Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 976486, 14 pages
http://dx.doi.org/10.1155/2014/976486

Research Article

An Improved Particle Swarm Optimization Algorithm Based on Centroid and Exponential Inertia Weight

Shouwen Chen,[1,2] Zhuoming Xu,[1] Yan Tang,[1] and Shun Liu[2]

[1]*College of Computer and Information, Hohai University, Nanjing, Jiangsu 210098, China*
[2]*College of Mathematics and Information, Chuzhou University, Chuzhou, Anhui 239000, China*

Correspondence should be addressed to Shouwen Chen; cshouwen@163.com

Received 10 April 2014; Accepted 23 July 2014; Published 30 December 2014

Academic Editor: Roque J. Saltarén

Particle swarm optimization algorithm (PSO) is a global stochastic tool, which has ability to search the global optima. However, PSO algorithm is easily trapped into local optima with low accuracy in convergence. In this paper, in order to overcome the shortcoming of PSO algorithm, an improved particle swarm optimization algorithm (IPSO), based on two forms of exponential inertia weight and two types of centroids, is proposed. By means of comparing the optimization ability of IPSO algorithm with BPSO, EPSO, CPSO, and ACL-PSO algorithms, experimental results show that the proposed IPSO algorithm is more efficient; it also outperforms other four baseline PSO algorithms in accuracy.

1. Introduction

Particle swarm optimization (PSO) has been proposed originally by Kennedy and Eberhart in 1995, which is a population-based stochastic optimization techniques inspired by social behavior of bird flocking or fish schooling [1]. The PSO algorithm is easy to implement and has been empirically shown to perform well on many optimization problems [2–5]. The development of PSO can be classified into three categories in general. The first category emphasizes the variants of PSO mechanism itself, both in mathematics [6] and in topology [7]. The second one hybridizes other optimization techniques into PSO, such as ACO [8], GA [9, 10], Tabu [11], and simulated annealing [12]. The third one leverages the advantages of Chaos Maps, such as certainty, ergodicity, and stochastic property [13]. The hybridization of chaos with PSO has also become an active direction in recent research activities [14, 15].

With the concept of the center of gravity in Physics, Song et al. designed a centroid particle swarm based on each particle's best position and proposed a centroid PSO (CPSO) algorithm [16] to enhance individual and group collaboration and information sharing capabilities. Also, Gou et al. proposed an ACL-PSO algorithm [17] with a population centroid based on every particle's current position. Their experimental results showed that ACL-PSO algorithm improved the global searching capability and effectively avoided the premature convergence. Inertia weight [18], in the form of linear decreasing one, was embedded into the original PSO firstly by Shi and Eberhart [18]. Based on their work, a conclusion can be drawn that a large inertia weight facilitates a global search while a small one facilitates a local search. After that, different kinds of inertia weights were introduced and expressed in exponential formalities [19] or other nonlinear formalities [20–22]. Recently, Ting et al. [23] proposed an exponential inertia weight frame. After their carefully analysis of the effect of local attractor and global attractor, they presented suggestions for adjusting these attractors in order to improve the performance of PSO algorithm.

In this paper, in order to prevent PSO from falling in a local optimum, we propose an improved PSO algorithm (IPSO) based on two forms of exponential inertia weight proposed by Ting et al. and two kinds of centroids, population centroid and the best individual centroid, which are based on a new weighted linear combination of each particle's current position and a linear combination of each particle's best position, respectively. Therein, the proposed IPSO algorithm provides two velocity updating forms, which are selected

by roulette wheel for every particle at each of evolution iterations. Besides one particle's own extreme position and the global extreme position, one of velocity updating forms is based on population centroid, another is based on best individual centroid. By means of comparing its optimization ability with other four PSO algorithms, the experiment results show that the proposed IPSO algorithm can reach more excellent optima.

The remainder of this paper is organized as follows. Basic PSO algorithm (BPSO) [18], exponential inertia weight PSO (EPSO) [23], center PSO (CPSO) [16], and self-adaptive comprehensive learning PSO (ACL-PSO) [17] are proposed in Section 2. We present our improved PSO algorithm (IPSO) model in Section 3. Section 4 shows our experimental results. Finally, we conclude our work in the last section.

2. Background

The basic PSO (BPSO) algorithm is a useful tool for optimization. Each particle's position stands for a candidate solution to the problem which will be solved. The BPSO, EPSO, CPSO, and ACL-PSO are proposed below.

2.1. BPSO. Denote f, S and CS as the fitness function, the scale of swarm, and the maximum iteration number, respectively. Let $X_i(t) = (x_{i1}^{(t)}, \ldots, x_{id}^{(t)}, \ldots, x_{iD}^{(t)})$, $v_i(t) = (v_{i1}^{(t)}, \ldots, v_{id}^{(t)}, \ldots, v_{iD}^{(t)})$, and $f_i^{(t)} = f(X_i(t))$ be the position, velocity, and fitness of the ith particle at the tth iteration, respectively. In addition, let $f_i^{(r)}$ be the ith particle's best fitness and let p_i be the corresponding position, St. $r = \arg\max_t\{f_i^{(t)}\}$. Also, let $f_g^{(r)}$ be the swarm's best fitness and let p_g be the corresponding position, St. $g = \arg\max_{i\in\{1,2,\ldots,S\}}\{f_i^{(r)}\}$.

