Hierarchical Particle Swarm Optimization Based on Mean Value

Chunfeng Wang, Pengpeng Shang, and Xiaodi Wu

Abstract—**Particle swarm optimization (PSO) has attracted the attention of many scholars due to its outstanding performance. However, PSO has the defects of easily falling into local optimum and low precision. To alleviate these defects, this paper presents a hierarchical particle swarm optimization based on mean value (mHPSO). Firstly, based on the mean value of the population, the whole population is divided automatically into low or high level. Secondly, according to the characteristics of particles at different levels, different speed and position updating strategies are designed, respectively, which are used to balance the global and local search capabilities and improve the accuracy of the solution. Thirdly, a random neighbor selection mechanism is embedded into the update process of the high-level particle to keep the population diversity. Finally, compared with other PSO variants, mHPSO has better performance and faster convergence speed in solving some benchmark test functions with different types. Moreover, by combining mHPSO with Otsu, mPSO also shows good performance in image segmentation.**

Index Terms—**Particle swarm optimization; Population mean; Random neighbor selection mechanism, Image segmentation.**

I. Introduction

In recent years, many complex and high-dimensional optimization problems have appeared in many fields. How to solve these problems efficiently has become a hot research topic. Since the deterministic methods have some limitations in solving complex practical problems, the swarm intelligence optimization algorithms have been highly praised by many scholars. In recent decades, many excellent swarm intelligence algorithms have been proposed, such as artificial bee colony algorithm (ABC) [1-3], firefly algorithm (FA) [4], particle swarm algorithm (PSO) [5], ant colony algorithm (ACO) [6], bat algorithm [7], and so on.

As a member of the swarm intelligence algorithms, P-SO comes from the coordinated and cooperative foraging behaviors of birds in nature. It was proposed by Kennedy and Eberhart in 1995 [8]. Since it was proposed, it has been studied and applied by many scholars. Furthermore, PSO has been successfully applied to sole different problems, such as multi-objective optimization [9], image processing [10], artificial neural network [11] and other fields [12-14].

However, PSO has the disadvantages of premature convergence, low accuracy and slow convergence speed. To overcome these shortcomings, many PSO variants have been proposed. For example, Liang et al. presented a comprehensive learning PSO [15]. In their method, the previous optimal information of other particles were used to update the velocity of each particle. In [16], Zhang et al. proposed a PSO variant with an adaptive learning strategy, which designed different learning strategies for different subgroups. By using adaptive strategy, a modified PSO was described in [17]. In this algorithm,to improve its comprehensive performance, chaos map, stochastic and mainstream learning strategies, adaptive position updating strategy and terminal replacement mechanism were combined. Inspired by natural phenomena, Xia et al. constructed an expanded PSO based on multi-exemplar and forgetting ability [18]. Aiming to choose the proper exemplars and design an efficient learning model for each particle, a triple archives PSO variant was proposed by Xia et al. [19]. By selecting more meaningful individuals as learning samples of particles, Song and Hua presented a multi-exemplar PSO to maintain the population diversity [20]. To well balance the exploration and exploitation, Bo et al. suggested a PSO variant based on multiple adaptive strategies [21]. A heterogeneous comprehensive learning PSO variant was proposed by Nandar et al., in which the population was divided into two groups: one to explore and the other to focus on exploitation [22]. There are also many excellent PSO variants that mix other swarm intelligence algorithms [23-25].

Although PSO has been deeply studied, the defects of premature convergence and low solution accuracy still exist. Aiming to solve these deficiencies, a hierarchical particle swarm optimization based on mean value (mHPSO) is proposed. First, the whole population is stratified by means of their fitness. Each particle belongs to either a high level or a low level. Second, the velocity and position updating strategies are designed respectively for the particles according to the different levels. these two strategies are used to balance the global and local search capabilities and accelerate the convergence. Third, to maintain the diversity of the population, a random neighbor selection mechanism is added into the high-level individuals. On the whole, the proposed algorithm can dynamically adjust the speed and position step size with iterations, which are also more conducive to its convergence.

The structure of the article are arranged as follows: the research status of PSO are presented in Section I. The process of the basic PSO is explained in Section II. Section III introduces the details of mHPSO. The results of numerical experiments and three image segmentation problems are

Manuscript received October 28, 2022; revised March 29, 2023. This work was supported by the National Natural Science Foundation of China(11801430); the Key cultivation project of Xianyang Normal University (XSYK21044); the Nature Science Fundation of Ningxia Province, PR China (2020AAC03237); the Doctoral Scientific Research Foundation of Xianyang Normal University (1052003610).

Chunfeng Wang is a professor of the School of Mathematics and Statistics, Xianyang Normal University, Xianyang, 712000, PR China; College of Mathematics and Information, Henan Normal University, Xinxiang, 453007, PR China, Email:wangchunfeng09@126.com

Pengpeng Shang is a graduate student of the College of Mathematics and Information Science, Henan Normal University, Xinxiang, 453007, PR China. Email: 783977427@qq.com

Xiaodi Wu is a lecturer of the Faculty of Mathematical and Physics, Guangxi University for Nationalities, Nanning, 530006, PR China. Email: 1710703068@qq.com

shown in Section IV, And the conclusion and the next work are given in Section V.

