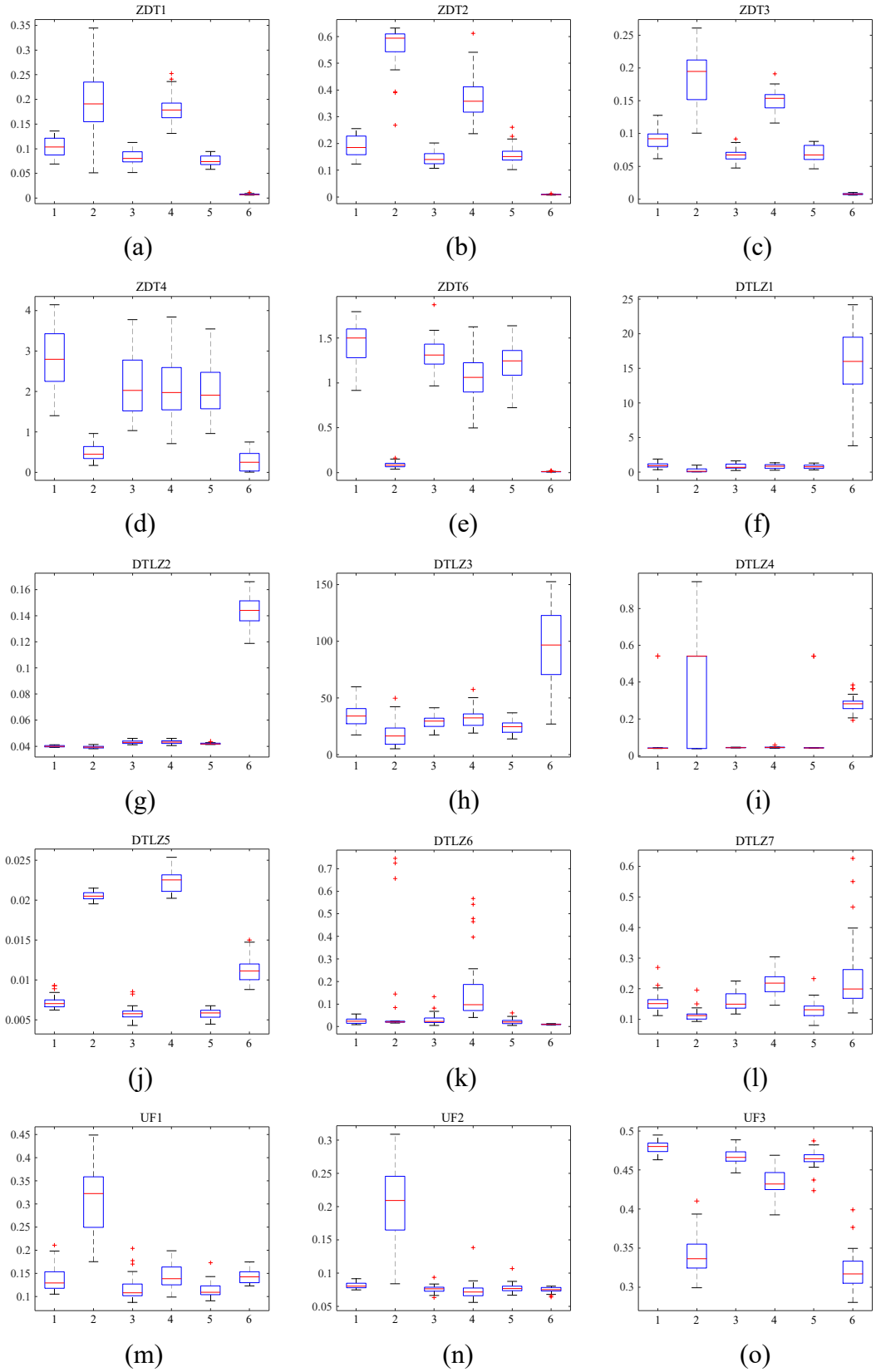
strategy. Figure 11 demonstrates that none of the other five MOEAs cover the true Pareto front, except for IMOPSOCE, which almost completely covers the true Pareto front of ZDT2. The two figures above show that only IMOPSOCE can be evenly distributed on the true Pareto front when compared to the other five algorithms. The primary reason for this is that IMOPSOCE employs an external archive maintenance strategy that combines inflection point distance and distribution coefficient, allowing it to measure the performance of non-dominated solutions from two perspectives of convergence and diversity and retain solutions with good comprehensive performance. The good diversity and convergence of IMOPSOCE allows it to better approximate the true Pareto front in the majority of test functions.

Table 5. IGD values of IMOPSOCE and five MOEAs on the test functions ZDT1-ZDT4 and ZDT6, DTLZ1-DTLZ7 and UF1-UF10.

