sensors

MDPI

Article

Double-Group Particle Swarm Optimization and Its Application in Remote Sensing Image Segmentation

Liang Shen *, Xiaotao Huang and Chongyi Fan

College of Electronic Science, National University of Defense Technology, Changsha 410000, China; xthuang@nudt.edu.cn (X.H.); chongyifan@nudt.edu.cn (C.F.)

* Correspondence: shenliang16@nudt.edu.cn; Tel.: +86-181-0739-5279

Received: 16 March 2018; Accepted: 27 April 2018; Published: 1 May 2018

Abstract: Particle Swarm Optimization (PSO) is a well-known meta-heuristic. It has been widely used in both research and engineering fields. However, the original PSO generally suffers from premature convergence, especially in multimodal problems. In this paper, we propose a double-group PSO (DG-PSO) algorithm to improve the performance. DG-PSO uses a double-group based evolution framework. The individuals are divided into two groups: an advantaged group and a disadvantaged group. The advantaged group works according to the original PSO, while two new strategies are developed for the disadvantaged group. The proposed algorithm is firstly evaluated by comparing it with the other five popular PSO variants and two state-of-the-art meta-heuristics on various benchmark functions. The results demonstrate that DG-PSO shows a remarkable performance in terms of accuracy and stability. Then, we apply DG-PSO to multilevel thresholding for remote sensing image segmentation. The results show that the proposed algorithm outperforms five other popular algorithms in meta-heuristic-based multilevel thresholding, which verifies the effectiveness of the proposed algorithm.

Keywords: particle swarm optimization; multilevel thresholding; remote sensing image segmentation; meta-heuristic; swarm intelligence

1. Introduction

Particle Swarm Optimization (PSO) is an evolutionary optimization algorithm based on swarm intelligence. It is originally proposed by Kennedy and Eberhart in 1995 [1] and is known for its effectiveness and simplicity. It has been proved to be outstanding in solving many complex optimization problems such as power systems [2], neural network training [3], global path planning [4], and feature selection [5].

However, PSO also suffers from two limitations. One is that the original PSO tends to converge to the local optima when applied to complex problems. On the other hand, the convergence speed of the original PSO and most of its variants is slow, especially on high-dimensional problems [6]. Therefore, accelerating the convergence speed and avoiding the local optima convergence have become the two most important and appealing goals in particle swarm optimization studies [7,8]. Specifically, the studies can be classified into three strategies: parameter selection strategy, topology strategy and learning strategy.

The parameter selection refers to the optimization of the inertial weight factor, convergence factor, and the acceleration constant. The inertial weight factor is introduced by Shi and Eberhart to improve the update of velocity [9]. Further studies also show that applying linear decreasing [10], nonlinear [11], exponential [12] and Gaussian [13] strategy to optimize the inertia weight can enhance the overall performance. The convergence factor is proposed by Clerc and Kennedy to enhance the final convergence [14]. In addition, detailed studies [15–17] show that the acceleration constant takes an important role on convergence performance.

The topology strategy is generally employed to improve exploration and avoid premature convergence. In topology strategy, individuals learn from the neighborhood rather than the whole swarm. Therefore, more information would be shared during the search process, which is useful to improve optimization performance. A number of topologies including ring or circle topology, wheel topology, star topology, pyramid topology, Von Neumann topology and random topology are suggested by Kennedy in [18]. Generally, a large neighborhood is good for simple problems, whereas a small neighborhood is helpful for avoiding premature convergence on complex problems [19]. Reference [20] studied the topology extensively, which provides a useful guide of topology selection. It points out that an optimal topology is both problem-specific and computational-budget-dependent and two formulas have been introduced to estimate optimal topology parameters based on numerical experiments.

In the original PSO, all individuals keep learning from the global best solution and their individual best experience in the whole search process. This may lead to premature convergence [21]. To overcome the problem, some novel learning strategies have been developed in recent years. A comprehensive learning strategy is developed to improve the performance on complex multimodal functions in [22]. Reference [23] introduces a cooperative approach to solve high-dimensional optimization problems with multiple swarms. A cooperatively coevolving strategy is proposed in [24] to further improve the performance. Sun et al. introduce a global guaranteed convergence optimizer called quantum behaved particle swarm optimization, which improves the performance by increasing the population diversity [25]. A variant with double learning patterns is developed in [26], which employs the master swarm and the slave swarm with different learning patterns to achieve a trade-off between the convergence speed and the swarm diversity.

However, the three strategies above still face the following shortcomings. In parameter selection, some strategies do improve the overall performance in many cases, but the effect is limited [19], and it is hard to obtain an optimal parameter for all cases. In topology strategy and learning strategy, although the exploration is improved to avoid premature convergence, the convergence speed is reduced at the same time.

In this paper, we design a double-group particle swarm optimization (DG-PSO) to improve the performance. The whole population is divided into two groups: an advantaged group and a disadvantaged group. The modification is focused on the disadvantaged group. A novel learning strategy is developed based on the comprehensive learning strategy and the self-pollination strategy in another popular metaheuristic called Flower Pollination Algorithm (FPA). In addition, a diversity enhancing strategy is also designed to avoid premature convergence. Compared with those published works, the main contribution in this paper is that a novel variant called DG-PSO is proposed which shows remarkable performance compared with five other popular variants and two meta-heuristics. Two new ideas are developed in DG-PSO: a learning strategy, which combines the comprehensive learning strategy [22] and the self-pollination strategy [27], and a diversity enhancing strategy, which adds disturbance to the individuals in the disadvantaged group to avoid premature convergence in multimodal problem. In addition, we also apply the algorithm to multilevel thresholding for image segmentation, which verifies the effectiveness of DG-PSO and provides a good choice of the metaheuristic algorithm to implement multilevel thresholding. The rest of the paper is organized as follows: Section 2 reviews the original PSO and some related works. The strategies and framework of the proposed algorithm are presented in detail in Section 3, followed by the experiments in Section 4. Then, the further application on multilevel thresholding for image segmentation is shown in Section 5.

2. Background

In this section, firstly, we outline the original PSO. Then, two basic works for our algorithm including the comprehensive learning strategy and the self-pollination strategy in FPA are introduced, respectively.

2.1. Particle Swarm Optimization

Similar to other meta-heuristics, PSO is based on swarm intelligence. The swarm is composed of a set of particles $i \in [1, 2, \ldots n]$. A particle moves in the search space with a velocity. The position and velocity of the particle are dynamically adjusted according to its own and its companion's historical experience. Each particle's position is associated with a candidate solution to the problem, and better solutions are obtained via evolution. The performance of a solution is judged by a given fitness function (e.g., smaller fitness function values indicate better solutions for the minimization problem). For a D-dimensional problem, there are four main vectors:

1. The velocity ($v_i = [v_i^1, v_i^2 \ldots v_i^D]$): v_i denotes the moving speed and direction of the particle i.
2. Position ($x_i = [x_i^1, x_i^2 \ldots x_i^D]$): x_i is the current position of particle i. It is updated using its velocity v_i. It can be regarded as a candidate solution.
3. Previous best position ($pbest_i = [pbest_i^1, pbest_i^2 \ldots pbest_i^D]$): $Pbest_i$ represents the historical best position of particle i. It is updated using the best position that particle i has ever found.
4. Global best position ($gbest = [gbest^1, gbest^2 \ldots gbest^D]$): $gbest$ is the best position the swarm has ever found. It is updated with the best $pbest$ in each generation. The final $gbest$ corresponds to the final solution of the whole algorithm.

Note that each of the solutions or candidate solutions represent a set of to-be-optimized parameters, where D is the number of parameters. In the original PSO, the position and velocity are updated by learning from the current global best solution and its previous best solution according to Equations (1) and (2):

$$v_i^d(t+1) = w \times v_i^d(t) + c_1 r_1 \times \left(pbest_i^d - x_i^d(t)\right) + c_2 r_2 \times (gbest^d - x_i^d(t)), \quad (1)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \quad (2)$$

where $i \in [1, 2, \ldots n]$ is the ith particle; c_1, c_2 are called acceleration constants; r_1, r_2 are two uniformly distributed random number within $[0, 1]$; and w is the inertial weight factor.

2.2. Comprehensive Learning Strategy

In the strategy of original PSO, the particles only learn from the global best solution, while its personal best solution may lead to premature convergence. As a consequence, the Comprehensive Learning PSO (CLPSO) is developed to improve the learning strategy [22]. It employs a comprehensive learning strategy, which allows each particle to learn from many particles. Specifically, each dimension of the particle in CLPSO learns from a random particle in the swarm as Equation (3) shows:

$$v_i^d(t+1) = w \times v_i^d(t) + c \times r_i^d \times \left(pbest_{f(i,d)}^d - x_i^d(t)\right), \quad (3)$$

where $f(i, d)$ is the function to define which particle's $pbest$ we should choose (for the ith particle to follow and learn from). Specifically, for each dimension d of particle i, a random number is generated. If the number is larger than a certain threshold, then the corresponding dimension will learn from its own $pbest$. Otherwise, $f(i, d)$ works as follows:

1. Randomly choose two particles out of the whole population excluding the particle whose velocity is updated;
2. Compare the fitness of the two particles' $pbest$ and choose the better one;
3. Use the winner's $pbest$ as the exemplar for the dth dimension of the particle to learn from using Equation (3).

Specially, if all the winners are the $pbest$ of their own ($pbest_i$), it will randomly choose one dimension from the $pbest$ of another particle to learn from. The framework of CLPSO is very similar to

the original PSO, and it has been well tested that CLPSO is effective in optimizing benchmark functions and real-world problems [22,28–31].

2.3. Self-Pollination Strategy in the Flower Pollination Algorithm

Flower pollination algorithm is a popular nature inspired meta-heuristic in [27]. It has been widely used in many fields such as sizing optimization of truss structures [27], economic load dispatch problem in power systems [32], Sudoku Puzzles [33] and feature selection [34] since being published in 2012.

As a swarm-based metaheuristic algorithm, each individual i in the swarm is called a pollen individual. Each pollen individual is associated with a candidate solution $(sol_i = [sol_i^1, sol_i^2 \dots sol_i^D])$ in the search space. FPA searches using the global and local search techniques, where the local search simulates the self-pollination process. The self-pollination strategy is one of the basic ideas in FPA (the other one is cross-pollination). Self-pollination occurs when there are no pollen vectors (Pollen vectors, or called pollinators, can be very diverse. It is estimate there are at least 200,000 variety of pollen vectors such as insects, bats and birds [27]) such as wind or insects or when the pollen individuals are pollinated within the same plant. Such self-pollination behaviors are concluded in the following two rules below:

1. Self-pollination corresponds to the local pollination.
2. Pollinators can develop flower constancy, which is regarded as a reproduction probability that is proportional to the similarity of two flowers involved.

Based on the two rules above, the self-pollination strategy is drawn as Equation (4) shows. Different from PSO, sol_i is the only vector that associates with each pollen individual. sol_i not only represents the position of the pollen individual i, but also plays the role of the best solution this individual has ever found (to understand what is the "*sol*", we can refer to the solutions in PSO such as the position x and the previous best solution *pbest*). It generates the new solutions by using the previous one and two other solutions chosen randomly from the population:

$$sol_i^d = sol_i^d + \varepsilon(sol_{r1}^d - sol_{r2}^d), \tag{4}$$

where sol_{r1} and sol_{r2} are two random solutions in the current generation, which mimics the flower constancy in a limited neighborhood. ε is a uniformly distributed random number within [0,1] used to implement a local random walk. As rule 1 indicates, the self-pollination is considered as local pollination, which often occurs in a limited neighborhood of the particle itself. It can be regarded as the local search around the current position of the pollen individual.

3. The Proposed Algorithm

In this section, we describe the proposed algorithm. Figure 1 shows the overall flowchart, where the process colored by yellow is the core idea of our algorithm. Different from the original PSO, we separate all particles into two groups in DG-PSO: an advantaged group (with the population of $x_1, x_2 \dots x_m$) and a disadvantaged group (with the population of $x_{m+1}, x_{m+2} \dots x_n$, where $n > m$). The advantaged group evolves according to the same theory as the original PSO (Equations (1) and (2)), while the disadvantaged group is updated with two novel strategies: a learning strategy and a diversity enhancing strategy. We focus on the explanation of how the disadvantaged group works. As shown in Figure 2, the two new strategies work as two sequential processing stages in the update of the disadvantaged group, which will be discussed carefully in the following two subsections. In addition, the detailed steps and the whole framework of the proposed method are given in Section 3.3. Finally, we discuss and compare the proposed algorithm with other related works in Section 3.4.

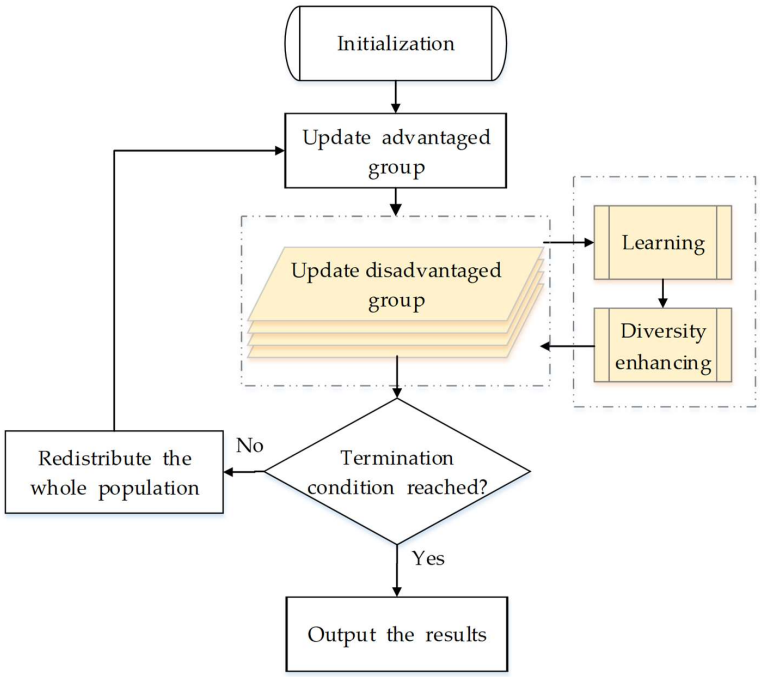


Figure 1. The overall framework of the proposed algorithm.

3.1. The Learning Strategy

The learning strategy is based on the self-pollination strategy introduced in Section 2. We firstly employ the previous best solution *pbest* to be the solution "*sol*" in Equation (4) (rather than the position x, this is because *pbest* represent the best historical experience of each particle, which is more worthy to learn from compared with the position x). Then, it becomes Equation (5) for the particle i:

$$x_i^d = pbest_i^d + \varepsilon \times (pbest_{r1}^d - pbest_{r2}^d), \tag{5}$$

where $i = m+1, \ldots, n$ denotes the particles in the disadvantaged group; $pbest_{r1}$ and $pbest_{r2}$ are two solutions chosen randomly from the *pbest* of the whole population (Specifically, $r1$ and $r2$ are two random integers chosen from sequence $1, 2, \ldots n$ ($r1 \neq r2$). These two parameters keep the same for all dimensions when updating a particle i. In addition, they are regenerated for different particles.). ε represents the scaling factor to perform a random walk satisfying a uniform distributed within $[0, 1]$. Similar to the original self-pollination strategy, Equation (5) can be considered as the local search around the solution (position) $pbest_i$.