In BPSO, tracking the two positions p_g and p_i, each particle flows through the multidimensional searching space to update its inertia weight, velocity, and position according to (1)–(3), respectively. Consider

$$w(t) = w_{\max} - \frac{t \times (w_{\max} - w_{\min})}{CS}, \tag{1}$$

$$v_{id}^{(t+1)} = w(t)\, v_{id}^{(t)} + c_1 r_1(t)\left(p_{id} - x_{id}^{(t)}\right) + c_2 r_2(t)\left(p_{gd} - x_{id}^{(t)}\right), \tag{2}$$

$$x_{id}^{(t+1)} = x_{id}^{(t)} + v_{id}^{(t+1)}. \tag{3}$$

In (1), $w(t)$ stands for inertia weight at the tth iteration and $w_{\max}$ and $w_{\min}$ are the initial inertia weight and the final inertia weight, respectively. In (2), c_1 and c_2 are accommodation parameters and random numbers $r_1(t)$ and $r_2(t)$ belong to the interval of 0 and 1.

2.2. EPSO. Inertia weight, which had been introduced firstly by Shi and Eberhart is one of the important parameters in PSO algorithm [18]. Since that, linearly decreasing inertia weight and nonlinearly decreasing one have been used widely in literature. Chauhan et al. [22] summarized different inertia weight forms. However, there is no clear justification of how this parameter can be adjusted to improve the performance of PSO algorithm. In [23], Ting et al. have investigated the property for an exponential inertia weight inspired by adaptive crossover rate (ACR) used in differential evolution algorithm. The ACR is defined as

$$\mathrm{CR} = \mathrm{CR}_0 \cdot e^{-a(t/CS)^b}, \tag{4}$$

where CR_0 is the initial crossover rate, t is the current generation number, and CS is the maximum number of generations. The adaptive function for the crossover rate is simply crafted based on the logic of high CR at the beginning of run to prevent premature convergence and low CR at the end of run to enhance the local search. This concept is exactly the same for the case of inertia weight w in BPSO. Thus, Ting et al. defined their exponential inertia weight as follows:

$$w(t) = w_{\max} \cdot e^{-a(t/CS)^b}, \tag{5}$$

where $w_{\max}$ is the initial inertia weight. Parameters a and b in (5) are named the local search attractor and the global search attractor, respectively. On the one hand, while parameter b is set to 1, parameter a has the ability to push down the value of w. On the other hand, when b is increased, it has the ability to pull up the value of the w. Note that when a is zero, the inertia weight becomes a static value $w_{\max}$; when b is zero, it is in fact a static w with the value approximate to 0.32 on condition that a is set to 1.

Substituting (5) for (1) in BPSO algorithm, Ting et al. proposed an exponential inertia weight PSO algorithm (EPSO). After testing on 23 benchmark problems, they draw a conclusion from simulation results that EPSO algorithm was capable of global search and local search when $b > a$ and $a > b$, respectively. At the same time, their simulation results showed that the EPSO algorithm had better performance in comparison to the BPSO algorithm, which was used widely in many significant works.

2.3. CPSO. In order to enhance interparticle cooperation and information sharing capabilities, Song et al. proposed a centroid PSO (CPSO) [16] algorithm. In CPSO algorithm, the centroid of particle swarm is defined as (6), and (2) in BPSO algorithm is substituted for (7). Consider

$$C = \frac{1}{S}\sum_{i=1}^{S} p_i, \tag{6}$$

$$\begin{aligned} v_{ij}^{(t+1)} = {} & w(t)\, v_{ij}^{(t)} + c_1 r_1\left(p_{ij} - x_{ij}^{(t)}\right) \\ & + c_2 r_2\left(p_{gj} - x_{ij}^{(t)}\right) + c_3 r_3\left(C_j - x_{ij}^{(t)}\right). \end{aligned} \tag{7}$$

In the abovementioned (7), c_3 is a positive constant similar to c_1 and c_2; r_3 is a random number between 0 and 1.

In this way, the running track of each particle is not only interrelated with the individual best position and the best position of the swarm but also interrelated with the centroid of the whole particle swarm. Their experimental

Algorithm. IPSO $(x_{\min}, x_{\max}, v_{\min}, v_{\max}, w_{\max}, c_1, c_2, c_3, D, S, CS)$
Input: $x_{\min}, x_{\max}, v_{\min}, v_{\max}, w_{\max}, c_1, c_2, c_3, D, S, CS$
Output: $p_g, f(p_g)$
Begin

(1) Initialize the parameters including population size S, $x_{\min}, x_{\max}, v_{\min}, v_{\max}, w_{\max}, c_1, c_2, c_3$, the dimension size D, the maximum iteration number CS, and the current iterative count $t = 0$

