S. Mirjalili, A. Lewis, A. S. Sadiq, Autonomous Particles Groups for Particle Swarm Optimization, *Arabian Journal for Science and Engineering*, vol. 39, no. 6, pp 4683-4697, DOI: http://dx.doi.org/10.1007/s13369-014-1156-x

Autonomous Particles Groups for Particle Swarm Optimization

Seyedali Mirjalili [b] , Andrew Lewis [b] , Ali Safa Sadiq [c]

[a] School of Information and Communication Technology, Griffith University, Nathan, Brisbane, QLD 4111, Australia
[b] Faculty of Computer Science and Information Systems Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia

seyedali.mirjalili@griffithuni.edu.au, a.lewis@griffith.edu.au, alisafa09@gmail.com

Abstract: In this paper a modified Particle Swarm Optimization (PSO) algorithm called Autonomous Groups Particles Swarm Optimization (AGPSO) is proposed to further alleviate the two problems of trapping in local minima and slow convergence rate in solving high dimensional problems. The main idea of AGPSO algorithm is inspired by individuals' diversity in bird flocking or insect swarming. In natural colonies, individuals are not basically quite similar in terms of intelligence and ability, but they all do their duties as members of a colony. Each individual's ability can be useful in a particular situation. In this paper a mathematical model of diverse particles groups called autonomous groups is proposed. In other words different functions with diverse slopes, curvatures, and interception points are employed to tune the *social and cognitive* parameters of the PSO algorithm to give particles different behaviors as in natural colonies. The results show that PSO with autonomous groups of particles outperforms the conventional and some recent modifications of PSO in terms of escaping local minima and convergence speed. The results also indicate that dividing particles in groups and allowing them to have different individual and social thinking can improve the performance of PSO significantly.

Keywords: PSO; Social behavior; Social coefficient; Cognitive coefficient; Function optimization; Autonomous Particles Groups

1 Introduction

Particle Swarm Optimization (PSO) is one of the most widely used evolutionary algorithms inspired by the social behavior of animals [1, 2]. The simplicity and inexpensive computational cost make this algorithm very popular. Due to these advantages, PSO has been applied to many domains such as medical detection [3], grid scheduling [4], robot path planning [5], video abstraction [6], optical buffer design [7, 8], and Neural Networks [9, 10]. In spite of these advantages, trapping in local minima and slow convergence rate are two unavoidable problems [11, 12]. These two problems deteriorate with increased problem dimensionality.

There are many methods in the literature to combat these problems. Some of them focus on the hybridization of PSO with other algorithms such as PSO-Genetic Algorithm (GA) [13], PSO-Gravitational Search Algorithm (GSA) [10, 14], and PSO-Ant Colony Optimization (ACO) [15]. Some studies manipulate the interaction neighborhood topology of PSO to do this [16-18]. Regardless of their promising results, increased computational cost is the main problem of these methods.

Using dynamic parameter tuning is a method that increases the performance of PSO without suffering from high computational cost [19-24]. The main parameters of PSO are the weighting factor (w), cognitive coefficient (c_1) and social coefficient (c_2). The similarity of these approaches is that the parameters are tuned with the same strategy for all particles. Therefore, all the particles follow the same pattern in their social and individual behaviors. In other words, the particles are obliged to search without any self-determination and intelligence. In this paper, we propose a new approach of utilizing autonomous groups to give particles a sort of independence with the purpose of increasing performance.

The rest of the paper is organized as follows: Section 2 describes the related works. Section 3 discusses the basic principles of the PSO algorithm. The proposed method is explained in Section 4. The experimental results are demonstrated in section 5. Finally, Section 6 concludes the work and suggests some directions for future research.

2 Related works

In order to improve the PSO algorithm's performance, recently some modified algorithms have been proposed. In 2009, Cai [19] proposed a new modified PSO based on the black stork foraging process. He defined two types of particles inspired from the foraging behavior of adult and infant black storks. These two types of particles have different cognitive coefficients (c_1) that are a function of best fitness value in the current iteration, worst value in the entire swarm, and current fitness values. The results show that the modified PSO has better performance than the conventional PSO when dealing with high-dimensional, multi-modal optimization problems.

Cai *et al.* also proposed a new setting for the social factor (c_2) to improve the convergence speed [20, 21]. The social coefficient is a function of the best fitness value in the current iteration, the worst fitness value in the entire swarm, and the current fitness value. This method can be considered as a PSO algorithm with *N* different particles in terms of following social consensus. This algorithm suffers from trapping in local minima more than the conventional PSO. For this reason the authors equipped the algorithm with a mutation strategy.

There are some studies which have used time-varying coefficients for both cognitive and social coefficients. In 2009, Ziyu and Dingxue [22] introduced an exponentially time-varying acceleration function for adjusting both cognitive and social coefficients in order to control the global search ability and convergence to the global best solution. In 2009, Bao and Mao suggested an asymmetric time-varying acceleration coefficient adjustment strategy [23]. They tried to utilize this strategy to balance local search and global search. They used some linear time-varying acceleration functions to adjust social and cognitive coefficients. In 2008, Ciu *et al.* [24] employed three non-linear time-varying cognitive adjustment strategies as well as a time-varying social coefficient adjustment strategy. The social factor was a function of the cognitive factor. The authors tried to find effective non-linear time-varying strategies for c_1 and c_2 in order to solve complex function optimization. The results showed that the PSO with the proposed time-varying adjustment strategy was superior to the conventional PSO.

Due to the complex nature of optimization problems, constant and linear time-varying values for cognitive and social factor may not work well in many cases. Using a non-linear time-varying coefficient for PSO could yield better performance in some cases. However, one non-linear time-varying strategy for all particles may not lead to a general optimizer with good performance. In this paper, we propose autonomous groups of particles for PSO which have different social and individual behaviors to improve local minima avoidance and convergence speed.

3 Overview of the PSO algorithm

PSO is an evolutionary computation technique that was proposed by Kennedy and Eberhart [1, 2]. It was inspired from the social behavior of bird flocking which uses a number of individuals (particles) flying around the search space to find the best solution. The particles trace the best location (best solution) in their paths over the course of iterations. In other words, particles are influenced by their own best locations found as well as the best solution obtained by the swarm These concepts have been mathematically modeled [1] using a position vector (x) and velocity vector (v) of length D, where D indicates the dimension (number of variables) of the problem. In the course of iterations, a particle adjusts its position and velocity as follows:

$$v_i^{t+1} = wv_i^t + c_1 \times \ rand \times (pbest_i - x_i^t) + \ c_2 \times \ rand \times (gbest - x_i^t) \quad (1)$$

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

where w is the inertial weight which is responsible for controlling the PSO algorithm's stability and usually is in [0.4, 0.9], c_1 is the cognitive coefficient that controls the influence of the individual memory of good solutions found, conventionally selected in (0, 2], c_2 is the social factor also conventionally chosen from the range (0, 2] which controls the extent to which a particle's motion is influenced by the best solution found by the whole swarm, *rand* is a random number between 0 and 1 which tries to give PSO more randomized search ability, and *pbest* and *gbest* are two variables to store the best solutions obtained so far by each particle and the whole swarm respectively. As can be observed, there are three main coefficients, w, c_1, and c_2. Dynamic tuning of these parameters is a way to give particles different behaviors as the algorithm proceeds. In this work c_1 and c_2 are targeted to increase the performance of PSO.

4 Proposed method

4.1 Motivation of proposed method

Finding the global minimum is a common, challenging task among all minimization methods [25]. In population-based optimization methods, generally the desirable way to converge towards the global minimum can be divided into two basic phases. In the early stages of the optimization, the individuals should be encouraged to scatter throughout the entire search space. In other words, they should try to explore the whole search space instead of clustering around local minima. In the latter stages, the individuals have to exploit information gathered to converge on the global minimum. In PSO, with fine-adjusting of the parameters c_1 and c_2, we can balance these two phases in order to find global minimum with fast convergence speed.