On the other hand, as the comprehensive learning strategy generally defines a more suitable solution for the particles to learn from, we additionally replaced the $pbest_i^d$ in Equation (5) with $pbest_{f(i,d)}^d$ given in Equation (6), where $f(i,d) \in [1, 2 \ldots m]$ is the strategy to identify a particle's *pbest* for the dth dimension of particle i to learn from:

$$x_i^d = pbest_{f(i,d)}^d + \varepsilon \times (pbest_{r1}^d - pbest_{r2}^d), \tag{6}$$

$f(i,d)$ works according to the comprehensive learning strategy. For the dth dimension of particle i, the specific procedure to identify the $pbest_{f(i,d)}^d$ is shown as follows:

1. Randomly choose two particles out of the advantaged group;
2. Compare the fitness of the two particles' *pbest* and choose the better one;
3. Use the dth dimension of the winner's *pbest* as the $pbest_{f(i,d)}^d$ for the corresponding dimension of the ith particle to learn from.

Then, a new position is generated using Equation (6) for the particle i in the disadvantaged group to update. Using (6), the particles in the disadvantaged group can learn from the information derived from different particles' historical best position. The strategy is different from the original self-pollination because we perform local search around the new generated position $pbest^d_{f(i,d)}$ rather than the particle itself. The reason is that always searching the area around the position itself may reduce the search efficiency because some particles may be located in the low-promising area. In contrast, making more use of the good information from the advantaged group (using the comprehensive learning strategy) is inductive to the search efficiency.

3.2. The Diversity Enhancing Strategy

PSO often suffers from premature convergence, especially when optimizing the multimodal problem. It is because the original PSO algorithm only employs an attraction phase Equation (1), in which all particles in the swarm move quickly to the same area and the diversity decreases quickly [35]. This generally leads to converging to the local optima due to the loss of diversity [22]. In such case, improving diversity becomes an important issue in PSO research [22,36]. As diversity is lost due to particles getting clustered together [37], adding disturbance to the particles is helpful for them to escape from the local optimal and enhance diversity. Therefore, we developed a strategy to push the particle away from their current position by adding disturbance given in Equation (7):

$$x_{i,d} = x_{i,d} + rand_1 \times s, \tag{7}$$

where s is the scaling factor that controls the intensity of the disturbance. As shown in Equation (8), it is identified using the whole search range of the corresponding dimension (which denotes the strong disturbance) or the Euclidean distance of the two *pbest* chosen in the learning strategy (which denotes a relatively weak disturbance). The strong disturbance is designed for the case that the particle falls $rand_1$ into a large-area local optimum. Therefore, a big jump is needed to escape. The weak disturbance is designed for the case that the particle is close to the global optimum. In such case, a small random walk is more helpful to approaching the optimum.

$$s = \begin{cases} |Ub_d - Lb_d|, & rand_2 < 0.5, \\ \|pbest_{r1} - pbest_{r2}\|, & otherwise, \end{cases} \tag{8}$$

where, $rand_1$ and $rand_2$ are two random number uniformly generated within $[0, 1]$ and Ub and Lb represents the upper and lower bounds of the search space.

Specifically, the strategy works as follows. For each dimension of particle i, we generate a random number within $[0, 1]$. If the number is smaller than the given threshold P, the diversity of the corresponding dimension will be enhanced by adding a random disturbance using (7) and (8). With the disturbance, the particles are more capable to escape from the local optimal and avoid premature convergence.

3.3. The Framework

Algorithm 1 shows the detailed steps of updating the disadvantaged group, which is the core of our modification. Apart from Algorithm 1, another minor modification in the proposed algorithm is that all particles in the two groups should be redistributed according to their fitness at the end of each generation. m particles with better fitness (for minimization problem, "better" means "smaller") are distributed to the advantaged group, whereas others are distributed to the disadvantaged group. The overall framework and the detailed steps are shown in Figure 1 and Algorithm 2, respectively, where *MaxFEs* is the maximum number of function evaluations that represent the maximum computation cost.

Algorithm 1. The Steps for Updating the Disadvantaged Group

For $i = m + 1 : n$
  Randomly choose two $pbest$:$pbest_{r2}$ and $pbest_{r2}$ out of the whole population;
  / Learning stage */*
  For $d = 1 : D$
    Generate two different integers a and b within [1,2 ... m]; [1]
    If $fpbest_a < fpbest_b$ [2]
      $pbest^d_{f(i,d)} = pbest^d_a$;
    Else
      $pbest^d_{f(i,d)} = pbest^d_b$;
    End
    $x^d_i = pbest^d_{f(i,d)} + \varepsilon \times (pbest^d_{r1} - pbest^d_{r2})$;
  End
  / Diversity Enhancing stage */*
  For $j = 1 : D$
    If $rand < p$
      Draw a scaling factor using Equation (8);
      Add disturbance for the current dimension using Equation (7);
    End
  End
End

[1] This step aims to choose two *pbest* from the advantaged group; [2] *fpbest* stands for the fitness value of the *pbest*, which has been recorded before.

Algorithm 2. The Steps of the Proposed Algorithm

Randomly initialize n particles;
m particles with better fitness value for the advantaged group; others for the disadvantaged;
While $fes < MaxFEs$
  For $i = 1 : m$
    Update the particle i in the advantaged group using Equations (1) and (2);
  End
  Evaluate the fitness of the advantaged group;
  Update *pbest* and record the corresponding fitness as *fpbest*.
  Update the disadvantaged group using Algorithm I;
  Evaluate the fitness of the disadvantaged group;
  Update *pbest* and record the corresponding fitness as *fpbest*.
  $fes = fes + n$;
  Redistribute the whole population;
End

3.4. Discussion and Comparison of the Proposed Algorithm with Other Related Works

As mentioned above, we combined the current existing comprehensive learning strategy with the self-pollination strategy in FPA. Specifically, we firstly applied the self-pollination strategy to PSO. Then, the comprehensive learning strategy is used to identify an exemplar for the particles in the disadvantaged group to learn from. Note that we choose the exemplar in the advantaged group rather than in the whole swarm. This strategy aims to improve the learning efficiency of the disadvantaged group. Obviously, such strategy is different from CLPSO (because CLPSO uses the comprehensive learning to modify the learning strategy of the original PSO as introduced in Section 2, whereas we proposed a new learning strategy).

Based on the analysis above, CLPSO, FPA would be used to compare with the proposed one. In addition, since we also developed a diversity enhancing strategy to further improve the performance, it is also necessary to evaluate its effectiveness. We firstly define:

1. dg-PSO: the proposed algorithm that only employs the learning strategy;
2. DG-PSO: the proposed algorithm that employs both the learning strategy and the diversity enhancing strategy.

Then, the effectiveness of the diversity enhancing strategy can be evaluated by comparing the performance of dg-PSO with DG-PSO.

4. Experiments on Benchmark Functions

In this section, we first describe the 20 benchmark functions used for performance evaluation. Then, the algorithms and the necessary parameters for comparison are introduced. Finally, the results are shown and discussed in detail.

4.1. The Benchmark Functions

The 20 benchmark functions employed in the experiments are presented in Table 1. All the functions are the minimization problem, which is defined according to [38,39] in the search space $[-100, 100]$. The functions can be categorized into four classes, namely (1) basic problems; (2) rotated problems; (3) shifted problems; and (4) complex problems. The basic problems include not only the basic unimodal and multimodal problems, but also a noisy problem (F4), an expanded (F8) and an expanded hybrid problem (F9). The rotated problems are designed to overcome the drawback in the basic functions that the variables are separable and the local optima are regularly distributed. In these rotated problems, the original variable x is rotated by left multiplying the orthogonal matrix M, i.e.,$y = M \times x$. Shifted problems are designed to overcome two other problems (in basic functions) including: each dimension value of the global optimum is always the same, and the global optimum is usually located at the centre of the search space. In addition, the complex problems include both rotation and shift.

4.2. Algorithms and Parameters

Table 2 shows the five PSO variants and two other popular meta-heuristics used in the comparison. These algorithms include not only the algorithms we mentioned before (CLPSO, FPA), but also some other state-of-the-art algorithms, which are chosen according to the three strategies introduced in Section 1. We give a brief description of them here. First, Modified PSO (MPSO) [36] uses parameter selection based strategy, of which the population size and inertial weight are adaptively adjusted within the search process. Second, Unified PSO (UPSO) [40] and Fully Informed PSO (FIPS) [41] are two neighbourhood topology strategy based variants. UPSO represents the unified PSO, which is a combination of the original PSO and the topology strategy based PSO. FIPS means the fully informed PSO, which employs the fully informed neighbourhood topology. Finally, Fitness-distance-Ratio PSO (FDR-PSO) [42] and CLPSO [22] are chosen from learning strategy based variants, where FDR-PSO employs a fitness-distance-ratio strategy to identify a "fittest-and-closest" particle to modify the learning strategy. In addition, another novel meta-heuristic called Social Spider Optimization (SSO) [43] is also chosen to give the comparison as comprehensive as possible. In addition, DG-PSO and dg-PSO are the proposed algorithms, where only DG-PSO has diversity enhancing strategy.

The parameters of the involved algorithm are set as follows. For dg-PSO and DG-PSO, the population size of the advantaged group and the disadvantage group are set to 30 and 25 respectively; the possibility p of diversity enhancing is set to $1/D$. The population size for other PSO variants are set to 40 [44], except MPSO, which employs the adaptive population strategy (initial value, minimum and maximum are 5, 5 and 40, respectively) [36]. Other parameters are listed in Table 2. We performed the evaluation in both 30 dimensions with $MaxFEs = 4 \times 10^5$ [45] and

50 dimensions with $MaxFEs = 7 \times 10^5$. Thirty runs are conducted for each function, and the mean fitness error and the corresponding deviation are calculated (the error is defined by the difference between the fitness function value and the minimum, i.e., $Error = Fitness - F_{\min}$). All the experiments are carried out using MATLAB 2016 on the same machine with an Intel I5-4590 CPU @ 3.3 GHz processor (Intel, Santa Clara, CA, USA), 4.00 GB memory, and Windows 7 Professional operating system (Microsoft, Redmond, WA, USA).

4.3. Results and Discussion

The mean fitness error values and the corresponding standard deviation are shown in Tables 3 and 4, respectively, where "Mean" represents the mean fitness error of which the best one in each case is shown in bold; "Std" means the standard deviation. We perform the Wilcoxon Signed Rank test to give a rigorous comparison, in which the significance level is set as 0.05. The results are represented by "C" in the tables, where the three kinds of symbols indicate the performance of DG-PSO: "+" means DG-PSO is relatively better, "=" means insignificant and "-" means DG-PSO is relatively worse. We make a sum of the comparison results and showed the final results in the form of "W/T/L" in the bottom of each table, where "W/T/L" means the number of problems DG-PSO win, tie and lose respectively compared with the corresponding algorithm.

We firstly compare DG-PSO with other published algorithms. According to the statistical results in Table 3 (30D), DG-PSO showed better or close performance in all functions when comparing with CLPSO (W/T/L = 18/2/0) and FPA (W/T/L = 19/1/0). In addition, in the comparison with other PSO variants, FDR-PSO (W/T/L = 18/1/1) seems to be the most competitive one to the proposed algorithm (except dg-PSO), however, it only wins in one case. In addition, DG-PSO even wins in all 20 comparisons compared with the recently published meta-heuristic SSO. Similar results are obtained in 50D where DG-PSO still shows great advantages over all other algorithms. The most competitive algorithm to the proposed algorithm is FDR-PSO (W/T/L = 17/2/1) (except dg-PSO), but, obviously, the results still show the superiority of our algorithm. Then, comparing DG-PSO with dg-PSO, the results are W/T/L = 13/1/6 in both 30D and 50D. Specifically, DG-PSO shows much better performance on multimodal problems such as F3, F5, F6, F8, F9, F14, F15, F16, F19 and F20. However, by comparing DG-PSO with dg-PSO, we can find that the diversity enhancing also brings significant inefficiency to DG-PSO on unimodal problems (F1, F2, F4, F10, F17 and F18). This is mainly because the diversity enhancing strategy weakens the exploitation.

Table 1. Description of the benchmark functions.

No.	Name	Definition	$F_{\min}$	Modality
F1	Schwefel 1.2	$F_1(x) = \sum_{d=1}^{D} \left(\sum_{j=1}^{d} x_j \right)^2$	0	Unimodal
F2	Bent Cigar	$F_2(x) = x_1^2 + 10^6 \cdot \sum_{d=2}^{D} x_d^2$	0	Unimodal
F3	Modified Schwefel	$F_3(x) = 418.9829 \cdot D - \sum_{d=1}^{D} g(z_d), z_d = x_d + 4.209687462275036e^2$ $g(z_d) = \begin{cases} z_d \sin\left(\lvert z_d \rvert^{1/2}\right) \\ (500 - \mathrm{mod}(z_d, 500)) \sin \sqrt{\lvert 500 - \mathrm{mod}(z_d, 500) \rvert} - \frac{(z_d - 500)^2}{10000D} \\ (\mathrm{mod}(\lvert z_d \rvert, 500) - 500) \sin \sqrt{\lvert \mathrm{mod}(\lvert z_d \rvert, 500) - 500 \rvert} - \frac{(z_d + 500)^2}{10000D} \end{cases}$	0	Multimodal
F4	Schwefel 1.2 with Noise	$F_4(x) = F_1 \cdot (1 + 0.4 \cdot \lvert N(0,1) \rvert)$	0	Unimodal
F5	Rosenbrock	$F_5(x) = \sum_{d=1}^{D-1} \left(100 \left(x_d^2 - x_{d+1} \right)^2 + (x_d - 1)^2 \right)$	0	Multimodal
F6	Rastrigin	$F_6(x) = \sum_{d=1}^{D} \left(x_d^2 - 10 \cos(2\pi x_d) + 10 \right)$	0	Multimodal
F7	Katsuura	$F_7(x) = \frac{10}{D^2} \prod_{d=1}^{D} \left(1 + d \sum_{j=1}^{32} \frac{2^j x_d - round\left(2^j x_d\right)}{2^j} \right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^{1.2}}$	0	Multimodal
F8	Expanded Scaffer F6	$F_8(x) = g(x_1, x_2) + g(x_2, x_3) + \ldots + g(x_D, x_1)$ *where* $g(x, y) = 0.5 + \frac{\left(\sin^2\left(\sqrt{x^2 + y^2} \right) - 0.5 \right)}{\left(1 + 0.001 (x^2 + y^2) \right)^2}$	0	Multimodal
F9	Expanded Griewank plus Rosenbrock Function	$F_9(x) = g(F_5(x_1, x_2)) + g(F_5(x_2, x_3)) + \ldots + g(F_5(x_D, x_1))$ *where* $g(y) = \sum_{d=1}^{D} \frac{y_d^2}{4000} - \prod_{d=1}^{D} \cos\left(\frac{y_d}{\sqrt{d}} \right) + 1$	0	Multimodal
F10	Rotated Bent Cigar	$F_{10}(x) = F_2(z), z = M \times x$	0	Unimodal
F11	Rotated Rosenbrock	$F_{11}(x) = F_4(z), z = M \times x$	0	Multimodal
F12	Rotated Expanded Scaffer F6	$F_{12}(x) = F_8(z), z = M \times x$	0	Multimodal
F13	Rotated Expanded Griewank plus Rosenbrock	$F_{13}(x) = F_9(z), z = M \times x$	0	Multimodal

Table 1. *Cont.*

No.	Name	Definition	$F_{\min}$	Modality
F14	Shifted Rastrigin	$F_{14}(x) = F_6(z) + f_{bias1}, z = x - o, f_{bias1} = 800$	800	Multimodal
F15	Shifted Expanded Scaffer F6	$F_{15}(x) = F_8(z) + f_{bias2}, z = x - o, f_{bias2} = 1600$	1600	Multimodal
F16	Shifted Expanded Griewank plus Rosenbrock	$F_{16}(x) = F_9(z) + f_{bias3}, z = x - o, f_{bias3} = 1500$	1500	Multimodal
F17	Shifted Rotated Bent Cigar	$F_{18}(x) = F_2(x) + f_{bias5}, z = M(x - o), f_{bias5} = 200$	200	Unimodal
F18	Shifted Rotated Discus	$F_{17}(x) = g(x) + f_{bias4}, z = M(x - o), f_{bias4} = 200$ $g(y) = 10^6 \cdot y_1^2 + \sum_{d=2}^{D} y_d^2$	300	Unimodal
F19	Shifted Rotated Expanded Scaffer F6	$F_{19}(x) = F_8(x) + f_{bias6}, z = M(x - o), f_{bias6} = 1600$	1600	Multimodal
F20	Shifted Rotated Expanded Griewank plus Rosenbrock	$F_{20}(x) = F_9(x) + f_{bias7}, z = M(x - o), f_{bias7} = 1500$	1500	Multimodal

Table 2. Parameters and references of the involved algorithms.

