

Quantum-behaved particle swarm optimization with dynamic grouping searching strategy

You, Q., Sun, J., Palade, V. & Pan, F

Author post-print (accepted) deposited by Coventry University's Repository

Original citation & hyperlink:

You, Q, Sun, J, Palade, V & Pan, F 2023, 'Quantum-behaved particle swarm optimization with dynamic grouping searching strategy', *Intelligent Data Analysis*, vol. 27, no. 3, pp. 769-789. <https://doi.org/10.3233/ida-226753>

DOI 10.3233/ida-226753

ISSN 1088-467X

ESSN 1571-4128

Publisher: IOS Press

Copyright © and Moral Rights are retained by the author(s) and/ or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This item cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

This document is the author's post-print version, incorporating any revisions agreed during the peer-review process. Some differences between the published version and this version may remain and you are advised to consult the published version if you wish to cite from it.

Quantum-behaved Particle Swarm Optimization with Dynamic Grouping Searching Strategy

Qi You^a, Jun Sun^{a,*}, Vasile Palade^b and Feng Pan^c

^a Jiangsu Provincial Engineering Laboratory of Pattern Recognition and Computational Intelligence, Wuxi, China

^b Centre for Computational Science and Mathematical Modeling, Coventry University, Coventry, UK

^c Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi, China

Abstract. The quantum-behaved particle swarm optimization (QPSO) algorithm, a variant of particle swarm optimization (PSO), has been proven to be an effective tool to solve various of optimization problems. However, like other PSO variants, it often suffers a premature convergence, especially when solving complex optimization problems. Considering this issue, this paper proposes a hybrid QPSO with dynamic grouping searching strategy, named QPSO-DGS. During the search process, the particle swarm is dynamically grouped into two subpopulations, which are assigned to implement the exploration and exploitation search, respectively. In each subpopulation, a comprehensive learning strategy is used for each particle to adjust its personal best position with a certain probability. Besides, a modified opposition-based computation is employed to improve the swarm diversity. The experimental comparison is conducted between the QPSO-DGS and other seven state-of-art PSO variants on the CEC'2013 test suit. The experimental results show that QPSO-DGS has a promising performance in terms of the solution accuracy and the convergence speed on the majority of these test functions, and especially on multimodal problems.

Keywords: Quantum-behaved particle swarm optimization, premature convergence, exploration, exploitation

1. Introduction

In the past decades, many swarm intelligence algorithms have been proposed to solve complex benchmark and real-world optimization problems, such as particle swarm optimization (PSO) [1], artificial bee colony (ABC) [2], [3], [4], ant colony optimization (ACO) [5], grey wolf optimization (GWO) [6], [7], [8], cuckoo search (CS) [9], etc. PSO is one of the most popular algorithms due to its simple implementation and effectiveness, and has been widely applied in many real-world optimization problems, such as flexible job shop scheduling [10], path planning [11], circuit design [12], vehicle routing [13], feature selection [14], [15], [16], data clustering [17] and so on.

In PSO, each individual in the population, called a particle, represents a potential solution to an optimization problem. During the optimization process, particles adjust their flying trajectories by learning from their own experience and other partners' experience which can help them fly towards a better search

*Corresponding author. E-mail: sunjun_wx@hotmail.com.

space and quickly converge to the optimum. However, the algorithm may suffer from the problem of premature convergence which means it gets trapped into the local optimum easily especially when solving complicated problems. This is because that the best experience of the swarm (i.e., the global best) is shared amongst all particles, which can lead the particles to gather around the global best. It would become difficult for the particles to escape from the local optimum if the global best is located near it. On the consideration of this problem, a lot of studies have been done and different variants of PSO have been put forward in the past decades. For example, Shi and Eberhart [18] first introduced the inertia weight in the velocity update equation of PSO in order to balance the exploration behavior of global search and the exploitation behavior of local search. Clerc and Kennedy [19] then added a constriction factor to control the convergence tendency of the particle swarm. Mendes et al. [20] developed a fully informed PSO (FIPS) algorithm in which information from a fully connected neighborhood is used. Later in 2006, Liang et al. [21] proposed a comprehensive learning PSO (CLPSO) for global optimization of multimodal functions, in which the velocity update of a particle is guided by the best information from all other particles. Zhan et al. [22] developed the orthogonal learning strategy to guide particles to fly in better directions by constructing an efficient exemplar. Nasir et al. [23] proposed a dynamic neighborhood learning particle swarm optimizer (DNLPSO), which modified the learning strategy in CLPSO and used it to generate exemplars from a predefined neighborhood, for the purpose of enhancing the explorative nature of the algorithm. In the same year, Li et al. [24] proposed a self-learning particle swarm optimizer, called SLPSO, in which a particle can adaptively adjust its search behavior during the search space. Specifically, there were four different strategies in SLPSO guiding particles to converge to the current global best, exploit the area of a local optimum, jump out of a local optimum, and explore new promising areas. Particles can automatically choose an appropriate learning objective at an appropriate moment during the search process. Besides, there also are some other PSO variants proposed in recent years that have been proven to be effective in solving complex optimization problems[25], [26], [27], [28], [29], [30].

Quantum-behaved particle swarm optimization (QPSO), proposed by Sun et al. in 2004 [31], is a variant of PSO inspired by quantum mechanics and the trajectory analysis of PSO. Each particle in QPSO is assumed to have quantum behavior and is further assumed to be attracted by a quantum delta potential well centered on its local focus. Compared to PSO, QPSO has no velocity vectors for particles to update and needs fewer parameters to adjust, making it easier to implement. Besides, the mean best position employed in this algorithm enables the particles to be more intelligent and cooperative, and thus the global search ability is enhanced. Given these advantages, QPSO has been widely used in many optimization problems [32], [33], [34], [35]. Nevertheless, it still succumbs to the issue of converging too fast and falling into the local optimum when solving complex multimodal problems, just like other PSO variants.

Based on the above analysis, in this paper, we proposed an improved QPSO with dynamic grouping searching strategy, named QPSO-DGS, in order to keep a good trade-off between exploration and exploitation. In this algorithm, the particle swarm is dynamically divided into two subgroups, and each subgroup is assigned to conduct the exploration and exploitation search, respectively. Each particle in each subgroup adjusts its own personal best position using a comprehensive learning method with a certain learning probability. Besides, a modified opposition-based computation is used during the search process to maintain the swarm diversity and thus further enhance the performance of the proposed algorithm.

The rest of this paper is organized as follows. Section 2 gives a brief introduction of PSO and QPSO. Section 3 describes the details of our proposed QPSO-DGS algorithm. The experimental results and

analysis on several benchmark functions are presented in Section 4. Finally, the work is concluded in Section 5.

2. Related work

2.1. Particle swarm optimization

Particle swarm optimization (PSO) is a population-based optimization technique originally developed by Kennedy and Eberhart in 1995 [1], which imitates the swarm behavior of birds' flocking. In PSO, each particle is treated as a potential solution to a given problem, and all particles follow their own experiences and the current optimal particle to fly through the solution space.

In the PSO with N particles, each particle i ($i = 1, \dots, N$) has a position vector $\mathbf{X}_i = (X_{i,1}, \dots, X_{i,D})$ and a velocity vector $\mathbf{V}_i = (V_{i,1}, \dots, V_{i,D})$. D is the dimension of the search space. During each iteration t , the particle i in the swarm is updated according to its personal best (*pbest*) position $\mathbf{P}_i = (P_{i,1}, \dots, P_{i,D})$ and the global best (*gbest*) position $\mathbf{G} = (G_1, \dots, G_D)$ found by the whole swarm. The update strategy is given in Eqs. (1) and (2).

$$V_{i,j}(t+1) = \omega V_{i,j}(t) + c_1 r_1 (P_{i,j}(t) - X_{i,j}(t)) + c_2 r_2 (G_j(t) - X_{i,j}(t)) \quad (1)$$

$$X_{i,j}(t+1) = X_{i,j}(t) + V_{i,j}(t+1) \quad (2)$$

for $i = 1, 2, \dots, N$; $j = 1, 2, \dots, D$, where ω is the inertia weight, c_1, c_2 represent the cognitive learning and the social learning factors, respectively. r_1, r_2 are two random variables uniformly distributed on $(0, 1)$.

2.2. Quantum-behaved particle swarm optimization

Quantum-behaved particle swarm optimization (QPSO) algorithm, a variant of PSO, was inspired by quantum mechanics and the trajectory analysis of PSO [19]. It utilizes a strategy based on a quantum delta potential well model to sample around the previous best points [31].

In the QPSO with N particles, each particle has a position vector $\mathbf{X}_i = (X_{i,1}, \dots, X_{i,D})$ and a personal best (*pbest*) position $\mathbf{P}_i = (P_{i,1}, \dots, P_{i,D})$. D is the dimension of the search space. During the iteration t , the position of particle i in the swarm is updated according to the following equation

$$X_{i,j}(t+1) = q_{i,j}(t) \pm \alpha |C_j(t) - X_{i,j}(t)| \cdot \ln(1/u_{i,j}(t)), \quad (3)$$

for $j = 1, 2, \dots, D$, where $u_{i,j}$ is a random number distributed uniformly on $(0, 1)$, $\mathbf{q}_i = (q_{i,1}, \dots, q_{i,D})$ is the local attractor of particle i , calculated by

$$\mathbf{q}_i = \varphi \cdot \mathbf{P}_i + (1 - \varphi) \cdot \mathbf{G}, \quad (4)$$

φ is a random number distributed on $(0, 1)$ uniformly, $\mathbf{G} = (G_1, \dots, G_D)$ is the global best (*gbest*) position found by the whole particle swarm during the search process. The contraction-expansion coefficient

α was designed to control the convergence speed of the QPSO algorithm. C is the mean of the personal best positions of all the particles, namely, the *mbest* position, and it can be calculated as below.

$$C = \frac{1}{N} \sum_{i=1}^N P_i = \left(\frac{1}{N} \sum_{i=1}^N P_{i,1}, \frac{1}{N} \sum_{i=1}^N P_{i,2}, \dots, \frac{1}{N} \sum_{i=1}^N P_{i,D} \right) \quad (5)$$

Since it was first introduced, some improved versions of QPSO have been reported. For example, in [32], Coelho proposed a variant of QPSO using Gaussian mutation (GQPSO) and applied it to constrained engineering problems. Sun et al. [36] proposed the QPSO with local attractor point subject to a Gaussian probability distribution (GAQPSO). Li et al. [37] presented a cooperative QPSO (CQPSO) using Monte Carlo method. In [38], the generalized space transformation search was employed in QPSO for population initialization and generation jumping. In [39], Bhatia et al. proposed a hybrid QPSO with Cauchy operator and natural selection mechanism (QPSO-CD) from evolutionary computations and claimed its outperformance in context of stability and convergence. Chen et al. [40] proposed an improved Gaussian distribution based QPSO and applied it to engineering shape design problems with multiple constraints.

3. The proposed QPSO algorithm with dynamic grouping searching strategy

In this section, the proposed QPSO algorithm with dynamic grouping searching strategy (QPSO-DGS) is described in detail. During the search process, the swarm is randomly divided into two subpopulations. For these subpopulations, the dynamic grouping searching strategy is introduced in this work for the purpose of maintaining a balance between exploration and exploitation. In addition, a new opposition-based computation is periodically used to improve the swarm diversity and to increase the chance of finding a solution close to the optimal one. Section 3.1 and Section 3.2 present this new opposition-based computation and the dynamic grouping searching strategy in detail, respectively. Then the framework of the proposed QPSO-DGS algorithm is illustrated in Section 3.3.