II. Basic PSO algorithm

PSO algorithm is a process in which individuals cooperate and compete with each other. Each particle learns the successful experience from itself and the entire population, and finally searches the global optimum. In PSO, each particle represents to a solution in the Euclidean space, and the fitness value is the objective function value. Like most swarm intelligence algorithms, the initial population (SN) in PSO is generated randomly in the feasible search space. At tth iteration, let $x_i^t = \{x_{i1}, x_{i2}, \cdots, x_{iD}\}$ and $v_i^t = \{v_{i1}, v_{i2}, \cdots, x_{iD}\}$ be the position vector and velocity vector of ith particle, respectively. D represents the dimension of the search space. Then, its position and velocity are updated according to the following formulas:

$$v_i^{t+1} = w^t * v_i^t + c1 * r1 * (pbest_i^t - x_i^t) + c2 * r2 * (gbest_i^t - x_i^t), \quad (1)$$

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

where $i \in \{1, 2, \cdots, SN\}$; $c1$ and $c2$ are two learning factors, which are usually two fixed values; $r1$ and $r2$ are two random numbers in [0,1]; $pbest_i$ is the historical optimal individual of the ith particle; $gbest_i$ represents the best individual of the current population. w, called inertia weight factor, controls how much the particles inherit from the current speed and has the ability to balance global and local search of the algorithm. In this paper, to dynamically adjust the velocity, the inertia weight factor that decreases linearly with the number of iterations is used [26]:

$$w^t = w_{max} - \frac{w_{max} - w_{min}}{MaxDt} * t, \quad (3)$$

where w_{max} and w_{min} represent the preset maximum and minimum inertia weight values, respectively; $MaxDt$ is maximum number of the iterations.

III. Hierarchical particle swarm optimization based on mean value (mHPSO)

Although the basic PSO can obtain the optimal solution in theory, it often performs poorly when dealing with some practical problems with limitations [27]. To improve the performance of PSO, this paper proposes a method mPSO. In mPSO, the population is stratified according to the mean value of the population firstly. And then, a new velocity and a position update formulas for individuals at different levels are designed. Finally, a random neighbor selection mechanism is embedded into the high level particle velocity update formula to maintain population diversity.

A. Hierarchical population based on mean value

Each particle in the population has its own characteristics. To highlight the advantage of the high-quality individuals and accelerate the convergence of the ordinary individuals, this paper divides the individuals according to the mean value of the whole population. At tth iteration, the mean value of the population is defined as follows:

$$Mean(t) = \frac{\sum_{i=1}^{N} F(x_i)}{SN}, \quad (4)$$

where $F(x_i)$ is the fitness value of the ith particle. For minimization problems, the high-level particles are those whose fitness values are less than $Mean(t)$, and the low-level particles are those whose fitness values are greater than $Mean(t)$. The opposite is true for maximization problems.

Generally, the high-level particles have a greater chance to search for the global optimum during the evolution process, and the successful experience of the population is mostly derived from them. However, when dealing with the complex high-dimensional multimodal optimization problems, overly aggressive particles tend to sacrifice population diversity and increase the risk of falling into local optimum. Therefore, strengthening the information exchange with other individuals may increase the opportunity to get rid of oscillation. To achieve this goal, the good neighbors can be selected as the individuals of information exchange. By this way, particles can not only learn good information from them, but also maintain the diversity of the population to a certain extent. For the low-level particles, although the population diversity has been maintained, they greatly slow down the convergence speed of the algorithm. In addition, as pointed out in [28], the dynamic step size may be more suitable for the movement and update of the particles.

Based on the above analysis, we design different update strategies for particles at different levels. At tth iteration, the details of particles movement are given as follows:

Case 1: $F(x_i) \leq Mean(t)$. Under this condition, the individual i is a high-level particle, which is updated and moved by Eq. (5) and Eq. (6) below:

$$\begin{aligned} v_i^{t+1} = {} & w^t * v_i^t + c1 * r1 * (pbest_i^t - x_i^t) \\ & + c2 * r2 * (gbest^t - x_i^t) + \phi * r3 * (x_n^t - x_i^t), \end{aligned} \quad (5)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \quad (6)$$

where x_n is a randomly neighbor selected from $\{F(x_j) < F(x_i) | j \in \{1, 2, \cdots, SN\}\}$; ϕ is a preset acceleration factor to control the exchange of information with neighbors; $r3$ is a random number in [0,1]. The addition of excellent neighbor not only ensures that particle i learn from other excellent individuals, but also increase the possibility of jumping out of local optimum and maintain population diversity.

Case 2: $F(x_i) > Mean(t)$. In this case, it means that particle i is a low-level individual. According to its characteristic, the following update rules is presented:

$$\begin{aligned} v_i^{t+1} = {} & w^t * v_i^t + c1 * r1 * (pbest_i^t - x_i^t) \\ & + c2 * r2 * (gbest^t - x_i^t), \end{aligned} \quad (7)$$

$$x_i^{t+1} = (1 - w^t) * x_i^t + w^t * v_i^{t+1}, \quad (8)$$

where w^t is consistent with the above. The original velocity update ensures that the low-level particles can fly globally as much as possible to explore the entire feasible space. However, it cannot reasonably allocate computing resources in PSO, so it will slow down the convergence speed. By referring to [14], we propose a position movement equation as shown in Eq. (8). From the definition of w^t, we can see that it has a greater value at earlier stage, which means the velocity v_i^{t+1} is valued by the low-level individuals to achieve the purpose of expanding the flight range. As the iteration progresses, w^t gradually decreases, which means that particles pay more attention to position x_i^t. Thus, it will enhance the local search ability and solution accuracy.

From the above analysis, we can learn that using different strategies for individuals with different characteristics may help the algorithm achieve better results.The pseudo-code of the proposed algorithm mHPSO is given as follows:

Algorithom 1. mHPSO

01. Initialize population SN and set the maximum number of iterations $MaxDt$.
02. Compute the fitness of $\{F(x_i)|i = 1, \cdots, SN\}$ and determine $pbest_i^t$ and $gbest^t$.
03. While $t \leq MaxDt$ **do**
04. **For** $i = 1$ **to** SN **do**
05. Compute the population mean $Mean(t)$.
06. **If** $F(x_i) \leq Mean(t)$
07. Randomly find a high quality neighbor x_n and update x_i by (5) and (6).
08. **Else** $F(x_i) > Mean(t)$
09. update x_i by (7) and (8).
10. **End if**
11. Update $pbest_i^t$ and $gbest^t$.
12. **End for**
13. t=t+1.
14. End While

IV. Experiments analysis

To comprehensively verify the performance of mPSO, 16 different types of benchmark functions are selected for comparative experiments [29-30]. In these benchmark functions, f_1-f_6 are unimodal functions; f_7-f_{12} are complex multimodal functions; f_{13}-f_{16} are rotation and translation functions. And the details of these functions are shown in Table I. All the experiments are coded on Matlab R2017a and executed on a computer with an Intel (R) Core (TM) i5-3250M CPU @ 2.60 GHz, 4 GB memory, Windows 7 system.