PSO Variants	Parameters		Reference
FDR-PSO	$w = 0.9 - 0.5 \times g/Max\_iter; c_1 = c_2 = 2.0$		[42]
UPSO	$w = 0.7298; c_1 = c_2 = 1.49445$		[46]
FIPS	$w = 0.7298; c_1 = c_2 = 1.49445$		[41]
CLPSO	$w = 0.9 - 0.5 \times g/Max\_iter; c_1 = c_2 = 2.0$		[22]
MPSO	$w_{\min} = 0.3\ w_{\max} = 0.9; c_1 = c_2 = 2.0;$		[36]
Other State-of-the-Art Meta-Heuristics	**Parameters**		**Reference**
FPA	Population size Switch possibility	$n = 25$ $p = 0.8$	[27]
SSO	Population size The threshold	$n = 50$ $PF = 0.7$	[47]
The Proposed Variants	**Parameters**		**Reference**
dg-PSO	$w = 0.9 - 0.5 \times g/Max\_iter; c_1 = c_2 = 2.0$		Ours (without diversity enhancing)
DG-PSO	$w = 0.9 - 0.5 \times g/Max\_iter; c_1 = c_2 = 2.0$		Ours (with diversity enhancing)

Table 3. Statistical results on 30 dimensions.

No.	Item	FDR	UPSO	FIPS	CLPSO	MPSO	FPA	SSO	dg-PSO	DG-PSO
F1	Mean	3.07×10^{-19}	3.72×10^{-12}	$7.05 \times 10^{+00}$	7.10×10^{-03}	1.12×10^{-02}	3.60×10^{-03}	2.71×10^{-02}	**1.23×10^{-39}**	3.45×10^{-29}
	Std	1.25×10^{-19}	1.94×10^{-12}	$2.47 \times 10^{+00}$	4.96×10^{-03}	5.30×10^{-03}	2.57×10^{-03}	1.41×10^{-02}	2.72×10^{-39}	1.79×10^{-29}
	C	+	+	+	+	+	+	+	-	/
F2	Mean	5.90×10^{-206}	9.59×10^{-176}	4.32×10^{-27}	6.75×10^{-78}	9.84×10^{-15}	3.81×10^{-05}	$8.18 \times 10^{+02}$	**1.28×10^{-226}**	1.68×10^{-126}
	Std	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	1.14×10^{-27}	3.68×10^{-78}	6.75×10^{-15}	3.59×10^{-05}	$1.67 \times 10^{+02}$	$0.00 \times 10^{+00}$	1.88×10^{-126}
	C	-	-	+	+	+	+	+	-	/
F3	Mean	$3.00 \times 10^{+02}$	$2.48 \times 10^{+03}$	$2.58 \times 10^{+03}$	$3.79 \times 10^{+02}$	$2.43 \times 10^{+03}$	$2.49 \times 10^{+03}$	$1.50 \times 10^{+03}$	$1.47 \times 10^{+03}$	**$1.66 \times 10^{+02}$**
	Std	$1.71 \times 10^{+02}$	$2.75 \times 10^{+02}$	$2.28 \times 10^{+02}$	$1.35 \times 10^{+02}$	$1.73 \times 10^{+02}$	$7.24 \times 10^{+01}$	$2.60 \times 10^{+02}$	$7.50 \times 10^{+02}$	$7.96 \times 10^{+01}$
	C	+	+	+	+	+	+	+	+	/
F4	Mean	$2.14 \times 10^{+03}$	$1.12 \times 10^{+03}$	$1.80 \times 10^{+02}$	$3.24 \times 10^{+02}$	$5.00 \times 10^{+03}$	$4.39 \times 10^{+01}$	$4.88 \times 10^{+00}$	**7.73×10^{-03}**	1.37×10^{-01}
	Std	$1.56 \times 10^{+03}$	$4.92 \times 10^{+02}$	$5.49 \times 10^{+01}$	$1.23 \times 10^{+02}$	$2.49 \times 10^{+03}$	$8.42 \times 10^{+00}$	$4.00 \times 10^{+00}$	1.20×10^{-02}	6.15×10^{-02}
	C	+	+	+	+	+	+	+	-	/
F5	Mean	4.01×10^{-04}	$1.49 \times 10^{+00}$	$2.20 \times 10^{+01}$	$2.04 \times 10^{+01}$	$3.19 \times 10^{+01}$	$6.60 \times 10^{+00}$	$1.00 \times 10^{+01}$	$1.05 \times 10^{+00}$	**4.22×10^{-16}**
	Std	1.51×10^{-04}	$1.02 \times 10^{+00}$	2.55×10^{-01}	6.99×10^{-01}	$1.39 \times 10^{+01}$	$4.55 \times 10^{+00}$	8.67×10^{-01}	$1.95 \times 10^{+00}$	2.16×10^{-16}
	C	+	+	+	+	+	+	+	+	/
F6	Mean	$3.04 \times 10^{+01}$	$6.93 \times 10^{+01}$	$6.11 \times 10^{+01}$	$7.76 \times 10^{+00}$	$3.42 \times 10^{+01}$	$3.67 \times 10^{+01}$	$4.31 \times 10^{+01}$	$3.08 \times 10^{+01}$	**$0.00 \times 10^{+00}$**
	Std	$2.65 \times 10^{+00}$	$8.09 \times 10^{+00}$	$4.66 \times 10^{+00}$	$1.66 \times 10^{+00}$	$4.86 \times 10^{+00}$	$3.33 \times 10^{+00}$	$7.74 \times 10^{+00}$	$1.41 \times 10^{+01}$	$0.00 \times 10^{+00}$
	C	+	+	+	+	+	+	+	+	/
F7	Mean	**$0.00 \times 10^{+00}$**	6.70×10^{-02}	$1.89 \times 10^{+00}$	**$0.00 \times 10^{+00}$**	1.96×10^{-01}	**$0.00 \times 10^{+00}$**	8.92×10^{-01}	**$0.00 \times 10^{+00}$**	**$0.00 \times 10^{+00}$**
	Std	$0.00 \times 10^{+00}$	1.39×10^{-02}	1.40×10^{-01}	$0.00 \times 10^{+00}$	1.10×10^{-01}	$0.00 \times 10^{+00}$	1.14×10^{-01}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	C	=	+	+	=	+	=	+	=	/
F8	Mean	$5.34 \times 10^{+00}$	$9.98 \times 10^{+00}$	$9.98 \times 10^{+00}$	$3.36 \times 10^{+00}$	$8.27 \times 10^{+00}$	$1.13 \times 10^{+01}$	$8.64 \times 10^{+00}$	$4.27 \times 10^{+00}$	**3.43×10^{-01}**
	Std	$1.26 \times 10^{+00}$	9.82×10^{-01}	3.99×10^{-01}	$1.11 \times 10^{+00}$	$1.93 \times 10^{+00}$	2.69×10^{-01}	2.43×10^{-01}	$2.84 \times 10^{+00}$	2.39×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F9	Mean	$2.99 \times 10^{+00}$	$6.43 \times 10^{+00}$	$1.15 \times 10^{+01}$	$2.18 \times 10^{+00}$	$3.35 \times 10^{+00}$	$8.53 \times 10^{+00}$	$1.95 \times 10^{+01}$	$5.77 \times 10^{+00}$	**9.93×10^{-01}**
	Std	7.01×10^{-01}	$1.41 \times 10^{+00}$	9.73×10^{-01}	6.75×10^{-01}	8.39×10^{-01}	$2.92 \times 10^{+00}$	$2.29 \times 10^{+00}$	$1.23 \times 10^{+00}$	1.53×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F10	Mean	$2.62 \times 10^{+00}$	$4.77 \times 10^{+02}$	$1.15 \times 10^{+03}$	$6.10 \times 10^{+00}$	$1.98 \times 10^{+01}$	$2.22 \times 10^{+00}$	$1.78 \times 10^{+03}$	**4.30×10^{-26}**	7.30×10^{-09}
	Std	$1.74 \times 10^{+00}$	$3.15 \times 10^{+02}$	$2.59 \times 10^{+02}$	$2.75 \times 10^{+00}$	$5.26 \times 10^{+00}$	5.07×10^{-01}	$9.43 \times 10^{+01}$	4.13×10^{-26}	4.83×10^{-09}
	C	+	+	+	+	+	+	+	-	/
F11	Mean	$1.29 \times 10^{+01}$	$1.98 \times 10^{+01}$	$2.51 \times 10^{+01}$	$5.05 \times 10^{+01}$	$3.22 \times 10^{+01}$	$2.06 \times 10^{+00}$	$2.37 \times 10^{+01}$	$2.15 \times 10^{+01}$	**1.19×10^{-01}**
	Std	$3.64 \times 10^{+00}$	$1.24 \times 10^{+00}$	3.49×10^{-01}	$1.61 \times 10^{+01}$	$1.84 \times 10^{+01}$	$1.06 \times 10^{+00}$	$7.37 \times 10^{+00}$	$1.82 \times 10^{+01}$	6.20×10^{-02}
	C	+	+	+	+	+	+	+	+	/

Table 3. *Cont.*

No.	Item	FDR	UPSO	FIPS	CLPSO	MPSO	FPA	SSO	dg-PSO	DG-PSO
F12	Mean	$5.42 \times 10^{+00}$	$6.47 \times 10^{+00}$	$1.46 \times 10^{+01}$	$5.28 \times 10^{+00}$	$1.20 \times 10^{+01}$	$9.28 \times 10^{+00}$	$2.00 \times 10^{+01}$	$1.10 \times 10^{+01}$	**$2.84 \times 10^{+00}$**
	Std	9.36×10^{-01}	4.13×10^{-01}	5.10×10^{-01}	6.47×10^{-01}	$2.18 \times 10^{+00}$	7.55×10^{-01}	$1.22 \times 10^{+00}$	9.25×10^{-01}	3.91×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F13	Mean	$9.38 \times 10^{+00}$	$1.14 \times 10^{+01}$	$1.13 \times 10^{+01}$	$1.01 \times 10^{+01}$	$1.02 \times 10^{+01}$	$1.21 \times 10^{+01}$	$8.76 \times 10^{+00}$	$1.68 \times 10^{+01}$	**$7.71 \times 10^{+00}$**
	Std	5.11×10^{-01}	1.20×10^{-01}	2.99×10^{-01}	3.96×10^{-01}	7.00×10^{-01}	1.54×10^{-01}	3.22×10^{-01}	$2.22 \times 10^{+00}$	5.43×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F14	Mean	$2.13 \times 10^{+01}$	$8.88 \times 10^{+01}$	$6.23 \times 10^{+01}$	$3.05 \times 10^{+01}$	$1.29 \times 10^{+02}$	$7.82 \times 10^{+01}$	$9.59 \times 10^{+01}$	$7.76 \times 10^{+01}$	**9.09×10^{-14}**
	Std	$2.10 \times 10^{+00}$	$4.66 \times 10^{+00}$	$4.09 \times 10^{+00}$	$4.22 \times 10^{+00}$	$2.74 \times 10^{+01}$	$6.16 \times 10^{+00}$	$1.27 \times 10^{+01}$	$2.36 \times 10^{+01}$	2.54×10^{-14}
	C	+	+	+	+	+	+	+	+	/
F15	Mean	$3.07 \times 10^{+00}$	$7.44 \times 10^{+00}$	$1.23 \times 10^{+01}$	$4.61 \times 10^{+00}$	$1.23 \times 10^{+01}$	$1.37 \times 10^{+01}$	$2.10 \times 10^{+01}$	$7.55 \times 10^{+00}$	**$1.09 \times 10^{+00}$**
	Std	2.63×10^{-01}	6.44×10^{-01}	5.58×10^{-01}	$1.95 \times 10^{+00}$	$8.38 \times 10^{+00}$	$2.09 \times 10^{+00}$	$1.69 \times 10^{+00}$	6.22×10^{-01}	1.21×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F16	Mean	$7.15 \times 10^{+00}$	$1.08 \times 10^{+01}$	$1.04 \times 10^{+01}$	$3.69 \times 10^{+00}$	$1.08 \times 10^{+01}$	$1.20 \times 10^{+01}$	$1.22 \times 10^{+01}$	$3.70 \times 10^{+00}$	**2.91×10^{-01}**
	Std	5.17×10^{-01}	2.38×10^{-01}	1.36×10^{-01}	6.79×10^{-01}	7.00×10^{-01}	9.60×10^{-02}	1.15×10^{-01}	7.70×10^{-01}	2.65×10^{-12}
	C	+	+	+	+	+	+	+	+	/
F17	Mean	$2.89 \times 10^{+08}$	$1.39 \times 10^{+04}$	$1.60 \times 10^{+03}$	$4.79 \times 10^{+08}$	$1.11 \times 10^{+09}$	$6.83 \times 10^{+03}$	$6.98 \times 10^{+07}$	**6.11×10^{-12}**	1.88×10^{-02}
	Std	$1.62 \times 10^{+08}$	$6.35 \times 10^{+03}$	$5.97 \times 10^{+02}$	$2.14 \times 10^{+08}$	$5.35 \times 10^{+08}$	$3.55 \times 10^{+03}$	$1.19 \times 10^{+07}$	6.16×10^{-12}	7.65×10^{-03}
	C	+	+	+	+	+	+	+	-	/
F18	Mean	$6.14 \times 10^{+00}$	$5.65 \times 10^{+02}$	$1.63 \times 10^{+03}$	$1.78 \times 10^{+03}$	$2.08 \times 10^{+04}$	$1.50 \times 10^{+01}$	$2.13 \times 10^{+04}$	**6.82×10^{-13}**	3.55×10^{-07}
	Std	$3.43 \times 10^{+00}$	$4.07 \times 10^{+02}$	$1.50 \times 10^{+02}$	$1.12 \times 10^{+03}$	$1.15 \times 10^{+04}$	$2.81 \times 10^{+00}$	$1.87 \times 10^{+03}$	3.38×10^{-13}	1.75×10^{-07}
	C	+	+	+	+	+	+	+	-	/
F19	Mean	$1.05 \times 10^{+01}$	$1.18 \times 10^{+01}$	$1.18 \times 10^{+01}$	$1.02 \times 10^{+01}$	$1.22 \times 10^{+01}$	$1.21 \times 10^{+01}$	$1.24 \times 10^{+01}$	$1.15 \times 10^{+01}$	**$1.01 \times 10^{+01}$**
	Std	2.05×10^{-01}	2.31×10^{-01}	8.86×10^{-02}	3.08×10^{-01}	5.40×10^{-01}	1.25×10^{-01}	1.04×10^{-01}	3.52×10^{-01}	3.28×10^{-01}
	C	+	+	+	=	+	+	+	+	/
F20	Mean	$7.67 \times 10^{+00}$	$7.22 \times 10^{+00}$	$1.33 \times 10^{+01}$	$5.70 \times 10^{+00}$	$1.56 \times 10^{+01}$	$1.65 \times 10^{+01}$	$2.38 \times 10^{+01}$	$4.77 \times 10^{+00}$	**$4.18 \times 10^{+00}$**
	Std	$1.93 \times 10^{+00}$	$1.10 \times 10^{+00}$	7.55×10^{-01}	$1.01 \times 10^{+00}$	$5.62 \times 10^{+00}$	$2.66 \times 10^{+00}$	$2.29 \times 10^{+00}$	3.01×10^{-01}	4.65×10^{-01}
	C	+	+	+	+	+	+	+	+	/
	W/T/L	**18/1/1**	**19/0/1**	B	**18/2/0**	**20/0/0**	**19/1/0**	**20/0/0**	**13/1/6**	**/**

Note: the gray bacground highlights the best result on each function.