Problem	NSGAIII	MOEAD	MOEAIGDNS	SPEAR	VaEA	IMOPSOCE
ZDT1	1.0361e-1- (1.84e-2)	1.9868e-1- (6.78e-2)	8.2561e-2- (1.44e-2)	1.7980e-1- (2.99e-2)	7.5846e-2- (1.07e-2)	**7.4580e-3 (1.13e-3)**
ZDT2	1.9225e-1- (4.02e-2)	5.5920e-1- (8.25e-2)	1.4279e-1- (2.38e-2)	3.6717e-1- (8.42e-2)	1.5661e-1- (3.52e-2)	**8.5043e-3 (1.40e-3)**
ZDT3	9.1523e-2- (1.77e-2)	1.8201e-1- (4.41e-2)	6.7850e-2- (9.96e-3)	1.5037e-1- (1.60e-2)	6.9368e-2- (1.15e-2)	**7.7408e-3 (1.15e-3)**
ZDT4	2.8099e+0- (7.61e-1)	4.8205e-1- (1.88e-1)	2.1566e+0- (7.86e-1)	2.0743e+0- (7.66e-1)	2.0778e+0- (6.82e-1)	**2.7971e-1 (2.29e-1)**
ZDT6	1.4438e+0- (2.45e-1)	8.8457e-2- (3.19e-2)	1.3222e+0- (1.91e-1)	1.0811e+0- (2.39e-1)	1.2357e+0- (2.14e-1)	**1.0595e-2 (3.76e-3)**
DTLZ1	9.7952e-1+ (3.79e-1)	**2.6621e-1+ (2.97e-1)**	8.4112e-1+ (3.89e-1)	8.3578e-1+ (3.05e-1)	7.7967e-1+ (2.94e-1)	1.5513e+1 (5.12e+0)
DTLZ2	3.9978e-2+ (6.22e-4)	**3.9402e-2+ (7.93e-4)**	4.3083e-2+ (1.16e-3)	4.3246e-2+ (1.49e-3)	4.1957e-2+ (5.72e-4)	1.4368e-1 (1.17e-2)
DTLZ3	3.4821e+1+ (9.38e+0)	**1.8789e+1+ (1.13e+1)**	2.9226e+1+ (5.78e+0)	3.1891e+1+ (8.74e+0)	2.4318e+1+ (5.57e+0)	9.4523e+1 (3.49e+1)
DTLZ4	7.4209e-2+ (1.27e-1)	3.8392e-1- (2.85e-1)	**4.3842e-2+ (1.19e-3)**	4.5053e-2+ (3.00e-3)	7.5419e-2+ (1.27e-1)	2.7893e-1 (4.56e-2)
DTLZ5	7.2490e-3+ (8.41e-4)	2.0509e-2- (5.16e-4)	5.8694e-3+ (8.95e-4)	2.2481e-2- (1.48e-3)	**5.8008e-3+ (5.95e-4)**	1.1122e-2 (1.53e-3)
DTLZ6	2.7042e-2- (1.29e-2)	9.5240e-2- (2.10e-1)	3.4208e-2- (2.73e-2)	1.7202e-1- (1.57e-1)	2.3555e-2- (1.34e-2)	**1.0169e-2 (1.79e-3)**
DTLZ7	1.5684e-1+ (3.27e-2)	**1.1484e-1+ (1.99e-2)**	1.5805e-1+ (3.32e-2)	2.1368e-1≈ (3.85e-2)	1.3228e-1+ (2.92e-2)	2.3897e-1 (1.21e-1)
UF1	1.3915e-1≈ (2.90e-2)	3.1483e-1- (7.21e-2)	1.1863e-1+ (2.74e-2)	1.4317e-1≈ (2.66e-2)	**1.1503e-1+ (1.78e-2)**	1.4230e-1 (1.39e-2)
UF2	8.1327e-2- (4.31e-3)	2.0440e-1- (5.90e-2)	7.5905e-2≈ (5.84e-3)	**7.3285e-2≈ (1.48e-2)**	7.7204e-2≈ (7.36e-3)	7.4755e-2 (4.15e-3)
UF3	4.7968e-1- (7.98e-3)	3.4122e-1- (2.74e-2)	4.6613e-1- (1.06e-2)	4.3357e-1- (1.58e-2)	4.6417e-1- (1.18e-2)	**3.2163e-1 (2.43e-2)**
UF4	9.4798e-2- (2.58e-3)	1.1593e-1- (4.67e-3)	8.8198e-2- (3.79e-3)	8.5335e-2- (2.06e-3)	9.0046e-2- (3.10e-3)	**6.2371e-2 (7.02e-3)**
UF5	1.4574e+0+ (3.05e-1)	1.3741e+0+ (2.69e-1)	**1.1081e+0+ (2.61e-1)**	1.1259e+0+ (2.17e-1)	1.1652e+0+ (3.55e-1)	1.5851e+0 (1.68e-1)
UF6	7.0048e-1≈ (1.37e-1)	6.0102e-1+ (2.25e-1)	5.8255e-1+ (9.23e-2)	6.7400e-1≈ (9.11e-2)	**5.8149e-1+ (8.09e-2)**	6.7761e-1 (8.19e-2)
UF7	2.0268e-1- (8.77e-2)	4.1262e-1- (1.39e-1)	1.7940e-1- (8.98e-2)	1.8238e-1- (6.99e-2)	1.6080e-1- (8.48e-2)	**1.0721e-1 (1.10e-2)**
UF8	3.3309e-1≈ (2.84e-2)	5.4074e-1- (2.52e-1)	3.9258e-1- (3.15e-2)	3.1871e-1≈ (1.58e-2)	3.4251e-1≈ (4.05e-2)	**3.1105e-1 (8.65e-2)**
UF9	5.0027e-1- (4.79e-2)	5.5908e-1- (7.95e-2)	5.0022e-1- (4.85e-2)	4.7566e-1- (6.23e-2)	4.7798e-1- (6.02e-2)	**1.3122e-1 (1.25e-2)**
UF10	2.5472e+0- (3.75e-1)	7.5099e-1- (9.55e-2)	1.5340e+0- (3.15e-1)	1.9784e+0- (3.88e-1)	1.4437e+0- (3.24e-1)	**4.2001e-1 (9.46e-2)**
+ / - / ≈	7/12/3	6/16/0	9/12/1	5/12/5	9/11/2	-
Best/all	0/22	4/22	2/22	1/22	3/22	12/22

Table 6. HV values of IMOPSOCE and five MOEAs on the test functions ZDT1-ZDT4 and ZDT6, DTLZ1-DTLZ7 and UF1-UF10.