Several PSO variants are used to compare with mPSO on these benchmark functions, which include PSO [8], CLPSO [15] and MPSO [16]. For the sake of fairness, $SN = 50$, $D = 30$, $MaxDt = 2000$ are used on all these comparison algorithms. In this paper, the acceleration factors $c1 = c2 = 1.49618$ and flight speed $v \in 0.2 * [x_{min}, x_{max}]$. The other parameters of PSO, CLPSO and MPSO are consistent with the original literatures. All these algorithms are independently run 30 times on each test function, and the minimum (Min), mean (Mean) and standard deviation (Std) of the 30 experimental results are counted as the comparison indicators. The experiments are divided into three parts: A is the sensitivity test of ϕ; B is the contrast experiment of PSO variants; C is the experiment on multithreshold image segmentation.

A. Sensitivity test: ϕ

In mPSO, ϕ indicates that how much knowledge is learned from the high-quality neighborhood, which directly affects the convergence speed and solution stability of the algorithm. So, it is critical to determine a reasonable ϕ. To this end, the values of $\phi \in \{0.5, 0.75, 1.0, 1.25, 1.5\}$ run on all the benchmark functions, respectively. The Min and Mean results obtained by mPSO with different ϕ are given in Table II.

From Table II, we can see that ϕ has little influence on f_7, f_8, f_9, f_{10} and f_{14}, because their optimal values are obtained. In addition, it is obvious that the solution accuracy of the objective function decreases with the increase of ϕ except for f_{15} and f_{16}. Although mPSO has the best performance with $\phi = 0.5$, it is inferior to on Mean when $\phi = 0.75$. $\phi = 1.5$ perform well on most functions in terms of Mean, especially on f_5 and f_6, but its solution accuracy is not satisfactory. Consequently, after comprehensive consideration, $\phi = 0.75$ is adopted in this paper.

B. Comparative experiment of PSO variants

In this subsection, different PSO variants are tested on all benchmark functions. The obtained Min, Mean and Std of each algorithm by running 30 times on each function, which are displayed in Table III and the optimums are shown in bold. Meanwhile, to illustrate the convergence of these algorithms more intuitively, Figures 1 and 2 depict the convergence curves of each algorithm.

From Table III, in terms of Min, mHPSO outperforms far better than its competitors in most functions, but it is inferior to CLPSO on f_5, f_{12} and f_{15}. Both MPSO and mHPSO can obtain the optimum values on f_7, f_8, f_9 and f_{14}. The statistical results of the Min and Mean winning rates of mHPSO are 75% and 56.25%, respectively, which are the highest among all these algorithms. From Figures 1 and 2, on f_1, f_4, f_6, f_{13} and f_{16}, mHPSO shows good ability to jump out of the local optimum while PSO, CLPSO and MPSO are all trapped in the local optimum. All the above indicates that mPSO is more effective.

C. Experiment on image segmentation

In digital image processing, image segmentation is a key step, which is to segment the image into several non overlapping regions with the same features in the region but different features between regions according to certain segmentation rules [31]. Threshold segmentation is a very popular method in image segmentation, which has been widely studied and applied by many researchers due to its simple calculation and high efficiency [32]. The selection of threshold is very important in the threshold segmentation method, which determines the quality of the final segmentation results. Otsu method is a typical threshold selection method [33]. However, the operation time and calculation complexity of Otsu will increase exponentially with the increase of thresholds. Therefore, it is very meaningful to apply swarm intelligence optimization algorithm to threshold screening and select the thresholds that can optimally segment the image [34-35]. In this paper, to improve the effect of image segmentation, the proposed algorithm mHPSO is combined with Otsu.

In order to test the effective of our method, several classic images from standard image library are used, which include Cameraman, Plane and Lena. The other image details, the results of one-threshold and multi-threshold image segmentation obtained by Otsu-mHPSO are shown in Figures 3, 4 and 5. Meanwhile, to verify performance of Otsu-mHPSO,

TABLE I: Benchmark test functions

Test functions	Range	Optimal
$f_1 = \sum_{i=1}^{D} x_i^2$	[-100,100]	0
$f_2 = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	[-10,10]	0
$f_3 = \sum_{i=1}^{D} \lvert x_i\rvert^{i+1}$	[-1,1]	0
$f_4 = \sum_{i=1}^{D} 10^{6\frac{i-1}{D-1}} x_i$	[-100,100]	0
$f_5 = -\exp(-0.5 * \sum_{i=1}^{D} x_i^2)$	[-1,1]	-1
$f_6 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	[-100,100]	0
$f_7 = \sum_{i=1}^{D} (x_i^2 - 10cos(2\pi x_i) + 10)$	[-5.12,5.12]	0
$f_8 = 20 + e - 20\exp(-0.2 * \sqrt{\sum_{i=1}^{D} x_i^2/D}) - \exp(\sum_{i=1}^{D} cos(\frac{2\pi x_i}{D}))$	[-32,32]	0
$f_9 = \frac{1}{4000}\sum_{i=1}^{D} x_i - \prod_{i=1}^{D} cos\frac{x_i}{\sqrt{i}} + 1$	[-600,600]	0
$f_{10} = \sum_{i=1}^{D} (y_i^2 - 10cos(2\pi y_i) + 10), \begin{cases} y_i = x_i, & \lvert x_i\rvert < \frac{1}{2} \\ y_i = \frac{\lfloor 2x_i \rfloor}{2}, & \lvert x_i\rvert \geq \frac{1}{2} \end{cases}$	[-50,50]	0
$f_{11} = 0.5 + \frac{\sin(\sqrt{\sum_{i=1}^{D} x_i^2})^2 - 0.5}{(1+0.01\sum_{i=1}^{D} x_i^2)^2}$	[-100,100]	0
$f_{12} = \frac{1}{D}[10\sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2(1 + 10\sin^2(\pi y_i + 1) + (y_D - 1)^2)]$ $+ \sum_{i=1}^{D} u(x_i, 10, 100, 4), where\ y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	[-50,50]	0
$f_{13} = \sum_{i=1}^{D} z_i^2, z = x * M$	[-500,500]	0
$f_{14} = \frac{1}{4000}\sum_{i=1}^{D} z_i - \prod_{i=1}^{D} cos\frac{z_i}{\sqrt{i}} + 1, z = x * M$	[-600,600]	0
$f_{15} = \sum_{i=1}^{D} (z_i^2 - 10cos(2\pi z_i) + 10) - 330, z = x - o$	[-5.12,5.12]	-330
$f_{16} = \sum_{i=1}^{D} (\sum_{j=1}^{i} z_j)^2 - 450, z = x - o$	[-100,100]	-450