3.1. A new opposition-based computation

Opposition-based learning (OBL) [41] is a popular concept in computational intelligence, and has been successfully used in various metaheuristics to enhance their performance [42], [43], [44], [45], [46], [47]. Its main idea is to consider an estimate and its corresponding opposite estimate at the same time, so as to achieve a more accurate approximation to the current candidate solution [48]. Some basic definitions of OBL are presented as follows.

Opposite number [41]: Let $x \in [a, b]$ be a real number. The opposite of x is defined by

$$x^* = a + b - x, \quad (6)$$

Opposite point [41]: Let $X = (x_1, x_2, \dots, x_D)$ be a point in a D-dimensional space, where $x_1, x_2, \dots, x_D \in R$ and $x_j \in [a_j, b_j], j = 1, 2, \dots, D$. The opposite point $X^* = (x_1^*, x_2^*, \dots, x_D^*)$ is defined by

$$x_j^* = a_j + b_j - x_j, j = 1, 2, \dots, D, \quad (7)$$

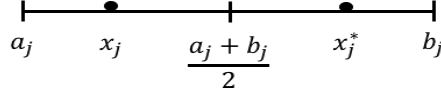


Fig. 1. The OBL concept

In this paper, we made a modification to the computation of the opposite solution in the OBL model, and used it during the search process in order to provide another chance for finding a solution closer to the global best as well as to improve the swarm diversity. This modified computation is stated as

$$x_j^{**} = (a_j + b_j - x_j) * r_1 * \ln(1/r_2), \quad (8)$$

where a_j, b_j are respectively the lower and upper bounds of x_j , which are obtained according to the minimal and the maximal values of the j th dimension of the current population. r_1, r_2 are two random numbers distributed uniformly on $(0, 1)$. To better understand this computation, we make a further description here. Assuming that x_j is the j th dimension of a particle, then its opposite location in the OBL model is x_j^* , as illustrated in Fig.1. It can be seen that the locations of x_j and x_j^* are totally symmetric in $[a_j, b_j]$. After multiplying r_1 and $\ln(1/r_2)$, just as shown in Eq. (8), the location of x_j^{**} would move to the one near or far away from x_j^* . This, to some degree, would increase the randomness of the opposite locations, and would further enhance the possibility of finding a promising candidate solution. Besides, the random number r_1 is used here to adjust the movement scope of x_j^{**} . As we know that the function $y = \ln(1/x)$ is a decreasing function and that the value of y is larger than 1, if $0 < x < 0.37$. This means that there would be a certain number of opposite solutions jumping out of the scope $[a_j, b_j]$ if we only multiply by $\ln(1/r_2)$. Therefore, the utilization of r_1 in Eq. (8) could help the algorithm find more suitable opposite solutions within the scope.

3.2. The dynamic grouping searching strategy

As stated in Section 2, the movement of each particle in the canonical QPSO is influenced by the local attractor and the *mbest* position. The local attractor attracts the particles to itself, and the *mbest* makes particles wait the lagged ones when they converge to the local attractor [31]. In short, the local attractor and the *mbest* position together make particles have strong ability of global search. Besides, according to the Eq. (4), the local attractor q_i of particle i is produced by the mutual interaction between the personal best position P_i and the global best position G . The value of the random number φ_i can be regarded as the weight of the influence that P_i has on the location of q_i . Similarly, the value of $1 - \varphi_i$ represents the weight of the influence that G has on the location of q_i . If the value of φ_i is larger than that of $1 - \varphi_i$, then the location of q_i is decided mainly by the personal best position of particle i . In this case, most of particles in the swarm search in the vicinity of their own personal best positions, bringing relatively great swarm diversity. In contrast, if the value of φ_i is smaller than that of $1 - \varphi_i$, then the location of q_i is decided mainly by the best position found by the whole particle swarm. In this case, most of particles in the swarm move to and further exploit the area of this best position. Therefore, on the basis of the above considerations, we randomly divide the entire particle swarm into two groups with the same size, and reset the local attractor for each group. This procedure is illustrated in Algorithm 1.

Algorithm 1 Computing the local attractors

```

1: // For group 1;
2: for particle  $i$  in group 1 do
3:    $\varphi_{1,i} = 1 - \varphi_i * \eta_1$ ;
4:    $\mathbf{q}_{1,i} = \varphi_{1,i} * \mathbf{P}_i + (1 - \varphi_{1,i}) * \mathbf{P}_{lb1}$ ;
5: end for
6: // For group 2;
7: for particle  $i$  in group 2 do
8:   if  $rand < \delta$  then
9:      $\varphi_{2,i} = \varphi_i * \eta_2$ ;
10:  else
11:     $\varphi_{2,i} = \varphi_i$ ;
12:  end if
13:   $\mathbf{q}_{2,i} = \varphi_{2,i} * \mathbf{P}_i + (1 - \varphi_{2,i}) * \mathbf{G}$ ;
14: end for

```

In Algorithm 1, for group 1, we set the weight of particle i in this group, i.e., $\varphi_{1,i}$, to be a larger value, and use the local best position \mathbf{P}_{lb1} instead of the global best position \mathbf{G} to generate the local attractor $\mathbf{q}_{1,i}$. Thus, the particles in group 1 are explorative and group 1 has good ability of exploration. Unlike group 1, we generate a random number for each particle in group 2. Empirically, if this random number is smaller than a predefined threshold δ , the weight of particle i in this group, i.e., $\varphi_{2,i}$, is set to be a smaller value; otherwise, it remains to be φ_i , as defined in the Eq. (4). Therefore, the particles in group 2 are exploitative and group 2 has good ability of exploitation. Besides, the values of η_1, η_2 and δ in this paper are set to 0.3, 0.3, 0.7 respectively according to the simulation results.

Meanwhile, the comprehensive learning method [21] with a learning probability P_c is adopted here to adjust the personal best positions of particles in each subpopulation. More precisely, we randomly select two particles in each subpopulation, then the one with better fitness value is chosen as the exemplar. The exemplar for each dimension is determined according to the learning probability P_c , and the P_c value for particle i is calculated as

$$P_{c_i} = 0.05 + 0.45 * \frac{\exp(10(i-1)/(N-1)) - 1}{\exp(10) - 1}, \quad (9)$$

where N is the population size. A random number is generated here for each dimension, and if this random number is smaller than the P_{c_i} value, the corresponding dimension of particle i is adjusted according to another particle's $pbest$. Otherwise, the corresponding dimension of particle i remains to be its own $pbest$. Particularly, if all dimensions come from the same particle, we randomly choose one dimension to be modified according to another particle in its subpopulation. In order to ensure the effectiveness of this learning method and to minimize the time wasted on poor directions, we set a refreshing gap m as suggested in [21], that is to say, a new $pbest$ is generated if there is no improvement for m consecutive iteration steps.

After the local attractors of two groups are determined, the position of each particle in both groups is updated according to the Eq. (3). Then the personal best positions of all particles, the local best positions of two groups and the global best position of the whole particle swarm are updated successively.

Furthermore, we define a regrouping gap R as the criterion of dynamic grouping, that is, the whole particle swarm is regrouped into two new subpopulations every R iteration steps. In detail, if the regrouping gap is met, then we employ the modified opposition-based computation stated in Section 3.1 for each particle so as to produce an opposite population. Next, the fittest particles that have the best fitness values are picked as a new population from the original population and the opposite population. This new population is then randomly regrouped into two subpopulations in the subsequent search process. This procedure is listed as shown in Algorithm 2.

Algorithm 2 Regrouping

```

1: // Update the boundaries of each dimension;
2: for  $j = 1 : D$  do
3:    $a_j = \min[X_{i,j}]$ ;
4:    $b_j = \max[X_{i,j}]$ ;
5: end for
6: // Executing the opposition-based computation;
7: for  $i = 1 : N$  do
8:   for  $j = 1 : D$  do
9:      $r_1 = \text{rand}$ ;
10:     $r_2 = \text{rand}$ ;
11:     $X_{i,j}^{**} = (a_j + b_j - X_{i,j}) * r_1 * \ln(1/r_2)$ ;
12:   end for
13:   Repair  $X_i^{**}$  if it is beyond the search scope;
14:   Evaluate  $f(X_i^{**})$ ;
15: end for
16: Sort all the fitness values and select the best  $N$  particles as the new population;
17: Update  $G$ ;
18: //Regrouping the swarm;
19: Divide the swarm into two groups randomly;
20: Update the local best position  $P_{lb1}, P_{lb2}$ ;

```

3.3. QPSO-DGS algorithm

Based on the above description, the procedure of implementing our proposed QPSO-DGS algorithm is given in Algorithm 3. The search process stops when the termination criterion is satisfied.

4. Experimental studies

A variety of experiments are carried out in this section to evaluate the performance of the proposed QPSO-DGS algorithm. We first describe the test problems and then conduct a set of experiments to find out how different values of the regrouping gap R affect the performance of the algorithm. Next, we compare the proposed QPSO-DGS with other variants of PSO in terms of solution accuracy and convergence speed. Besides, we further investigate the effectiveness of the strategies used in QPSO-DGS.

Algorithm 3 QPSO-DGS**Input:** N (the swarm size), $[X_{min}, X_{max}]$ (the search scope), D (dimension)**Output:** Optimal solution

```

1: Initialize:  $X_{i,j} = X_{min} + rand \cdot (X_{max} - X_{min}), i = 1, \dots, N, j = 1, \dots, D;$ 
2:  $P_i = X_i;$ 
3:  $G = P_g$ , where  $g = \arg \min_{1 \leq i \leq N} [f(P_i)];$ 
4: Divide the swarm into two groups randomly;
5: Find the local best positions  $P_{lb1}, P_{lb2};$ 
6: Set  $\alpha$  in Eq. (3);
7: while termination criterion is not fulfilled do
8:   Compute the mbest positions of two groups using Eq. (5);
9:   for  $i = 1 : N$  do
10:    Adjust  $P_i$  if the refreshing gap is met;
11:    Compute the local attractors  $q_{1,i}, q_{2,i}$  according to Algorithm 1;
12:    Update  $X_i$  using Eq. (3);
13:    Update  $P_i, P_{lb1}, P_{lb2};$ 
14:    Update  $G;$ 
15:   end for
16:   if the condition is satisfied then
17:     Regrouping the particle swarm according to Algorithm 2;
18:   end if
19: end while

```

4.1. Test problems

In order to assess the performance of the QPSO-DGS algorithm, we use CEC'2013 test suit in the following experiments, which contains 28 test problems. A summary of this test suit is given in Table 1. As we can see that the CEC'2013 test functions can be divided into three classes according to their properties: unimodal functions ($f_1 - f_5$), basic multimodal functions ($f_6 - f_{20}$) and composition functions ($f_{21} - f_{28}$). All the problems used in this paper are minimization problems. More details about the CEC'2013 functions can be found in [49].

4.2. Parameter sensitive analysis

The value of the regrouping gap R may have influence on the performance of QPSO-DGS. Therefore, we studied the performance of QPSO-DGS under variant R values so as to select a better value for this parameter. In the experiments, five different values of R , i.e., $R = 10, 20, 50, 100, 200$, are tested.