Problem	NSGAIII	MOEAD	MOEAIGDNS	SPEAR	VaEA	IMOPSOCE
ZDT1	5.8567e-1- (2.27e-2)	5.3373e-1- (4.98e-2)	6.1280e-1- (1.87e-2)	4.8930e-1- (3.15e-2)	6.2123e-1- (1.38e-2)	**7.1622e-1 (1.31e-3)**
ZDT2	2.1776e-1- (3.43e-2)	9.3077e-2- (2.80e-2)	2.6429e-1- (2.31e-2)	8.5507e-2- (4.68e-2)	2.5202e-1- (2.98e-2)	**4.4006e-1 (1.76e-3)**
ZDT3	5.3936e-1- (1.31e-2)	5.9420e-1≈ (7.06e-2)	5.5647e-1- (7.52e-3)	5.0460e-1- (1.55e-2)	5.5441e-1- (7.13e-3)	**5.9870e-1 (3.73e-4)**
ZDT4	0.0000e+0- (0.00e+0)	2.1224e-1- (1.40e-1)	0.0000e+0- (0.00e+0)	1.8871e-3- (1.03e-2)	0.0000e+0- (0.00e+0)	**5.0728e-1 (1.61e-1)**
ZDT6	0.0000e+0- (0.00e+0)	2.7290e-1- (3.68e-2)	0.0000e+0- (0.00e+0)	9.3888e-4- (5.14e-3)	0.0000e+0- (0.00e+0)	**3.8190e-1 (3.20e-3)**
DTLZ1	3.3512e-3≈ (1.28e-2)	**4.2524e-1+ (3.50e-1)**	1.5097e-2≈ (5.44e-2)	8.5317e-3+ (3.06e-2)	6.3437e-3+ (1.96e-2)	0.0000e+0 (0.00e+0)
DTLZ2	**5.6197e-1+ (1.45e-3)**	5.6176e-1+ (2.22e-3)	5.5642e-1+ (2.33e-3)	5.5625e-1+ (2.29e-3)	5.5991e-1+ (1.34e-3)	3.9933e-1 (1.82e-2)
DTLZ3	0.0000e+0≈ (0.00e+0)	0.0000e+0≈ (0.00e+0)	0.0000e+0≈ (0.00e+0)	0.0000e+0≈ (0.00e+0)	0.0000e+0≈ (0.00e+0)	0.0000e+0 (0.00e+0)
DTLZ4	5.4659e-1+ (5.49e-2)	4.0946e-1≈ (1.37e-1)	**5.5842e-1+ (2.32e-3)**	5.5316e-1+ (3.27e-3)	5.4574e-1+ (5.41e-2)	3.8637e-1 (3.76e-2)
DTLZ5	1.9646e-1+ (7.70e-4)	1.8770e-1- (1.93e-3)	1.9751e-1+ (5.88e-4)	1.8595e-1- (1.47e-3)	**1.9803e-1+ (4.79e-4)**	1.9343e-1 (2.04e-3)
DTLZ6	1.8312e-1- (9.26e-3)	1.6005e-1- (5.77e-2)	1.7772e-1- (1.55e-2)	9.8452e-2- (5.37e-2)	1.8418e-1- (9.04e-3)	**1.9827e-1 (7.60e-4)**
DTLZ7	2.1469e-1+ (1.16e-2)	**2.2775e-1+ (1.13e-2)**	2.1571e-1+ (1.12e-2)	1.9571e-1+ (1.22e-2)	2.2557e-1+ (1.12e-2)	1.0280e-1 (2.29e-2)
UF1	5.0877e-1≈ (3.80e-2)	4.1510e-1- (4.49e-2)	5.4354e-1+ (3.41e-2)	5.0749e-1≈ (3.22e-2)	**5.4417e-1+ (2.34e-2)**	5.0140e-1 (2.34e-2)
UF2	6.1511e-1- (5.12e-3)	5.6447e-1- (2.51e-2)	6.2684e-1- (5.97e-3)	6.2648e-1- (1.09e-2)	6.2559e-1- (6.00e-3)	**6.3620e-1 (4.11e-3)**
UF3	1.8484e-1- (8.11e-3)	2.9905e-1- (3.59e-2)	1.9296e-1- (1.01e-2)	2.2237e-1- (1.54e-2)	1.9665e-1- (1.02e-2)	**3.2391e-1 (1.96e-2)**
UF4	3.1399e-1- (3.36e-3)	2.8478e-1- (5.58e-3)	3.2394e-1- (4.70e-3)	3.2451e-1- (2.65e-3)	3.2108e-1- (3.76e-3)	**3.5712e-1 (8.97e-3)**
UF5	0.0000e+0≈ (0.00e+0)	0.0000e+0≈ (0.00e+0)	7.6429e-4≈ (3.46e-3)	1.5758e-3≈ (6.02e-3)	**7.1833e-3+ (2.14e-2)**	0.0000e+0 (0.00e+0)
UF6	1.2068e-2≈ (1.48e-2)	**7.8446e-2+ (7.11e-2)**	3.0343e-2+ (2.15e-2)	1.0186e-2≈ (1.61e-2)	2.9364e-2+ (2.28e-2)	6.1517e-3 (9.61e-3)
UF7	3.2904e-1- (7.88e-2)	2.4808e-1- (7.27e-2)	3.6112e-1- (7.10e-2)	3.5550e-1- (5.37e-2)	3.7978e-1- (7.92e-2)	**4.2467e-1 (1.43e-2)**
UF8	2.1302e-1- (4.47e-2)	1.5648e-1- (5.02e-2)	2.5428e-1- (4.27e-2)	1.8835e-1- (2.88e-2)	2.6155e-1- (3.42e-2)	**3.3260e-1 (2.37e-2)**
UF9	2.5073e-1- (4.13e-2)	2.6915e-1- (4.41e-2)	2.6735e-1- (4.05e-2)	2.6781e-1- (5.23e-2)	2.8537e-1- (4.42e-2)	**6.2740e-1 (1.61e-2)**
UF10	0.0000e+0- (0.00e+0)	3.0154e-2- (2.78e-2)	0.0000e+0- (0.00e+0)	0.0000e+0- (0.00e+0)	0.0000e+0- (0.00e+0)	**2.2771e-1 (6.40e-2)**
+ / - / ≈	4/13/5	4/14/4	6/13/3	4/14/4	8/13/1	-
Best/all	1/22	3/22	1/22	0/22	3/22	13/22