Considering these points, we propose the autonomous groups concept as a modification of the conventional PSO. In this method, each group of particles autonomously tries to search the problem space with its own strategy, based on tuning c_1 and c_2. The groups' strategies can contain constant, linear time-varying, exponential, or logarithmic time-varying values for c_1 and c_2 as shown in Fig. 1.

4.2 Autonomous groups and AGPSO algorithm

The concept of autonomous groups is inspired by the individuals' diversity in animals flocking or insects swarming. In any gathering, individuals are not quite similar in terms of intelligence and ability, but they all do their duties as a member of the group. Each individual's ability can be useful in a particular situation. In a termite colony, for instance, there are four types of termites such as soldier, worker, babysitter, and queen. They all have diverse abilities, but these differences are necessary for survival of their colony. Soldiers have greater bulk with giant jaws in order to fight with enemies. Workers are smaller than soldiers, so they can move around very fast to find and provide food for the colony. They also have the ability of excavating to build the nest. The queen and babysitters reproduce and raise children. These four types of termite can be considered as four autonomous groups which have a common goal of promoting the colony's survival.

In conventional PSO, all particles behave the same in terms of local and global search, so particles can be considered as a group with one strategy. However, using diverse autonomous groups with a common goal in any population-based optimization algorithm theoretically could result in more randomized and directed search simultaneously. In this paper, we mathematically model the autonomous groups, utilizing different strategies for updating c_1 and c_2. In other words, the groups behave differently in terms of the extent to which they follow individual and social leads.

Updating strategies of autonomous groups could be implemented with any continuous function whose range is in the interval [0,L]. Fig.1 represents some of the functions that can be used for updating cognitive and social factors. These functions consist of ascending or descending linear and polynomial, as well as exponential and logarithmic functions. In Fig. 1, the blue and red curves can be used for updating c_1 and c_2 respectively. As may be observed c_1 is decreased over the iteration, whereas c_s is increased. It is clear that particles tend to have higher local search ability when c_1 is greater than c_2.. In contrast, particles search the search space more globally when c_2 is greater than c_1. Finding a good balance between c_1 and c_2 and considering them as dynamic coefficients is investigated in this study.

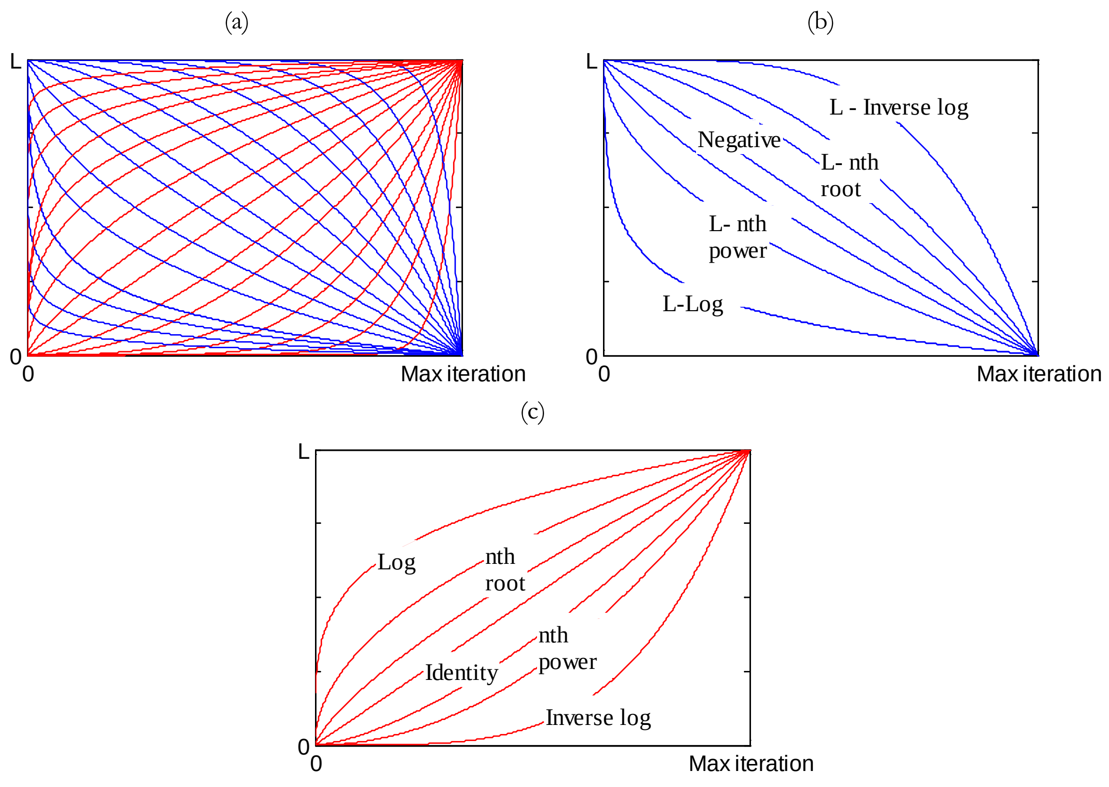

Figure 1. (a) Some samples of all posible functions for updating c_1 and c_2, (b) specific functions for updating c_1, and (c) specific functions for updating c_2 *where L is the upper bound of the c_1 and c_2*

We define four groups based on termite colonies which have their own patterns to search the problem search space locally and globally. We also develop three different versions of PSO with different autonomous groups named AGPSO1, AGPSO2, and AGPSO3. The dynamic coefficients of these algorithms are presented in Table 1 and Fig. 2 to Fig. 4. In this table, T indicates the maximum number of iterations and t is the current iteration. We try to use a diverse range of functions to investigate their effects on the performance of PSO. These functions have been chosen with different slopes, curvatures, and interception points in order to investigate the efficiency of these characteristics in improving the performance of PSO. For instance, the particles of APSO1 in group 1 tend to follow social behavior earlier than other groups, followed by group 2. In contrast, the particles in group 4 prefer to search individually in the majority of the iterations since the intersection points of c1 and c2 are close to the last iterations.

Table 1 Updating stategies

Algorithm		Updating formula	
		C1	*C2*
AGPSO1	Group1	$(-2.05/T)t+2.55$	$(1/T)t+1.25$
	Group2	$(-2.05/T)t+2.55$	$(2t^3/T)+0.5$
	Group3	$(-2t^3/T^3)+2.5$	$(1/T)t+1.25$
	Group4	$(-2t^3/T^3)+2.5$	$(2t^3/T^3)+0.5$
AGPSO2	Group1	$2.5-(2\log(t)/\log(T))$	$(2\log(t)/\log(T))+0.5$
	Group2	$(-2t^3/T^3)+2.5$	$(2t^3/T^3)+0.5$
	Group3	$0.5+2\exp[-(4t/T)^2]$	$2.2-2\exp[-(4t/T)^2]$
	Group4	$2.5+2(t/T)^2-2(2t/T)$	$0.5-2(t/T)^2+2(2t/T)$
AGPSO3	Group1	$1.95-2t^{1/3}/T^{1/3}$	$2t^{1/3}/T^{1/3}+0.05$
	Group2	$(-2t^3/T^3)+2.5$	$(2t^3/T^3)+0.5$
	Group3	$1.95-2t^{1/3}/T^{1/3}$	$(2t^3/T^3)+0.5$
	Group4	$(-2t^3/T^3)+2.5$	$2t^{1/3}/T^{1/3}+0.05$