For other parameters in QPSO-DGS, we used the following settings. The population size (N) was set to 40. The number of particles in group 1 and group 2 were 15 and 25, respectively. α linearly decreases from 0.7 to 0.3. The dimension of each test function (D) was 30, then the maximal number of fitness evaluations (Max_FEs) was $D \times 10000$, as suggested in [49]. All the experiments were conducted 30 runs for each test functions.

The mean and standard deviation (Std) values are shown in Table 2 where the best ones for each test function are highlighted in bold background. It can be seen that $R = 50$ obtains the most favorable

Table 1
Summary of the CEC'2013 test suit.

| Functions | Name | Type | $f_i^* = f_i(x^*)$ |
|-------------------------------|---|-------------|--------------------|
| f_1 | Sphere | Unimodal | -1400 |
| f_2 | Rotated High Conditioned Elliptic | | -1300 |
| f_3 | Rotated Bent Cigar | | -1200 |
| f_4 | Rotated Discus | | -1100 |
| f_5 | Different Powers | | -1000 |
| f_6 | Rotated Rosenbrock | Multimodal | -900 |
| f_7 | Rotated Schaffers F7 | | -800 |
| f_8 | Rotated Ackley | | -700 |
| f_9 | Rotated Weierstrass | | -600 |
| f_{10} | Rotated Griewank | | -500 |
| f_{11} | Rastrigin | | -400 |
| f_{12} | Rotated Rastrigin | | -300 |
| f_{13} | Non-Continuous Rotated Rastrigin | | -200 |
| f_{14} | Schwefel | | -100 |
| f_{15} | Rotated Schwefel | | 100 |
| f_{16} | Rotated Katsuura | | 200 |
| f_{17} | Lunacek Bi_Rastrigin | | 300 |
| f_{18} | Rotated Lunacek Bi_Rastrigin | | 400 |
| f_{19} | Expanded Griewank plus Rosenbrock | | 500 |
| f_{20} | Expanded Scaffer's F6 | | 600 |
| f_{21} | Composition Function 1 (n=5, Rotated) | Composition | 700 |
| f_{22} | Composition Function 2 (n=3, Unrotated) | | 800 |
| f_{23} | Composition Function 3 (n=3, Rotated) | | 900 |
| f_{24} | Composition Function 4 (n=3, Rotated) | | 1000 |
| f_{25} | Composition Function 5 (n=3, Rotated) | | 1100 |
| f_{26} | Composition Function 6 (n=5, Rotated) | | 1200 |
| f_{27} | Composition Function 7 (n=5, Rotated) | | 1300 |
| f_{28} | Composition Function 8 (n=5, Rotated) | | 1400 |
| Search Range: $[-100, 100]^D$ | | | |

x^* stands for the global optima. n is the number of functions composed.

performance especially on benchmark functions from f_6 to f_{28} . Therefore, the value of the regrouping gap R is set to 50 in the following experiments.

4.3. Comparative study with PSO variants

In this section, QPSO-DGS is compared with seven other PSO variants, including PSO with constriction factor (PSO-cf) [50], QPSO, comprehensive learning PSO (CLPSO) [21], dynamic neighborhood learning PSO (DNLPSO) [23], enhanced leader PSO (ELPSO) [25], self regulating PSO (SRPSO) [27], and MPSO [28].

The first two algorithms are the canonical PSO and QPSO. In CLPSO, a comprehensive learning strategy is used to improve the performance on complex multimodal problems. DNLPSO is an improved variant of CLPSO which introduces neighborhood based selection of the exemplar for velocity updating to enhance its exploration ability. ELPSO uses a five-staged successive mutation strategy to the swarm

Table 2
Sensitivity of R .

| | R=10 | | R=20 | | R=50 | | R=100 | | R=200 | |
|----------|-----------------|----------|----------|----------|-----------------|-----------------|-----------------|-----------------|----------|----------|
| | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| f_1 | 6.82E-14 | 1.06E-13 | 6.82E-14 | 1.06E-13 | 6.06E-14 | 1.02E-13 | 9.85E-14 | 1.15E-13 | 9.85E-14 | 1.15E-13 |
| f_2 | 3.72E+06 | 1.30E+06 | 3.32E+06 | 1.59E+06 | 2.92E+06 | 1.03E+06 | 3.67E+06 | 1.83E+06 | 3.42E+06 | 1.33E+06 |
| f_3 | 3.80E+07 | 4.09E+07 | 3.95E+07 | 3.36E+07 | 2.17E+07 | 1.91E+07 | 4.27E+07 | 6.25E+07 | 3.35E+07 | 3.63E+07 |
| f_4 | 4.34E+04 | 5.16E+03 | 4.50E+04 | 5.01E+03 | 4.20E+04 | 4.97E+03 | 4.56E+04 | 5.26E+03 | 4.34E+04 | 3.97E+03 |
| f_5 | 1.71E-13 | 5.78E-14 | 1.63E-13 | 5.73E-14 | 1.52E-13 | 5.45E-14 | 1.74E-13 | 6.50E-14 | 1.71E-13 | 5.78E-14 |
| f_6 | 2.92E+01 | 1.86E+01 | 3.38E+01 | 2.09E+01 | 2.85E+01 | 1.94E+01 | 3.47E+01 | 2.34E+01 | 4.06E+01 | 2.87E+01 |
| f_7 | 1.51E+01 | 7.88E+00 | 1.55E+01 | 9.16E+00 | 1.35E+01 | 6.93E+00 | 1.59E+01 | 6.57E+00 | 1.63E+01 | 9.66E+00 |
| f_8 | 2.13E+01 | 7.32E-02 | 2.13E+01 | 7.05E-02 | 2.13E+01 | 5.41E-02 | 2.13E+01 | 7.03E-02 | 2.13E+01 | 6.33E-02 |
| f_9 | 1.46E+01 | 2.45E+00 | 1.39E+01 | 2.30E+00 | 1.33E+01 | 2.58E+00 | 1.43E+01 | 2.30E+00 | 1.35E+01 | 2.33E+00 |
| f_{10} | 3.80E-01 | 1.37E-01 | 4.54E-01 | 2.43E-01 | 3.47E-01 | 1.55E-01 | 4.41E-01 | 2.00E-01 | 3.75E-01 | 2.24E-01 |
| f_{11} | 1.34E+01 | 4.46E+00 | 1.54E+01 | 4.35E+00 | 1.41E+01 | 3.84E+00 | 1.65E+01 | 6.21E+00 | 1.43E+01 | 4.34E+00 |
| f_{12} | 2.41E+01 | 8.09E+00 | 2.23E+01 | 6.77E+00 | 2.34E+01 | 6.09E+00 | 2.48E+01 | 7.94E+00 | 2.42E+01 | 8.98E+00 |
| f_{13} | 5.77E+01 | 1.79E+01 | 6.35E+01 | 1.98E+01 | 5.02E+01 | 1.98E+01 | 5.82E+01 | 1.86E+01 | 6.20E+01 | 2.14E+01 |
| f_{14} | 1.02E+03 | 3.86E+02 | 1.03E+03 | 3.59E+02 | 9.04E+02 | 2.99E+02 | 1.01E+03 | 3.54E+02 | 1.01E+03 | 3.34E+02 |
| f_{15} | 2.72E+03 | 7.47E+02 | 2.98E+03 | 6.55E+02 | 2.79E+03 | 7.08E+02 | 2.95E+03 | 5.75E+02 | 2.75E+03 | 5.68E+02 |
| f_{16} | 5.48E+00 | 7.29E-01 | 5.34E+00 | 1.01E+00 | 5.11E+00 | 1.30E+00 | 5.58E+00 | 9.39E-01 | 5.50E+00 | 7.59E-01 |
| f_{17} | 4.25E+01 | 3.17E+00 | 4.46E+01 | 4.78E+00 | 4.32E+01 | 3.61E+00 | 4.38E+01 | 3.80E+00 | 4.43E+01 | 3.25E+00 |
| f_{18} | 6.22E+01 | 1.25E+01 | 5.93E+01 | 9.61E+00 | 5.87E+01 | 9.64E+00 | 5.87E+01 | 1.20E+01 | 5.98E+01 | 1.03E+01 |
| f_{19} | 3.55E+00 | 6.33E-01 | 3.25E+00 | 5.85E-01 | 3.14E+00 | 6.50E-01 | 3.45E+00 | 4.69E-01 | 3.32E+00 | 5.77E-01 |
| f_{20} | 1.28E+01 | 4.79E-01 | 1.24E+01 | 9.56E-01 | 1.25E+01 | 6.52E-01 | 1.24E+01 | 8.39E-01 | 1.26E+01 | 8.81E-01 |
| f_{21} | 3.25E+02 | 8.01E+01 | 3.12E+02 | 5.81E+01 | 2.94E+02 | 8.55E+01 | 3.20E+02 | 7.70E+01 | 3.01E+02 | 5.92E+01 |
| f_{22} | 7.25E+02 | 2.54E+02 | 6.77E+02 | 2.35E+02 | 6.82E+02 | 2.64E+02 | 6.48E+02 | 2.16E+02 | 6.59E+02 | 2.64E+02 |
| f_{23} | 2.74E+03 | 7.93E+02 | 2.66E+03 | 6.80E+02 | 2.50E+03 | 7.85E+02 | 2.66E+03 | 6.45E+02 | 2.66E+03 | 6.77E+02 |
| f_{24} | 2.34E+02 | 9.67E+00 | 2.36E+02 | 1.12E+01 | 2.33E+02 | 1.04E+01 | 2.34E+02 | 9.93E+00 | 2.36E+02 | 1.12E+01 |
| f_{25} | 2.61E+02 | 6.72E+00 | 2.60E+02 | 7.10E+00 | 2.59E+02 | 5.73E+00 | 2.59E+02 | 6.75E+00 | 2.61E+02 | 6.17E+00 |
| f_{26} | 2.62E+02 | 6.76E+01 | 2.67E+02 | 6.85E+01 | 2.39E+02 | 6.10E+01 | 2.56E+02 | 6.47E+01 | 2.64E+02 | 6.56E+01 |
| f_{27} | 6.52E+02 | 8.18E+01 | 6.63E+02 | 7.35E+01 | 6.50E+02 | 7.65E+01 | 6.55E+02 | 1.01E+02 | 6.71E+02 | 9.44E+01 |
| f_{28} | 3.74E+02 | 2.80E+02 | 3.00E+02 | 0.00E+00 | 2.93E+02 | 3.65E+01 | 3.00E+02 | 0.00E+00 | 3.00E+02 | 0.00E+00 |

leader at each iteration considering the problem of premature convergence. SRPSO, inspired by learning principles found in human cognitive psychology, employs the self-regulating inertia weight to the best particle for stronger exploration and the self-perception of the global search direction to the rest of the particles for stronger exploitation. MPSO is a modified PSO, in which particles can adaptively choose the strategies of position updating according to the corresponding conditions, and thus achieving a better balance between exploration and exploitation.

For the fairness of fairness, all the above algorithms are tested on all 28 test functions and run 30 times independently, with the population size (N) and the dimension of each test problem were set to 40 and 30, respectively. The termination criterion of all algorithms is the maximal number of fitness evaluations (Max_FEs) set as $D \times 10000$. The settings of other parameters for all algorithms are shown in Table 3.