Continued on next page

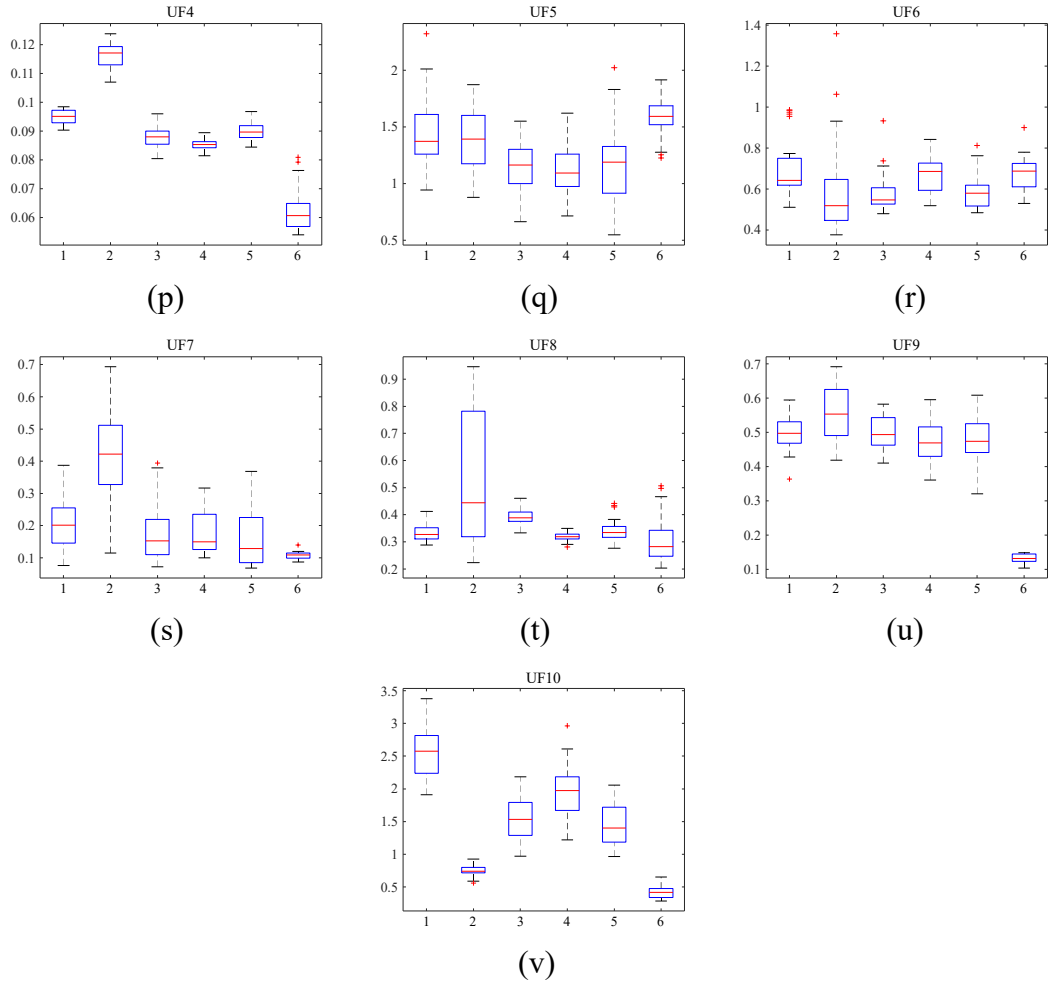


Figure 8. Box statistical plots of IGD on the 22 test functions.