Table 4. Statistical results on 50 dimensions.

No.	Item	FDR	UPSO	FIPS	CLPSO	MPSO	FPA	SSO	dg-PSO	DG-PSO
F1	Mean	6.98×10^{-08}	2.52×10^{-05}	$3.22 \times 10^{+03}$	$1.27 \times 10^{+02}$	$7.10 \times 10^{+00}$	4.24×10^{-01}	1.13×10^{-02}	**7.35×10^{-22}**	6.46×10^{-18}
	Std	3.53×10^{-08}	1.22×10^{-05}	$8.23 \times 10^{+02}$	$5.25 \times 10^{+01}$	$5.24 \times 10^{+00}$	2.32×10^{-01}	6.12×10^{-03}	7.50×10^{-22}	2.54×10^{-18}
	C	+	+	+	+	+	+	+	-	/
F2	Mean	4.28×10^{-164}	5.86×10^{-200}	2.00×10^{-22}	8.77×10^{-74}	6.04×10^{-13}	6.44×10^{-05}	$1.50 \times 10^{+03}$	**6.55×10^{-216}**	1.44×10^{-121}
	Std	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	5.27×10^{-23}	6.05×10^{-74}	2.97×10^{-13}	3.84×10^{-05}	$1.85 \times 10^{+02}$	3.63×10^{-216}	9.20×10^{-122}
	C	-	-	+	+	+	+	+	+	/
F3	Mean	$1.31 \times 10^{+03}$	$5.27 \times 10^{+03}$	$7.77 \times 10^{+03}$	$7.57 \times 10^{+02}$	$4.11 \times 10^{+03}$	$4.81 \times 10^{+03}$	$3.38 \times 10^{+03}$	$3.68 \times 10^{+03}$	**$4.50 \times 10^{+02}$**
	Std	$4.86 \times 10^{+02}$	$4.04 \times 10^{+02}$	$2.29 \times 10^{+02}$	$2.90 \times 10^{+02}$	$6.46 \times 10^{+02}$	$1.55 \times 10^{+02}$	$5.58 \times 10^{+02}$	$1.88 \times 10^{+03}$	$1.21 \times 10^{+02}$
	C	+	+	+	+	+	+	+	+	/
F4	Mean	$5.10 \times 10^{+03}$	$1.66 \times 10^{+04}$	$1.05 \times 10^{+04}$	$1.25 \times 10^{+04}$	$4.37 \times 10^{+02}$	$3.76 \times 10^{+02}$	$5.04 \times 10^{+03}$	**$8.21 \times 10^{+01}$**	$1.45 \times 10^{+02}$
	Std	$3.72 \times 10^{+03}$	$4.01 \times 10^{+03}$	$1.34 \times 10^{+03}$	$5.04 \times 10^{+03}$	$1.74 \times 10^{+02}$	$1.09 \times 10^{+02}$	$2.49 \times 10^{+03}$	$3.57 \times 10^{+01}$	$7.05 \times 10^{+01}$
	C	+	+	+	+	+	+	+	-	/
F5	Mean	$2.49 \times 10^{+01}$	1.92×10^{-01}	$4.22 \times 10^{+01}$	$9.21 \times 10^{+01}$	7.25×10^{-02}	$9.03 \times 10^{+00}$	$3.79 \times 10^{+01}$	$1.85 \times 10^{+00}$	**7.21×10^{-02}**
	Std	$1.27 \times 10^{+01}$	9.25×10^{-02}	1.96×10^{-01}	$2.78 \times 10^{+01}$	4.75×10^{-02}	$5.00 \times 10^{+00}$	$3.11 \times 10^{+00}$	$1.72 \times 10^{+00}$	4.46×10^{-01}
	C	+	+	+	+	=	+	+	+	/
F6	Mean	$6.29 \times 10^{+01}$	$1.31 \times 10^{+02}$	$1.67 \times 10^{+02}$	$1.43 \times 10^{+01}$	$7.02 \times 10^{+01}$	$6.94 \times 10^{+01}$	$6.95 \times 10^{+01}$	$5.94 \times 10^{+01}$	**6.04×10^{-15}**
	Std	$6.33 \times 10^{+00}$	$1.72 \times 10^{+01}$	$9.78 \times 10^{+00}$	5.67×10^{-01}	$1.65 \times 10^{+01}$	$8.82 \times 10^{+00}$	$7.57 \times 10^{+00}$	$1.48 \times 10^{+01}$	7.94×10^{-16}
	C	+	+	+	+	+	+	+	+	/
F7	Mean	**$0.00 \times 10^{+00}$**	1.45×10^{-01}	$2.60 \times 10^{+00}$	**$0.00 \times 10^{+00}$**	3.05×10^{-01}	**$0.00 \times 10^{+00}$**	$1.46 \times 10^{+00}$	**$0.00 \times 10^{+00}$**	**$0.00 \times 10^{+00}$**
	Std	$0.00 \times 10^{+00}$	1.28×10^{-02}	9.52×10^{-02}	$0.00 \times 10^{+00}$	1.75×10^{-01}	$0.00 \times 10^{+00}$	1.25×10^{-01}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	C	=	+	+	=	+	=	+	+	/
F8	Mean	$1.16 \times 10^{+01}$	$2.00 \times 10^{+01}$	$1.96 \times 10^{+01}$	$4.93 \times 10^{+00}$	$1.98 \times 10^{+01}$	$1.91 \times 10^{+01}$	$1.54 \times 10^{+01}$	$1.05 \times 10^{+01}$	**8.74×10^{-01}**
	Std	$2.37 \times 10^{+00}$	4.87×10^{-01}	5.52×10^{-01}	$1.73 \times 10^{+00}$	7.46×10^{-01}	9.88×10^{-01}	5.30×10^{-01}	4.85×10^{-01}	1.76×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F9	Mean	$7.66 \times 10^{+00}$	$1.58 \times 10^{+01}$	$2.83 \times 10^{+01}$	$3.85 \times 10^{+00}$	$1.08 \times 10^{+01}$	$2.21 \times 10^{+01}$	$3.53 \times 10^{+01}$	$6.36 \times 10^{+00}$	**$1.88 \times 10^{+00}$**
	Std	$1.16 \times 10^{+00}$	$3.02 \times 10^{+00}$	$1.05 \times 10^{+00}$	9.01×10^{-01}	$2.79 \times 10^{+00}$	$3.53 \times 10^{+00}$	$2.68 \times 10^{+00}$	9.90×10^{-01}	3.46×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F10	Mean	$1.02 \times 10^{+02}$	$1.46 \times 10^{+03}$	$8.12 \times 10^{+03}$	$4.46 \times 10^{+02}$	$2.81 \times 10^{+02}$	$1.14 \times 10^{+02}$	$3.08 \times 10^{+03}$	**2.69×10^{-07}**	9.16×10^{-05}
	Std	$1.68 \times 10^{+01}$	$4.09 \times 10^{+02}$	$7.63 \times 10^{+02}$	$7.17 \times 10^{+01}$	$1.01 \times 10^{+02}$	$5.14 \times 10^{+01}$	$1.87 \times 10^{+02}$	2.24×10^{-07}	3.04×10^{-05}
	C	+	+	+	+	+	+	+	+	/
F11	Mean	$6.08 \times 10^{+01}$	$5.25 \times 10^{+01}$	$4.49 \times 10^{+01}$	$1.16 \times 10^{+02}$	$5.98 \times 10^{+01}$	4.61×10^{-02}	$5.53 \times 10^{+01}$	$1.70 \times 10^{+01}$	**2.02×10^{-02}**
	Std	$1.35 \times 10^{+01}$	$1.27 \times 10^{+01}$	3.52×10^{-01}	$4.43 \times 10^{+01}$	$1.63 \times 10^{+01}$	1.62×10^{-02}	$7.07 \times 10^{+00}$	$1.32 \times 10^{+01}$	1.05×10^{-02}
	C	+	+	+	+	+	+	+	+	/

Table 4. *Cont.*

No.	Item	FDR	UPSO	FIPS	CLPSO	MPSO	FPA	SSO	dg-PSO	DG-PSO
F12	Mean	$1.15 \times 10^{+01}$	$2.42 \times 10^{+01}$	$3.13 \times 10^{+01}$	$1.19 \times 10^{+01}$	$2.38 \times 10^{+01}$	$2.19 \times 10^{+01}$	$4.23 \times 10^{+01}$	**$1.78 \times 10^{+00}$**	$8.18 \times 10^{+00}$
	Std	$1.33 \times 10^{+00}$	$3.59 \times 10^{+00}$	2.74×10^{-01}	$1.09 \times 10^{+00}$	$2.57 \times 10^{+00}$	8.40×10^{-01}	$3.23 \times 10^{+00}$	6.58×10^{-01}	6.69×10^{-01}
	C	+	+	+	+	+	+	+	-	/
F13	Mean	$1.73 \times 10^{+01}$	$2.03 \times 10^{+01}$	$2.13 \times 10^{+01}$	$1.89 \times 10^{+01}$	$1.87 \times 10^{+01}$	$2.11 \times 10^{+01}$	$1.60 \times 10^{+01}$	$2.62 \times 10^{+01}$	**$1.58 \times 10^{+01}$**
	Std	5.78×10^{-01}	4.40×10^{-01}	1.64×10^{-01}	3.40×10^{-01}	7.67×10^{-01}	1.84×10^{-01}	5.61×10^{-01}	$6.18 \times 10^{+00}$	5.60×10^{-01}
	C	+	+	+	+	+	+	=	+	/
F14	Mean	$7.73 \times 10^{+01}$	$1.93 \times 10^{+02}$	$1.73 \times 10^{+02}$	$6.77 \times 10^{+01}$	$2.82 \times 10^{+02}$	$1.55 \times 10^{+02}$	$2.83 \times 10^{+02}$	$2.48 \times 10^{+02}$	**1.59×10^{-13}**
	Std	$7.70 \times 10^{+00}$	$1.76 \times 10^{+01}$	$1.20 \times 10^{+01}$	$4.42 \times 10^{+00}$	$3.04 \times 10^{+01}$	$1.07 \times 10^{+01}$	$2.23 \times 10^{+01}$	$2.04 \times 10^{+01}$	3.11×10^{-14}
	C	+	+	+	+	+	+	+	+	/
F15	Mean	$2.50 \times 10^{+01}$	$2.16 \times 10^{+01}$	$2.71 \times 10^{+01}$	$5.84 \times 10^{+03}$	$1.07 \times 10^{+01}$	$5.38 \times 10^{+01}$	$4.95 \times 10^{+01}$	$2.14 \times 10^{+01}$	**$1.62 \times 10^{+00}$**
	Std	$1.03 \times 10^{+01}$	$3.16 \times 10^{+00}$	9.78×10^{-01}	$4.23 \times 10^{+03}$	$1.73 \times 10^{+00}$	$9.85 \times 10^{+00}$	$1.06 \times 10^{+00}$	3.81×10^{-01}	7.70×10^{-02}
	C	+	+	+	+	+	+	+	+	/
F16	Mean	$1.26 \times 10^{+01}$	$2.02 \times 10^{+01}$	$2.03 \times 10^{+01}$	$5.43 \times 10^{+00}$	$2.09 \times 10^{+01}$	$2.10 \times 10^{+01}$	$2.17 \times 10^{+01}$	$1.40 \times 10^{+01}$	**8.70×10^{-01}**
	Std	$1.01 \times 10^{+00}$	2.30×10^{-01}	6.67×10^{-02}	5.09×10^{-01}	9.02×10^{-01}	2.92×10^{-01}	8.77×10^{-02}	$5.39 \times 10^{+00}$	2.86×10^{-01}
	C	+	+	+	+	+	+	+	+	/
F17	Mean	$1.41 \times 10^{+09}$	**$1.96 \times 10^{+03}$**	$4.83 \times 10^{+04}$	$3.83 \times 10^{+09}$	$2.03 \times 10^{+10}$	$1.51 \times 10^{+04}$	$2.89 \times 10^{+08}$	$7.71 \times 10^{+08}$	$1.58 \times 10^{+04}$
	Std	$7.06 \times 10^{+08}$	$9.96 \times 10^{+02}$	$3.28 \times 10^{+04}$	$1.15 \times 10^{+09}$	$1.38 \times 10^{+10}$	$5.62 \times 10^{+03}$	$6.63 \times 10^{+07}$	$8.60 \times 10^{+08}$	$8.68 \times 10^{+03}$
	C	+	-	+	+	+	=	+	+	/
F18	Mean	$1.88 \times 10^{+03}$	$4.90 \times 10^{+03}$	$1.09 \times 10^{+04}$	$3.05 \times 10^{+03}$	$3.73 \times 10^{+03}$	$1.48 \times 10^{+03}$	$6.35 \times 10^{+04}$	**6.23×10^{-09}**	3.37×10^{-03}
	Std	$1.28 \times 10^{+03}$	$5.95 \times 10^{+02}$	$9.56 \times 10^{+02}$	$1.00 \times 10^{+03}$	$1.57 \times 10^{+03}$	$3.26 \times 10^{+02}$	$3.80 \times 10^{+03}$	2.11×10^{-09}	1.52×10^{-03}
	C	+	+	+	+	+	+	+	-	/
F19	Mean	$1.95 \times 10^{+01}$	$2.12 \times 10^{+01}$	$2.17 \times 10^{+01}$	**$1.92 \times 10^{+01}$**	$2.07 \times 10^{+01}$	$2.17 \times 10^{+01}$	$2.22 \times 10^{+01}$	$2.07 \times 10^{+01}$	$1.93 \times 10^{+01}$
	Std	5.51×10^{-01}	1.08×10^{-01}	1.44×10^{-01}	3.62×10^{-01}	1.55×10^{-01}	2.64×10^{-01}	6.31×10^{-02}	8.82×10^{-01}	2.37×10^{-01}
	C	=	+	+	=	+	+	+	+	
F20	Mean	$7.71 \times 10^{+01}$	$2.70 \times 10^{+01}$	$3.19 \times 10^{+01}$	$1.15 \times 10^{+02}$	$3.94 \times 10^{+02}$	$4.45 \times 10^{+01}$	$5.55 \times 10^{+01}$	$1.19 \times 10^{+01}$	**$8.64 \times 10^{+00}$**
	Std	$4.93 \times 10^{+01}$	$4.94 \times 10^{+00}$	6.65×10^{-01}	$5.86 \times 10^{+01}$	$1.99 \times 10^{+02}$	$7.92 \times 10^{+00}$	$4.90 \times 10^{+00}$	$7.90 \times 10^{+00}$	9.42×10^{-01}
	C	+	+	+	+	+	+	+	+	/
	W/T/L	**17/2/1**	**19/0/1**	**20/0/0**	**19/1/0**	**19/1/0**	**19/2/0**	**19/1/0**	**13/1/6**	**/**

Note: the gray bacground highlights the best result on each function.