the image results of segmentation obtained by the original Ostu method and the Otsu method combined with the basic PSO (Otsu-PSO) are compared with that of mPSO. The comparison results are given in Table IV. The maximum inter class variance is used as the function value to evaluate the effect of optimal threshold. The higher the value, the better the segmentation effect. Running time represents the time consumed by each algorithm during image segmentation. The longer the time, the higher the computational complexity of the algorithm. In this paper, $SN = 100$, $MaxDt = 100$, $v \in [-2.5, 2.5]$ and $c1 = c2 = 1.49618$ are used in Otsu-mHPSO and Otsu-PSO, and the operating environment is the same as numerical experiments.

In Table IV, we can see that the function value of Otsu-mHPSO is equal to Otsu but better than Otsu-PSO in these three segmentation situations, which means that our proposed algorithm can obtain appropriate thresholds and perform better than basic PSO. In terms of Running time, as the number of thresholds increases, the running time of all the algorithms increase. It is obvious that Otsu spends far more time than Otsu combined with intelligent algorithms. However, Otsu-mHPSO takes slightly more time than Otsu-PSO, which may be caused by the different particle update strategies. Meanwhile, from Figures 3, 4 and 5, we can see that with the increase of the number of thresholds, the segmented images using our approach are significantly better. All the above results show that mHPSO is better than the comparison algorithms in image segmentation.

V. Conclusion

A hierarchical particle swarm optimization algorithm based on mean (mHPSO) is presented in this paper. After the population is stratified according to the mean value, we designed two types of adaptive strategies for the particles with different characteristics to improve the performance of PSO. To maintain population diversity, the high quality neighbor selection mechanism was integrated into the high-level particles. Furthermore, the numerical experiments proved that mHPSO is superior to its comparison algorithms. By comparing Otsu and Otsu-PSO, our method can obtain effective thresholds in image segmentation, which verified its potential application in image processing. In future work, we will further study and apply it to more practical problems.

References

[1] D. Karaboga, B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problem", *Journal of Global Optimization*, vol. 4529, pp. 789-798, 2007.

[2] C.F. Wang, Y.H. Zhang, "An improved artificial bee colony algorithm for solving optimization problems", *IAENG International Journal of Computer Science*, vol. 43, no. 3, pp. 336-343, 2016.

[3] C.F. Wang, P.P. Shang, L.X. Liu, "Improved artificial bee colony algorithm guided by experience", *Engineering Letters*, vol. 30, no. 1, pp. 261-265, 2022.

TABLE II: Different ϕ

Test functions	Indicator	$\phi = 0.5$	$\phi = 0.75$	$\phi = 1.0$	$\phi = 1.25$	$\phi = 1.5$
f_1	Min	5.56E-219	2.16E-202	3.81E-183	7.79E-160	6.73E-137
	Mean	3.56E-156	1.46E-172	1.13E-170	3.03E-153	2.75E-130
f_2	Min	4.21E-127	1.33E-120	3.88E-112	1.59E-104	1.95E-91
	Mean	3.23E-29	3.36E-27	3.70E-89	1.07E-69	1.31E-54
f_3	Min	6.07E-269	4.08E-261	2.64E-242	3.23E-207	3.27E-178
	Mean	1.58E-174	2.79E-163	6.43E-17	5.64E-22	3.87E-20
f_4	Min	3.40E-213	1.85E-196	1.07E-176	4.51E-146	1.01E-127
	Mean	9.40E+04	1.14E+04	5.31E+03	5.16E+03	8.93E+03
f_5	Min	-9.999E-01	-9.999E-01	-9.998E-01	-9.998E-01	-9.999E-01
	Mean	-9.995E-01	-9.995E-01	-9.995E-01	-9.995E-01	-9.997E-01
f_6	Min	2.34E-213	5.79E-193	1.34E-176	2.88E-156	4.13E-130
	Mean	4.148E+01	3.561E+01	2.052E+01	1.096E+01	1.31E-19
f_7	Min	0	0	0	0	0
	Mean	4.957E+01	3.569E+01	2.981E+01	2.359E+01	1.396E+01
f_8	Min	-8.88E-16	-8.88E-16	-8.88E-16	-8.88E-16	-8.88E-16
	Mean	1.72E-15	5.92E-17	7.70E-16	1.01E-15	1.13E-15
f_9	Min	0	0	0	0	0
	Mean	0	0	0	0	0
f_{10}	Min	0	0	0	0	0
	Mean	5.641E+01	9.303E+01	3.37E+01	1.034E+01	4.06E-13
f_{11}	Min	9.7E-03	1.14E-13	9.7E-03	9.7E-03	9.7E-03
	Mean	6.239E-02	9.3E-02	5.87E-02	2.67E-02	2.6E-02
f_{12}	Min	1.69E-02	1.1E-02	2.2E-03	2.7E-03	4.5E-03
	Mean	1.19E-02	8.04E-02	3.13E-02	2E-02	1.45E-02
f_{13}	Min	1.40E-217	5.35E-202	1.24E-179	7.68E-160	2.00E-135
	Mean	2.88E-190	7.38E-185	3.25E-174	1.08E-151	3.04E-130
f_{14}	Min	0	0	0	0	0
	Mean	0	0	0	0	0
f_{15}	Min	-246.5599	-261.289	-283.7745	-288.604	-286.0836
	Mean	-177.9537	-211.7176	-235.5671	-2.40E+02	-260.1992
f_{16}	Min	377.6325	-118.2399	-30.2941	-169.8491	-219.6242
	Mean	9.46E+03	4.79E+03	3.22E+03	2.86E+03	1.80E+03

[4] P. Hu, W.H. Zhu, et al. "Enhancing firefly algorithm with courtship learning", *Information Sciences* , vol. 543, pp. 18-42, 2021.