The comparison results of mean and standard deviation (Std) values are listed in Table 4, Table 5 and Table 6, where the best results performed by those algorithms on each benchmark functions are marked with bold font. In these tables, the performance of each algorithm is ranked in terms of both the mean and standard deviation values, and the final rank for each algorithm is given according to its average rank

Table 3
Parameters setting for different algorithms.

| Algorithms | Year | Parameters settings |
|-------------|------|--|
| PSO-cf [50] | 2000 | $\chi = 0.729, c_1 = c_2 = 2.05$ |
| QPSO [31] | 2004 | $\alpha = 1.0 - 0.5$ |
| CLPSO [21] | 2006 | $\omega = 0.9 - 0.4, c = 1.49445, m = 7$ |
| DNLPSO [23] | | $\omega = 0.9 - 0.4, c_1 = c_2 = 1.49445, m = 3, g = 10$ |
| ELPSO [25] | 2015 | $\omega = 0.9 - 0.4, c_1 = c_2 = 2.0$ |
| SRPSO [27] | 2015 | $\omega = 1.05 - 0.5, c_1 = c_2 = 1.49445, \eta = 1, \lambda = 0.5$ |
| MPSO [28] | 2020 | $\omega = 0.9 - 0.4, c_1 = c_2 = 2.0$ |
| QPSO-DGS | | $\alpha = 0.7 - 0.3, \eta_1 = \eta_2 = 0.3, \delta = 0.7, R = 50, m = 7$ |

value over 28 test problems. Besides, in order to evaluate the statistical difference between QPSO-DGS and the other PSO variants, the Wilcoxon signed rank test results with a significance level of 0.05 are presented in Table 7. In this table, the symbol “+” means that QPSO-DGS performs significantly better than the compared algorithms, “ \approx ” means that there is no significant difference between QPSO-DGS and the compared algorithms, and “−” means that the compared algorithms perform significantly better than QPSO-DGS.

As shown in Table 4, on unimodal functions (i.e., functions from f_1 to f_5), we can see that the proposed QPSO-DGS algorithm outperformed its competitors on functions f_1 and f_3 , and that the DNLPSO was the worst on all of these benchmark functions. PSO-cf obtained the best solution on function f_2 , while QPSO and QPSO-DGS were respectively the second best and the third best. According to the statistical result in Table 7, there is no significant difference between the performance of QPSO and QPSO-DGS on function f_2 . MPSO did the very best on function f_4 while QPSO-DGS performed the second worst, just better than DNLPSO. As for function f_5 , most of these algorithms got relatively approximate mean values except algorithms DNLPSO, ELPSO and SRPSO.

On multimodal and composition problems, QPSO-DGS yielded the best mean values on functions $f_7, f_9, f_{12}, f_{13}, f_{15}, f_{18}, f_{20}, f_{23}, f_{24}, f_{27}, f_{28}$, i.e., 11 out of 23 test functions. CLPSO provided the best performance on functions $f_8, f_{11}, f_{14}, f_{17}, f_{19}, f_{21}, f_{22}, f_{26}$, i.e., 8 out of 23 test functions. PSO-cf obtained the best mean values on functions f_6 and f_{10} , while MPSO was the best on function f_{16} . In terms of QPSO-DGS, it got the second best performance on functions $f_{11}, f_{14}, f_{17}, f_{19}, f_{21}, f_{22}, f_{25}$. For function f_6 , the mean value of QPSO was slightly lower than that of QPSO-DGS, while the corresponding statistical result in Table 7 shows that there is no significant difference between QPSO and QPSO-DGS on this benchmark function. For function f_8 , QPSO-DGS got the second worst performance, which is just better than DNLPSO. But it should be noted that the mean values obtained by these test algorithms are all around $2.10\text{E}+01$, and that the difference between the best value and the value of QPSO-DGS is only 0.4. For function f_{10} , QPSO was the second best among all of these compared algorithms, followed by MPSO and QPSO-DGS. Specially, we can see that PSO-cf and QPSO performed better than other PSO variants on both functions f_6 and f_{10} . This may be related to the unique characteristics and landscapes of these functions, and different strategies used in different algorithms may affect their effectiveness of dealing with these two functions. For function f_{16} , which is a non-separable and asymmetrical multimodal problem, having plenty of local optimum, the mean value of QPSO-DGS was larger than most of its competitors except DNLPSO. Although QPSO got the minimal mean value on function f_{25} ,

Table 4: Comparison results of mean and standard deviation (Std) values on CEC'2013 benchmark functions. The best values are highlighted in bold background.

| Functions | Criteria | PSO-cf | QPSO | CLPSO | DNLPSO | ELPSO | SRPSO | MPSO | QPSO-DGS |
|-----------|----------|-----------------|----------|-----------------|----------|----------|----------|-----------------|-----------------|
| f_1 | Mean | 9.47E-13 | 2.35E-13 | 2.05E-13 | 9.96E+04 | 3.07E+03 | 2.27E-09 | 1.74E-13 | 6.06E-14 |
| | Std | 4.70E-13 | 4.15E-14 | 6.94E-14 | 1.64E+04 | 1.92E+03 | 5.33E-09 | 9.78E-14 | 1.02E-13 |
| | Rank | 5 | 4 | 3 | 8 | 7 | 6 | 2 | 1 |
| f_2 | Mean | 1.84E+06 | 2.47E+06 | 1.96E+07 | 2.58E+09 | 2.71E+07 | 2.74E+07 | 4.68E+06 | 2.92E+06 |
| | Std | 1.12E+06 | 1.40E+06 | 4.33E+06 | 8.42E+08 | 2.93E+07 | 2.22E+07 | 8.24E+06 | 1.03E+06 |
| | Rank | 1 | 2 | 5 | 8 | 6 | 7 | 4 | 3 |
| f_3 | Mean | 6.34E+08 | 3.83E+07 | 3.59E+08 | 3.81E+21 | 4.55E+10 | 1.52E+09 | 6.63E+07 | 2.17E+07 |
| | Std | 8.33E+08 | 7.68E+07 | 1.61E+08 | 1.12E+22 | 4.27E+10 | 2.02E+09 | 1.20E+08 | 1.91E+07 |
| | Rank | 5 | 2 | 4 | 8 | 7 | 6 | 3 | 1 |
| f_4 | Mean | 9.72E+02 | 1.58E+03 | 2.46E+04 | 9.35E+06 | 7.82E+03 | 2.30E+04 | 6.20E+01 | 4.20E+04 |
| | Std | 6.91E+02 | 7.47E+02 | 4.85E+03 | 1.71E+07 | 4.98E+03 | 5.52E+03 | 1.93E+02 | 4.97E+03 |
| | Rank | 2 | 3 | 6 | 8 | 4 | 5 | 1 | 7 |
| f_5 | Mean | 8.30E-13 | 2.24E-13 | 3.56E-13 | 7.53E+04 | 1.49E+03 | 1.98E-05 | 1.36E-13 | 1.52E-13 |
| | Std | 2.88E-13 | 4.70E-14 | 4.94E-14 | 2.94E+04 | 1.23E+03 | 2.85E-05 | 4.63E-14 | 5.45E-14 |
| | Rank | 5 | 3 | 4 | 8 | 7 | 6 | 1 | 2 |
| f_6 | Mean | 2.05E+01 | 2.61E+01 | 3.05E+01 | 2.76E+04 | 1.96E+02 | 1.51E+02 | 3.31E+01 | 2.85E+01 |
| | Std | 1.39E+01 | 2.02E+01 | 7.11E+00 | 7.24E+03 | 7.16E+01 | 5.04E+01 | 2.59E+01 | 1.94E+01 |
| | Rank | 1 | 2 | 4 | 8 | 7 | 6 | 5 | 3 |
| f_7 | Mean | 1.58E+02 | 2.59E+01 | 7.74E+01 | 7.89E+07 | 1.29E+02 | 6.38E+01 | 6.70E+01 | 1.35E+01 |
| | Std | 5.80E+01 | 1.79E+01 | 1.03E+01 | 1.15E+08 | 4.90E+01 | 1.96E+01 | 1.67E+01 | 6.93E+00 |
| | Rank | 7 | 2 | 5 | 8 | 6 | 3 | 4 | 1 |
| f_8 | Mean | 2.09E+01 | 2.09E+01 | 2.09E+01 | 2.14E+01 | 2.09E+01 | 2.09E+01 | 2.09E+01 | 2.13E+01 |
| | Std | 6.65E-02 | 4.75E-02 | 4.65E-02 | 6.49E-02 | 5.32E-02 | 5.26E-02 | 4.99E-02 | 5.41E-02 |
| | Rank | 6 | 2 | 1 | 8 | 5 | 4 | 3 | 7 |
| f_9 | Mean | 2.67E+01 | 1.79E+01 | 2.73E+01 | 5.03E+01 | 2.06E+01 | 3.84E+01 | 2.74E+01 | 1.33E+01 |
| | Std | 4.59E+00 | 6.39E+00 | 1.96E+00 | 2.00E+00 | 2.86E+00 | 3.31E+00 | 4.42E+00 | 2.58E+00 |
| | Rank | 4 | 2 | 5 | 8 | 3 | 7 | 6 | 1 |
| f_{10} | Mean | 1.04E-01 | 1.81E-01 | 3.68E+00 | 1.57E+04 | 6.03E+02 | 3.91E+00 | 2.48E-01 | 3.47E-01 |
| | Std | 5.60E-02 | 9.88E-02 | 7.74E-01 | 3.45E+03 | 3.88E+02 | 4.26E+00 | 1.69E-01 | 1.55E-01 |
| | Rank | 1 | 2 | 5 | 8 | 7 | 6 | 3 | 4 |

Table 5: Comparison results of mean and standard deviation (Std) values on CEC'2013 benchmark functions. The best values are highlighted in bold background.