To rank the algorithms clearly, the Friedman test is used to compare the involved algorithms using all the data of mean fitness error values on the 20 problems. The Friedman test is the best-known procedure for testing the differences between more than two related samples [48], which can detect significant differences between the behavior of two or more algorithms. We conduct two tests that rank the algorithms on the 30D and 50D, respectively. The significance level is set to 0.05. Table 5 presents the numerical rankings obtained by the test. In addition, the corresponding graphical ranking results are shown in Figures 2 and 3, where the center square indicates the average rank of the corresponding algorithm and the line denotes the confidence intervals. Smaller ranks mean better performance and, when there is no overlap on the intervals of any two algorithms, they are significantly different. The results in these two figures clearly demonstrate that the proposed algorithm outperforms all other algorithms including dg-PSO, CLPSO and FPA in both 30D and 50D.

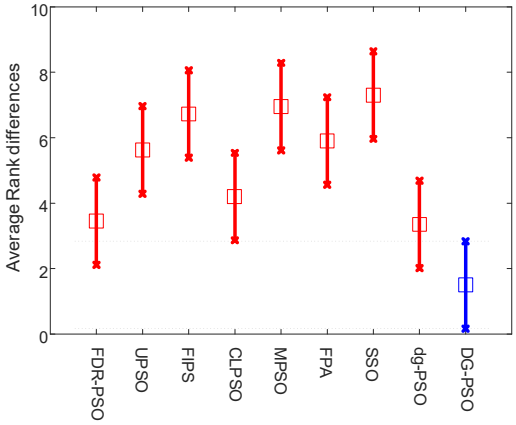


Figure 2. Friedman test of 30D problems.

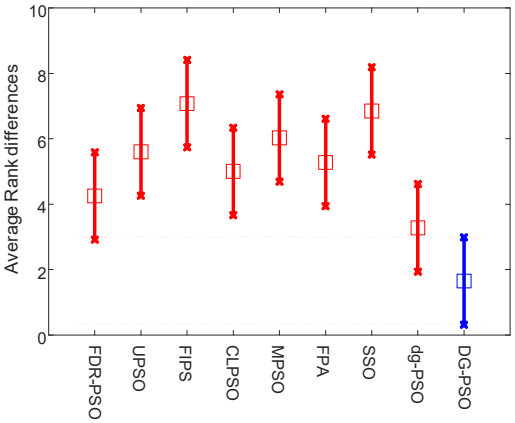


Figure 3. Friedman test of 50D problems.

Table 5. Numerical rankings of the Friedman test.

Dimensions	FDR	UPSO	FIPS	CLPSO	MPSO	FPA	SSO	dg-PSO	DG-PSO
30-*D*	3.45	5.625	6.725	4.2	6.95	5.9	7.3	3.35	**1.5**
50-*D*	4.25	5.6	7.075	5	6.025	5.275	6.85	3.275	**1.65**

For further evaluation, the convergence performance and average time consumption are also compared in Figures 4 and 5, respectively. The results of F8, F10, and F14 in 50-*D* are given to exemplify the performance. From Figure 4, we observe that DG-PSO has outstanding performance on the multimodal problems (F8 and F14), while dg-PSO obtained the best result in unimodal function F10.

From Figure 5, it can be found that DG-PSO consumes slightly more than the two related algorithms: CLPSO and FPA. However, the time consumption of DG-PSO is still acceptable when compared with the other algorithms such as FDR-PSO, UPSO, MPSO and SSO.

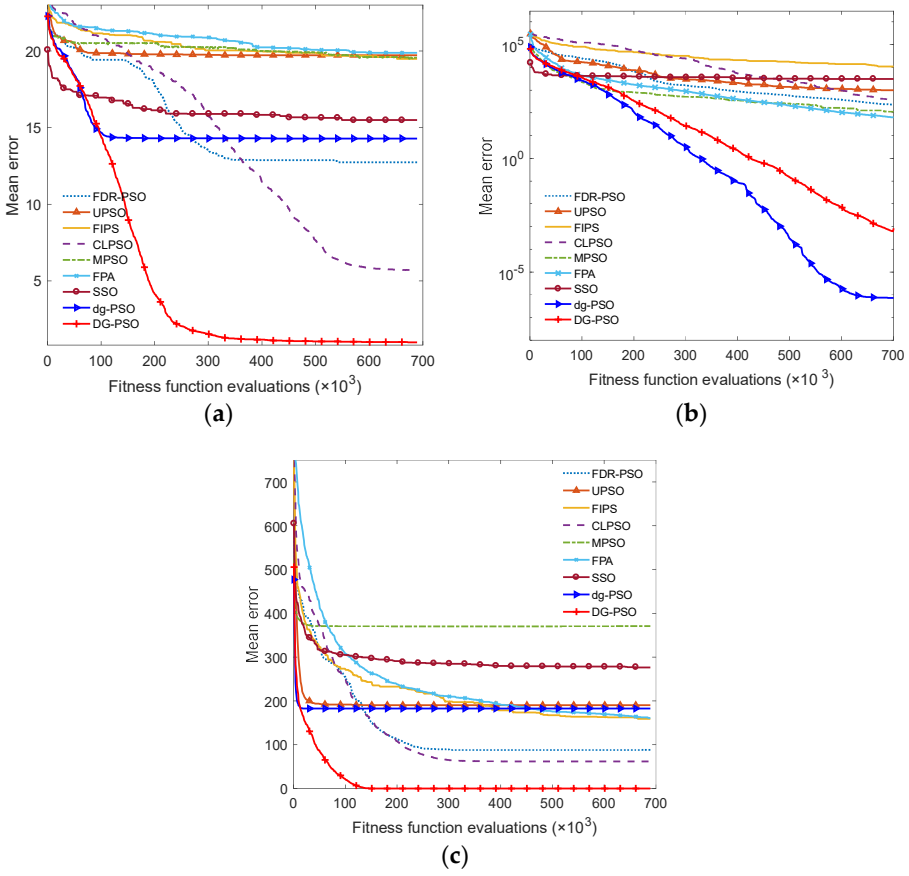


Figure 4. Convergence performance. (**a**) F8, multimodal; (**b**) F10, rotated unimodal; (**c**) F14, shifted multimodal.

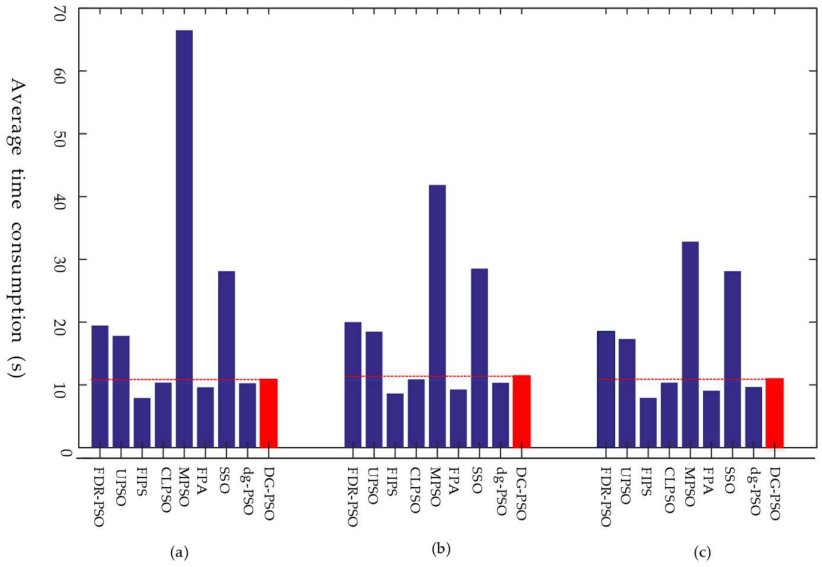


Figure 5. Average time consumption. (**a**) F8, multimodal; (**b**) F10, rotated unimodal; (**c**) F14, shifted multimodal.

5. DG-PSO Based Remote Sensing Image Segmentation

Image segmentation is a fundamental task in remote sensing applications [49], such as change detection and object-based classification. It is used with the expectation that it will divide the image into semantically significant regions, or objects, to be recognized by further processing steps [50]. This work attracts a lot of researchers in the past decade but is still an intractable problem [51]. In terms of all the existing segmentation methods, one of the most popular segmentation techniques is thresholding due to its simplicity, robustness and accuracy [52].

The thresholding methods can be divided into two categories: the bi-level thresholding and multilevel thresholding. If the object in an image is separated from the background using a single threshold value, it is called the bi-level thresholding. In contrast, the multilevel thresholding means that the given image are classified into several different regions according to multiple thresholds. In remote sensing image segmentation, bi-level thresholding does not give appropriate performance, and there are strong requirements of multilevel thresholding [53]. Therefore, numerous studies have been reported [47,53–58] in multilevel thresholding.

The most popular way [53–61] to search the optimal thresholds is to maximize some discriminating criteria (fitness function). The traditional method searches the optimal thresholds using exhaustive search strategies, which lead to high computation costs. In recent years, meta-heuristics based methods gained the attention of researchers because of the high computation inefficiency. Quantities of algorithms have been introduced to this area such as PSO [36], Differential Evolution (DE) [62], Artificial Bee Colony (ABC) [59,63,64], Wind Driven Optimization (WDO) [56], Cuckoo Search (CS) [65] and SSO [47]. However, the remote sensing images are very difficult to segment accurately due to multimodality of the histograms [53]. Therefore, improving the performance of the metaheuristic algorithms is necessary for the remote sensing image segmentation.

In this section, we applied the proposed algorithm to multilevel thresholding for optical remote sensing image segmentation. We first describe the problem. Then, the experimental setup is introduced carefully in Section 5.2. Finally, the results and analysis are given in detail.

5.1. Problem Definition

This subsection deals with the problem definition of multilevel thresholding problem. As we mention above, multilevel thresholding methods generally search the optimal thresholds by maximizing some criteria. In the literature, Otsu's criterion [66] has been widely employed [36,67,68]. It generally provides image segmentation with satisfactory results [69] and is known for its simplicity and effectivity with respect to uniformity and shape measures and can usually obtain optimal global threshold value [58].

Let $l \in [0, 1 \dots L-1]$ be the gray level of a given an image I, where L is the total gray levels, the problem is then defined as follows. Firstly, the image histogram is calculated and normalized, which is denoted by P_l, $l = 0, 1, \dots L-1$. For the $(D+1)-$class thresholding problem, there are D thresholds k_d, $(d = 1, 2, \dots D)$ that segment the image into $D+1$ classes. Assume that $k_0 (k_0 = 0)$ and $k_{D+1} (k_{D+1} = L)$ denote the upper and lower bound. Then, the thresholds can be sorted with $k_0 < k_1 < \dots < k_d < \dots k_{D+1}$, and the problem is defined using (9):

$$(k_1^*, k_2^* \dots k_D^*) = \underset{k_0 < k_1 < \dots < k_d < \dots k_{D+1}}{\arg \max} \{F(k_1, k_2 \dots k_D)\}, \tag{9}$$

where $F = \sum_{d=0}^{D} \omega_d (\mu_d - \mu_T)^2$, $\mu_d = \sum_{l=k_d}^{k_{d+1}} \frac{l \cdot P_l}{\omega_d}$. Here, $\omega_d = \sum_{l=k_d}^{k_{d+1}} P_l$ is the probability of the occurrence of the dth class. $\mu_T = \sum_{l=1}^{L} l \cdot P_l$ is the total mean intensity of the original image.

5.2. Experimental Setup

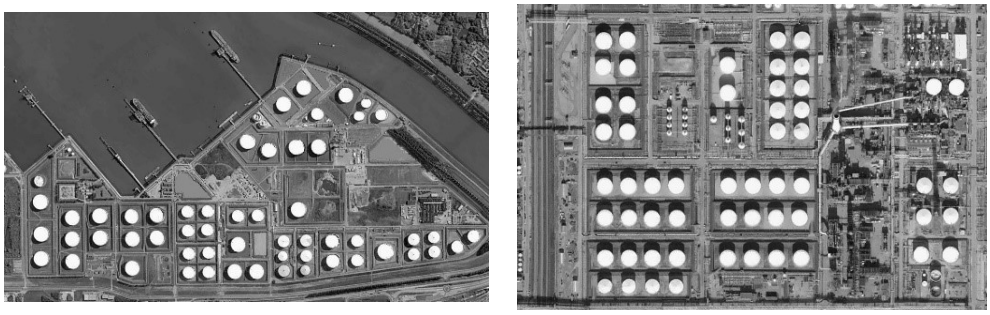
To demonstrate the superiority of the proposed method, five popular meta-heuristic algorithms in multilevel thresholding including DE, ABC, CS, MPSO, SSO are chosen to compare with the

proposed algorithm. All of these algorithms are demonstrated to have good performance in multilevel thresholding in the corresponding reference in Table 6. Specifically, ABC performs better than PSO when the level of thresholds is higher than two in [59]. Reference [53] demonstrates that CS showed remarkable performances in multilevel thresholding problems and could outperform the other known algorithms, such as DE, PSO, WDO and ABC. MPSO shows better performance than Genetic Algorithm (GA) and the original PSO [36]. SSO is applied to multilevel thresholding in [47] and it clearly outperforms PSO, BAT algorithm and FPA in [47]. The parameters of these algorithms are set according to the corresponding work shown in Table 6. The parameters of our proposed algorithm are the same as that in Section 4.