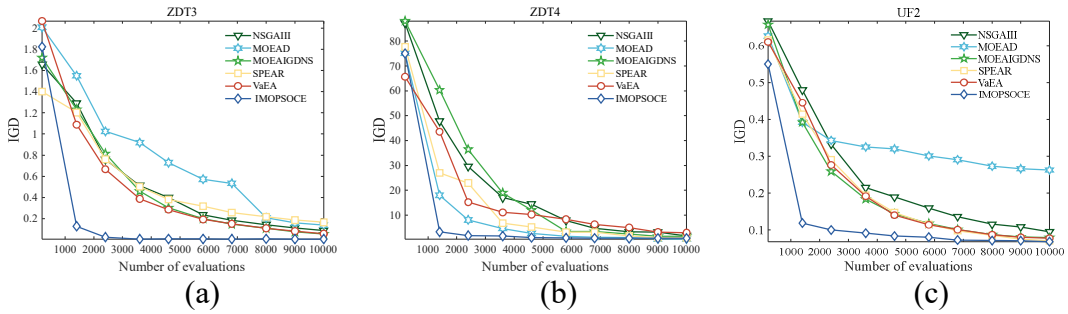


Figure 9. IGD convergence trajectories of IMOPSOCE and five MOEAs on ZDT3, ZDT4 and UF2.

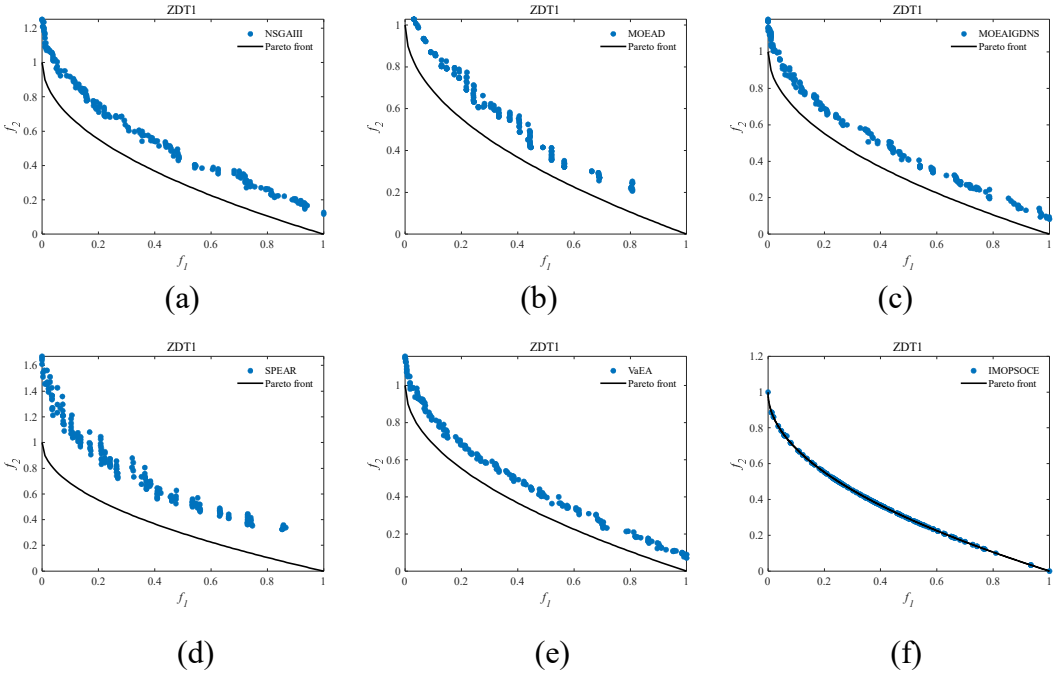


Figure 10. Approximate Pareto front of IMOPSOCE and MOEAs on the bi-objective test function ZDT1. (a) NSGAIII, (b) MOEAD, (c) MOEAIGDNS, (d) SPEAR, (e) VaEA and (f) IMOPSOCE.