(2) Generate the initial population and initial velocity. The initial population and initial velocity for each particle are randomly generated as follows:

$$\text{Population} = \begin{bmatrix} X_1(t) \\ X_2(t) \\ \vdots \\ X_S(t) \end{bmatrix} \quad X_i(t) = \left(x_{i1}^{(t)}, \ldots, x_{id}^{(t)}, \ldots, x_{iD}^{(t)}\right) \; x_{id}^{(t)} \in \left[x_{\min}, x_{\max}\right] \; i = 1, 2, \ldots, S \; d = 1, 2, \ldots, D$$

$$\text{Velocity} = \begin{bmatrix} v_1(t) \\ v_2(t) \\ \vdots \\ v_S(t) \end{bmatrix} \quad v_i(t) = \left(v_{i1}^{(t)}, \ldots, v_{id}^{(t)}, \ldots, v_{iD}^{(t)}\right) \; v_{id}^{(t)} \in \left[v_{\min}, v_{\max}\right] \; i = 1, 2, \ldots, S \; d = 1, 2, \ldots, D$$

where $X_i(t)$ and $v_i(t)$ are the position and velocity at the tth iteration for the ith particle, respectively. D is the dimension of each particle. $x_{\min}$, $x_{\max}$ are the minimum and maximum value of each point belonging to the dth dimension, respectively. $v_{\min}$, $v_{\max}$ are the minimum and maximum value of each point belonging to the dth dimension, respectively.

(3) Calculate the fitness value of each particle and record it as: $f_i^{(t)} \leftarrow f\left(X_i(t)\right)$, record the ith particle's best fitness and its corresponding position p_i as: $f_i^{(r)} \leftarrow f_i^{(t)}$, $p_i \leftarrow X_i(t)$, St. $r = \arg\min_t\{f_i^{(t)}\}$, $i = 1, \ldots, S$

(4) Calculate the population's best fitness $f_g^{(r)}$ and its corresponding position p_g, St. $g = \arg\min_{i\in\{1,2,\ldots,S\}}\{f_i^{(r)}\}$

(5) Compute population centroid by (9) with the weighted coefficient $\text{Pecent}_i(t)$ computed by (15), compute best individual centroid by (6).

(6) **For** $t = 1$ to CS do

(6.1) Compute R_t by (10), and generate a random number $P \in [0, 1]$;

(6.2) **If** $P \le R_t$

(6.2.1) Update $w_2(t)$ by (19)

(6.2.2) **For** $i = 1$ to S do

Update $v_i(t)$ and $X_i(t)$ by (12) and (3) respectively, compute $f_i^{(t)} \leftarrow f\left(X_i(t)\right)$

If $f_i^{(t)} < f_i^{(r)}$ then $f_i^{(r)} \leftarrow f_i^{(t)}$, $p_i \leftarrow X_i(t)$, $r = \arg\min_t\{f_i^{(t)}\}$,

End If

End

Else

(6.2.3) Update $w_1(t)$ by (19)

(6.2.4) **For** $i = 1$ to S do

Update $v_i(t)$ and $X_i(t)$ by (7) and (3) respectively, compute $f_i^{(t)} \leftarrow f\left(X_i(t)\right)$

If $f_i^{(t)} < f_i^{(r)}$ then $f_i^{(r)} \leftarrow f_i^{(t)}$, $p_i \leftarrow X_i(t)$, $r = \arg\min_t\{f_i^{(t)}\}$,

End If

End

End If

(6.3) Compute $f_g^{(r)}$ and set $p_g \leftarrow X_g(r)$, where $g = \arg\min_{i\in\{1,2,\ldots,S\}}\{f_i^{(r)}\}$

(6.4) Compute population centroid by (9) with the weighted coefficient $\text{Pecent}_i(t)$ computed by (15), compute best individual centroid by (6).

(6.5) let $t = t + 1$

End

(7) output $p_g, f(p_g)$

End

Algorithm 1: The execution process of IPSO algorithm.

results show that the CPSO algorithm not only enhances the local searching efficiency and global searching performance but also has an ability to avoid the premature convergence problem effectively.

2.4. ACL-PSO. After introducing population centroid learning mechanism into the BPSO, Gou et al. proposed an ACL-PSO algorithm [17] based on self-adapted comprehensive learning. They defined the fitness proportion of the ith

Table 1: Descriptions of four benchmark functions.

Function	Mathematical formula	Range and dim.	Minima	Characteristics
Griewank (f_1)	$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^{20}$	0	Highly multimodal
Rastrigin (f_2)	$f(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right) + 10$	$[-5.12, 5.12]^{20}$	0	Highly multimodal
Rosenbrock (f_3)	$f(x) = \sum_{i=1}^{n-1} \left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right)$	$[-30, 30]^{20}$	0	Multiple local optima
Sphere (f_4)	$f(x) = \sum_{i=1}^{n} x_i^2$	$[-5.12, 5.12]^{20}$	0	Unimodal convex

Table 2: Disjoint distribution of (S, c_3) for each function.

Function		f_1	f_2	f_3	f_4
	10	0.5	1	2	0.5
S	20	0.5	0.5	2	0.5
	30	0.5	0.5	2	0.5

particle as (8), and the population centroid corresponding to the tth iteration as (9). Consider

$$\text{Pecent}_i(t) = \frac{f(X_i(t))}{\sum_{k=1}^{S} f(X_k(t))}, \tag{8}$$

$$p_c(t) = \sum_{i=1}^{S} \text{Pecent}_i(t) * X_i(t). \tag{9}$$

Considering not only individual particle's own previous best position but also the evolution trend of populations, Gou et al. proposed two different evolution strategies at different evolution stage. Therein, $p_c(t)$ was used to guide the particle searching direction at the early stage of evolution, while p_g was used to guide its direction at the later stage. To realize these evolution strategies, they used (10) to compute R_t at the tth iteration and used a random number P between 0 and 1 to compare with R_t. Consider