| Functions | Criteria | PSO-cf | QPSO | CLPSO | DNLPSO | ELPSO | SRPSO | MPSO | QPSO-DGS |
|-----------|----------|----------|----------|-----------------|----------|----------|----------|-----------------|-----------------|
| f_{11} | Mean | 7.53E+01 | 1.66E+01 | 6.44E-14 | 1.58E+03 | 7.16E+01 | 3.00E+01 | 3.72E+01 | 1.41E+01 |
| | Std | 2.77E+01 | 4.96E+00 | 1.97E-14 | 3.19E+02 | 3.22E+01 | 8.59E+00 | 1.07E+01 | 3.84E+00 |
| | Rank | 7 | 3 | 1 | 8 | 6 | 4 | 5 | 2 |
| f_{12} | Mean | 1.09E+02 | 1.45E+02 | 1.14E+02 | 1.59E+03 | 1.86E+02 | 1.90E+02 | 1.04E+02 | 2.34E+01 |
| | Std | 3.85E+01 | 4.35E+01 | 1.45E+01 | 2.01E+02 | 4.72E+01 | 5.37E+01 | 2.92E+01 | 6.09E+00 |
| | Rank | 3 | 5 | 4 | 8 | 6 | 7 | 2 | 1 |
| f_{13} | Mean | 2.00E+02 | 1.60E+02 | 1.57E+02 | 1.61E+03 | 2.26E+02 | 2.12E+02 | 1.78E+02 | 5.02E+01 |
| | Std | 3.83E+01 | 3.63E+01 | 1.83E+01 | 2.21E+02 | 3.86E+01 | 3.81E+01 | 3.51E+01 | 1.98E+01 |
| | Rank | 5 | 3 | 2 | 8 | 7 | 6 | 4 | 1 |
| f_{14} | Mean | 2.26E+03 | 5.28E+03 | 2.34E+00 | 1.00E+04 | 2.11E+03 | 2.66E+03 | 1.74E+03 | 9.04E+02 |
| | Std | 4.83E+02 | 1.09E+03 | 1.16E+00 | 4.31E+02 | 5.83E+02 | 1.50E+03 | 6.05E+02 | 2.99E+02 |
| | Rank | 5 | 7 | 1 | 8 | 4 | 6 | 3 | 2 |
| f_{15} | Mean | 4.24E+03 | 7.26E+03 | 4.29E+03 | 1.02E+04 | 5.92E+03 | 7.32E+03 | 3.76E+03 | 2.79E+03 |
| | Std | 6.19E+02 | 2.42E+02 | 3.00E+02 | 4.79E+02 | 1.28E+03 | 3.54E+02 | 6.35E+02 | 7.08E+02 |
| | Rank | 3 | 6 | 4 | 8 | 5 | 7 | 2 | 1 |
| f_{16} | Mean | 1.81E+00 | 2.43E+00 | 1.55E+00 | 7.29E+00 | 2.01E+00 | 2.47E+00 | 1.54E+00 | 5.11E+00 |
| | Std | 5.30E-01 | 2.62E-01 | 2.43E-01 | 1.26E+00 | 3.32E-01 | 2.64E-01 | 5.11E-01 | 1.30E+00 |
| | Rank | 3 | 5 | 2 | 8 | 4 | 6 | 1 | 7 |
| f_{17} | Mean | 1.05E+02 | 1.55E+02 | 3.18E+01 | 2.81E+03 | 9.97E+01 | 1.15E+02 | 7.70E+01 | 4.32E+01 |
| | Std | 2.53E+01 | 2.40E+01 | 4.31E-01 | 3.21E+02 | 5.42E+01 | 2.98E+01 | 1.23E+01 | 3.61E+00 |
| | Rank | 5 | 7 | 1 | 8 | 4 | 6 | 3 | 2 |
| f_{18} | Mean | 1.40E+02 | 2.07E+02 | 1.88E+02 | 2.90E+03 | 2.63E+02 | 2.52E+02 | 1.08E+02 | 5.87E+01 |
| | Std | 3.85E+01 | 1.08E+01 | 1.30E+01 | 2.92E+02 | 4.01E+01 | 1.77E+01 | 2.30E+01 | 9.64E+00 |
| | Rank | 3 | 5 | 4 | 8 | 7 | 6 | 2 | 1 |
| f_{19} | Mean | 6.48E+00 | 7.76E+00 | 6.37E-01 | 1.39E+07 | 2.40E+03 | 1.15E+01 | 3.95E+00 | 3.14E+00 |
| | Std | 1.76E+00 | 3.46E+00 | 2.25E-01 | 7.57E+06 | 5.03E+03 | 5.74E+00 | 1.58E+00 | 6.50E-01 |
| | Rank | 4 | 5 | 1 | 8 | 7 | 6 | 3 | 2 |
| f_{20} | Mean | 1.47E+01 | 1.40E+01 | 1.38E+01 | 1.50E+01 | 1.43E+01 | 1.30E+01 | 1.31E+01 | 1.25E+01 |
| | Std | 7.16E-01 | 1.40E+00 | 3.62E-01 | 9.59E-06 | 7.65E-01 | 4.60E-01 | 7.27E-01 | 6.52E-01 |
| | Rank | 7 | 5 | 4 | 8 | 6 | 2 | 3 | 1 |

Table 6: Comparison results of mean and standard deviation (Std) values on CEC'2013 benchmark functions. The best values are highlighted in bold background.

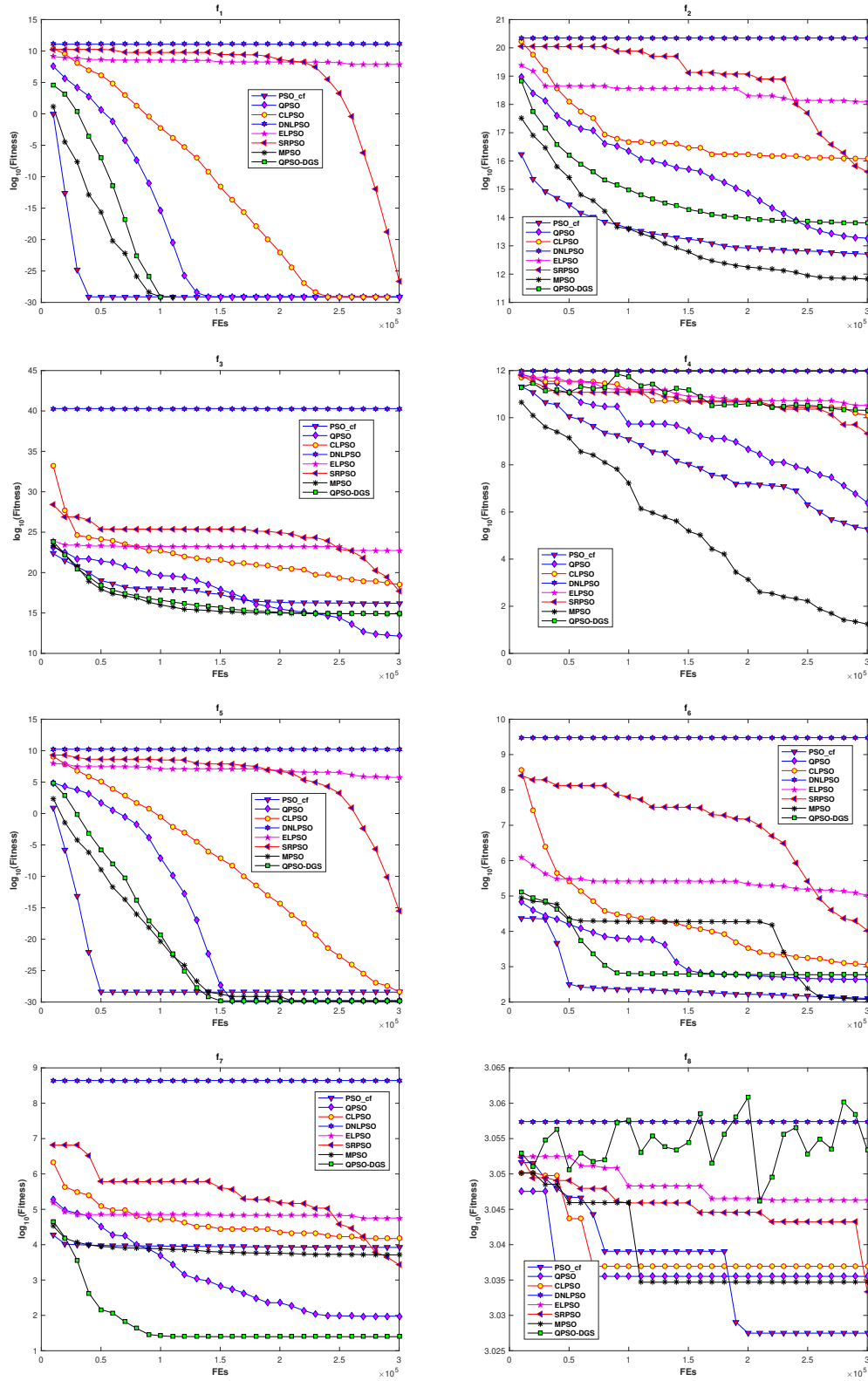
| Functions | Criteria | PSO-cf | QPSO | CLPSO | DNLPSO | ELPSO | SRPSO | MPSO | QPSO-DGS |
|---------------------|----------|----------|-----------------|-----------------|----------|----------|----------|----------|-----------------|
| f_{21} | Mean | 3.07E+02 | 3.13E+02 | 2.83E+02 | 6.53E+03 | 6.01E+02 | 3.01E+02 | 4.53E+02 | 2.94E+02 |
| | Std | 9.15E+01 | 9.65E+01 | 3.25E+01 | 8.38E+02 | 1.88E+02 | 5.92E+01 | 3.50E+02 | 8.55E+01 |
| | Rank | 4 | 5 | 1 | 8 | 7 | 3 | 6 | 2 |
| f_{22} | Mean | 2.24E+03 | 3.83E+03 | 1.09E+02 | 1.06E+04 | 2.40E+03 | 1.93E+03 | 1.56E+03 | 6.82E+02 |
| | Std | 5.54E+02 | 1.74E+03 | 2.64E+01 | 4.39E+02 | 8.29E+02 | 1.54E+03 | 4.99E+02 | 2.64E+02 |
| | Rank | 5 | 7 | 1 | 8 | 6 | 4 | 3 | 2 |
| f_{23} | Mean | 4.49E+03 | 7.10E+03 | 5.21E+03 | 1.05E+04 | 6.12E+03 | 7.81E+03 | 4.59E+03 | 2.50E+03 |
| | Std | 7.09E+02 | 3.24E+02 | 4.64E+02 | 4.00E+02 | 1.11E+03 | 3.13E+02 | 9.48E+02 | 7.85E+02 |
| | Rank | 2 | 6 | 4 | 8 | 5 | 7 | 3 | 1 |
| f_{24} | Mean | 2.66E+02 | 2.44E+02 | 2.74E+02 | 6.34E+02 | 2.80E+02 | 2.50E+02 | 2.82E+02 | 2.33E+02 |
| | Std | 9.58E+00 | 8.91E+00 | 7.48E+00 | 1.30E+02 | 9.03E+00 | 1.48E+01 | 1.25E+01 | 1.04E+01 |
| | Rank | 4 | 2 | 5 | 8 | 6 | 3 | 7 | 1 |
| f_{25} | Mean | 2.68E+02 | 2.58E+02 | 2.95E+02 | 4.76E+02 | 3.02E+02 | 3.32E+02 | 3.12E+02 | 2.59E+02 |
| | Std | 8.91E+00 | 8.34E+00 | 4.42E+00 | 4.23E+01 | 1.41E+01 | 2.40E+01 | 1.52E+01 | 5.73E+00 |
| | Rank | 3 | 1 | 4 | 8 | 5 | 7 | 6 | 2 |
| f_{26} | Mean | 3.48E+02 | 2.65E+02 | 2.01E+02 | 4.56E+02 | 3.02E+02 | 2.02E+02 | 2.98E+02 | 2.39E+02 |
| | Std | 5.09E+01 | 7.11E+01 | 6.22E-01 | 2.48E+01 | 7.49E+01 | 2.06E+00 | 8.24E+01 | 6.10E+01 |
| | Rank | 7 | 4 | 1 | 8 | 6 | 2 | 5 | 3 |
| f_{27} | Mean | 9.73E+02 | 7.11E+02 | 7.69E+02 | 2.10E+03 | 9.37E+02 | 9.86E+02 | 9.06E+02 | 6.50E+02 |
| | Std | 1.14E+02 | 7.83E+01 | 3.17E+02 | 2.20E+02 | 7.83E+01 | 2.93E+02 | 1.04E+02 | 7.65E+01 |
| | Rank | 6 | 2 | 3 | 8 | 5 | 7 | 4 | 1 |
| f_{28} | Mean | 3.82E+02 | 3.00E+02 | 3.00E+02 | 1.06E+04 | 2.03E+03 | 3.00E+02 | 4.27E+02 | 2.93E+02 |
| | Std | 3.13E+02 | 4.88E-13 | 1.11E-03 | 2.18E+03 | 3.63E+02 | 8.99E-03 | 5.15E+02 | 3.65E+01 |
| | Rank | 5 | 2 | 3 | 8 | 7 | 4 | 6 | 1 |
| Average rank | | 4.21 | 3.71 | 3.14 | 8.00 | 5.79 | 5.32 | 3.57 | 2.25 |
| Final rank | | 5 | 4 | 2 | 8 | 7 | 6 | 3 | 1 |
| Best/2nd Best/Worst | | 3/2/0 | 1/10/0 | 8/2/0 | 0/0/28 | 0/0/0 | 0/2/0 | 3/4/0 | 13/8/0 |