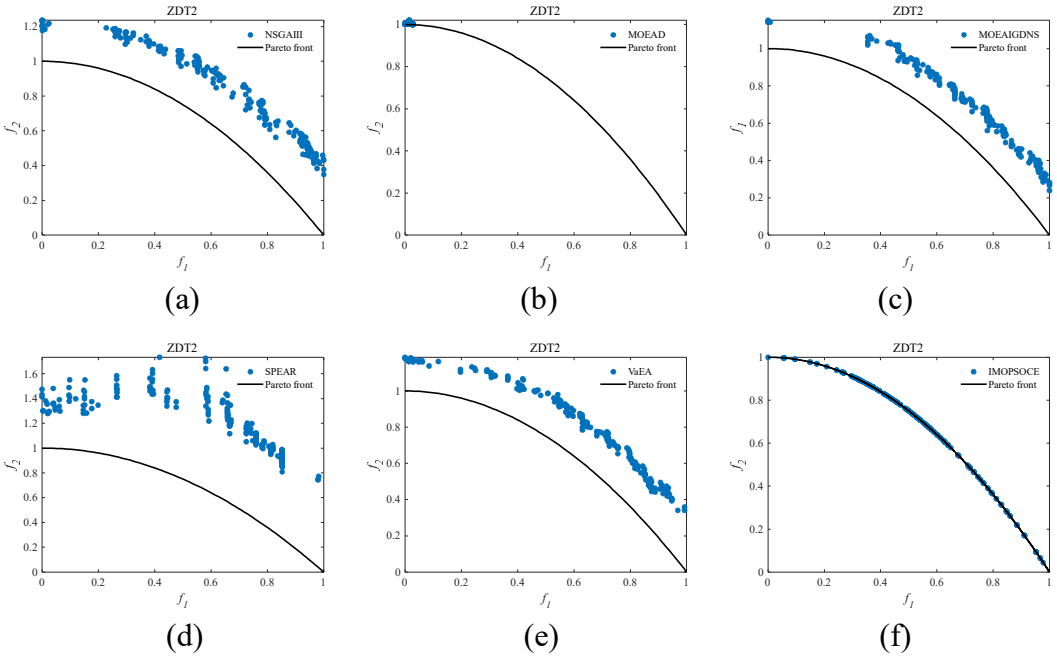


Figure 11. Approximate Pareto front of IMOPSOCE and MOEAs on the bi-objective test function ZDT2. (a) NSGAIII, (b) MOEAD, (c) MOEAIGDNS, (d) SPEAR, (e) VaEA and (f) IMOPSOCE.

4.4.3. Comparisons using non-parametric test

In order to obtain more comprehensive comparative statistics and determine whether the obtained results are correct, we conducted a Friedman rank test [43,44] on the experimental results to compare the performance of various algorithms.

Table 7. Friedman rank test results of IGD values for all comparison algorithms and IMOPSOCE.

Algorithm	ZDT:ZDT1-4,6		DTLZ:DTLZ1-7		UF:UF1-10		Total	
	Friedman test	Rank	Friedman test	Rank	Friedman test	Rank	Friedman test	Rank
MOPSO	9.80	11	9.29	11	9.40	11	9.45	11
dMOPSO	3.60	2	7.07	7	7.90	8	6.66	8
NMPSO	4.40	3	4.86	5	4.90	5	4.77	4
SMPSO	5.00	4	7.57	9	8.30	9	7.32	9
MPSOD	8.00	10	8.57	10	9.00	10	8.64	10
NSGAIII	7.40	8	4.43	4	5.85	6	5.75	6
MOEAD	7.40	8	4.36	3	6.90	7	6.20	7
MOEAIGDNS	5.80	5	3.86	2	4.05	4	4.39	3
SPEAR	7.20	7	5.43	6	3.80	3	5.09	5
VaEA	5.80	5	**3.14**	**1**	3.40	2	**3.86**	**1**
IMOPSOCE	**1.60**	**1**	7.43	8	**2.50**	**1**	**3.86**	**1**

Table 8. Friedman rank test results of HV values for all comparison algorithms and IMOPSOCE.

Algorithm	ZDT:ZDT1-4,6		DTLZ:DTLZ1-7		UF:UF1-10		Total	
	Friedman test	Rank	Friedman test	Rank	Friedman test	Rank	Friedman test	Rank
MOPSO	9.40	11	8.79	11	9.20	11	9.11	11
dMOPSO	2.80	2	6.00	7	7.00	8	5.73	5
NMPSO	3.60	3	**3.36**	**1**	3.65	2	**3.55**	**1**
SMPSO	5.00	4	5.71	6	8.50	9	6.82	9
MPSOD	7.20	7	8.64	10	8.90	10	8.43	10
NSGAIII	8.00	9	5.14	4	6.55	7	6.43	8
MOEAD	6.00	5	5.14	4	6.45	6	5.93	6
MOEAIGDNS	7.00	6	4.57	2	4.30	4	5.00	4
SPEAR	8.20	10	6.43	8	4.90	5	6.14	7
VaEA	7.20	7	4.71	3	3.70	3	4.82	3
IMOPSOCE	**1.60**	**1**	7.50	9	**2.85**	**1**	4.05	2

Table 7 presents the Friedman rank test rankings of the IGD values for all algorithms on the ZDT, DTLZ, and UF benchmark suites, with the highest ranking shown in bold. As shown in Table 7, IMOPSOCE ranks first on both the ZDT and UF benchmark suites. The total highest ranking is obtained by IMOPSOCE and VaEA, which are, in turn, MOEAIGDNS, NMPSO, SPEAR, NSGAIII, MOEAD, dMOPSO, SMPSO, MPSOD, and MOPSO. Table 8 presents the Friedman rank test

rankings of the HV values for all algorithms on the ZDT, DTLZ, and UF benchmark suites. Similar to Table 7, IMOPSOCE achieves the first ranking on the ZDT and UF benchmark suites. IMOPSOCE is ranked second in the total ranking, with only a slight gap between it and NMPSO, which is ranked first. Combining Tables 7 and 8, it can be seen that the IMOPSOCE proposed in this paper can obtain the best ranking on the ZDT and UF benchmark suites for both IGD and HV. The experimental results show that IMOPSOCE provides better overall performance compared to other comparison algorithms and has obvious advantages.

5. Conclusions

In this paper, IMOPSOCE, a novel MOPSO algorithm with a comprehensive indicator and layered particle swarm optimizer, has been proposed for solving MOPs. First, the CM indicator, which is based on inflection point distance and distribution coefficient, is used to measure the performance of non-dominated solutions in the external archive. Superior-performing non-dominated solutions are retained, while inferior-performing solutions are eliminated. Second, to balance the capacities for exploitation and exploration of particles, the random inertia weight strategy is used. Finally, the particles in the population are divided into two layers according to the levels after updating, and different flight modes are offered for the particles in different levels in order to improve the efficiency of the algorithm in solving problems and the optimization capability. Twenty-two test functions are used to validate the proposed IMOPSOCE. The results of the experiment demonstrate that, compared to five other MOPSOs and MOEAs, IMOPSOCE is better able to maintain the diversity of Pareto optimal solutions and make the Pareto optimal solutions converge to the true Pareto front.