All populations are uniformly randomly initialized. Thirty independent runs are carried out for each algorithm on each image on 2, 3, 4, 5, 7, 9, 15 and 20 thresholds [68,69], respectively. All algorithms are conducted with the same maximum function evaluation: $MaxFEs = 3000 * D$ in identical search space: $[0, 256)$. All methods are adapted for integer optimization problems using the rounding method. Specifically, the search space is defined as $[0, 256)$ for 8-bit gray-scale images, and the integer is obtained by rounding down (e.g., 255.6 is rounded to 255). Figure 6 shows the test images (These images are taken from a very-high-resolution remote sensing image dataset constructed by Gong Cheng et al. from Northwestern Polytechnical University [70].

Table 6. Parameters and references of the algorithms.

Algorithm	Parameters	Value	Reference
DE	Population size Scaling factor Crossover possibility	40 0.8 0.25	[62]
ABC	Swam size Max trial limit	20 50	[59]
CS	Number of nests Step size Mutation probability value Scale factor	25 1 0.25 1.5	[53]
MPSO	Maximum, minimum swarm size acceleration constants c_1, c_2 Maximum, minimum inertial weight	40, 5 2, 2 0.9, 0.3	[36]
SSO	Population size The threshold PF	50 0.7	[47]



(**a**) (**b**)

Figure 6. *Cont.*

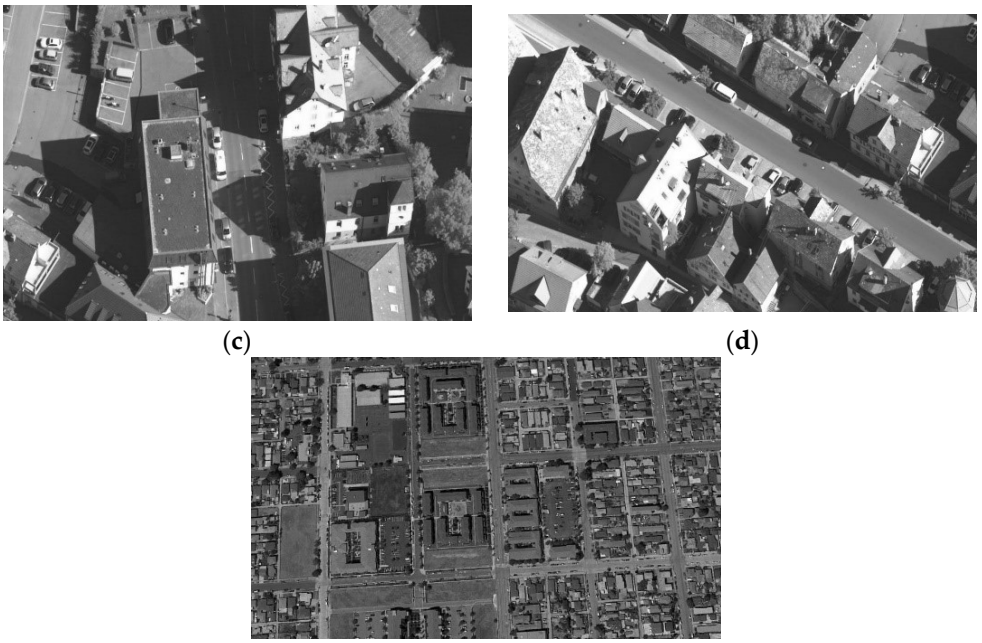


Figure 6. Images used in the experiments. (**a**) Image a; (**b**) Image b; (**c**) Image c; (**d**) Image d; (**e**) Image e.

5.3. Results and Discussion

In detail, the mean fitness and the corresponding standard deviation are given in Table 7, where the best one in each case of the mean fitness is shown in bold. It is easy to find that our algorithm obtains the best results in all cases in terms of the mean fitness, except the case of 7-level thresholding (D = 7) of image C. To evaluate the effectiveness of our algorithm's improvement over other ones, the involved algorithms are also ranked with the Friedman test. We conduct two tests that ranked the algorithms on the normal (D = 2, 3, 4 and 5) level and high level (The high level thresholding is popularly employed in multilevel thresholding [68,69]) (D = 7, 9, 15 and 20), respectively. Therefore, 40 variables ($i \times t \times m = 40$) are used in each comparison in each test, where $i = 5$ is the number of images, $t = 4$ denoted the number of levels, and $m = 2$ denoted the number of used measures including the meant fitness and the corresponding standard deviation. The significance level is set to 0.05. Table 8 and the two figures (Figures 7 and 8) present the numerical rankings and graphical results obtained by the test, where better performance is denoted by smaller ranks.

From the results of normal level thresholding shown in Figure 7, the proposed algorithm significantly outperforms DE, ABC and MPSO, and also showed an advantage over the other two algorithms. It can be observed from Figure 8 that the proposed algorithm ranks even better in high level thresholding, which showed a significant difference from all algorithms except CS (our algorithm also ranks better than CS). Figures 9 and 10 show the segmentation results. The pseudo color image shows the whole thresholding results, where each level of the image is represented by the regions with the same color. The binary images show some of the objects separated from the original image, which proved the effectiveness of the segmentation.

In conclusion, the results demonstrated that the proposed algorithm shows remarkable performance in multilevel thresholding when compared with other popular meta-heuristics in this research area.

Table 7. Statistical results.

Image	*D*	Item	DE	ABC	CS	MPSO	SSO	Ours
a	2	Mean	$\mathbf{1.79165 \times 10^{+03}}$	$\mathbf{1.79165 \times 10^{+03}}$	$\mathbf{1.79165 \times 10^{+03}}$	$\mathbf{1.79165 \times 10^{+03}}$	$\mathbf{1.79165 \times 10^{+03}}$	$\mathbf{1.79165 \times 10^{+03}}$
		Std	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}
	3	Mean	$1.94917 \times 10^{+03}$	$1.94918 \times 10^{+03}$	$\mathbf{1.94919 \times 10^{+03}}$	$\mathbf{1.94919 \times 10^{+03}}$	$\mathbf{1.94919 \times 10^{+03}}$	$\mathbf{1.94919 \times 10^{+03}}$
		Std	6.90×10^{-02}	1.59×10^{-02}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}
	4	Mean	$2.03403 \times 10^{+03}$	$2.03416 \times 10^{+03}$	$\mathbf{2.03427 \times 10^{+03}}$	$\mathbf{2.03427 \times 10^{+03}}$	$\mathbf{2.03427 \times 10^{+03}}$	$\mathbf{2.03427 \times 10^{+03}}$
		Std	6.01×10^{-01}	5.45×10^{-02}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	5	Mean	$2.06635 \times 10^{+03}$	$2.06618 \times 10^{+03}$	$\mathbf{2.06648 \times 10^{+03}}$	$\mathbf{2.06648 \times 10^{+03}}$	$\mathbf{2.06648 \times 10^{+03}}$	$\mathbf{2.06648 \times 10^{+03}}$
		Std	2.74×10^{-01}	1.79×10^{-01}	4.50×10^{-03}	5.84×10^{-03}	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	7	Mean	$2.09715 \times 10^{+03}$	$2.09653 \times 10^{+03}$	$2.09734 \times 10^{+03}$	$\mathbf{2.09739 \times 10^{+03}}$	$\mathbf{2.09739 \times 10^{+03}}$	$\mathbf{2.09739 \times 10^{+03}}$
		Std	2.25×10^{-01}	4.38×10^{-01}	4.63×10^{-02}	4.20×10^{-04}	1.42×10^{-03}	4.67×10^{-13}
	9	Mean	$2.11153 \times 10^{+03}$	$2.11097 \times 10^{+03}$	$2.11188 \times 10^{+03}$	$2.11158 \times 10^{+03}$	$\mathbf{2.11192 \times 10^{+03}}$	$\mathbf{2.11192 \times 10^{+03}}$
		Std	4.20×10^{-01}	3.52×10^{-01}	6.55×10^{-02}	2.43×10^{-01}	8.36×10^{-02}	9.06×10^{-02}
	15	Mean	$2.12912 \times 10^{+03}$	$2.12857 \times 10^{+03}$	$2.12965 \times 10^{+03}$	$2.12976 \times 10^{+03}$	$2.12922 \times 10^{+03}$	$\mathbf{2.12995 \times 10^{+03}}$
		Std	8.17×10^{-01}	2.30×10^{-01}	1.69×10^{-01}	3.25×10^{-01}	6.58×10^{-01}	1.40×10^{-01}
	20	Mean	$2.13280 \times 10^{+03}$	$2.13370 \times 10^{+03}$	$2.13444 \times 10^{+03}$	$2.13464 \times 10^{+03}$	$2.13333 \times 10^{+03}$	$\mathbf{2.13472 \times 10^{+03}}$
		Std	7.99×10^{-01}	2.01×10^{-01}	1.18×10^{-01}	5.27×10^{-01}	6.86×10^{-01}	1.44×10^{-01}
b	2	Mean	$\mathbf{2.27112 \times 10^{+03}}$	$\mathbf{2.27112 \times 10^{+03}}$	$\mathbf{2.27112 \times 10^{+03}}$	$\mathbf{2.27112 \times 10^{+03}}$	$\mathbf{2.27112 \times 10^{+03}}$	$\mathbf{2.27112 \times 10^{+03}}$
		Std	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}
	3	Mean	$2.50343 \times 10^{+03}$	$2.50346 \times 10^{+03}$	$\mathbf{2.50347 \times 10^{+03}}$	$\mathbf{2.50347 \times 10^{+03}}$	$\mathbf{2.50347 \times 10^{+03}}$	$\mathbf{2.50347 \times 10^{+03}}$
		Std	1.37×10^{-01}	1.40×10^{-02}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}
	4	Mean	$2.58706 \times 10^{+03}$	$2.58695 \times 10^{+03}$	$\mathbf{2.58710 \times 10^{+03}}$	$2.58709 \times 10^{+03}$	$\mathbf{2.58710 \times 10^{+03}}$	$\mathbf{2.58710 \times 10^{+03}}$
		Std	1.12×10^{-01}	1.30×10^{-01}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}	4.67×10^{-13}
	5	Mean	$2.62673 \times 10^{+03}$	$2.62643 \times 10^{+03}$	$2.62685 \times 10^{+03}$	$\mathbf{2.62686 \times 10^{+03}}$	$\mathbf{2.62686 \times 10^{+03}}$	$\mathbf{2.62686 \times 10^{+03}}$
		Std	1.49×10^{-01}	2.85×10^{-01}	1.39×10^{-02}	6.24×10^{-02}	9.33×10^{-13}	9.33×10^{-13}
	7	Mean	$2.66456 \times 10^{+03}$	$2.66401 \times 10^{+03}$	$2.66481 \times 10^{+03}$	$2.66488 \times 10^{+03}$	$\mathbf{2.66490 \times 10^{+03}}$	$\mathbf{2.66490 \times 10^{+03}}$
		Std	2.78×10^{-01}	2.76×10^{-01}	5.67×10^{-02}	2.08×10^{-01}	4.67×10^{-13}	4.67×10^{-13}
	9	Mean	$2.68400 \times 10^{+03}$	$2.68338 \times 10^{+03}$	$2.68448 \times 10^{+03}$	$2.68458 \times 10^{+03}$	$2.68457 \times 10^{+03}$	$\mathbf{2.68459 \times 10^{+03}}$
		Std	4.71×10^{-01}	4.59×10^{-01}	5.68×10^{-02}	3.55×10^{-01}	2.69×10^{-02}	8.48×10^{-03}
	15	Mean	$2.70493 \times 10^{+03}$	$2.70461 \times 10^{+03}$	$2.70564 \times 10^{+03}$	$2.70564 \times 10^{+03}$	$2.70535 \times 10^{+03}$	$\mathbf{2.70588 \times 10^{+03}}$
		Std	7.97×10^{-01}	2.72×10^{-01}	1.34×10^{-01}	3.15×10^{-01}	$1.10 \times 10^{+00}$	7.63×10^{-02}
	20	Mean	$2.70907 \times 10^{+03}$	$2.71046 \times 10^{+03}$	$2.71127 \times 10^{+03}$	$2.71168 \times 10^{+03}$	$2.71019 \times 10^{+03}$	$\mathbf{2.71153 \times 10^{+03}}$
		Std	$1.30 \times 10^{+00}$	2.09×10^{-01}	1.54×10^{-01}	5.25×10^{-01}	7.90×10^{-01}	2.79×10^{-01}