$$R_t = 1 - \frac{t}{CS}. \tag{10}$$

If $P \le R_t$, update the ith particle's velocity by (11) or (12), else by (2). Consider

$$v_{ij}^{(t+1)} = w(t) v_{ij}^{(t)} + c_1 r_1 \left(p_{ij} - x_{ij}^{(t)}\right) + c_2 r_2 \left(a_{ij} * p_{cj} - x_{ij}^{(t)}\right), \tag{11}$$

$$\begin{aligned} v_{ij}^{(t+1)} &= w(t) v_{ij}^{(t)} + c_1 r_1 \left(p_{ij} - x_{ij}^{(t)}\right) \\ &\quad + c_2 r_2 \left(p_{gj} - x_{ij}^{(t)}\right) + c_3 r_3 \left(a_{ij} * p_{cj} - x_{ij}^{(t)}\right). \end{aligned} \tag{12}$$

In the abovementioned formulas, inertia weight of ACL-PSO was computed by the following:

$$w(t) = w_{\max} \cdot e^{-t^2/2}, \tag{13}$$

where random number a_{ij} was in $[1/S, 1]$. In addition, c_3 was the weight coefficient of population centroid and r_3 was a random number between 0 and 1. The item $c_3 r_3 (a_{ij} * p_{cj} - x_{ij}^{(t)})$ was the entry of population centroid learning, which reflected an individual particle's social cognition learning and thinking.

Compared to other four improved PSO algorithms in terms of accuracy, convergence speed, and computational complexity, ACL-PSO converged faster, resulting in more robust and better optima [17].

3. Our Proposed Method IPSO

Define particle fitness f as object function directly, and adjust the IPSO algorithm search mechanism as follows:

$$g = \arg\min_{i \in \{1,2,\ldots,S\}} \left\{f_i^{(r)}\right\}, \qquad r = \arg\min_{t} \left\{f_i^{(t)}\right\}. \tag{14}$$

Combining the above EPSO algorithm's inertia weight, CPSO and ACL-PSO algorithms' population centroids, taking advantage of ACL-PSO algorithm's evolution strategies, we propose a new improved PSO algorithm (IPSO) as follows.

(1) Population Centroid and Best Individual Centroid. So as to CPSO and ACL-PSO algorithms, their centroids are different from each other. The centroid of CPSO algorithm is a linear combination of each particle's best position, while the other is a weighted linear combination of each particle's current position. Taking advantage of the two centroids, we embed them into BPSO to balance PSO algorithm's performance of local and global searching, and denote them as best individual centroid and population centroid, respectively.

If $f(X_i(t))$ is smaller, that is to say, the object value is smaller at the tth iteration, then the particle's current position makes more contribution to the construction of the population centroid corresponding to the same iteration. Thus, in order to show the degree of importance of the particle's current position, $X_i(t)$, in the population centroid, we define the ith particle proportion as (15) and compute the population centroid $p_c(t)$ by (9). At the same time, (16) can be obtained naturally. Consider

$$\text{Pecent}_i(t) = \frac{1}{S-1}\left[1 - \frac{f(X_i(t))}{\sum_{k=1}^{S} f(X_k(t))}\right], \tag{15}$$

$$\sum_{i=1}^{S} \text{Pecent}_i(t) = 1. \tag{16}$$

Table 3: Parameter settings of different algorithms.

Algorithms	Parameters
BPSO [18]	$w_{\min} = 0.4, w_{\max} = 0.9, c_1 = c_2 = 2, w(t) = w_{\max} - (w_{\max} - w_{\min}) * t/CS$
EPSO [23]	$w_{\max} = 0.9, c_1 = c_2 = 2, a = 1, b = 1$
CPSO [16]	$w_{\min} = 0.4, w_{\max} = 0.9, c_1 = c_2 = c_3 = 1.4, w(t) = w_{\max} - (w_{\max} - w_{\min}) * (t/CS)$
ACL-PSO [17]	$w_{\max} = 1.2, c_1 = c_2 = c_3 = 2, w(t) = w_{\max} * \exp(-0.5 * t * t)$
IPSO	$w_{\max} = 0.9, c_1 = c_2 = 1.4, w_1(t) = w_{\max} * \exp(-(t/CS)^2), w_2(t) = w_{\max} * \exp(-2 * t/CS)$

Table 4: Comparison of the results of different PSO algorithms ($D = 20$).