Some questions can be further studied in future work. First, in order to better balance exploration and exploitation capabilities, it would be interesting to design a better hierarchical structure. Second, employing learning strategies is highly efficient for improving MOPSO performance. To get better solutions, various learning strategies might be taken into account. Finally, we can optimize the suggested algorithm and attempt to utilize it to address specific issues in the real world. This is a brand-new subject that merits more research.

Use of AI tools declaration

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

Acknowledgments

This work was supported in part by Key Laboratory of Evolutionary Artificial Intelligence in Guizhou (Qian Jiaoji [2022] No. 059) and the Key Talens Program in digital economy of Guizhou Province.

Conflicts of interest

The authors declare there is no conflict of interest.

References

1. C. A. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, *IEEE Trans. Evol. Comput.*, **8** (2004), 256–279. https://doi.org/10.1109/TEVC.2004.826067
2. X. Zhao, J. Guo, M. He, Multiobjective optimization of multisource heating system based on improving diversification and implementation, *Energy Convers. Manage.*, **266** (2022), 115789. https://doi.org/10.1016/j.enconman.2022.115789
3. S. Chen, Y. Yang, M. Qin, Q. Xu, Coordinated multiobjective optimization of the integrated energy distribution system considering network reconfiguration and the impact of price fluctuation in the gas market, *Int. J. Electr. Power Energy Syst.*, **138** (2022), 107776. https://doi.org/10.1016/j.ijepes.2021.107776
4. W. Tan, X. Yuan, G. Huang, Z. Liu, Low-carbon joint scheduling in flexible open-shop environment with constrained automatic guided vehicle by multi-objective particle swarm optimization, *Appl. Soft Comput.*, **111** (2021), 107695. https://doi.org/10.1016/j.asoc.2021.107695
5. M. Li, S. Yang, M. Zhang, Power supply system scheduling and clean energy application based on adaptive chaotic particle swarm optimization, *Alexandria Eng. J.*, **61** (2022), 2074–2087. https://doi.org/10.1016/j.aej.2021.08.008
6. A. H. Barahimi, A. Eydi, A. Aghaie, Multi-modal urban transit network design considering reliability: multi-objective bi-level optimization, *Reliab. Eng. Syst. Saf.*, **216** (2021), 107922. https://doi.org/10.1016/j.ress.2021.107922
7. H. Li, S. Wang, Q. Chen, M. Gong, L. Chen, IPSMT: multi-objective optimization of multipath transmission strategy based on improved immune particle swarm algorithm in wireless sensor networks, *Appl. Soft Comput.*, **121** (2022), 108705. https://doi.org/10.1016/j.asoc.2022.108705
8. A. Mehto, S. Tapaswi, K. K. Pattanaik, Multi-objective particle swarm optimization based rendezvous point selection for the energy and delay efficient networked wireless sensor data acquisition, *J. Network Comput. Appl.*, **195** (2021), 103234. https://doi.org/10.1016/j.jnca.2021.103234
9. Y. Zhang, L. Yuan, Q. Zhang, X. Sun, Multi-objective optimization of building energy performance using a particle swarm optimizer with less control parameters, *J. Build. Eng.*, **32** (2020), 101505. https://doi.org/10.1016/j.jobe.2020.101505
10. M. A. Mosa, A novel hybrid particle swarm optimization and gravitational search algorithm for multi-objective optimization of text mining, *Appl. Soft Comput.*, **90** (2020), 106189. https://doi.org/10.1016/j.asoc.2020.106189
11. M. Kaucic, Equity portfolio management with cardinality constraints and risk parity control using multi-objective particle swarm optimization, *Comput. Oper. Res.*, **109** (2019), 300–316. https://doi.org/10.1016/j.cor.2019.05.014
12. 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
13. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