[5] C.F. Wang, W.X. Song, "A modified particle swarm optimization algorithm based on velocity updating mechanism", *Ain Shams Engineering Journal*, vol. 10, no. 4, pp. 847-866, 2019.

[6] M. Dorigo, S. Thomas, "Ant Colony Optimization: Overview and Recent Advances", *International Series in Operations Research & Management Science*, vol. 146, no. 0, pp. 227-263, 2010.

[7] C.F. Wang, W.X. Song, L.X. Liu, "An adaptive bat algorithm with memory for global optimization", *IAENG International Journal of Computer Science*, vol. 45, no. 2, pp. 320-327, 2018.

[8] J. Kennedy, R. Eberhart, "Particle swarm optimization", *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4, pp. 1942-1948, 1995.

[9] M. Reyes-Sierra, C.C. Coello, "Multi-objective particle swarm optimizers: A survey of the state-of-the-art", *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 287-308, 2006.

[10] S. Yadav, A. Ekbal, et al. "Feature selection for entity extraction from multiple biomedical corpora: A PSO-based approach", *Soft Computing*, vol. 22, no. 20, pp. 6881-6904, 2018.

[11] F. Wang, H.Q. Zhu, et al. "A hybrid convolution network for serial number recognition on banknotes", *Information Sciences*, vol. 512, pp. 952-963, 2020.

[12] M.R. Bonyadi, "A theoretical guideline for designing an effective adaptive particle swarm", *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 57-68, 2020.

[13] Y.F. Zhang, H.D. Chiang, "A novel consensus-based particle swarm optimization-assisted trust-tech methodology for large-scale global",*IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2717-2729, 2017.

[14] Y.D. Zhang, S.H. Wang, "Binary PSO with mutation operator for feature selection using decision tree applied to spam detection.", *Knowledge-Based Systems* , no. 1, pp. 22-31, 2014.

[15] J.J. Liang, A.K. Qin, et al. "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281-295, 2006.

[16] Y.F. Zhang, X.X. Liu, et al. "Particle swarm optimization with adaptive learning strategy", *Knowledge-Based Systems*, vol. 196, pp. 105789, 2020.

[17] H. Liu, X.W. Zhang, L.P. Tu, "A modified particle swarm optimization using adaptive strategy", *Expert Systems with Applications*, vol. 152, pp. 113353, 2020.

[18] X.W. Xia, L. Gui, et al. "An expanded particle swarm optimization based on multi-exemplar and forgetting ability", *Information Sciences*, vol. 508, pp. 105-120, 2020.

[19] X.W Xia, L. Gui, et al. "Triple archives particle swarm optimization", *IEEE Transactions on Cybernetics*, vol. 50, no. 12, pp. 4862-4875, 2019.

[20] W. Song, Z.Y. Hua, "Multi-Exemplar Particle Swarm Optimization", *IEEE Access*, vol. 8, pp. 176363-176374, 2020.

[21] W. Bo, X.W. Xia, et al. "Multiple adaptive strategies based particle swarm optimization algorithm", *Swarm and Evolutionary Computation*, vol. 57, pp. 100731, 2020.

[22] N. Lynn, P.N. Suganthan, "Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation", *Swarm and Evolutionary Computation*, vol. 24, pp. 11-24, 2015.

[23] S.H Wang, Y.Z. Li, H.Y. Yang, "Self-adaptive mutation differential evolution algorithm based on particle swarm optimization", *Applied Soft Computing*, vol. 81, pp. 105496, 2019.

[24] K. Chen, F.Y. Zhou, et al. "A hybrid particle swarm optimizer with sine cosine acceleration coefficients", *Information Sciences*, vol. 422, pp. 218-241, 2018.

[25] J. Vinita, B. Punam, "An improved hybrid ant particle optimization (IHAPO) algorithm for reducing travel time in VANETs", *Applied Soft Computing*, vol. 64, pp. 526-535, 2018.

[26] W.B. Liu, Z.D. Wang, et al. "A Novel Randomised Particle Swarm Optimizer", *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 2, pp. 529-540, 2021.

[27] F. van den Bergh, A.P. Engelbrecht, "A study of particle swarm optimization particle trajectories", *Information Sciences*, vol. 176, no. 8, pp. 937-971, 2006.

[28] R. Wang, K.R. Hao, et al. "A Novel Hybrid Particle Swarm Optimization Using Adaptive Strategy", *Information Sciences*, vol. 579, pp. 231-250, 2021.