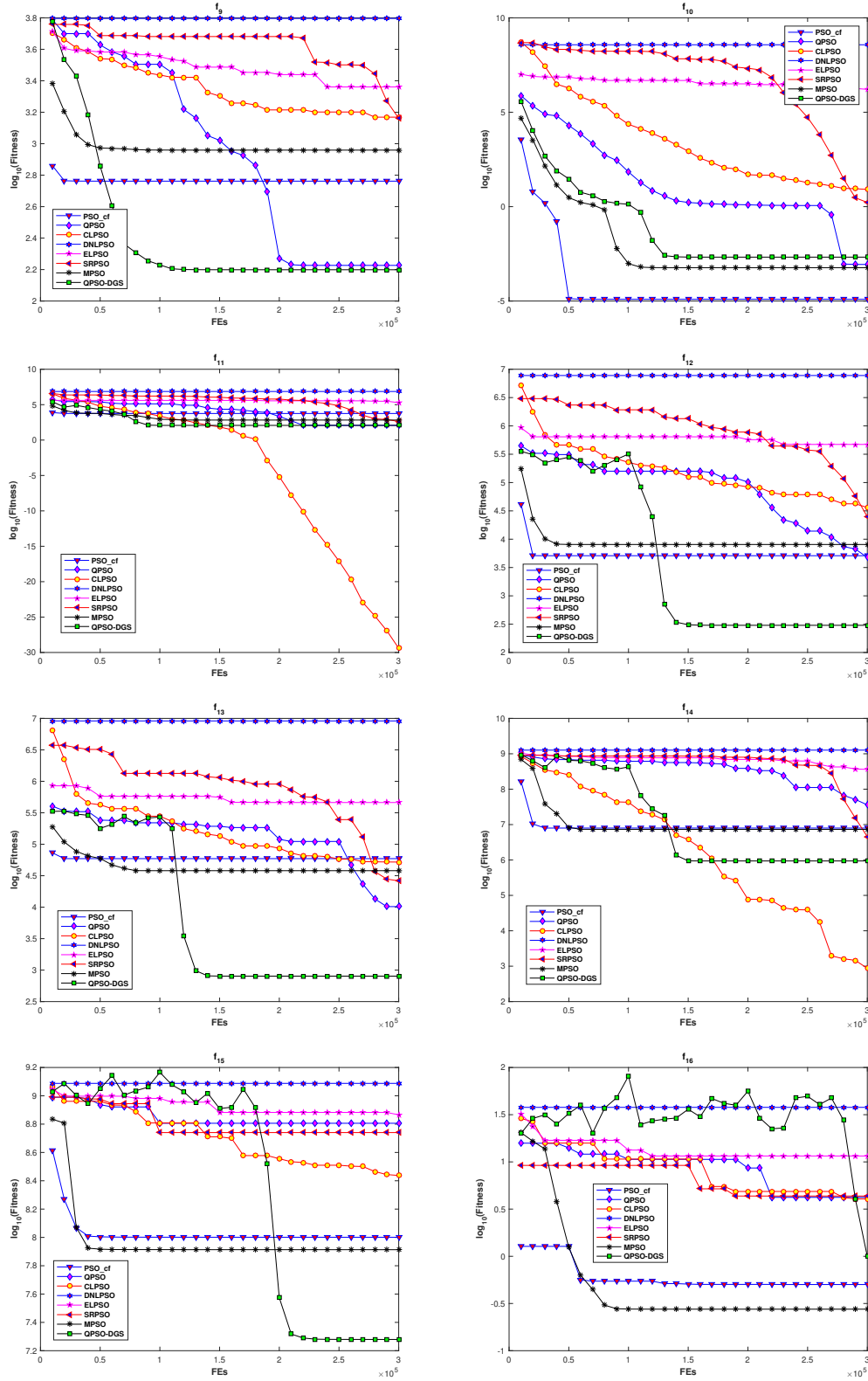
Table 7
Wilcoxon signed rank test results with a significance level of 0.05.

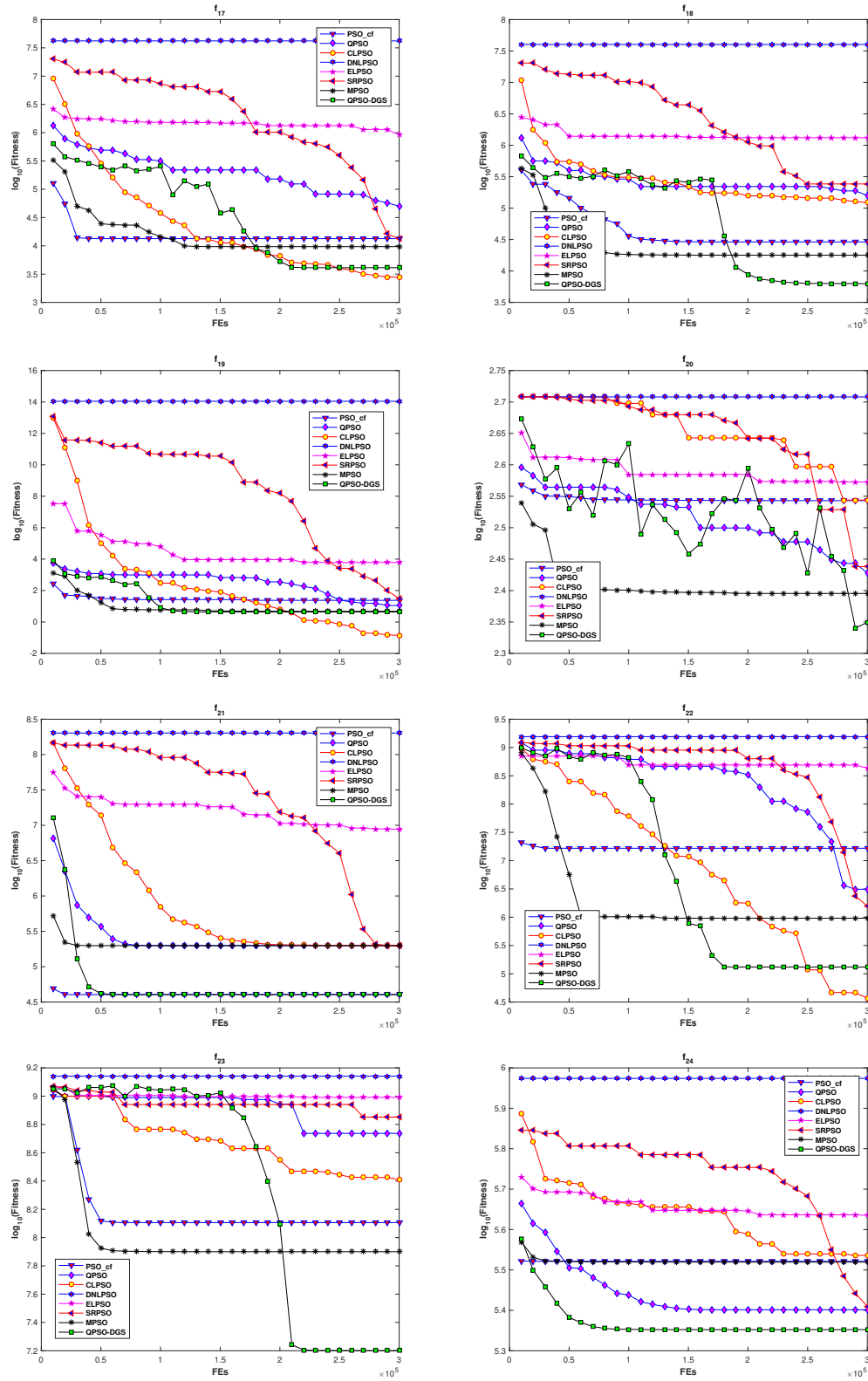
| Functions | QPSO-DGS versus | | | | | | |
|-------------------|-----------------|-----------|-----------|--------|--------|--------|-----------|
| | PSO-cf | QPSO | CLPSO | DNLPSO | ELPSO | SRPSO | MPSO |
| f_1 | + | + | + | + | + | + | + |
| f_2 | — | \approx | + | + | + | + | + |
| f_3 | + | + | + | + | + | + | + |
| f_4 | — | — | — | + | — | — | — |
| f_5 | + | + | + | + | + | + | — |
| f_6 | — | \approx | + | + | + | + | \approx |
| f_7 | + | + | + | + | + | + | + |
| f_8 | — | — | — | + | — | — | — |
| f_9 | + | + | + | + | + | + | + |
| f_{10} | — | — | + | + | + | + | — |
| f_{11} | + | + | — | + | + | + | + |
| f_{12} | + | + | + | + | + | + | + |
| f_{13} | + | + | + | + | + | + | + |
| f_{14} | + | + | — | + | + | + | + |
| f_{15} | + | + | + | + | + | + | + |
| f_{16} | — | — | — | + | — | — | — |
| f_{17} | + | + | — | + | + | + | + |
| f_{18} | + | + | + | + | + | + | + |
| f_{19} | + | + | — | + | + | + | \approx |
| f_{20} | + | + | + | + | + | + | + |
| f_{21} | — | + | — | + | + | + | + |
| f_{22} | + | + | — | + | + | + | + |
| f_{23} | + | + | + | + | + | + | + |
| f_{24} | + | + | + | + | + | + | + |
| f_{25} | + | \approx | + | + | + | + | + |
| f_{26} | + | \approx | — | + | + | — | \approx |
| f_{27} | + | + | \approx | + | + | + | + |
| f_{28} | + | + | + | + | + | + | + |
| + / \approx / — | 21/0/7 | 20/4/4 | 17/1/10 | 28/0/0 | 25/0/3 | 24/0/4 | 20/3/5 |