Functions	Algorithms	Indicators				
		Min	Mean	Max	Deviation	ACIN
f_1	BPSO	$1.0397E+00$	$5.3844E+00$	$2.6224E+01$	$4.9320E+00$	$9.9424E+02$
	EPSO	$5.8010E-01$	$1.0862E+00$	$1.6439E+00$	$2.1640E-01$	$9.5420E+02$
	CPSO	$9.8400E-01$	$3.3640E+00$	$9.2279E+00$	$2.0551E+00$	$9.7812E+02$
	ACL-PSO	$2.0906E-06$	$6.0700E-02$	$5.8570E-01$	$1.4610E-01$	$9.9998E+02$
	IPSO	0	0	0	0	$5.5164E+02$
f_2	BPSO	$3.2302E+01$	$8.0862e+01$	$1.2456e+02$	$2.0199E+01$	$9.8684E+02$
	EPSO	$1.8100E+01$	$5.7588E+01$	$1.2552E+02$	$2.3422E+01$	$9.9958e+02$
	CPSO	$3.0708E+01$	$6.3367E+01$	$1.1899E+02$	$2.0263E+01$	$9.6404E+02$
	ACL-PSO	$1.3242E-08$	$5.0000E-03$	$4.5100E-02$	$1.0100E-02$	$1.0000E+03$
	IPSO	0	0	0	0	$5.4176E+02$
f_3	BPSO	$4.1348E+01$	$3.7374E+04$	$3.0661E+05$	$6.1177E+04$	$9.9250E+02$
	EPSO	$2.7485E+01$	$5.6346E+02$	$3.5538E+03$	$8.1005E+02$	$9.9520E+02$
	CPSO	$2.4717e+01$	$1.7367E+04$	$9.5022E+04$	$2.5152E+04$	$9.8090E+02$
	ACL-PSO	$1.8851E+01$	$1.8965E+01$	$1.9363E+01$	$8.4700E-02$	$1.0000E+03$
	IPSO	$1.8605E+01$	$1.8878E+01$	$1.8942E+01$	$5.5500E-02$	$9.9974E+02$
f_4	BPSO	$4.1700E-02$	$2.0394E+00$	$1.1227E+01$	$2.9060E+00$	$9.8864E+02$
	EPSO	$3.2000E-02$	$6.0300E-02$	$1.0026E+00$	$1.3970E-01$	$9.9978E+02$
	CPSO	$9.2000E-03$	$5.5700E-01$	$5.8396E+00$	$9.1550E-01$	$9.7898E+02$
	ACL-PSO	$2.1674E-17$	$2.7549E-05$	$2.1864E-04$	$4.9741E-05$	$1.000E+03$
	IPSO	$9.6962E-84$	$6.2702E-61$	$1.6301E-59$	$2.7260E-60$	$9.9994E+02$

Gou et al. [17] have proved that their proposed population centroid will be the convergence position on condition that ACL-PSO algorithm converges to local minimum or global convergence. Similarly, Let P^* satisfy (17), (18) indicates that $p_c(t)$ is the convergence position of IPSO algorithm on condition that the algorithm converges to local minimum or global convergence. Consider

$$P^* = \lim_{t\to\infty} X_i(t), \tag{17}$$

$$\begin{aligned}
&\lim_{t\to\infty} p_c(t) \\
&= \lim_{t\to\infty} \sum_{i=1}^{S} \frac{1}{S-1}\left[1 - \frac{f(X_i(t))}{\sum_{k=1}^{S} f(X_k(t))}\right] * X_i(t) \\
&= \frac{1}{S-1}\sum_{i=1}^{S}\left[1 - \frac{\lim_{t\to\infty} f(X_i(t))}{\sum_{k=1}^{S} \lim_{t\to\infty} f(X_k(t))}\right] * \lim_{t\to\infty} X_i(t) \\
&= \frac{1}{S-1}\sum_{i=1}^{S}\left[1 - \frac{f(P^*)}{\sum_{k=1}^{S} f(P^*)}\right] * P^* \\
&= P^*.
\end{aligned} \tag{18}$$

(2) Inertia Weight. Based on the exponential inertia weight proposed by Ting et al. [23], we select two pairs of a and b. One pair is that a is 1 and b is 2. Another one is contrary. Denote them as (19) at the tth iteration. These inertia weights work in with different evolution strategies, the same as ACL-PSO algorithm. Consider

$$w_1(t) = w_{\max} \cdot e^{-(t/CS)^2}, \qquad w_2(t) = w_{\max} \cdot e^{-2t/CS}. \tag{19}$$

(3) Update Velocity. It can be drawn from (18) that population centroid will coincide with the population globally optimal location. So, we use Gou et al.'s method [17] to help the ith particle to select velocity updating formula. If $P \le R_t$, update

Table 5: Comparison of experimental results of success rate and average convergence iteration.

Functions	Target precision	Indicators	BPSO	EPSO	CPSO	ACL-PSO	IPSO
f_1	0.1	SR	0	0	0	86	100
		ACI	1000	1000	1000	901.28	676.66
	0.6	SR	0	4	0	98	100
		ACI	1000	974.80	1000	747.80	607.82
	1	SR	0	26	4	100	100
		ACI	1000	818.98	989.02	439.76	472.48
	1.5	SR	16	100	64	100	100
		ACI	991.82	327	840.20	23.10	269.64
f_2	10^{-7}	SR	0	0	0	48	100
		ACI	1000	1000	1000	967.60	369.50
	20	SR	0	4	0	100	100
		ACI	1000	971.98	1000	40.92	262.20
	30	SR	0	8	2	100	100
		ACI	1000	945.64	998.82	42.42	267.02
	40	SR	2	30	12	100	100
		ACI	982.44	775.36	944.36	28.62	254.82
f_3	19	SR	0	0	0	70	100
		ACI	1000	1000	1000	932.98	336.88
	30	SR	0	8	0	100	100
		ACI	1000	947.38	1000	444.52	275.40
	40	SR	0	14	2	100	100
		ACI	1000	903.22	983	333.14	608.60
	500	SR	6	62	30	100	100
		ACI	998.72	530.34	943.50	36.18	28.40
f_4	10^{-10}	SR	0	0	0	78	100
		ACI	1000	1000	1000	920.22	330.84
	0.01	SR	0	10	8	100	100
		ACI	1000	932.46	980.92	229.54	255.48
	0.05	SR	4	68	18	100	100
		ACI	998.68	504.24	959.46	53.96	225.48
	0.1	SR	14	90	52	100	100
		ACI	994	294.80	899.40	34.64	209.18

the ith particle's velocity by (12) with $w_2(t)$, else by (7) with $w_1(t)$. The execution process of IPSO algorithm is shown in Algorithm 1.