[29] L.Z. Cui, G.H. Li, et al. "A ranking-based adaptive artificial bee colony

TABLE III: The results obtained by PSO variants

Test function	Indicator	PSO	CLPSO	MPSO	mHPSO
f_1	Min	6.4522	7.70E-04	1.01E+03	**3.27E-203**
	Mean	30.2303	0.0023	1.43E+03	**1.98E-195**
	Std	17.9568	8.12E-04	786.7879	**0**
f_2	Min	0.7226	0.0045	1.79E-76	**2.59E-120**
	Mean	3.9578	0.0092	**4.63E-72**	1.49E-34
	Std	2.952	0.0022	**1.48E-71**	8.15E-34
f_3	Min	2.95E-11	5.17E-24	**0**	2.49E-257
	Mean	1.95E-08	3.27E-22	**0**	6.57E-23
	Std	2.59E-08	7.96E-22	**0**	3.60E-22
f_4	Min	1.28E+05	1.9975	4.64E+06	**7.92E-186**
	Mean	3.66E+06	**6.8485**	1.46E+07	3.97E+03
	Std	9.95E+06	**2.3654**	1.18E+07	1.43E+04
f_5	Min	-0.9998	**-1**	**-1**	-0.9998
	Mean	-0.9986	**-1**	-0.9002	-0.9995
	Std	9.35E-04	**5.23E-08**	0.1885	4.24E-04
f_6	Min	237.2076	3.45E+03	1.00E+03	**2.25E-199**
	Mean	810.5094	4.75E+03	3.08E+03	**46.9321**
	Std	517.2023	766.1018	1.47E+03	**137.7649**
f_7	Min	30.7979	4.6738	**0**	**0**
	Mean	52.4272	8.7061	**0.9992**	36.8705
	Std	13.0013	**1.566**	3.7275	27.6008
f_8	Min	3.1533	0.0285	**-8.88E-16**	**-8.88E-16**
	Mean	5.6495	0.0462	2.31E-15	**7.70E-16**
	Std	1.2155	0.0101	1.23E-15	**1.80E-15**
f_9	Min	1.0528	0.0038	**0**	**0**
	Mean	1.2232	0.0151	**0**	**0**
	Std	0.1474	0.006	**0**	**0**
f_{10}	Min	201.0938	14.0237	**0**	**0**
	Mean	380.0556	20.153	235.2213	**8.8813**
	Std	115.8384	**2.7146**	374.8992	16.4979
f_{11}	Min	0.2727	0.0793	**0.0097**	**0.0097**
	Mean	0.3793	0.1365	0.086	**0.0758**
	Std	0.0438	0.0245	**0.0893**	0.1686
f_{12}	Min	0.908	**1.35E-05**	0.0067	0.0064
	Mean	2.5016	**3.14E-05**	86.2644	0.0779
	Std	1.3098	**1.08E-05**	472.389	0.0628
f_{13}	Min	149.523	0.0316	9.74E+03	**7.65E-203**
	Mean	545.5113	0.0597	4.00E+04	**6.14E-194**
	Std	244.1802	0.0175	1.74E+04	**0**
f_{14}	Min	1.0739	0.0235	**0**	**0**
	Mean	1.2435	0.0498	**0**	**0**
	Std	0.184	0.0147	**0**	**0**
f_{15}	Min	-246.1994	**-327.1845**	-259.3573	-261.7201
	Mean	-185.6635	**-323.0636**	-176.6674	-203.3313
	Std	28.5113	**2.2891**	34.6393	27.7229
f_{16}	Min	316.5411	5.68E+03	3.39E+04	**-82.6397**
	Mean	7.20E+03	1.00E+04	8.28E+04	**4.98E+03**
	Std	7.01E+03	6.71E+03	3.50E+04	**6.13E+03**
Winning Rate	Min Winning	0%	18.75%	50%	**75%**
	Meang Winning	0%	25 %	31.25 %	**56.25 %**

algorithm for global numerical optimization", *Information Sciences*, vol. 417, pp. 169-185, 2017.

[30] C.F. Wang, P.P. Shang, P.P. Shen, "An improved artificial bee colony algorithm based on Bayesian estimation", *Complex & Intelligent Systems*, vol. 417, pp. 169-185, 2022.

[31] R.P. Nikhil, K.P. Sankar, "A review on image segmentation techniques", *Pattern Recognition*, vol. 26, no. 9, pp. 1277-1294, 1993.

[32] M. Sezgin, B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation", *Journal of Electronic Imaging* , vol. 13, no. 1, pp. 146-165, 2004.

[33] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems*, vol. 9, no. 1, pp. 62-66, 1979.

[34] C.Q. Wang, J.P Yang, et al. "Otsu multi-threshold image segmentation algorithm based on improved particle swarm optimization",*IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP)*, pp. 440-443, 2019.

[35] S.A. Ludwig, "Improved glowworm swarm optimization algorithm applied to multi-level thresholding",*2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1533-1540, 2016.