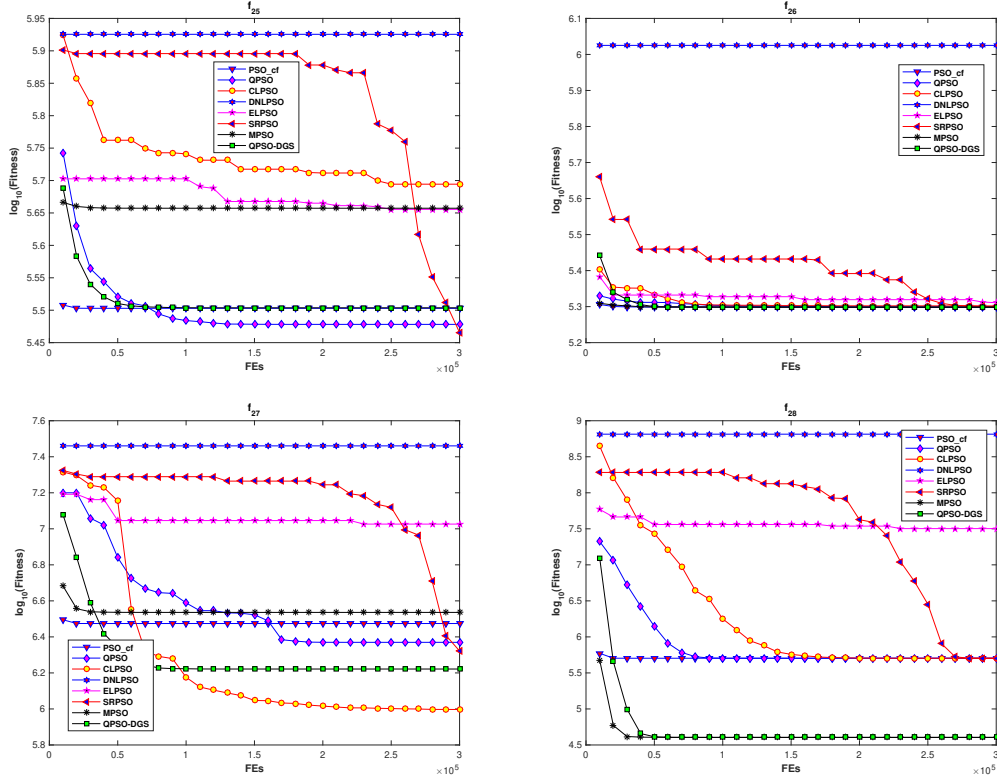
it did not show significant difference between QPSO and QPSO-DGS according to the corresponding results in Table 6 as well as the statistical result in Table 7. For function f_{26} , SRPSO was the second best algorithm among all of these compared ones, followed by QPSO-DGS, QPSO and MPSO. And the statistical results in Table 7 suggests that there is no significant difference among algorithms QPSO-DGS, QPSO and MPSO. Besides, it can be seen that DNLPSO performed the worst on all of these benchmark functions, followed by ELPSO and SRPSO.

The number of “Best/2nd Best/Worst” is counted for each algorithm in the last row of Table 6. We can see that the proposed QPSO-DGS algorithm outperformed the others on 13 out of 28 benchmark functions and obtained the second best mean values on 8 out of 28 benchmark functions. Overall, it ranked the first over the other PSO variants. CLPSO ranked the second, followed by MPSO, while DNLPSO ranked the last. Meanwhile, according to the statistical results in Table 7, we can also conclude that QPSO-DGS is evidently superior to its competitors and that DNLPSO is the worst among all of these

Fig. 2. Convergence graphs of different algorithms on functions $f_1 - f_8$.

Fig. 3. Convergence graphs of different algorithms on functions $f_9 - f_{16}$.

Fig. 4. Convergence graphs of different algorithms on functions $f_{17} - f_{24}$.

Fig. 5. Convergence graphs of different algorithms on functions $f_{25} - f_{28}$.

tested algorithms in solving the CEC'2013 test suit.

Figures from Fig. 2 to Fig. 5 show the convergence curves of the above eight algorithms. It should be noted that the convergence curves in these figures are the result of the minimum rather than the average of 30 independent runs. From these figures, we can see that the proposed QPSO-DGS can converge with a good convergence speed on most of the benchmark functions, except on functions f_4 and f_8 , which is consistent with the results in tables from Table 4 to Table 7. This might be due to the unique characteristics of these test functions and the possibility that the comprehensive learning and dynamic grouping scheme in QPSO-DGS make the swarm lack of exploitation ability in solving these problems, and thus getting trapped into the local optimum. In particular, for some benchmark functions, i.e., functions f_{12} , f_{13} , f_{14} , f_{15} , f_{18} , f_{22} , f_{23} , the convergence curves of QPSO-DGS in Fig. 3 and Fig. 4 can be roughly divided into two segments. Particles in the former half segment have relatively strong diversity and explore the whole search space to approach the global optimal area. Once reaching the vicinity of the global optimum, they converge quickly in the latter half segment. Besides, we can see that the convergence graphs of QPSO-DGS for functions f_{16} and f_{20} in Fig. 3 and Fig. 4 have a similar trend of change. For the convergence graph of function f_{16} in Fig. 3, it is obvious that the fitness value of QPSO-DGS fluctuates within a certain range during nearly the whole search process, and decline sharply to its minimum at the end. For the convergence graph of function f_{20} in Fig. 4, the fitness value of QPSO-DGS fluctuates but tends to decrease with the iteration step increases, and then reaches to a value lower than those of other PSO variants.

Table 8

Mean and Standard deviation (Std) values obtained by QPSO, DGS1, DGS2 and QPSO-DGS. The best values are highlighted in bold background.

| | QPSO | | DGS1 | | DGS2 | | QPSO-DGS | |
|----------|-----------------|----------|-----------------|----------|-----------------|-----------------|-----------------|----------|
| | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| f_1 | 2.20E-13 | 4.15E-14 | 2.05E-13 | 6.94E-14 | 8.94E-13 | 3.82E-13 | 6.06E-14 | 1.02E-13 |
| f_2 | 1.95E+06 | 1.10E+06 | 2.85E+06 | 1.09E+06 | 2.57E+06 | 1.33E+06 | 2.92E+06 | 1.03E+06 |
| f_3 | 3.75E+07 | 3.64E+07 | 1.33E+08 | 1.87E+08 | 6.77E+08 | 9.65E+08 | 2.17E+07 | 1.91E+07 |
| f_4 | 1.63E+03 | 8.03E+02 | 5.18E+04 | 4.00E+03 | 2.95E+03 | 3.05E+03 | 4.20E+04 | 4.97E+03 |
| f_5 | 2.35E-13 | 5.92E-14 | 1.25E-13 | 3.47E-14 | 9.17E-13 | 4.89E-13 | 1.52E-13 | 5.45E-14 |
| f_6 | 2.23E+01 | 1.27E+01 | 3.12E+01 | 2.49E+01 | 4.99E+01 | 2.82E+01 | 2.85E+01 | 1.94E+01 |
| f_7 | 2.38E+01 | 1.29E+01 | 3.12E+01 | 1.45E+01 | 7.71E+01 | 2.52E+01 | 1.35E+01 | 6.93E+00 |
| f_8 | 2.09E+01 | 4.66E-02 | 2.13E+01 | 6.67E-02 | 2.09E+01 | 4.26E-02 | 2.13E+01 | 5.41E-02 |
| f_9 | 2.02E+01 | 6.73E+00 | 1.66E+01 | 3.19E+00 | 1.89E+01 | 2.60E+00 | 1.33E+01 | 2.58E+00 |
| f_{10} | 2.28E-01 | 2.04E-01 | 5.06E-01 | 3.42E-01 | 3.68E-01 | 3.19E-01 | 3.47E-01 | 1.55E-01 |
| f_{11} | 1.85E+01 | 1.44E+01 | 2.52E+01 | 8.00E+00 | 4.51E+01 | 1.38E+01 | 1.41E+01 | 3.84E+00 |
| f_{12} | 1.50E+02 | 3.93E+01 | 4.35E+01 | 1.07E+01 | 8.75E+01 | 2.82E+01 | 2.34E+01 | 6.09E+00 |
| f_{13} | 1.67E+02 | 2.89E+01 | 1.13E+02 | 2.39E+01 | 1.76E+02 | 3.10E+01 | 5.02E+01 | 1.98E+01 |
| f_{14} | 5.15E+03 | 1.32E+03 | 1.70E+03 | 4.05E+02 | 1.38E+03 | 3.26E+02 | 9.04E+02 | 2.99E+02 |
| f_{15} | 7.19E+03 | 2.44E+02 | 3.99E+03 | 6.11E+02 | 4.22E+03 | 1.43E+03 | 2.79E+03 | 7.08E+02 |
| f_{16} | 2.45E+00 | 2.68E-01 | 2.02E+00 | 1.19E+00 | 2.41E+00 | 3.44E-01 | 5.11E+00 | 1.30E+00 |
| f_{17} | 1.55E+02 | 2.49E+01 | 5.46E+01 | 6.74E+00 | 7.39E+01 | 1.37E+01 | 4.32E+01 | 3.61E+00 |
| f_{18} | 2.07E+02 | 1.28E+01 | 8.68E+01 | 1.37E+01 | 1.96E+02 | 3.32E+01 | 5.87E+01 | 9.64E+00 |
| f_{19} | 7.02E+00 | 3.58E+00 | 2.99E+00 | 6.60E-01 | 4.07E+00 | 1.37E+00 | 3.14E+00 | 6.50E-01 |
| f_{20} | 1.41E+01 | 1.41E+00 | 1.42E+01 | 7.15E-01 | 1.46E+01 | 1.03E+00 | 1.25E+01 | 6.52E-01 |
| f_{21} | 2.87E+02 | 8.47E+01 | 3.36E+02 | 9.06E+01 | 3.18E+02 | 8.61E+01 | 2.94E+02 | 8.55E+01 |
| f_{22} | 3.75E+03 | 1.80E+03 | 1.24E+03 | 4.27E+02 | 1.27E+03 | 3.12E+02 | 6.82E+02 | 2.64E+02 |
| f_{23} | 7.19E+03 | 2.90E+02 | 5.36E+03 | 1.15E+03 | 4.19E+03 | 1.66E+03 | 2.50E+03 | 7.85E+02 |
| f_{24} | 2.43E+02 | 7.67E+00 | 2.42E+02 | 1.14E+01 | 2.67E+02 | 1.08E+01 | 2.33E+02 | 1.04E+01 |
| f_{25} | 2.56E+02 | 6.53E+00 | 2.68E+02 | 8.60E+00 | 2.87E+02 | 9.17E+00 | 2.59E+02 | 5.73E+00 |
| f_{26} | 2.64E+02 | 6.99E+01 | 2.88E+02 | 6.86E+01 | 3.32E+02 | 5.30E+01 | 2.39E+02 | 6.10E+01 |
| f_{27} | 6.90E+02 | 9.06E+01 | 7.46E+02 | 8.59E+01 | 8.68E+02 | 9.37E+01 | 6.50E+02 | 7.65E+01 |
| f_{28} | 3.72E+02 | 2.75E+02 | 3.38E+02 | 2.07E+02 | 5.44E+02 | 6.56E+02 | 2.93E+02 | 3.65E+01 |

Above all, our proposed QPSO-DGS algorithm outperforms the other PSO variants in terms of the mean values and the convergence speed on most of the benchmark functions, especially on multimodal and composition ones.

4.4. Further study on DGS

In order to investigate the effect of the dynamic grouping searching strategy employed in the QPSO-DGS algorithm, we carried out a set of experiments to study the performance of QPSO-DGS without comprehensive learning, QPSO-DGS with comprehensive learning only, and QPSO-DGS, and compared these algorithms with QPSO. For the purpose of simplicity, we denote the former two algorithms as DGS1 and DGS2. As for the parameters, we used the same settings as described in Section 4.2. The CEC'2013 test suit was used and each algorithm was conducted 30 runs for each benchmark function.