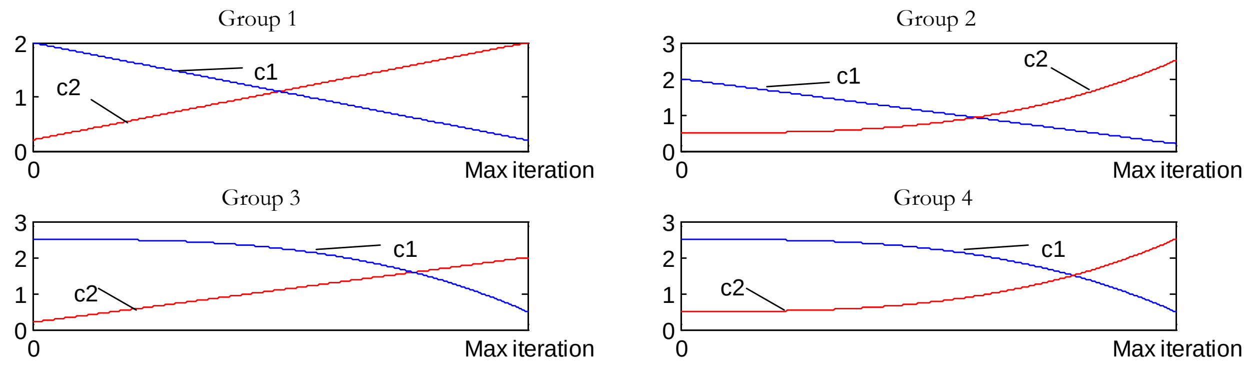

Figure 2. Matematical models of autonomous groups for APSO1

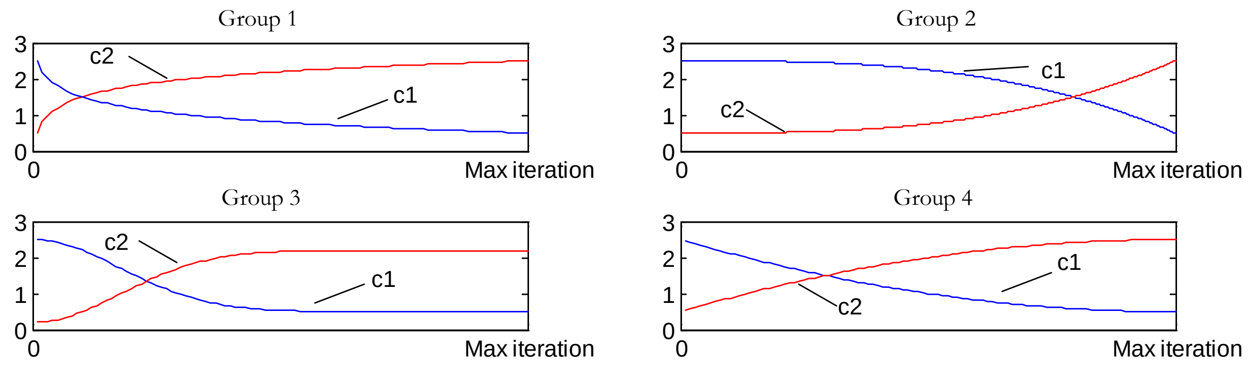

Figure 3. Matematical models of autonomous groups for APSO2

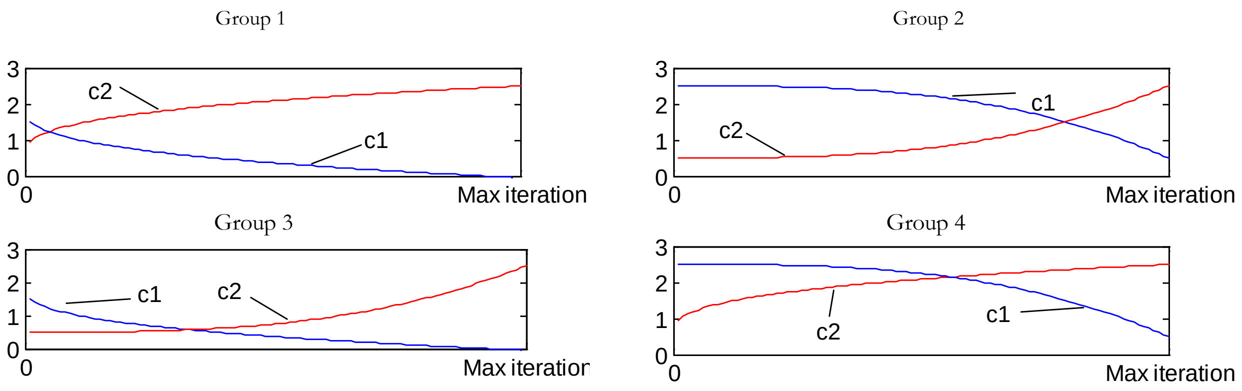

Figure 4. Matematical models of autonomous groups for APSO3

It may be observed that AGPSO1 uses two linear functions for group 1, linear and cubic functions for groups 2 and 3, and two cubic functions for group 3. AGPSO2 employs two logarithmic, two cubic, two exponential, and two quadratic functions for groups 1 to 4 respectively. It is worth mentioning that these functions have different patterns, changing during the course of the iterations. For instance, the particles in group 1 of AGPSO 2 tend to change the global and local search ability much earlier than group 2. AGPSO3 utilizes two principal third root, two cubic functions for groups 1 and 2 as well as one principal third root and cubic functions for groups 3 and 4

In PSO with autonomous groups (AGPSO), at first all particles are randomly placed in the problem search space. After that the particles are randomly divided into some predefined autonomous groups. At each iteration *gBest*, *pBest,* and the fitness of the particles are defined. For each particle the coefficients c_1 and c_2 are updated using its group's strategy. After calculating c_1 and c_2, the velocities and positions of particles will be updated using equations (1) and (2). Fig. 5 shows the pseudo-code of AGPSO.

```
Create and initialize a D-dimensional PSO
Divide particles randomly into autonomous groups
Repeat
  Calculate particles' fitness, Gbest, and Pbest
  For each particle:
      Extract the particle's group
      Use its group strategy to update $c_1$ and $c_2$
      Use $c_1$ and $c_2$ to update velocities (1)
      Use new velocities to define new positions (2)
  End for
Until stopping condition is satisfied
```

Figure 5. Pseudo-code for the proposed modification of PSO algorithm (APSO)

To see how autonomous groups are effective in AGPSO some points may be noted:

- Autonomous groups have different strategies to update c_1, so particles could explore the search space locally with different capability than the convectional PSO.
- Autonomous groups have different strategies to update c_2, so particles could follow social behavior more autonomously than the conventional PSO.
- Dynamic and diverse patterns of c_1 and c_2 cause balancing between local and global search during the course of iterations.
- Autonomous groups contain nonlinear patterns such as exponential and logarithmic functions for c_1 and c_2, so they could be more effective than the conventional PSO in solving complex optimization problems.
- PSO with autonomous groups has diverse strategies for updating c_1 and c_2, so it perhaps could more adaptable than the conventional PSO in solving a wider range of optimization problems.

These points theoretically could give AGPSO the potential of having high performance. In the following section the effectiveness of AGPSO is investigated and proved.

5 Experimental results and discussions

5.1 Selected benchmark functions

As shown in Tables 2 to 4, twenty-three standard benchmark functions are employed in order to testify the performance of AGPSO [26-30]. The objective is to find the global minimum. These benchmark functions can be divided into three groups: unimodal, multimodal, and fixed-dimension multimodal. In these tables *Dim* indicates the dimension of the function, *Range* gives the boundaries of the search space, and f_{min} is the minimum value of the function. Note that unimodal and multimodal functions with 300 dimensions have been chosen to examine the performance of the proposed method in dealing with problems of high dimensionality. Additionally, ten fixed-dimension benchmark functions have also been selected to provide a comprehensive study. A detailed description of the benchmark functions is available in the Appendix.