4. Experiments and Results

In this section, BPSO, EPSO, CPSO, and ACL-PSO algorithm are compared as four benchmark functions to verify the feasibility of IPSO algorithm. The descriptions of those test functions, which can be divided into unimodal and multimodal function, are shown in Table 1. Using the object function in Table 1 to evaluate each particle's fitness, the smaller the function value the higher the fitness.

Experiments use the following methods. Firstly, determine the parameter pairs of IPSO algorithm, such as S and c_3. Secondly, fixing the number of iterations, with different number of particles, evaluate performances of those five algorithms by average object value corresponding to the tth iteration. At last, setting the maximum iteration number and different target accuracies of these functions, success rate and average convergence iteration number are compared.

The average object value according to the tth iteration of all the r turns is as (20) for each algorithm. Therein, $f_j^{(i)}(t)$ stands for the global best fitness for the jth algorithm, corresponding to the tth iteration at the ith turn. Consider

$$y(t) = \frac{1}{r}\sum_{i=1}^{r} f_j^{(i)}(t), \quad t = 1, \ldots, CS, \ j = 1, \ldots, 5. \tag{20}$$

(1) Determine the Parameter Pairs of S and c_3 in IPSO Algorithm. With $v_{\min} = x_{\min}$, $v_{\max} = x_{\max}$, $c_1 = c_2 = 1.4$, $CS = 1000$, let S be 10, 20, and 30 each time and let c_3 be updated by (21) for each S. Run the IPSO algorithm $r = 50$ turns per time. Its $(t-\log(y))$ curves of above four benchmark functions are shown in Figures 1, 2, 3, and 4. Consider

$$c_3(k) = 0.5 \times (k-1) \quad k = 1, \ldots, 5. \tag{21}$$

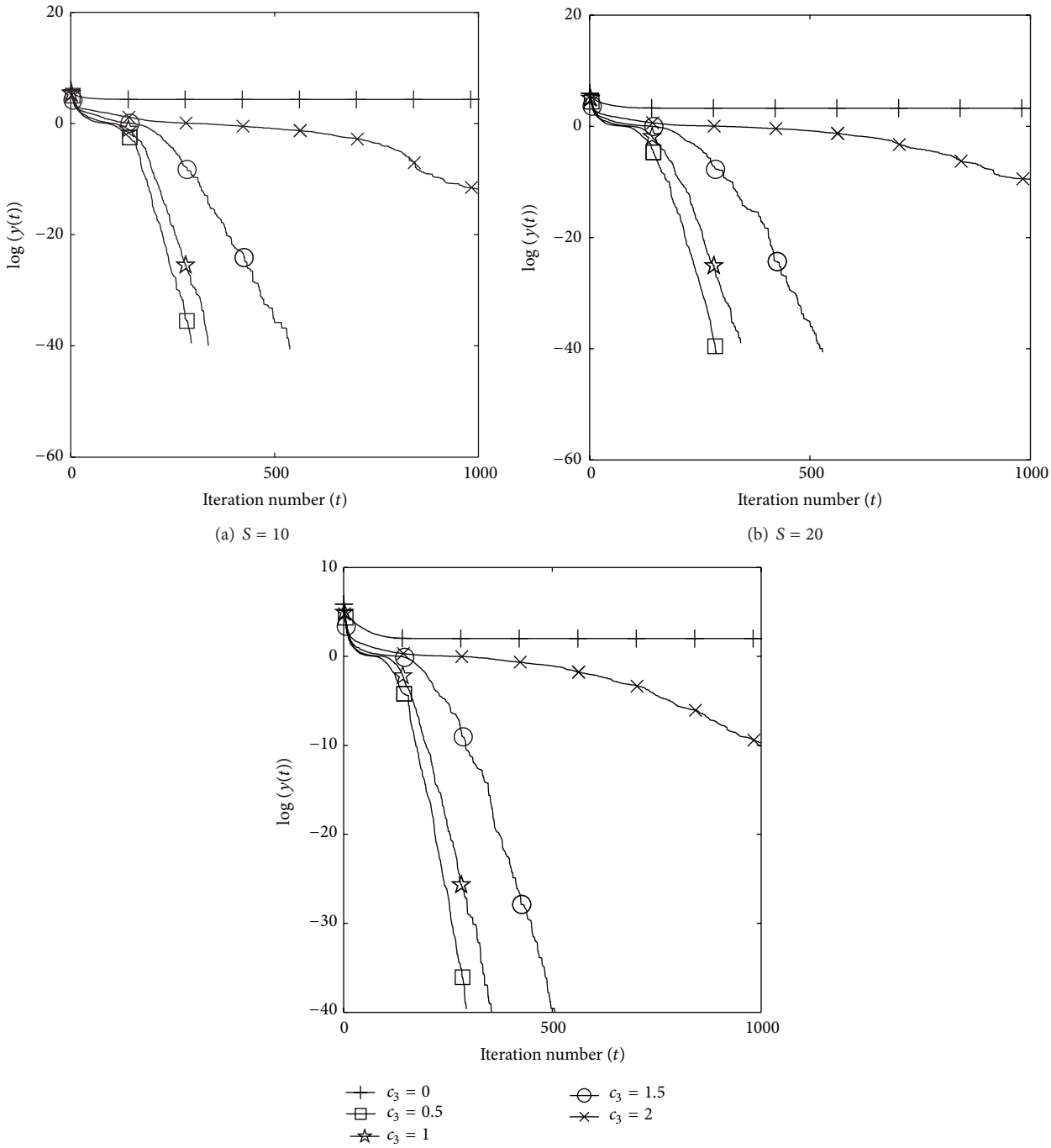

FIGURE 1: $(t - \log(y))$ curves of f_1 according to different S.