Table 7. *Cont.*

Image	*D*	Item	DE	ABC	CS	MPSO	SSO	Ours
c	2	Mean	**$2.67669 \times 10^{+03}$**	**$2.67669 \times 10^{+03}$**	**$2.67669 \times 10^{+03}$**	**$2.67669 \times 10^{+03}$**	**$2.67669 \times 10^{+03}$**	**$2.67669 \times 10^{+03}$**
		Std	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}
	3	Mean	$2.87034 \times 10^{+03}$	$2.87040 \times 10^{+03}$	**$2.87042 \times 10^{+03}$**	**$2.87042 \times 10^{+03}$**	**$2.87042 \times 10^{+03}$**	**$2.87042 \times 10^{+03}$**
		Std	2.80×10^{-01}	2.94×10^{-02}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}	9.33×10^{-13}
	4	Mean	$2.93107 \times 10^{+03}$	$2.93110 \times 10^{+03}$	$2.93128 \times 10^{+03}$	**$2.93129 \times 10^{+03}$**	**$2.93129 \times 10^{+03}$**	**$2.93129 \times 10^{+03}$**
		Std	5.24×10^{-01}	1.69×10^{-01}	2.87×10^{-03}	2.36×10^{-01}	1.40×10^{-12}	1.40×10^{-12}
	5	Mean	$2.96630 \times 10^{+03}$	$2.96622 \times 10^{+03}$	**$2.96645 \times 10^{+03}$**	**$2.96645 \times 10^{+03}$**	**$2.96645 \times 10^{+03}$**	**$2.96645 \times 10^{+03}$**
		Std	3.08×10^{-01}	1.72×10^{-01}	1.90×10^{-03}	1.65×10^{-01}	1.40×10^{-12}	1.57×10^{-03}
	7	Mean	$2.99969 \times 10^{+03}$	$2.99923 \times 10^{+03}$	**$2.99982 \times 10^{+03}$**	$2.99979 \times 10^{+03}$	$2.99980 \times 10^{+03}$	$2.99981 \times 10^{+03}$
		Std	2.09×10^{-01}	2.68×10^{-01}	4.74×10^{-02}	9.48×10^{-03}	8.60×10^{-02}	8.25×10^{-02}
	9	Mean	$3.01430 \times 10^{+03}$	$3.01377 \times 10^{+03}$	$3.01450 \times 10^{+03}$	$3.01462 \times 10^{+03}$	$3.01467 \times 10^{+03}$	**$3.01474 \times 10^{+03}$**
		Std	4.31×10^{-01}	3.59×10^{-01}	2.06×10^{-01}	3.49×10^{-01}	3.39×10^{-01}	2.45×10^{-01}
	15	Mean	$3.02911 \times 10^{+03}$	$3.02886 \times 10^{+03}$	$3.02963 \times 10^{+03}$	$3.02973 \times 10^{+03}$	$3.02936 \times 10^{+03}$	**$3.02989 \times 10^{+03}$**
		Std	8.23×10^{-01}	1.91×10^{-01}	1.53×10^{-01}	2.02×10^{-01}	8.09×10^{-01}	2.10×10^{-01}
	20	Mean	$3.03250 \times 10^{+03}$	$3.03354 \times 10^{+03}$	$3.03409 \times 10^{+03}$	$3.03443 \times 10^{+03}$	$3.03310 \times 10^{+03}$	**$3.03447 \times 10^{+03}$**
		Std	7.16×10^{-01}	1.50×10^{-01}	1.07×10^{-01}	5.52×10^{-01}	6.31×10^{-01}	1.05×10^{-01}
d	2	Mean	**$3.50826 \times 10^{+03}$**	**$3.50826 \times 10^{+03}$**	**$3.50826 \times 10^{+03}$**	**$3.50826 \times 10^{+03}$**	**$3.50826 \times 10^{+03}$**	**$3.50826 \times 10^{+03}$**
		Std	1.40×10^{-12}	1.40×10^{-12}	1.40×10^{-12}	1.40×10^{-12}	1.40×10^{-12}	1.40×10^{-12}
	3	Mean	$3.65657 \times 10^{+03}$	$3.65661 \times 10^{+03}$	**$3.65665 \times 10^{+03}$**	**$3.65665 \times 10^{+03}$**	**$3.65665 \times 10^{+03}$**	**$3.65665 \times 10^{+03}$**
		Std	2.31×10^{-01}	5.15×10^{-02}	2.33×10^{-12}	2.05×10^{-02}	2.33×10^{-12}	2.33×10^{-12}
	4	Mean	$3.73546 \times 10^{+03}$	$3.73537 \times 10^{+03}$	**$3.73562 \times 10^{+03}$**	**$3.73562 \times 10^{+03}$**	**$3.73562 \times 10^{+03}$**	**$3.73562 \times 10^{+03}$**
		Std	2.28×10^{-01}	1.56×10^{-01}	4.67×10^{-13}	2.78×10^{-02}	4.67×10^{-13}	4.67×10^{-13}
	5	Mean	$3.78074 \times 10^{+03}$	$3.78052 \times 10^{+03}$	**$3.78100 \times 10^{+03}$**	$3.78099 \times 10^{+03}$	**$3.78100 \times 10^{+03}$**	**$3.78100 \times 10^{+03}$**
		Std	4.10×10^{-01}	2.56×10^{-01}	1.23×10^{-02}	3.15×10^{-01}	9.33×10^{-13}	9.33×10^{-13}
	7	Mean	$3.81857 \times 10^{+03}$	$3.81818 \times 10^{+03}$	**$3.81897 \times 10^{+03}$**	$3.81864 \times 10^{+03}$	$3.81865 \times 10^{+03}$	$3.81883 \times 10^{+03}$
		Std	6.02×10^{-01}	4.10×10^{-01}	3.38×10^{-02}	4.35×10^{-01}	$1.11 \times 10^{+00}$	8.17×10^{-01}
	9	Mean	$3.83628 \times 10^{+03}$	$3.83555 \times 10^{+03}$	$3.83665 \times 10^{+03}$	$3.83681 \times 10^{+03}$	$3.83680 \times 10^{+03}$	**$3.83682 \times 10^{+03}$**
		Std	5.38×10^{-01}	4.48×10^{-01}	8.98×10^{-02}	5.02×10^{-01}	1.62×10^{-02}	3.18×10^{-03}
	15	Mean	$3.85501 \times 10^{+03}$	$3.85465 \times 10^{+03}$	$3.85558 \times 10^{+03}$	$3.85598 \times 10^{+03}$	$3.85543 \times 10^{+03}$	**$3.85593 \times 10^{+03}$**
		Std	9.03×10^{-01}	2.48×10^{-01}	1.44×10^{-01}	8.12×10^{-01}	6.33×10^{-01}	1.62×10^{-01}
	20	Mean	$3.85891 \times 10^{+03}$	$3.86036 \times 10^{+03}$	$3.86097 \times 10^{+03}$	$3.86131 \times 10^{+03}$	$3.86011 \times 10^{+03}$	**$3.86135 \times 10^{+03}$**
		Std	9.91×10^{-01}	2.52×10^{-01}	1.64×10^{-01}	6.12×10^{-01}	7.65×10^{-01}	1.97×10^{-01}

Table 7. *Cont.*

Image	*D*	Item	DE	ABC	CS	MPSO	SSO	Ours
e	2	Mean	**$1.07218 \times 10^{+03}$**	**$1.07218 \times 10^{+03}$**	**$1.07218 \times 10^{+03}$**	**$1.07218 \times 10^{+03}$**	**$1.07218 \times 10^{+03}$**	**$1.07218 \times 10^{+03}$**
		Std	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$	$0.00 \times 10^{+00}$
	3	Mean	$1.17024 \times 10^{+03}$	$1.17021 \times 10^{+03}$	**$1.17025 \times 10^{+03}$**	$1.17024 \times 10^{+03}$	**$1.17025 \times 10^{+03}$**	**$1.17025 \times 10^{+03}$**
		Std	1.49×10^{-02}	4.97×10^{-02}	2.33×10^{-13}	2.33×10^{-13}	2.33×10^{-13}	2.33×10^{-13}
	4	Mean	$1.21391 \times 10^{+03}$	$1.21383 \times 10^{+03}$	$1.21398 \times 10^{+03}$	**$1.21399 \times 10^{+03}$**	**$1.21399 \times 10^{+03}$**	**$1.21399 \times 10^{+03}$**
		Std	1.29×10^{-01}	1.23×10^{-01}	6.60×10^{-03}	3.89×10^{-02}	2.33×10^{-13}	2.33×10^{-13}
	5	Mean	$1.24246 \times 10^{+03}$	$1.24221 \times 10^{+03}$	$1.24261 \times 10^{+03}$	**$1.24265 \times 10^{+03}$**	**$1.24265 \times 10^{+03}$**	**$1.24265 \times 10^{+03}$**
		Std	3.18×10^{-01}	1.81×10^{-01}	6.27×10^{-02}	1.95×10^{-01}	4.67×10^{-13}	4.67×10^{-13}
	7	Mean	$1.27623 \times 10^{+03}$	$1.27577 \times 10^{+03}$	$1.27656 \times 10^{+03}$	$1.27660 \times 10^{+03}$	**$1.27661 \times 10^{+03}$**	**$1.27661 \times 10^{+03}$**
		Std	5.52×10^{-01}	4.12×10^{-01}	5.43×10^{-02}	2.55×10^{-01}	2.33×10^{-13}	2.33×10^{-13}
	9	Mean	$1.29227 \times 10^{+03}$	$1.29152 \times 10^{+03}$	$1.29254 \times 10^{+03}$	$1.29267 \times 10^{+03}$	$1.29267 \times 10^{+03}$	**$1.29268 \times 10^{+03}$**
		Std	4.55×10^{-01}	4.68×10^{-01}	7.29×10^{-02}	3.04×10^{-01}	1.42×10^{-02}	3.65×10^{-03}
	15	Mean	$1.30947 \times 10^{+03}$	$1.30914 \times 10^{+03}$	$1.30986 \times 10^{+03}$	$1.31018 \times 10^{+03}$	$1.30972 \times 10^{+03}$	**$1.31035 \times 10^{+03}$**
		Std	7.66×10^{-01}	3.43×10^{-01}	2.49×10^{-01}	3.64×10^{-01}	6.56×10^{-01}	1.45×10^{-01}
	20	Mean	$1.31345 \times 10^{+03}$	$1.31412 \times 10^{+03}$	$1.31467 \times 10^{+03}$	$1.31501 \times 10^{+03}$	$1.31379 \times 10^{+03}$	**$1.31522 \times 10^{+03}$**
		Std	8.91×10^{-01}	2.70×10^{-01}	1.94×10^{-01}	4.25×10^{-01}	7.16×10^{-01}	1.09×10^{-01}

Note: the gray bacground highlights the best result on each function.

Table 8. Numerical rankings of the Friedman tests.

Level	DE	ABC	CS	MPSO	SSO	Ours
Normal level	5.25	4.775	2.8625	3.3125	2.775	**2.025**
High level	5.3125	4.7625	2.6625	3.1125	3.2	**1.95**

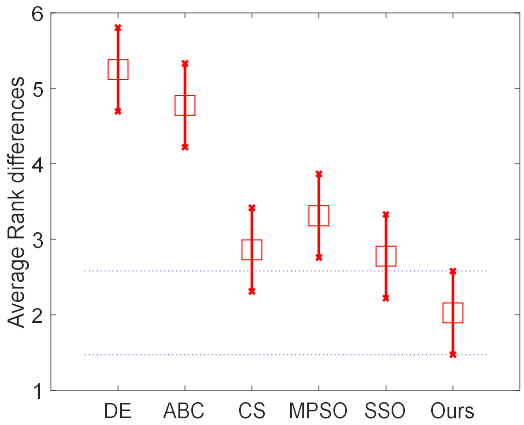


Figure 7. Friedman test of the normal level.

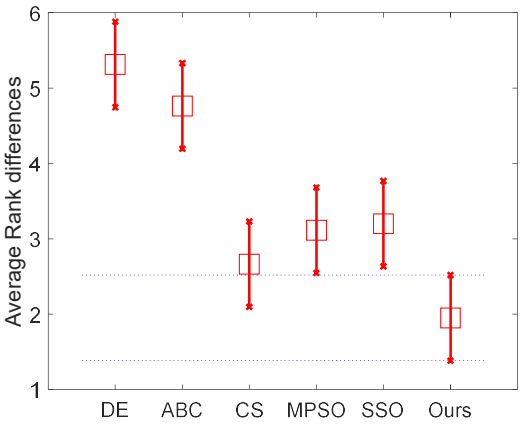
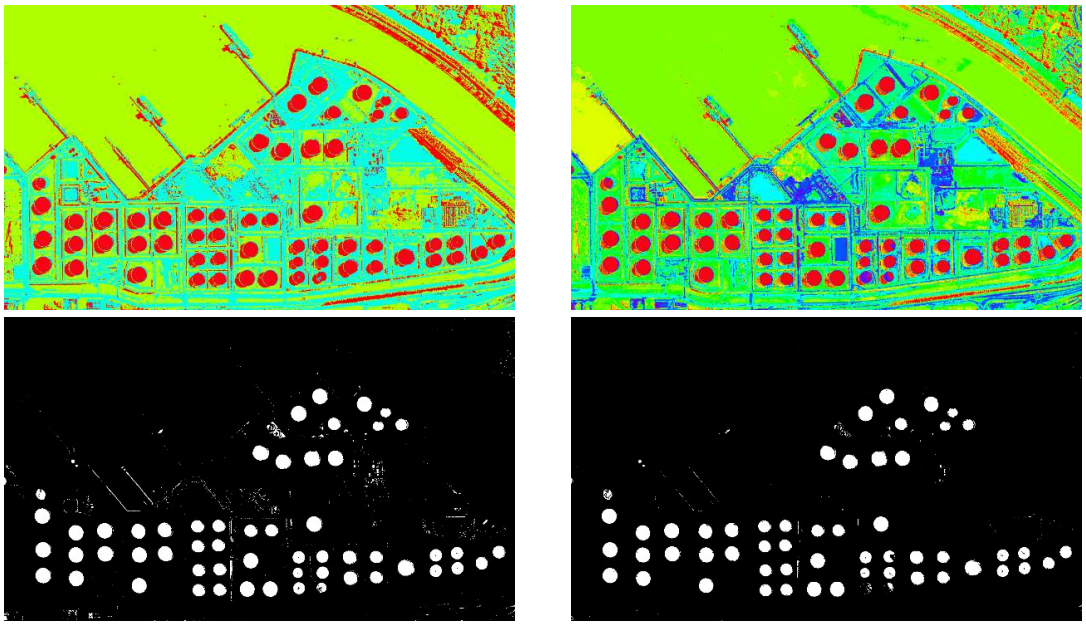


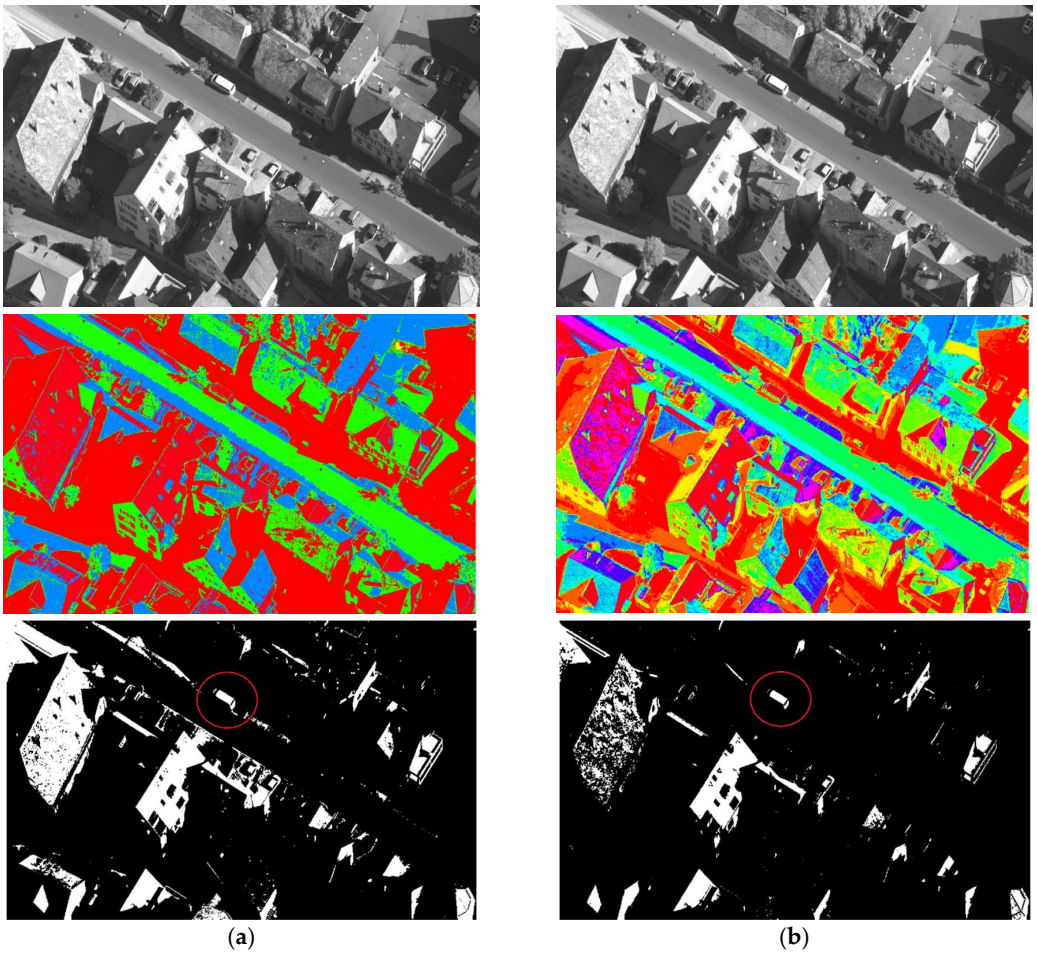
Figure 8. Friedman test of the high level.

Figure 9. *Cont.*



(**a**) (**b**)

Figure 9. The segmentation results of the Image a. (**a**) 3-level thresholding; (**b**) 9-level thresholding.



(**a**) (**b**)

Figure 10. The segmentation results of the Image d. (**a**) 3-level thresholding; (**b**) 9-level thresholding, which separated the target better.