Table 2 Unimodal benchmark functions

Function	Dim	Range	f_{min}
$F_1(x)=\sum_{i=1}^{n} x_i^2$	300	[-100,100]	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	300	[-10,10]	0
$F_3(x)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i} x_j\right)^2$	300	[-100,100]	0
$F_4(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$	300	[-100,100]	0
$F_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	300	[-30,30]	0
$F_6(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	300	[-100,100]	0
$F_7(x)=\sum_{i=1}^{n} ix_i^4+random[0,1)$	300	[-1.28,1.28]	0

Table 3 Multimodal benchmark functions

Function	Dim	Range	f_{min}
$F_8(x)=\sum_{i=1}^{n}-x_i sin\left(\sqrt{\lvert x_i\rvert}\right)$	300	[-500,500]	-418.9829×300
$F_9(x)=\sum_{i=1}^{n}[x_i^2-10cos(2\pi x_i)+10]$	300	[-5.12,5.12]	0
$F_{10}(x)=-20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right)+20+e$	300	[-32,32]	0
$F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	300	[-600,600]	0
$F_{12}(x)=\frac{\pi}{n}\{10sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n} u(x_i,10,100,4)$ $y_i=1+\frac{x_i+1}{4}$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	300	[-50,50]	0
$F_{13}(x)=0.1\{sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+sin^2(3\pi x_i+1)]+(x_n-1)^2[1+sin^2(2\pi x_n)]\}+\sum_{i=1}^{n} u(x_i,5,100,4)$	300	[-50,50]	0

Table 4 Fixed-dimension multimodal benchmark functions

Function	Dim	Range	f_{min}
$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[-65,65]	1
$F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[-5,5]	0.00030
$F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	[-5,5]	-1.0316
$F_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)cosx_1+10$	2	[-5,5]	0.398
$F_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	[-2,2]	3
$F_{19}(x)=-\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{3} a_{ij}(x_j-p_{ij})^2\right)$	3	[1,3]	-3.86
$F_{20}(x)=-\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{6} a_{ij}(x_j-p_{ij})^2\right)$	6	[0,1]	-3.32
$F_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	[0,10]	-10.1532
$F_{22}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	[0,10]	-10.4028
$F_{23}(x)=-\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	[0,10]	-10.5363

In order to validate the results of AGPSO, it is compared with the conventional and some recent modifications of PSO with time-varying accelerators such as TACPSO [22], MPSO [23], and IPSO [24].

5.2 Parameter setting

The coefficients of SPSO, TACPSO, MPSO, and IPSO are listed in Table 5. The inertial weight *w* for all the algorithms AGPSO1 to AGPSO3 are decreased linearly from 0.9 to 0.4. There are 100 particles, and the maximum iteration is 2000.

Table 5 Updating stategies

Algorithm	Updating formula	
	C1	*C2*
SPSO [2]	2	2
TACPSO [16]	$0.5+2\exp[-(4t/T)^2]$	2.2-2exp[-(4t/T)2]
MPSO [17]	$(-2.05/T)t+2.55$	$(1/T)t+1.25$
IPSO [18]	$2.5+2(t/T)^2 – 2(2t/T)$	$0.5-2(t/T)^2 + 2(2t/T)$

5.3 Performance analysis

In order to compare the performance of all algorithm, the results are collected over 30 independent runs. The average, median, and standard deviation of the best solution in the last iteration are reported in Tables 6 to 8. The best results are indicated in bold type.

Table 6 shows the results for unimodal functions. As may be seen from this table, AGPSO3 has the best results in five out of seven unimodal benchmark functions. Generally, the results of AGPSO1 to AGPSO3 are much better than the other algorithms. These results show that autonomous groups could improve the performance of PSO algorithm for these benchmark functions. Fig. 6 illustrates the convergence curves of the algorithms. As can be seen from these curves, AGPSO3 has the best convergence rates for most of the benchmark functions, followed by AGPSO1 and AGPSO2. It is worth noting that unimodal benchmark functions have only one global minimum and there are no local minima in the search space. So these kinds of functions are quite suitable for benchmarking the convergence ability of algorithms. Consequently, the results of the AGPSO algorithms indicate that autonomous groups could improve the convergence ability of the PSO algorithm significantly. The reason for the superior results is that the particles have diversity in the population and are able to exploit knowledge of the location of near optimal solutions effectively.

Table 6 Comparison results among all algorithms on unimodal benchmark functions

Test Function		AGPSO1	AGPSO2	AGPSO3	SPSO	MPSO	TACPSO	IPSO
	Average best so far	1.35E+04	1.63E+04	**4.15E+03**	8.69E+05	8.93E+04	3.53E+04	3.93E+04
F1	Median best so far	1.3393E+04	1.5894E+04	**4.2138E+03**	8.7488E+05	8.6431E+04	3.4401E+04	3.7548E+04
	Std dev best so far	3.3699E+03	3.7457E+03	**1.1488E+03**	1.9307E+04	1.3441E+04	6.4825E+03	6.3760E+03
	Average best so far	5.38E+04	3.47E+02	**1.10E+02**	7.21E+09	1.58E+06	1.21E+05	8.59E+03
F2	Median best so far	3.9440E+04	3.4272E+02	**1.1068E+02**	2.7663E+08	3.5944E+05	5.3477E+04	3.5306E+03
	Std dev best so far	4.6654E+04	6.4717E+01	**1.2805E+01**	3.3309E+10	6.1156E+06	1.4437E+05	1.3154E+04
	Average best so far	3.23E+05	2.74E+05	2.56E+05	5.03E+06	4.60E+05	**2.40E+05**	3.21E+05
F3	Median best so far	2.9637E+05	2.6753E+05	2.4995E+05	5.0515E+06	4.5800E+05	**2.3158E+05**	3.1431E+05
	Std dev best so far	6.1328E+04	6.4660E+04	**4.6873E+04**	1.7152E+06	7.4540E+04	5.7444E+04	8.0702E+04
	Average best so far	5.93E+01	6.28E+01	6.26E+01	9.78E+01	9.22E+01	**5.88E+01**	5.93E+01
F4	Median best so far	5.9099E+01	6.2104E+01	6.2783E+01	9.7744E+01	9.6449E+01	**5.8866E+01**	5.9099E+01
	Std dev best so far	3.4682E+00	3.3038E+00	2.4881E+00	4.5566E-01	8.8241E+00	**2.4359E+00**	3.4682E+00
	Average best so far	2.74E+06	2.80E+06	**4.63E+05**	3.82E+09	1.19E+08	2.54E+07	2.74E+06
F5	Median best so far	2.5478E+06	2.4039E+06	**4.6453E+05**	3.8447E+09	1.2626E+08	2.4604E+07	2.5478E+06
	Std dev best so far	9.5801E+05	1.4135E+06	**1.4947E+05**	1.6960E+08	3.0795E+07	7.6467E+06	9.5801E+05
	Average best so far	1.36E+04	1.64E+04	**4.26E+03**	8.72E+05	9.34E+04	3.54E+04	3.82E+04
F6	Median best so far	1.3293E+04	1.6556E+04	**4.0308E+03**	8.7876E+05	9.2104E+04	3.5586E+04	3.7781E+04
	Std dev best so far	2.2733E+03	3.4075E+03	**1.2613E+03**	2.3473E+04	1.8672E+04	5.2585E+03	6.5413E+03
	Average best so far	2.3529E+01	3.7165E+01	**1.1886E+01**	1.8920E+04	4.9963E+02	1.4398E+02	2.1712E+02
F7	Median best so far	2.2634E+01	3.4481E+01	**1.0926E+01**	1.9000E+04	4.8630E+02	1.3566E+02	2.0407E+02
	Std dev best so far	6.5380E+00	1.4691E+01	**3.4267E+00**	9.5780E+02	1.3928E+02	5.2471E+01	6.9669E+01