From Figures 1–4, we can get that each of the number of iteration, the value of parameter c_3, and swarm size S produces an effect on performance of IPSO algorithm. Let c_3 be 0.5 and let S be 30; $y(t)$ will arrive to minimum at later iteration stage for the four benchmark functions. If $y(t)$ is zero, $\log(y(t))$ will tend to be infinite, and the corresponding curves about $(t - \log(y))$ will not appear. The phenomenon happened in curves of functions f_1 and f_2, that is to say, IPSO algorithm can find target optima value for f_1 and f_2 at litter iteration number, this can be seen in Figures 1 and 2. While $y(t)$ is smaller, $\log(y(t))$ will become smaller too. From Figure 3, we can get that setting $c_3 = 2$ is more effective for using IPSO algorithm to search optimal solution of f_3 at later iteration stage. So as to f_4, setting $c_3 = 0.5$ is reasonable.

With different pairs of swarm size and c_3, $y(t)$ comes to the minimum at the later iteration. Table 2 lists the

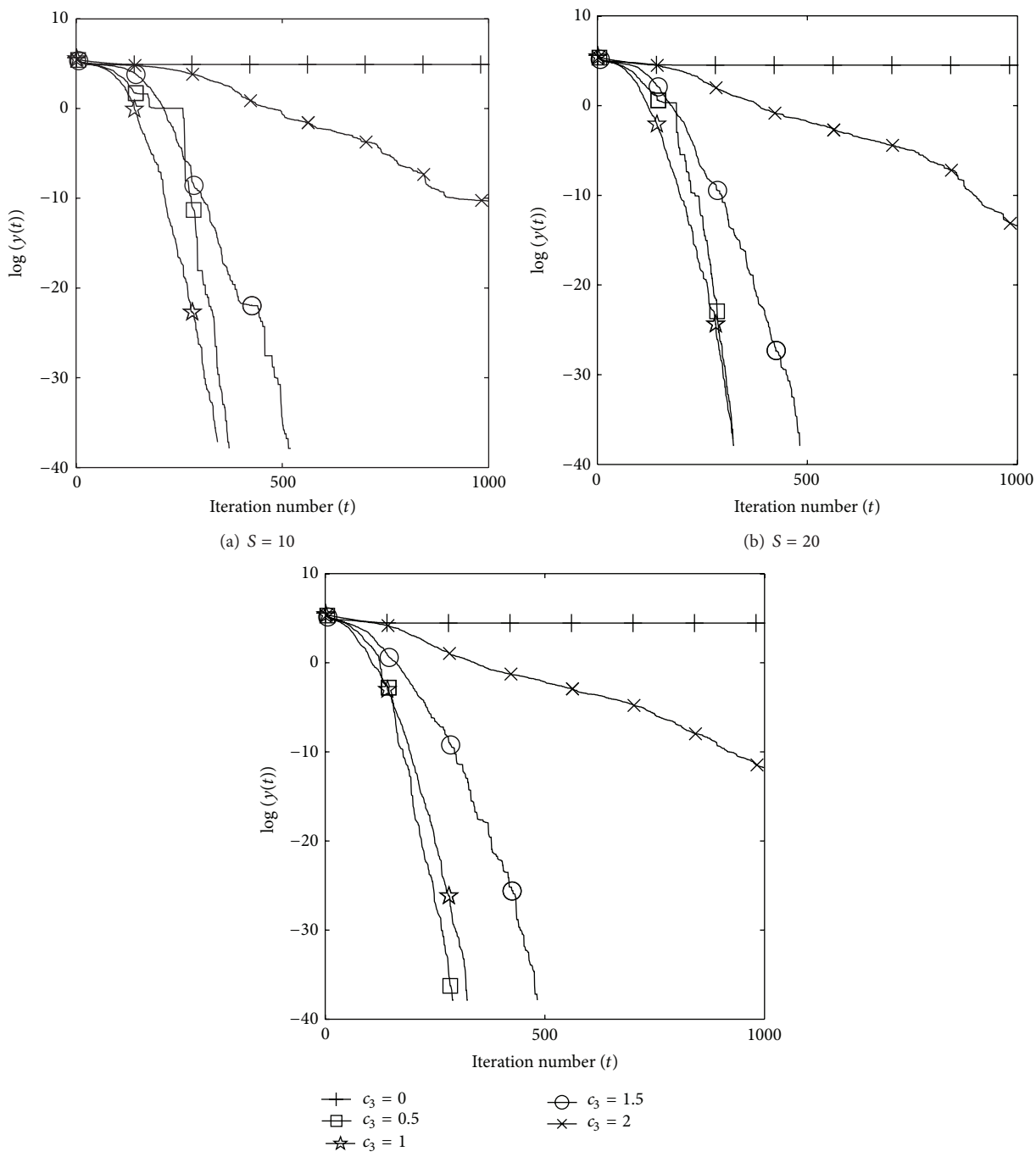

Figure 2: $(t - \log(y))$ curves of f_2 according to different S.