6. Conclusions

This paper proposes a variant of particle swarm optimization called DG-PSO. DG-PSO uses a double-group based evolution framework. The individuals in DG-PSO are divided into two groups according to their fitness values. Two main ideas are introduced in the evolution of the disadvantaged group: a hybrid strategy for learning and a diversity enhancing strategy for avoiding premature convergence. The experimental results on various benchmark functions demonstrate that: although DG-PSO consumes slightly more time than the two related algorithms of the proposed algorithm: CLPSO and FPA; DG-PSO achieves a significant improvement in terms of mean fitness error, the corresponding standard deviation and convergence performance over all contrast algorithms. In addition, we further apply the proposed algorithm to multilevel thresholding for remote sensing image segmentation. The results also show the effectiveness of DG-PSO.

Author Contributions: L.S. designed the experiments and write the paper; X.H. analyzed the data and results; C.F. performed the experiments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 1944, pp. 1942–1948.
2. Alrashidi, M.R.; El-Hawary, M.E. A Survey of Particle Swarm Optimization Applications in Electric Power Systems. *IEEE Trans. Evol. Comput.* **2009**, *13*, 913–918. [CrossRef]
3. Gisbert, S.; Michael, S.; Michael, M. Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training. *BMC Bioinf.* **2006**, *7*, 125.
4. Tang, B.; Zhu, Z.; Luo, J. A convergence-guaranteed particle swarm optimization method for mobile robot global path planning. *Assem. Autom.* **2017**, *37*, 114–129. [CrossRef]
5. Xue, B.; Zhang, M.; Browne, W.N. Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Appl. Soft Comput.* **2014**, *18*, 261–276. [CrossRef]
6. Eslami, M.; Shareef, H.; Khajehzadeh, M.; Mohamed, A. A Survey of the State of the Art in Particle Swarm Optimization. *Res. J. Appl. Sci. Eng. Technol.* **2012**, *4*, 1181–1197.
7. Zhang, Q.; Liu, W.; Meng, X.; Yang, B.; Vasilakos, A.V. Vector coevolving particle swarm optimization algorithm. *Inf. Sci.* **2017**, *394–395*, 273–298. [CrossRef]
8. Cui, Q.; Li, Q.; Li, G.; Li, Z.; Han, X.; Lee, H.P.; Liang, Y.; Wang, B.; Jiang, J.; Wu, C. Globally-Optimal Prediction-Based Adaptive Mutation Particle Swarm Optimization. *Inf. Sci.* **2017**, *418–419*, 186–217. [CrossRef]
9. Shi, Y.; Eberhart, R. A Modified particle swarm optimizer. In Proceedings of the International Conference on Evolutionary Computation Proceedings 1998 (IEEE ICEC Conference), Anchorage, AK, USA, 4–9 May 1998; pp. 69–73.
10. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99), Washington, DC, USA, 6–9 July 1999; Volume 321, pp. 320–324.
11. Li, L.; Xue, B.; Niu, B.; Chai, Y.; Wu, J. The novel nonlinear strategy of inertia weight in particle swarm optimization. In Proceedings of the International Conference on Bio-Inspired Computing (Bic-Ta 2009), Beijing, China, 16–19 October 2009; pp. 1–5.
12. Wu, J.; He, X.X.; Zhao, W.G.; Rui, W. Exponential inertia weight particle swarm algorithm for dynamics optimization of electromechanical coupling system. In Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems, Shanghai, China, 20–22 November 2009; pp. 479–483.
13. Pant, M.; Radha, T.; Singh, V.P. Particle Swarm Optimization Using Gaussian Inertia Weight. In Proceedings of the International Conference on Computational Intelligence and Multimedia Applications, Sivakasi, India, 13–15 December 2007; pp. 97–102.
14. Clerc, M.; Kennedy, J. The particle swarm—Explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
15. Bajpai, P.; Singh, S.N. Fuzzy Adaptive Particle Swarm Optimization for Bidding Strategy in Uniform Price Spot Market. *IEEE Trans. Power Syst.* **2007**, *22*, 2152–2160. [CrossRef]

16. Chang, X.Y.; Li, R.J. Experimental analysis of acceleration coefficient in particle swarm optimization algorithm. *Comput. Eng.* **2010**, *36*, 183–186.
17. Zhan, Z.H.; Zhang, J. Adaptive Particle Swarm Optimization. *IEEE Trans. Syst. Man Cybern.* **2009**, *39*, 1362–1381. [CrossRef] [PubMed]
18. Kennedy, J.; Mendes, R. Population structure and particle swarm performance. In Proceedings of the 2002 Congress on Evolutionary Computation (2002 CEC), Honolulu, HI, USA, 12–17 May 2002; pp. 1671–1676.
19. Gou, J.; Lei, Y.X.; Guo, W.P.; Wang, C.; Cai, Y.Q.; Luo, W. A novel improved particle swarm optimization algorithm based on individual difference evolution. *Appl. Soft Comput.* **2017**, *57*, 468–481. [CrossRef]
20. Liu, Q.; Wei, W.; Yuan, H.; Zhan, Z.H.; Li, Y. Topology selection for particle swarm optimization. *Inf. Sci.* **2016**, *363*, 154–173. [CrossRef]
21. Gong, Y.J.; Li, J.J.; Zhou, Y.; Li, Y.; Chung, H.S.; Shi, Y.H.; Zhang, J. Genetic Learning Particle Swarm Optimization. *IEEE Trans. Cybern.* **2016**, *46*, 2277–2290. [CrossRef] [PubMed]
22. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans. Evol. Comput.* **2006**, *10*, 281–295. [CrossRef]
23. Frans, V.D.B.; Engelbrecht, A.P. A Cooperative approach to particle swarm optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 225–239.
24. Li, X.; Yao, X. Cooperatively Coevolving Particle Swarms for Large Scale Optimization. *IEEE Trans. Evol. Comput.* **2012**, *16*, 210–224.
25. Sun, J.; Xu, W.; Fang, W. A Diversity-Guided Quantum-Behaved Particle Swarm Optimization Algorithm. In *Asia-Pacific Conference on Simulated Evolution and Learning*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 497–504.
26. Shen, Y.; Wei, L.; Zeng, C.; Chen, J. Particle Swarm Optimization with double learning patterns. *Comput. Intell. Neurosci.* **2016**, *2016*, 32. [CrossRef] [PubMed]
27. Yang, X.S. Flower pollination algorithm for global optimization. In *International Conference on Unconventional Computation and Natural Computation 2012*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7445, pp. 240–249.
28. Gao, L.; Hailu, A. Comprehensive Learning Particle Swarm Optimizer for Constrained Mixed-Variable Optimization Problems. *Int. J. Comput. Intell. Syst.* **2010**, *3*, 832–842. [CrossRef]
29. Hu, Z.; Bao, Y.; Xiong, T. Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. *Appl. Soft Comput.* **2014**, *25*, 15–25. [CrossRef]
30. Mahadevan, K.; Kannan, P.S. Comprehensive learning particle swarm optimization for reactive power dispatch. *Appl. Soft Comput.* **2010**, *10*, 641–652. [CrossRef]
31. Zhang, X.; Yu, X.; Qin, H. Optimal operation of multi-reservoir hydropower systems using enhanced comprehensive learning particle swarm optimization. *J. Hydro-Environ. Res.* **2016**, *10*, 50–63. [CrossRef]
32. Abdelaziz, A.Y.; Ali, E.S.; Elazim, S.M.A. Combined economic and emission dispatch solution using Flower Pollination Algorithm. *Int. J. Electr. Power Energy Syst.* **2016**, *80*, 264–274. [CrossRef]
33. Abdel-Raouf, O.; Abdel-Baset, M.; El-Henawy, I. A Novel Hybrid Flower Pollination Algorithm with Chaotic Harmony Search for Solving Sudoku Puzzles. *Int. J. Mod. Educ. Comput. Sci.* **2014**, *7*, 126–132.
34. Sayed, S.A.F.; Nabil, E.; Badr, A. A binary clonal flower pollination algorithm for feature selection. *Pattern Recognit. Lett.* **2016**, *77*, 21–27. [CrossRef]
35. Wang, H.; Sun, H.; Li, C.; Rahnamayan, S.; Pan, J.S. Diversity enhanced particle swarm optimization with neighborhood search. *Inf. Sci.* **2013**, *223*, 119–135. [CrossRef]
36. Liu, Y.; Mu, C.; Kou, W.; Liu, J. Modified particle swarm optimization-based multilevel thresholding for image segmentation. *Soft Comput.* **2015**, *19*, 1311–1327. [CrossRef]
37. Cheng, S.; Shi, Y.; Qin, Q. Population diversity based study on search information propagation in particle swarm optimization. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, Australia, 10–15 June 2012; pp. 1272–1279.
38. Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report 201311; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013.

39. García, S.; Molina, D.; Lozano, M.; Herrera, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *J. Heuristics* **2009**, *15*, 617–644. [CrossRef]
40. Parsopoulos, K.E.; Vrahatis, M.N. UPSO: A unified particle swarm optimization scheme. *Lect. Ser. Comput. Comput. Sci.* **2004**, *1*, 868–873.
41. Mendes, R.; Kennedy, J.; Neves, J. The fully informed particle swarm: Simpler, maybe better. *IEEE Trans. Evol. Comput.* **2004**, *8*, 204–210. [CrossRef]
42. Peram, T.; Veeramachaneni, K.; Mohan, C.K. Fitness-distance-ratio based particle swarm optimization. In Proceedings of the Swarm Intelligence Symposium (SIS 2003), Indianapolis, IN, USA, 24–26 April 2003; pp. 174–181.
43. Cuevas, E.; Cortés, M.A.D.; Navarro, D.A.O. A Swarm Global Optimization Algorithm Inspired in the Behavior of the Social-Spider. *Exp. Syst. Appl.* **2014**, *40*, 6374–6384. [CrossRef]
44. Zhang, Q.; Liu, W.; Meng, X.; Yang, B.; Vasilakos, A.V. Vector coevolving particle swarm optimization algorithm. *Inf. Sci.* **2017**, *394*, 273–298. [CrossRef]
45. Qin, Q.; Cheng, S.; Zhang, Q.; Li, L.; Shi, Y. Particle Swarm Optimization With Interswarm Interactive Learning Strategy. *IEEE Trans. Cybern.* **2016**, *46*, 2238–2251. [CrossRef] [PubMed]
46. Parsopoulos, K.E.; Vrahatis, M.N. A unified particle swarm optimization scheme. In *Proceedings of International Conference on Computational Methods in Sciences and Engineering*; Ser Lecture Series on Computer & Computational Sciences Attica; VSP International Science: Tripolis, Greece, 2003.
47. Ouadfel, S.; Taleb-Ahmed, A. Social spiders optimization and flower pollination algorithm for multilevel image thresholding: A performance study. *Exp. Syst. Appl.* **2016**, *55*, 566–584. [CrossRef]
48. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
49. Blaschke, T. Object based image analysis for remote sensing. *ISPRS J. Photogramm. Remote Sens.* **2010**, *65*, 2–16. [CrossRef]
50. Trias-Sanz, R.; Stamon, G.; Louchet, J. Using colour, texture, and hierarchial segmentation for high-resolution remote sensing. *ISPRS J. Photogramm. Remote Sens.* **2008**, *63*, 156–168. [CrossRef]
51. Li, N.; Huo, H.; Fang, T. A Novel Texture-Preceded Segmentation Algorithm for High-Resolution Imagery. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 2818–2828.
52. Sezgin, M.; Sankur, B. Survey over image thresholding techniques and quantitative performance evaluation. *J. Electron. Imaging* **2004**, *13*, 146–168.
53. Bhandari, A.K.; Kumar, A.; Singh, G.K. Tsallis entropy based multilevel thresholding for colored satellite image segmentation using evolutionary algorithms. *Exp. Syst. Appl.* **2015**, *42*, 8707–8730. [CrossRef]
54. Bhandari, A.K.; Kumar, A.; Chaudhary, S.; Singh, G.K. A novel color image multilevel thresholding based segmentation using nature inspired optimization algorithms. *Exp. Syst. Appl.* **2016**, *63*, 112–133. [CrossRef]
55. Bhandari, A.K.; Kumar, A.; Singh, G.K. Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions. *Exp. Syst. Appl.* **2015**, *42*, 1573–1601. [CrossRef]
56. Bhandari, A.K.; Singh, V.K.; Kumar, A.; Singh, G.K. Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. *Exp. Syst. Appl. Int. J.* **2014**, *41*, 3538–3560. [CrossRef]
57. Khaled, A.; Abdel-Kader, R.F.; Yasein, M.S. A Hybrid Color Image Quantization Algorithm Based on k-Means and Harmony Search Algorithms. *Appl. Artif. Intell.* **2016**, *30*, 331–351. [CrossRef]
58. Mala, C.; Sridevi, M. Multilevel threshold selection for image segmentation using soft computing techniques. *Soft Comput.* **2016**, *20*, 1793–1810. [CrossRef]
59. Akay, B. A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding. *Appl. Soft Comput.* **2013**, *13*, 3066–3091. [CrossRef]
60. Dey, S.; Saha, I.; Bhattacharyya, S.; Maulik, U. Multi-level thresholding using quantum inspired meta-heuristics. *Knowl. Based Syst.* **2014**, *67*, 373–400. [CrossRef]
61. He, L.; Huang, S. Modified firefly algorithm based multilevel thresholding for color image segmentation. *Neurocomputing* **2017**, *240*, 152–174. [CrossRef]

62. Cuevas, E.; Zaldivar, D.; Pérez-Cisneros, M. A novel multi-threshold segmentation approach based on differential evolution optimization. *Exp. Syst. Appl.* **2010**, *37*, 5265–5271. [CrossRef]
63. Horng, M.H. Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation. *Exp. Syst. Appl.* **2011**, *38*, 13785–13791. [CrossRef]
64. Osuna-Enciso, V.N.; Cuevas, E.; Sossa, H. A comparison of nature inspired algorithms for multi-threshold image segmentation. *Exp. Syst. Appl.* **2013**, *40*, 1213–1219. [CrossRef]
65. Pare, S.; Kumar, A.; Bajaj, V.; Singh, G.K. A multilevel color image segmentation technique based on cuckoo search algorithm and energy curve. *Appl. Soft Comput.* **2016**, *47*, 76–102. [CrossRef]
66. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
67. Ayala, H.V.H.; Santos, F.M.D.; Mariani, V.C.; Coelho, L.D.S. Image thresholding segmentation based on a novel beta differential evolution approach. *Exp. Syst. Appl.* **2015**, *42*, 2136–2142. [CrossRef]
68. Mlakar, U.; Potočnik, B.; Brest, J. A hybrid differential evolution for optimal multilevel image thresholding. *Exp. Syst. Appl.* **2016**, *65*, 221–232. [CrossRef]
69. Gao, H.; Kwong, S.; Yang, J.; Cao, J. Particle swarm optimization based on intermediate disturbance strategy algorithm and its application in multi-threshold image segmentation. *Inf. Sci.* **2013**, *250*, 82–112. [CrossRef]
70. Cheng, G.; Han, J.; Zhou, P.; Guo, L. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS J. Photogramm. Remote Sens.* **2014**, *98*, 119–132. [CrossRef]