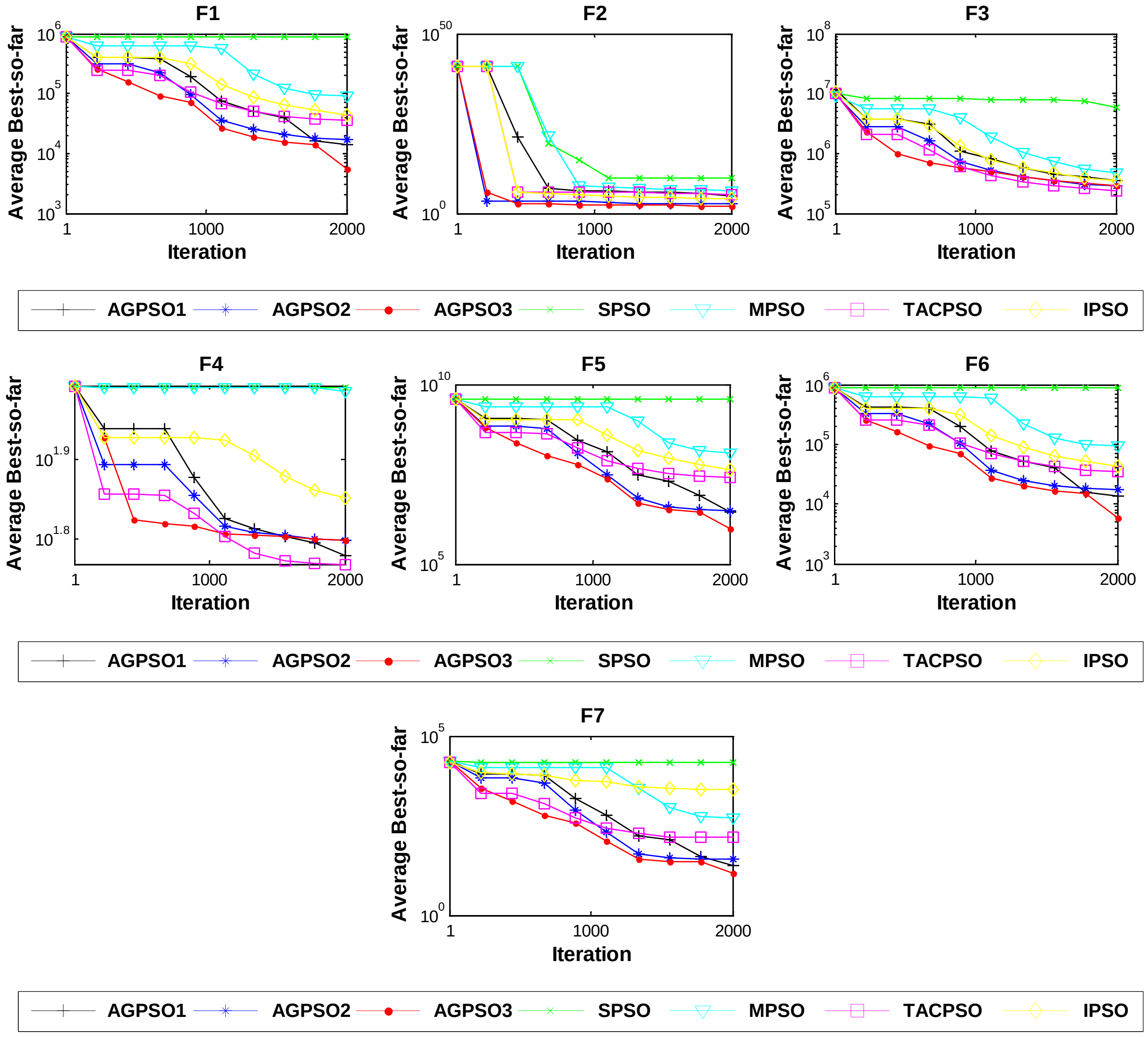

Figure 6. Convergence curves of the algorithms on unimodal benchmark functions

The results for the multimodal benchmark functions are provided in Table 7. In contrast to the unimodal functions, these benchmark functions have many local minima that increase exponentially with problem dimensionality. Therefore, they are suitable for benchmarking the capability of algorithms in avoiding local minima. As the results show, AGPSO3 performs better than the other algorithms in most of the multimodal benchmark functions. The only benchmark function on which AGPSO3 is not able to outperform TACPSO is F10, but the results of these two algorithms are very close. Generally, the AGPSO algorithms have the best results. The results of Table 7 show that the autonomous groups increased the performance of the PSO algorithm in terms of avoiding local minima. As may be observed in Fig. 7, similar to the results of unimodal benchmark functions the convergence rate of the AGPSO algorithms is better than the other algorithms. The AGPSO3 algorithm has the best convergence rates of the AGPSO algorithms. The reason for the improved ability in avoiding local minima is that the autonomous groups give AGPSO more randomized search in comparisons with the conventional and recent modifications of the PSO algorithm, so the particles are not easily trapped in local minima.

Table 7 Comparison results among all algorithms on multimodal benchmark functions

Test Function		AGPSO1	AGPSO2	AGPSO3	SPSO	MPSO	TACPSO	IPSO
	Average best so far	-6.662E+04	-5.90E+04	**-6.83E+04**	-3.83E+04	-4.81E+04	-6.63E+04	-6.39E+04
F8	Median best so far	-6.719E+04	-5.819E+04	**-6.896E+04**	-3.785E+04	-4.707E+04	-6.674E+04	-6.377E+04
	Std dev best so far	5.0885E+03	4.7799E+03	**4.2049E+03**	3.7701E+03	4.3025E+03	2.3123E+03	2.9516E+03
	Average best so far	1.4010E+03	1.4983E+03	**1.3493E+03**	3.1436E+03	2.3469E+03	1.3428E+03	1.6243E+03
F9	Median best so far	1.3944E+03	1.4794E+03	**1.3457E+03**	3.1595E+03	2.3642E+03	1.3181E+03	1.3944E+03
	Std dev best so far	7.8790E+01	1.2466E+02	**1.0394E+02**	1.6415E+02	9.5798E+01	9.4062E+01	7.8790E+01
	Average best so far	1.6990E+01	1.7313E+01	1.7089E+01	1.9966E+01	1.9356E+01	**1.5317E+01**	1.8093E+01
F10	Median best so far	1.6962E+01	1.7429E+01	1.7103E+01	1.9966E+01	1.9367E+01	**1.5267E+01**	1.8166E+01
	Std dev best so far	4.2914E-01	5.1447E-01	2.4298E-01	1.9365E-04	9.8243E-02	**8.9630E-01**	2.9006E-01
	Average best so far	1.9974E+02	1.7726E+02	**5.5658E+01**	2.8277E+03	1.2274E+03	2.5232E+02	2.7665E+02
F11	Median best so far	2.0386E+02	1.6590E+02	**4.2906E+01**	2.7726E+03	1.2170E+03	2.5856E+02	2.0386E+02
	Std dev best so far	6.7637E+01	5.3999E+01	**3.6911E+01**	2.9022E+02	1.7422E+02	5.6008E+01	6.7637E+01
	Average best so far	2.3642E+03	2.3793E+04	**5.6427E+01**	1.8896E+09	5.7484E+08	2.2419E+06	6.4022E+06
F12	Median best so far	3.7962E+02	1.6973E+04	**4.1806E+01**	1.8026E+09	5.4422E+08	1.9938E+06	5.2430E+06
	Std dev best so far	4.6017E+03	2.3723E+04	**3.8456E+01**	5.1837E+08	2.8703E+08	1.5327E+06	3.9894E+06
	Average best so far	3.7934E+05	1.2170E+06	**3.4986E+04**	3.6421E+09	1.1519E+09	2.2574E+07	5.3217E+07
F13	Median best so far	2.2776E+05	7.8352E+05	**2.5203E+04**	3.6631E+09	1.0375E+09	2.1180E+07	3.8380E+07
	Std dev best so far	3.9792E+05	1.3654E+06	**3.5129E+04**	6.7487E+08	5.0529E+08	1.1331E+07	7.9149E+07