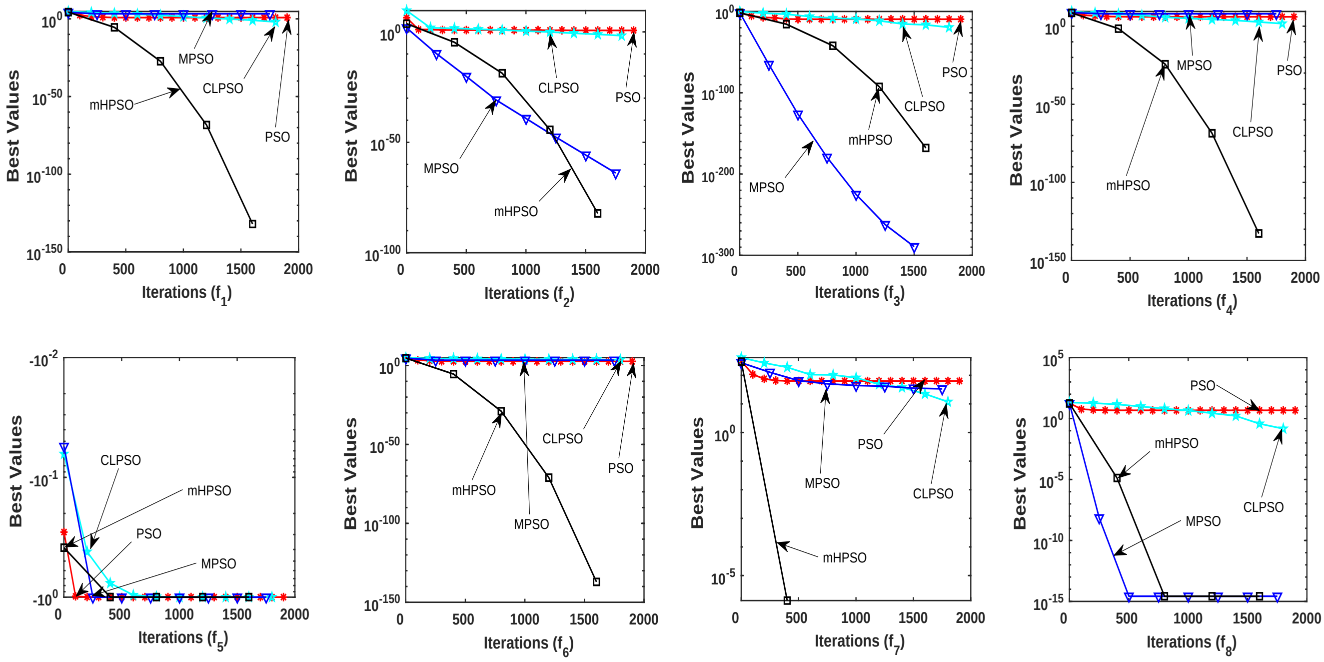

Figure 1: Convergence curves of f_1-f_8

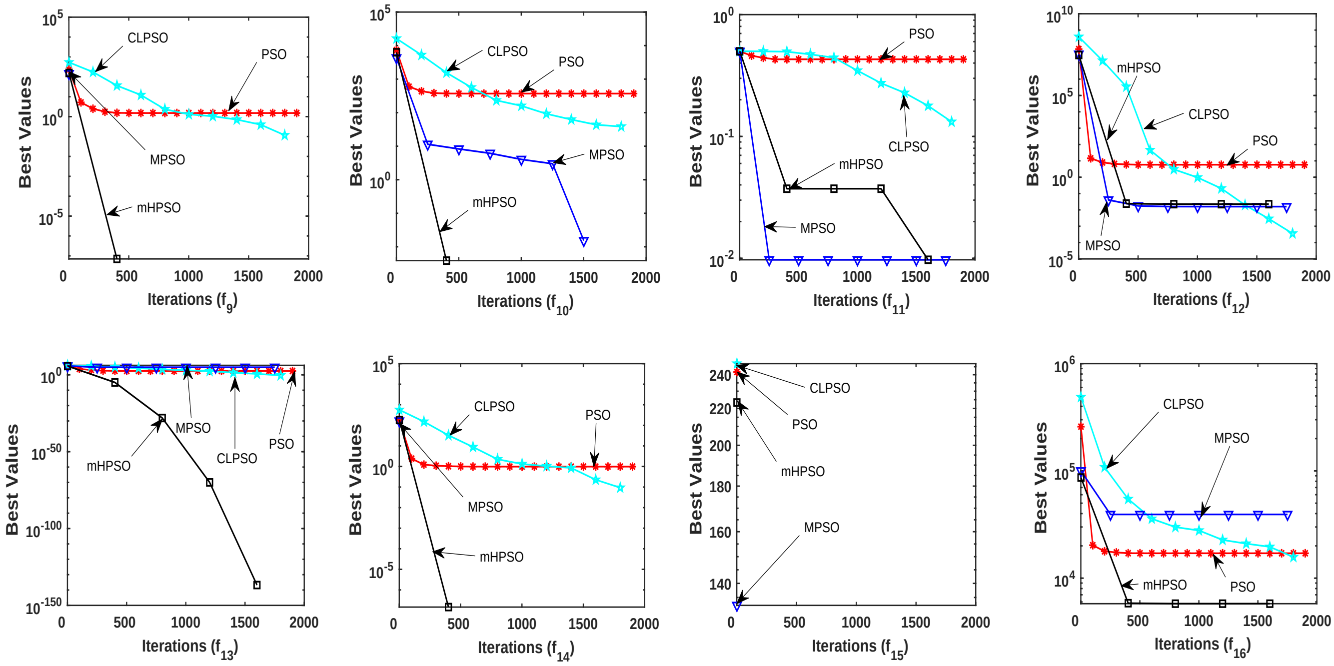

Figure 2: Convergence curves of f_9-f_{16}

TABLE IV: Segmentation thresholds and optimal functions value from three Otsu methods

Figures	Thresholds number	Segmentation thresholds			Function values			Running time (s)		
		Otsu	Otsu-PSO	Otsu-mHPSO	Otsu	Otsu-PSO	Otsu-mHPSO	Otsu	Otsu-PSO	Otsu-mHPSO
Cameraman	1	87	85	85	3272.6	3272.6	3272.6	0.2316	0.0765	0.0925
	2	70,144	67,144	68,141	3653.2	3653.1	3653.2	33.3242	0.0872	0.1043
	3	45,101,149	45,104,147	65,133,168	3730.4	3726.4	3730.4	3433.7523	0.0947	0.1137
37073	1	72	70	70	650.2291	650.2291	650.2291	0.4103	0.1084	0.1191
	2	71,141	73,140	70,139	721.1367	721.0163	721.1367	59.7271	0.1182	0.1372
	3	53,88,143	53,90,143	51,87,141	771.9984	771.9662	771.9984	5758.2963	0.1268	0.1479
Lena	1	117	115	115	1521.0	1521.0	1521.0	0.0821	0.0542	0.0581
	2	93,150	92,151	95,151	1861.1	1861.1	1861.1	9.4025	0.0651	0.0749
	3	81,125,169	81,126,171	83,123,169	2019.2	2019.0	2019.2	771.2325	0.0734	0.0884