14. S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, *Neural Comput. Appl.*, **27** (2016), 1053–1073. https://doi.org/10.1007/s00521-015-1920-1
15. C. A. C. Coello, M. S. Lechuga, MOPSO: a proposal for multiple objective particle swarm optimization, in *Proceedings of the 2002 Congress on Evolutionary Computation*, **2** (2002), 1051–1056. https://doi.org/10.1109/CEC.2002.1004388
16. H. T. Chen, W. C. Wang, X. N. Chen, L. Qiu, Multi-objective reservoir operation using particle swarm optimization with adaptive random inertia weights, *Water Sci. Eng.*, **13** (2020), 136–144. https://doi.org/10.1016/j.wse.2020.06.005
17. M. Roshanzamir, M. A. Balafar, S. N. Razavi, A new hierarchical multi group particle swarm optimization with different task allocations inspired by holonic multi agent systems, *Expert Syst. Appl.*, **149** (2020), 113292. https://doi.org/10.1016/j.eswa.2020.113292
18. K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multi-objective optimization, in *Evolutionary Multiobjective Optimization*, (2005), 105–145. https://doi.org/10.1007/1-84628-137-7_6
19. K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.*, **6** (2002), 182–197. https://doi.org/10.1109/4235.996017
20. C. R. Raquel, P. C. Naval, An effective use of crowding distance in multiobjective particle swarm optimization, in *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, (2005), 257–264. https://doi.org/10.1145/1068009.1068047
21. J. Liu, F. Li, X. Kong, P. Huang, Handling many-objective optimisation problems with R2 indicator and decomposition-based particle swarm optimiser, *Int. J. Syst. Sci.*, **50** (2019), 320–336. https://doi.org/10.1080/00207721.2018.1552765
22. J. Luo, A. Gupta, Y. S. Ong, Z. Wang, Evolutionary optimization of expensive multiobjective problems with co-sub-Pareto front Gaussian process surrogates, *IEEE Trans. Cybern.*, **49** (2018), 1708–1721. https://doi.org/10.1109/TCYB.2018.2811761
23. W. Hu, G. G. Yen, Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system, *IEEE Trans. Evol. Comput.*, **19** (2013), 1–18. https://doi.org/10.1109/TEVC.2013.2296151
24. H. Han, W. Lu, J. Qiao, An adaptive multiobjective particle swarm optimization based on multiple adaptive methods, *IEEE Trans. Cybern.*, **47** (2017), 2754–2767. https://doi.org/10.1109/TCYB.2017.2692385
25. P. K. Tripathi, S. Bandyopadhyay, S. K. Pal, Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients, *Inf. Sci.*, **177** (2007), 5033–5049. https://doi.org/10.1016/j.ins.2007.06.018
26. K. Shibata, H. Nakano, A. Miyauchi, A learning method for dynamic Bayesian network structures using a multi-objective particle swarm optimizer, *Artif. Life Rob.*, **16** (2011), 329–332. https://doi.org/10.1007/s10015-011-0943-7
27. K. Zou, Y. Liu, S. Wang, N. Li, Y. Wu, A multiobjective particle swarm optimization algorithm based on grid technique and multistrategy, *J. Math.*, **2021** (2021), 1626457. https://doi.org/10.1155/2021/1626457
28. S. Z. Martínez, C. A. C. Coello, A multi-objective particle swarm optimizer based on decomposition, in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, (2011), 69–76. https://doi.org/10.1145/2001576.2001587

29. Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, et al., Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems, *IEEE Trans. Evol. Comput.*, **22** (2018), 32–46. https://doi.org/10.1109/TEVC.2016.2631279
30. A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. C. Coello, F. Luna, E. Alba, SMPSO: a new PSO-based metaheuristic for multi-objective optimization, in *2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM)*, (2009), 66–73. https://doi.org/10.1109/MCDM.2009.4938830
31. C. Dai, Y. Wang, M. Ye, A new multi-objective particle swarm optimization algorithm based on decomposition, *Inf. Sci.*, **325** (2015), 541–557. https://doi.org/10.1016/j.ins.2015.07.018
32. K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints, *IEEE Trans. Evol. Comput.*, **18** (2013), 577–601. https://doi.org/10.1109/TEVC.2013.2281535
33. Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, *IEEE Trans. Evol. Comput.*, **11** (2007), 712–731. https://doi.org/10.1109/TEVC.2007.892759
34. Y. Tian, X. Zhang, R. Cheng, Y. Jin, A multi-objective evolutionary algorithm based on an enhanced inverted generational distance metric, in *2016 IEEE Congress on Evolutionary Computation (CEC)*, (2016), 5222–5229. https://doi.org/10.1109/CEC.2016.7748352
35. S. Jiang, S. Yang, A strength Pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization, *IEEE Trans. Evol. Comput.*, **21** (2017), 329–346. https://doi.org/10.1109/TEVC.2016.2592479
36. Y. Xiang, Y. Zhou, M. Li, Z. Chen, A vector angle-based evolutionary algorithm for unconstrained many-objective optimization, *IEEE Trans. Evol. Comput.*, **21** (2016), 131–152. https://doi.org/10.1109/TEVC.2016.2587808
37. E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, *Evol. Comput.*, **8** (2000), 173–195. https://doi.org/10.1162/106365600568202
38. K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable multi-objective optimization test problems, in *Proceedings of the 2002 Congress on Evolutionary Computation*, **1** (2002), 825–830. https://doi.org/10.1109/CEC.2002.1007032
39. Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: a regularity model-based multiobjective estimation of distribution algorithm, *IEEE Trans. Evol. Comput.*, **12** (2008), 41–63. https://doi.org/10.1109/TEVC.2007.894202
40. C. A. C. Coello, N. C. Cortés, Solving multiobjective optimization problems using an artificial immune system, *Genet. Program. Evolvable Mach.*, **6** (2005), 163–190. https://doi.org/10.1007/s10710-005-6164-x
41. L. While, P. Hingston, L. Barone, S. Huband, A faster algorithm for calculating hypervolume, *IEEE Trans. Evol. Comput.*, **10** (2006), 29–38. https://doi.org/10.1109/TEVC.2005.851275
42. Y. Tian, R. Cheng, X. Zhang, Y. Jin, PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [Educational Forum], *IEEE Comput. Intell. Mag.*, **12** (2017), 73–87. https://doi.org/10.1109/MCI.2017.2742868
43. M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, *J. Am. Stat. Assoc.*, **32** (1937), 675–701. https://doi.org/10.2307/2279372

44. Y. Cui, X. Meng, J. Qiao, A multi-objective particle swarm optimization algorithm based on two-archive mechanism, *Appl. Soft Comput.*, **119** (2022), 108532. https://doi.org/10.1016/j.asoc.2022.108532

AIMS Press