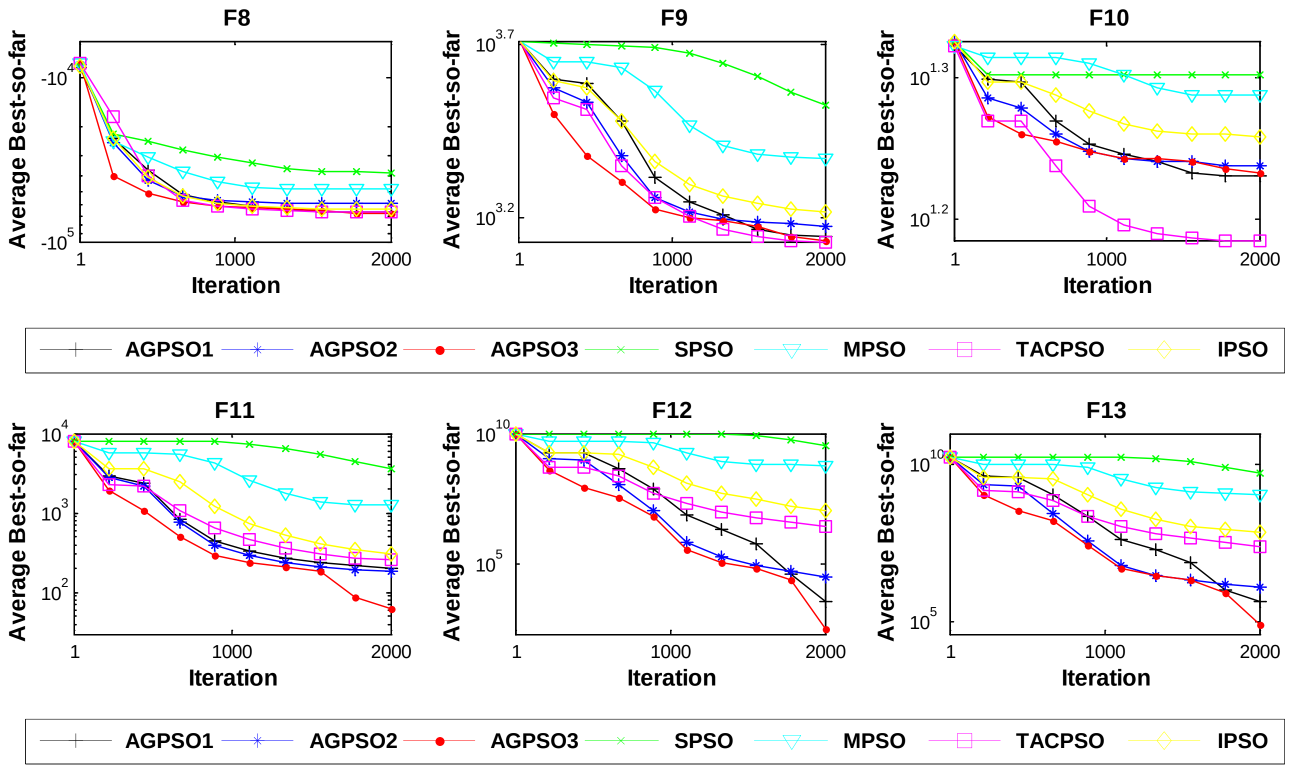

Figure 7. Convergence curves of the algorithms on multimodal benchmark functions

In contrast to the multimodal functions, the fixed-dimension multimodal benchmark functions have few local minima. As shown in Table 8, the results of all algorithms are equal on five of the functions. However, the AGPSO algorithms outperform the other algorithms on F15, F20, F21, and F22. AGPSO3 has the best results in three of these functions. Fig. 8 illustrates the convergence behavior of the algorithms dealing with fixed-dimension functions. All the algorithms have close convergence curves, slightly better for the AGPSO algorithms. The similarity of results and convergence curves are due to the low-dimensional characteristic of these benchmark functions; the effect of autonomous groups is more observable for the high-dimensional problems.

Table 8 Comparison results among all algorithms on fixed-dimension benchmark functions

Test Function		AGPSO1	AGPSO2	AGPSO3	SPSO	MPSO	TACPSO	IPSO
	Average best so far	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**
F14	Median best so far	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**
	Std dev best so far	**3.3876E-16**	**3.3876E-16**	**3.3876E-16**	**3.3876E-16**	**3.3876E-16**	**3.3876E-16**	**3.3876E-16**
	Average best so far	3.3824E-04	3.9905E-04	**3.6853E-04**	1.0108E-03	4.4773E-04	4.1489E-04	4.1489E-04
F15	Median best so far	3.0749E-04	3.0749E-04	**3.0749E-04**	7.8266E-04	3.0749E-04	3.0749E-04	**3.0749E-04**
	Std dev best so far	1.6714E-04	2.7940E-04	**2.3232E-04**	4.6281E-04	2.7642E-04	2.8739E-04	2.8739E-04
	Average best so far	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**
F16	Median best so far	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**	**-1.0316**
	Std dev best so far	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Average best so far	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
F17	Median best so far	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**	**0.3979**
	Std dev best so far	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Average best so far	**3**	**3**	**3**	**3**	**3**	**3**	**3**
F18	Median best so far	**3**	**3**	**3**	**3**	**3**	**3**	**3**
	Std dev best so far	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Average best so far	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**
F19	Median best so far	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**	**-3.8628**
	Std dev best so far	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	Average best so far	-3.2625	-3.2586	**-3.2824**	-3.2361	-3.2704	-3.2665	-3.2665
F20	Median best so far	-3.2625	-3.2031	**-3.3220**	-3.2031	**-3.3220**	**-3.3220**	**-3.3220**
	Std dev best so far	6.0463E-02	6.0328E-02	**5.7005E-02**	7.5065E-02	6.0063E-02	6.0328E-02	6.0328E-02
	Average best so far	-8.4675	**-9.3111**	-9.1412	-8.1262	-7.8784	-8.6360	-8.7238
F21	Median best so far	-10.1532	**-10.1532**	-10.1532	-10.1532	-10.1532	-10.1532	-10.1532
	Std dev best so far	2.4247	**1.9151**	2.0586	2.5251	2.6820	2.3573	2.4471
	Average best so far	-9.5225	**-10.0513**	**-10.0513**	-9.6984	-9.5212	-9.8755	-9.8742
F22	Median best so far	-10.4029	**-10.4029**	**-10.4029**	-10.4029	-10.4029	-10.4029	-10.4029
	Std dev best so far	2.0023	**1.3381**	**1.3381**	1.8271	2.0054	1.6093	1.6135
	Average best so far	-10.3577	-9.4643	-10.0003	-9.4627	-10.3561	-10.0003	**-10.5364**
F23	Median best so far	-10.5364	-10.5364	-10.5364	-10.5364	-10.5364	-10.5364	**-10.5364**
	Std dev best so far	0.9787	2.1810	1.6357	2.1842	0.9873	1.6357	**0**