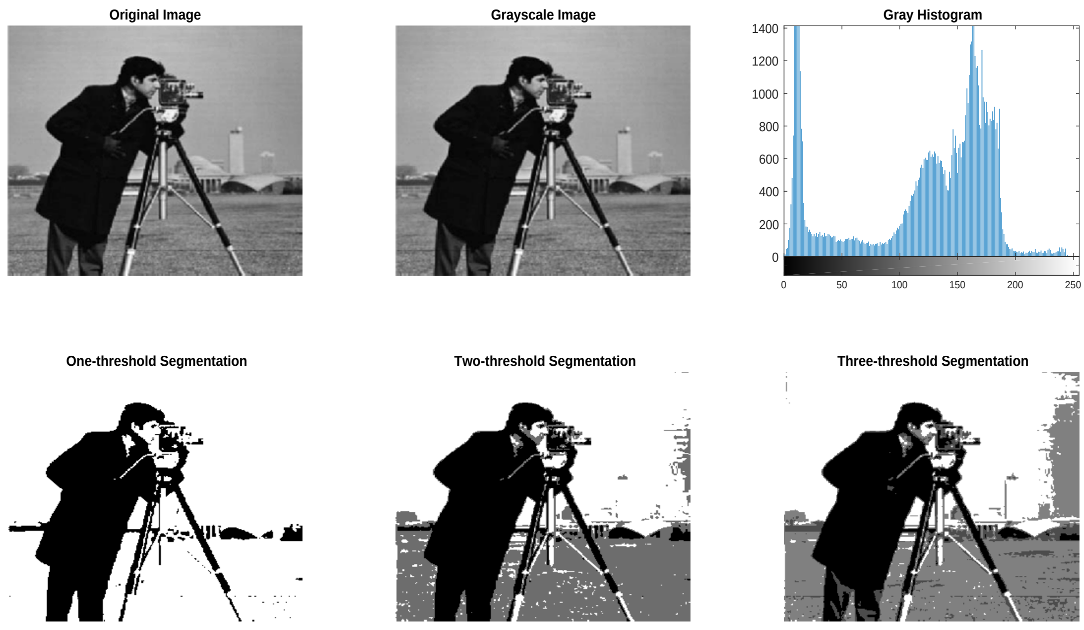

Figure 3: Cmeraman

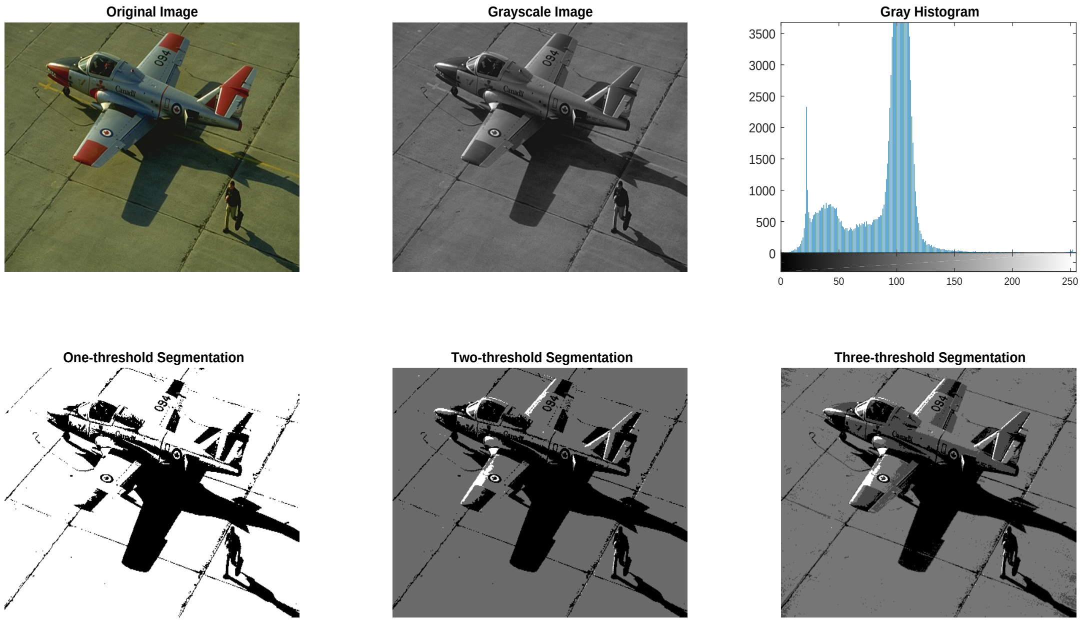

Figure 4: Plane

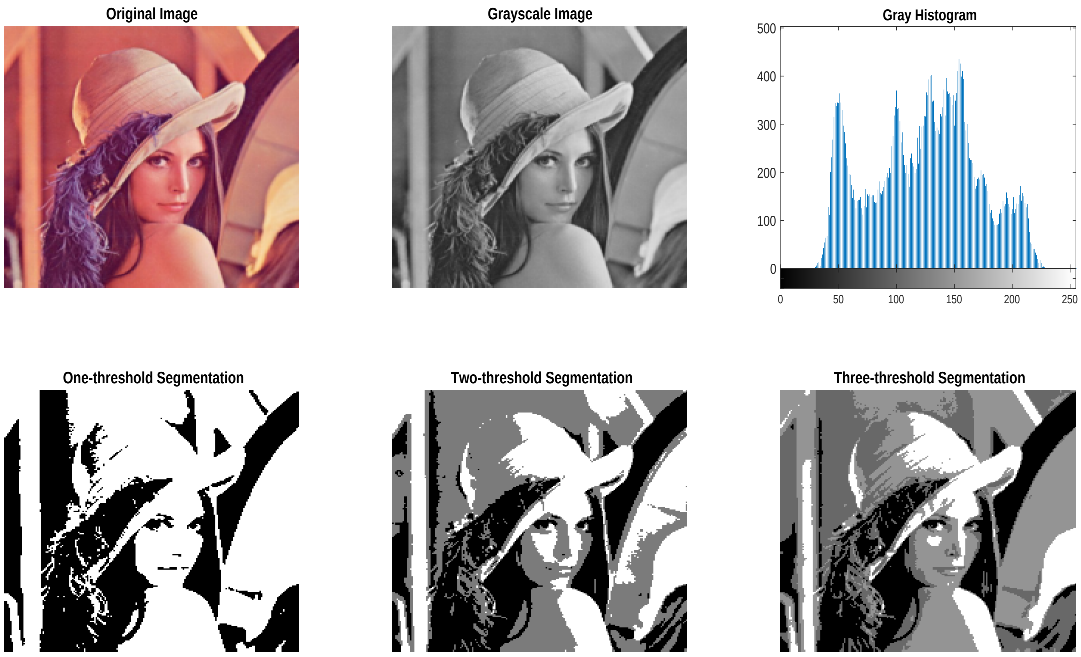

Figure 5: Lena