Table 8 illustrates the mean and standard deviation (Std) values of the above four algorithms, and the best results are highlighted in bold background. It is obvious that QPSO-DGS is the best among all competitors, for it obtained the minimal mean values on most of the benchmark functions, especially on multimodal and composition ones. According to the comparison results between QPSO and DGS2, we can see that DGS2 only outperformed the QPSO on functions $f_8, f_9, f_{12}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}, f_{19}, f_{22}, f_{23}$, i.e., 11 out of 28 benchmark functions, which means that using comprehensive learning only did not have significant effect on improving the performance of QPSO when solving every benchmark function in the CEC'2013 test suit, but did enhance the searching ability of global optimum for the complex ones. However, comparing the results between QPSO and DGS1, we can see that DGS1 performed better than QPSO on functions $f_1, f_5, f_9, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}, f_{19}, f_{22}, f_{23}, f_{24}, f_{28}$, i.e., 15 out of 28 benchmark functions, which means that the other parts in the dynamic grouping searching strategy did have positive influence on enhancing the performance of QPSO. Furthermore, combining comprehensive learning with the dynamic grouping searching strategy can effectively increase the chance of obtaining better fitness values, particularly on multimodal and composition problems, for the mean values obtained by QPSO-DGS were better than those of QPSO on functions $f_1, f_3, f_5, f_7, f_9, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{17}, f_{18}, f_{19}, f_{20}, f_{22}, f_{23}, f_{24}, f_{26}, f_{27}, f_{28}$, i.e., 20 out of 28 test problems.

Overall, we can conclude that the dynamic grouping searching strategy adopted in our proposed QPSO-DGS algorithm could effectively improve the performance of the QPSO algorithm in solving complicated optimization problems.

5. Conclusions

In this paper, we proposed a hybrid QPSO with a dynamic grouping searching strategy, named QPSO-DGS, to solve complex optimization problems. In this algorithm, the particle swarm is dynamically grouped into two subpopulations, and each subpopulation is assigned to conduct the exploration and exploitation search, respectively. The comprehensive learning method with a certain learning probability is adopted for particles in each subpopulation to adjust their personal best positions so as to generate proper exemplars to guide their movement. Besides, a modified opposition-based computation is used periodically during the search process to maintain the swarm diversity.

In order to testify the performance of QPSO-DGS, we carried out various experiments to compare it with other seven state-of-art PSO variants. The experimental results show that QPSO-DGS can get better solutions as well as faster convergence speed on most of the test functions. It also reveals the competitive performance of QPSO-DGS and the ability of dealing with complex optimization problems such as multimodal and composition ones. Furthermore, we investigated the effect that the dynamic grouping searching strategy has on enhancing the performance of our proposed algorithm. The experimental results demonstrated that this strategy can effectively improve the algorithmic performance of QPSO-DGS, particularly when dealing with multimodal and composition problems.

As future work, we will focus on strengthening the performance of QPSO-DGS to make it more effective in solving complex optimization problems. In addition, we will also try to apply QPSO-DGS to real-world optimization problems.

Acknowledgements

This study was funded in part by the National Natural Science Foundation of China (Projects Numbers: 61673194, 61672263, 61672265), and in part by the national first-class discipline program of Light Industry Technology and Engineering (Project Number: LITE2018-25).

References

- [1] J. Kennedy and R.C. Eberhart, Particle swarm optimization, in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.
- [2] D. Karaboga, An Idea Based on Honey BEE Swarm for Numerical Optimization, *Tech. rep., Computer Engineering Department, Erciyes University* (2005).
- [3] X. Zhou, H. Wang, M. Wang and J. Wan, Enhancing the modified artificial bee colony algorithm with neighborhood search, *Soft Comput* **21**(10) (2005), 1–11.
- [4] X. Zhou, Z. Wu, H. Wang and S. Rahnamayan, Gaussian bare-bones artificial bee colony algorithm, *Soft Comput. Fusion Found. Methodol. Appl* **20**(3) (2016), 907–924.
- [5] M. Dorigo, V. Maniezzo and A. Colomi, The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics* **26** (1996), 29–41.
- [6] S. Mirjalili, S. Mirjalili and A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw* **69** (2014), 46–61.
- [7] M. Sulaiman, Z. Mustaffa, M. Mohamed and O. Aliman, Using the gray wolf optimizer for solving optimal reactive power dispatch problem, *Appl. Soft Comput* **32** (2015), 286–292.
- [8] A. Khairuzzaman and S. Chaudhury, Multilevel thresholding using grey wolf optimizer for image segmentation, *Expert Syst. Appl* **86** (2017), 64–76.
- [9] U. Mlakar, I.J. Fister and I. Fister, Hybrid self-adaptive cuckoo search for global optimization, *Swarm Evol. Comput* **29** (2016), 47–72.
- [10] M. Singh, M. Singh, S. Mahapatra and N.e.a. Jagadev, Particle swarm optimization algorithm embedded with maximum deviation theory for solving multi-objective flexible job shop scheduling problem, *Int. J. Adv. Manuf. Technol* **85** (2016), 2353–2366.
- [11] Y. Zhang, L. Wu and S. Wang, Ucarv path planning by fitness-scaling adaptive chaotic particle swarm optimization, *Math. Probl. Eng* (2013), 1–9.
- [12] S. Mallick, R. Kar, S. Ghoshal and D. Mandal, Optimal sizing and design of cmos analogue amplifier circuits using craziness-based particle swarm optimization, *Int. J. Numer. Model. -Electron. Netw. Devices Fields* **29** (2016), 943–966.
- [13] Y. Marinakis, M. Marinaki and A. Migdalas, A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows, *Inform. Sci* **481** (2019), 311–329.
- [14] L. Chen, C. Su and K. Chen, An improved particle swarm optimization for feature selection, *Intelligent Data Analysis* **16** (2012), 167–182.
- [15] C. Qiu and F. Xiang, Feature selection using a set based discrete particle swarm optimization and a novel feature subset evaluation criterion, *Intelligent Data Analysis* **23** (2019), 5–21.
- [16] Q. Wu, Z. Ma, J. Fan, G. Xu and Y. Shen, A feature selection method based on hybrid improved binary quantum particle swarm optimization, *IEEE Access* **7** (2019), 80588–80601.
- [17] D. Tran, Z. Wu and C. Deng, An improved approach of particle swarm optimization and application in data clustering, *Intelligent Data Analysis* **19** (2015), 1049–1070.
- [18] Y. Shi and R. Eberhart, A modified particle swarm optimizer, in *Proceedings of the IEEE Congress on Evolutionary Computation*, 1998, pp. 69–73.
- [19] M. Clerc and J. Kennedy, The particle swarm – explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. Evol. Comput* **6**(1) (2002), 58–73.
- [20] R. Mendes, J. Kennedy and J. Neves, The fully informed particle swarm: Simpler, maybe better, *IEEE Trans. Evol. Comput* **8**(3) (2004), 204–210.
- [21] J. Liang, A. Qin, P. Suganthan and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Trans. Evol. Comput* **10** (2006), 281–295.
- [22] Z. Zhan, J. Zhang, Y. Lin and Y. Shi, Orthogonal learning particle swarm optimization, *IEEE Trans. Evol. Comput* **15**(6) (2011), 832–847.
- [23] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder and P. Suganthan, A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization, *Inform. Sci* **209** (2012), 16–36.

- [24] C. Li, S. Yang and T. Nguyen, A Self-Learning Particle Swarm Optimizer for Global Optimization Problems, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **42**(3) (2012), 627–646.
- [25] A. Jordehi, Enhanced leader PSO (ELPSO): A new PSO variant for solving global optimisation problems, *Appl. Soft Comput* **26** (2015), 401–417.
- [26] N. Lynn and P. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, *Swarm Evol. Comput* **24** (2015), 11–24.
- [27] M. Tanweer, S. Suresh and N. Sundararajan, Self regulating particle swarm optimization algorithm, *Inform. Sci* **294** (2015), 182–202.
- [28] H. Liu, X. Zhang and L. Tu, A modified particle swarm optimization using adaptive strategy, *Expert Syst. Appl* **152** (2020), 113353.
- [29] N. Lynn and P. Suganthan, Ensemble particle swarm optimizer, *Appl. Soft Comput* **55** (2017), 533–548.
- [30] Y. Zhang, X. Liu, F. Bao, J. Chi, C. Zhang and P. Liu, Particle swarm optimization with adaptive learning strategy, *Knowledge-Based Systems* **196** (2020), 105789.
- [31] J. Sun, B. Feng and W. Xu, Particle swarm optimization with particles having quantum behavior, in *IEEE Congress on Evolutionary Computation*, 2004, pp. 325–331.
- [32] L. Coelho, Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems, *Expert Syst. Appl* **37** (2010), 1676–1683.
- [33] A. Tharwat and A. Hassanien, Quantum-behaved particle swarm optimization for parameter optimization of support vector machine, *J Classif* **36** (2019), 576–598.
- [34] B. Du, Q. Wei and R. Liu, An improved quantum-behaved particle swarm optimization for endmember extraction, *IEEE Trans. Geosci. Remote. Sens* **57** (2019), 6003–6017.
- [35] Y. Li, J. Xiao, Y. Chen and L. Jiao, Evolving deep convolutional neural networks by quantum behaved particle swarm optimization with binary encoding for image classification, *Neurocomputing* **362** (2019), 156–165.
- [36] J. Sun, W. Fang, V. Palade, X. Wu and W. Xu, Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point, *Applied Mathematics and Computation* **218** (2011), 3763–3775.
- [37] Y. Li, R. Xiang, L. Jiao and R. Liu, An improved cooperative quantum-behaved particle swarm optimization, *Soft Comput* **16** (2012), 1061–1069.
- [38] Y. Zhang and Z. Jin, Quantum-behaved particle swarm optimization with generalized space transformation search, *Soft Comput* **24** (2020), 14981–14997.
- [39] A. Bhatia, M. Saggi, S. Zheng and S. Nayak, QPSO-CD: Quantum-behaved Particle Swarm Optimization Algorithm with Cauchy Distribution, *Quantum Inf Process* **19** (2020).
- [40] Q. Chen, J. Sun, V. Palade, X. Wu and X. Shi, An improved Gaussian distribution based quantum-behaved particle swarm optimization algorithm for engineering shape design problems, *Engineering Optimization* **54** (2022), 743–769.
- [41] H. Tizhoosh, Opposition-based learning: A new scheme for machine intelligence, in *Proc. Int. Conf. Comput. Intell. Modeling, Control Automat*, 2005, pp. 695–701.
- [42] S. Rahnamayan, H. Tizhoosh and M. Salama, Oppositionbased differential evolution, *IEEE Trans. Evol. Comput* **12**(1) (2008), 64–79.
- [43] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu and M. Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, *Inf. Sci* **181**(20) (2011), 4699–4714.
- [44] M. Elaziz, D. Oliva and S. Xiong, An improved opposition-based sine cosine algorithm for global optimization, *Expert Sys. Appl* **90** (2017), 484–500.
- [45] Y. Cao, S. Ji and Y. Lu, Improved artificial bee colony algorithm with opposition-based learning, *IET Image Process* **14** (2020), 3639–3650.
- [46] X. Yu, W. Xu and C. Li, Opposition-based learning grey wolf optimizer for global optimization, *Knowledge-Based Systems* **226** (2021), 107139.
- [47] T. Choi, J. Togelius and Y. Cheong, A Fast and efficient stochastic opposition-based learning for differential evolution in numerical optimization, *Swarm Evol. Comput* **60** (2021), 100768.
- [48] S. Rahnamayan, J. Jesuthasan, F. Bourennani, H. alehinejad and G. Naterer, Computing opposition by involving entire population, in *2014 IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp. 1800–1807.
- [49] J. Liang, B. Qu, P. Suganthan and A. Hernández-Díaz, Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization, *Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China* (2013).
- [50] R. Eberhart and Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, 2000, pp. 84–88.