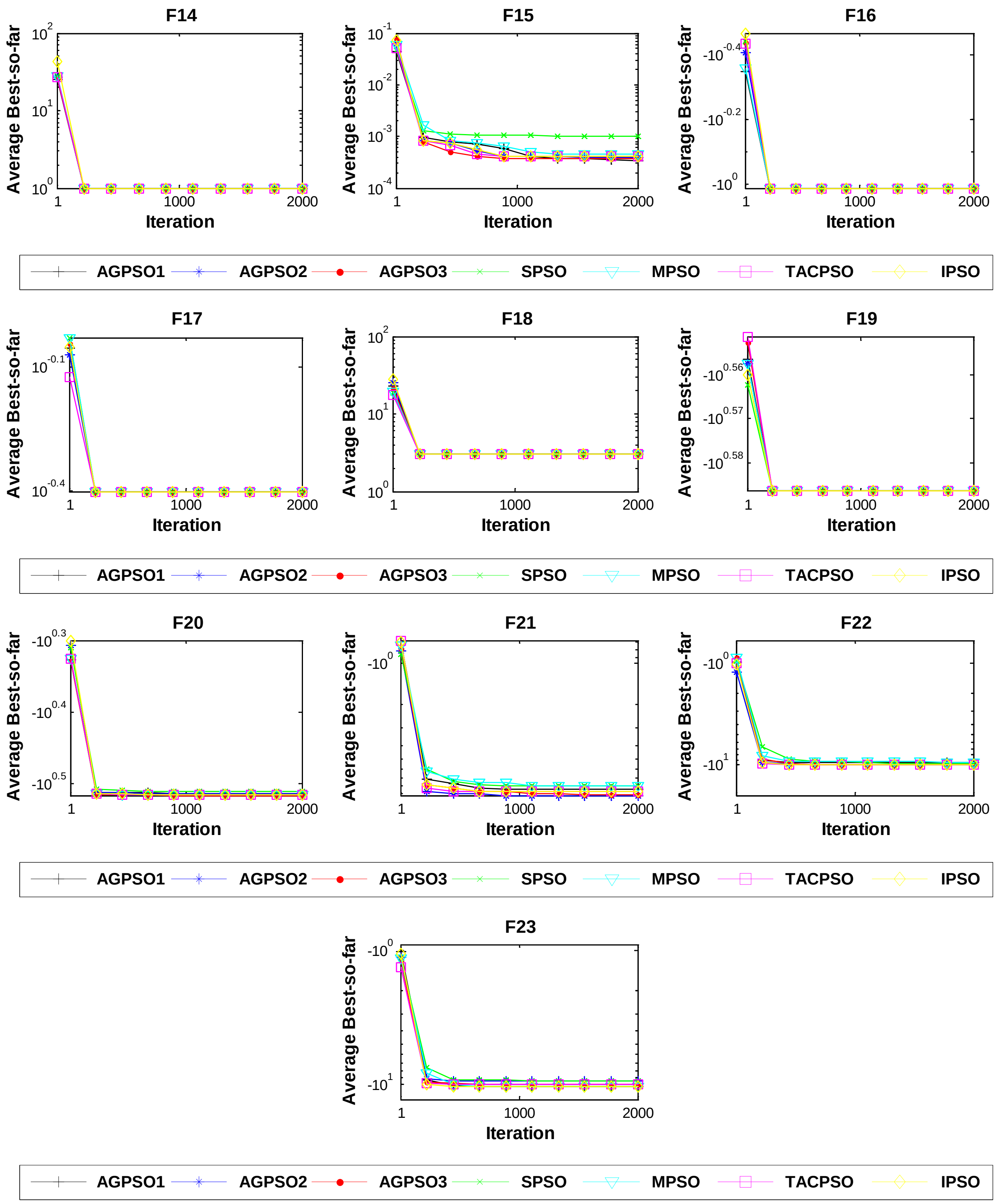

Figure 8. Convergence curves of the algorithms on fixed-dimension multimodal benchmark functions

To summarize, the results show that the proposed method is useful for the PSO algorithm in terms not only of avoiding local minima but also improved convergence rate. Statistically speaking, the AGPSO3 algorithm has the best results for seventeen out of twenty-three benchmark functions, more than half. The AGPSO1 and AGPSO2 algorithms also show the best results on 5 and 6 benchmark functions respectively. This shows that there is a significant superiority for the AGPSO3 algorithm compared to others. As can be seen in the tables, generally the results of the AGPSO algorithms are much better than those of the SPSO algorithms. The SPSO algorithm provides good results on 5 out of 23 test functions, mostly on low-dimensional functions. However, the AGPSO algorithms show much better results on high-dimensional unimodal and multimodal test functions. The results of unimodal functions revealed SPSO failed to provide fast convergence behavior in the high-dimensional problems, whereas the proposed approach allows AGPSO to provide high convergence rates because of different cognitive behaviors for particles. In addition, the results of high-dimensional multimodal functions indicated the poor ability of SPSO in avoiding local optima. The results of the AGPSO algorithms show that the proposed autonomous groups allow particles to have different patterns for following the social behavior of the whole swarm, resulting in higher local optima avoidance capability.

Among the three proposed groups, the third groups show much better results. As can be inferred from Fig. 2 to Fig. 4, group 1 of AGPSO3 has the most local search ability because the intersection point of c_1 and c_2 are close to the start of iterations. However, group 2 better allows particles to search globally because c_1 intersects c_2 after almost three quarters of the allowed iterations. Group 3 and group 4 also provide smooth transition between local and global search ability. This combination prevents particles from easily becoming trapped in local optima. This is the main reason for the superior results of the proposed autonomous groups (especially the third group). It should be noted that this remarkable improvement has been made just with dividing particles to autonomous groups and utilizing the new mathematical functions. There is no extra computational cost for the proposed method. The results support the contention that the proposed approach has merit for solving high dimensional problems.

6 Conclusion

In this paper, a new modification of PSO called AGPSO is proposed utilizing the concept of autonomous groups inspired by the diversity of individuals in natural colonies. Three versions of AGPSO with different autonomous groups were introduced. In order to evaluate their performance, twenty-three benchmark functions were employed, and the results compared with the conventional, and some recent modifications of PSO. The results show that AGPSO has merit compared to other algorithms in terms of improving avoidance of trapping in local minima and convergence speed, particularly for problems of higher dimensionality. The results also showed that dividing particles in groups and allowing them to have different individual and social behaviour can improve the performance of PSO significantly without any extra computational burden.

For future studies, it would be interesting to apply AGPSO in optimization problems to evaluate the efficiencies of AGPSO in solving real world problems. Increasing the number of autonomous groups is also worthy of investigation. Moreover, employing different types of function with greater variety of slopes, curvatures, and interception points is recommended for future study.

References

[1] R. C. Eberhart and J. Kennedy, "A new optimizer using particles swarm theory," in *Sixth Int. Symp. on Micro Machine and Human Science*, Nagoya, Japan, 1995, pp. 39–43.

[2] R. C. Eberhart and J. Kennedy, "Particle swarm optimization," in *IEEE Int. Conf. on Neural Network*, Perth, Australia, 1995, pp. 1942–1948.

[3] S. Chandra, R. Bhat, and H. Singh, "A PSO based method for detection of brain tumors from MRI," in *Nature & Biologically Inspired Computing*, Coimbatore, 2009, pp. 666 - 671.

[4] M. Mathiyalagan, U. Dhepthie, and S. Sivanandam, "Grid Scheduling Using Enhanced PSO Algorithm," vol. 2, pp. 140-145, 2010.

[5] E. Masehian and D. Sedighizadeh, "A multi-objective PSO-based algorithm for robot path planning," in *IEEE International Conference on Industrial Technology* Vi a del Mar, 2010, pp. 465 - 470

[6] M. Fayk, H. El Nemr, and M. Moussa, "Particle swarm optimisation based video abstraction," *Journal of Advanced Research,* vol. 1, pp. 163-167, 2010.

[7] S. M. Mirjalili, K. Abedi, and S. Mirjalili, "Light property and optical buffer performance enhancement using Particle Swarm Optimization in Oblique Ring-Shape-Hole Photonic Crystal Waveguide," in *Photonics Global Conference (PGC), 2012*, 2012, pp. 1-4.

[8] S. M. Mirjalili, K. Abedi, and S. Mirjalili, "Optical Buffer Performance Enhancement Using Particle Swarm Optimization in Ring-Shape-Hole Photonic Crystal Waveguide," *Optik - International Journal for Light and Electron Optics.*

[9] S. Mirjalili and A. S. Sadiq, "Magnetic Optimization Algorithm for training Multi Layer Perceptron," in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, 2011, pp. 42-46.

[10] S. Mirjalili, S. Z. Mohd Hashim, and H. Moradian Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation,* vol. 218, pp. 11125-11137, 2012.

[11] S. Mirjalili, S. M. Hashim, G. Taherzadeh, S. Mirjalili, and S. Salehi, "A Study of Different Transfer Functions for Binary Version of Particle Swarm Optimization," in *GEM'11*, 2011.

[12] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization," *Swarm and Evolutionary Computation,* vol. 9, pp. 1-14, 2013.

[13] K. Premalatha and A. M. Natarajan, "Hybrid PSO and GA for Global Maximization," *International Journal of Open Problems in Computer Science and Mathematics* vol. 2, pp. 597-608, 2009.

[14] S. Mirjalili and S. Z. Mohd Hashim, "A New Hybrid PSOGSA Algorithm for Function Optimization," in *International Conference on Computer and Information Application (ICCIA 2010)*, 2010, pp. 374-377.

[15] B. Shuang, J. Chen, and Z. Li, "Study on hybrid PS-ACO algorithm," *Applied Intelligence,* vol. 34, pp. 64-73, 2011.

[16] G. Toscano-Pulido, A. Reyes-Medina, and J. Ramirez-Torres, "A Statistical Study of the Effects of Neighborhood Topologies in Particle Swarm Optimization," *Computational Intelligence,* pp. 179-192, 2011.
[17] H. Matsushita and Y. Nishio, "Network-Structured Particle Swarm Optimizer with Various Topology and its Behaviors," *Advances in Self-Organizing Maps,* pp. 163-171, 2009.
[18] S. Mo, J. Zeng, and Y. Tan, "Particle swarm optimisation based on self-organisation topology driven by different fitness rank," *International Journal of Computational Science and Engineering,* vol. 6, pp. 24-33, 2011.
[19] X. Cai, "A new modified PSO based on black stork foraging process," in *8th IEEE International Conference on Cognitive Informatics*, Kowloon, Hong Kong 2009, pp. 509 - 513.
[20] X. Cai a, Z. Cui, J. Zeng, and Y. Tana, "Dispersed particle swarm optimization," *Information Processing Letters,* vol. 105, pp. 231-235, 2008.
[21] X. Cai, Y. Cui, and Y. Tan, "Predicted modified PSO with time-varying accelerator coefficients," *Int. J. Bio-Inspired Computation,* vol. 1, pp. 50-60, 2009.
[22] T. Ziyu and Z. Dingxue, "A Modified Particle Swarm Optimization with an Adaptive Acceleration Coefficients," in *Asia-Pacific Conference on Information Processing*, Shenzhen 2009, pp. 330 - 332
[23] G. Q. Bao and K. F. Mao, "Particle swarm optimization algorithm with asymmetric time varying acceleration coefficients," in *IEEE International Conference on Robotics and Biomimetics* Guilin 2009 pp. 2134 - 2139.
[24] Z. Cui, J. Zeng, and Y. Yin, "An Improved PSO with Time-Varying Accelerator Coefficients," in *Eighth International Conference on Intelligent Systems Design and Applications*, Kaohsiung 2008, pp. 638 - 643
[25] Y. Fukuyama and H. Yoshida, "A particle swarm optimization for reactive power and voltage control in electric power system," in *IEEE Congress on Evolutionary Computation*, Seoul, Korea, 2001, pp. 87-93.
[26] M. Molga and C. Smutnicki, in *Test functions for optimization needs*, ed, 2005.
[27] X.-S. Yang. ((Eds Xin-She Yang), John Wiley & Sons, 2010) Test problems in optimization. *An Introduction with Metaheuristic Applications*.
[28] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation,* vol. 3, pp. 82 - 102, 1999.
[29] J. G. Digalakis and K. G. Margaritis, "On benchmarking functions for genetic algorithms," *Intern. J. Computer Math,* vol. 77, pp. 481-506, 2001.
[30] S. Mirjalili and S. Z. M. Hashim, "BMOA: Binary Magnetic Optimization Algorithm," *International Journal of Machine Learning and Computing,* vol. 2, pp. 204-208, 2012.

Appendix

Table 9 to Table 16 contain the details of the benchmark functions.

Table 9 $a_{i,j}$ in F_{14}

$$a_{i,j} = \begin{pmatrix} -32,-16,0,16,32,-32,\dots,0,16,32 \\ -32,-32,-32,-32,-16,\dots,32,32,32 \end{pmatrix}$$

Table 10 a_i and b_i in F_{15}

i	1	2	3	4	5	6	7	8	9	10	11
a_i	0.1957	0.1947	0.1735	0.1600	0.0844	0.0627	0.0456	0.0342	0.0342	0.0235	0.0246
b_i^{-1}	0.25	0.5	1	2	4	6	8	10	12	14	16

Table 11 a_{ij} and c_i in F_{19}

i	a_{i1}	a_{i2}	a_{i3}	c_i
1	3	10	30	1
2	0.1	10	35	1.2
3	3	10	30	3
4	0.1	10	30	3.2

Table 12 p_{ij} in F_{19}

i	p_{i1}	p_{i2}	p_{i3}
1	0.3689	0.1170	0.2673
2	0.4699	0.4387	0.7470
3	0.1091	0.8732	0.5547
4	0.03815	0.5743	0.8828

Table 13 a_{ij} and c_i in F_{20}

i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}	a_{i6}	c_i
1	10	3	17	3.5	1.7	8	1
2	0.05	10	17	0.1	8	14	1.2
3	3	3.5	1.7	10	17	8	3
4	17	8	0.05	10	0.1	14	3.2

Table 14 p_{ij} in F_{20}

i	p_{i1}	p_{i2}	p_{i3}	p_{i4}	p_{i5}	p_{i6}
1	0.131	0.169	0.556	0.012	0.828	0.588
2	0.232	0.413	0.830	0.373	0.100	0.999
3	0.234	0.141	0.352	0.288	0.304	0.665
4	0.404	0.882	0.873	0.574	0.109	0.038

Table 15 a_{ij} and c_i in F_{21}, F_{22}, and F_{23}

i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	6	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

Table 16 Best solution for fixed-dimension multimodal functions

F	X_{output}	F_{min}
F_{14}	(-32,32)	1
F_{15}	(0.1928,0.1908,0.1231,0.1358)	0.00030
F_{16}	(0.089,-0.712), (-0.089, 0.712)	-1.0316
F_{17}	(-3.14, 12.27), (3.14, 2.275), (9.42, 2.42)	0.398
F_{18}	(0,_1)	3
F_{19}	(0.114, 0.556,0.852)	-3.86
F_{20}	(0.201, 0.15,0.477, 0.275, 0.311,0.657)	-3.32
F_{21}	5 local minima in a_{ij}, i = 1,2,3,4,5	-10.1532
F_{22}	7 local minima in a_{ij}, i = 1,2,3,4,5,6,7	-10.4028
F_{23}	10 local minima in a_{ij}, i = 1,2,3,4,5,6,7,8,9,10	-10.5363