disjoint distribution of (S, c_3) pairs according to $CS = 1000$. Larger population sizes require more function evaluations and increase the computing efforts for convergence, but increase the reliability of the algorithm. The problem is to find a compromise between cost and reliability. In order to make IPSO algorithm have better optimization capability, we will set swarm size $S = 30$, and let parameter c_3 take the corresponding values in the third row of Table 2 for the different benchmark functions in the following tests with the algorithm.

(2) Compare IPSO Algorithm with Other Four Algorithms. Using BPSO, EPSO, CPSO, and ACL-PSO algorithm to compare with IPSO algorithm on above four benchmark

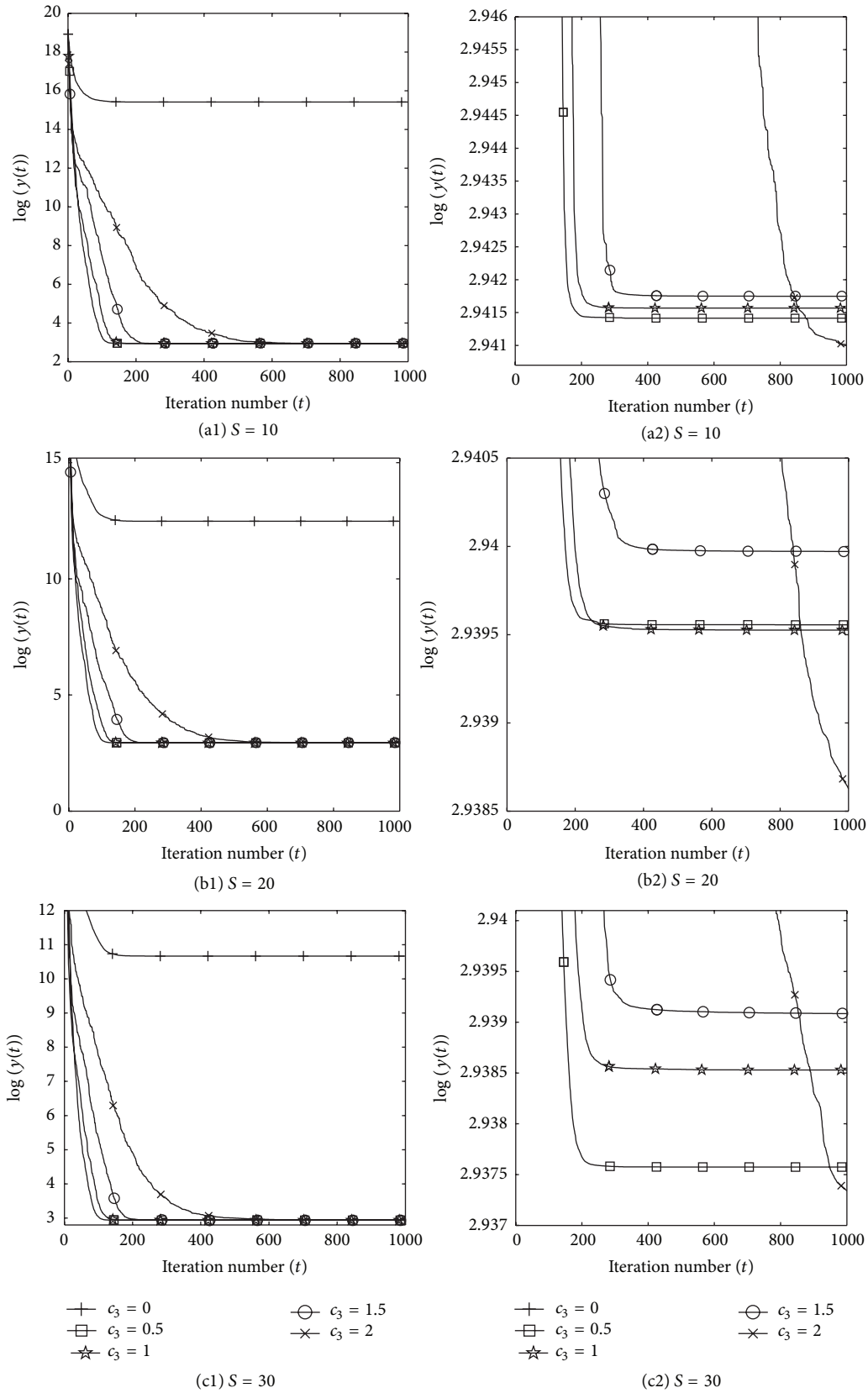

FIGURE 3: $(t - \log(y))$ curves of f_3 according to different S; (a2) is a partial graph of (a1) corresponding to $\log(y) \in [2.9407, 2.946]$; (b2) is a partial graph of (b1) corresponding to $\log(y) \in [2.9385, 2.9405]$; (c2) is a partial graph of (c1) corresponding to $\log(y) \in [2.937, 2.9401]$.

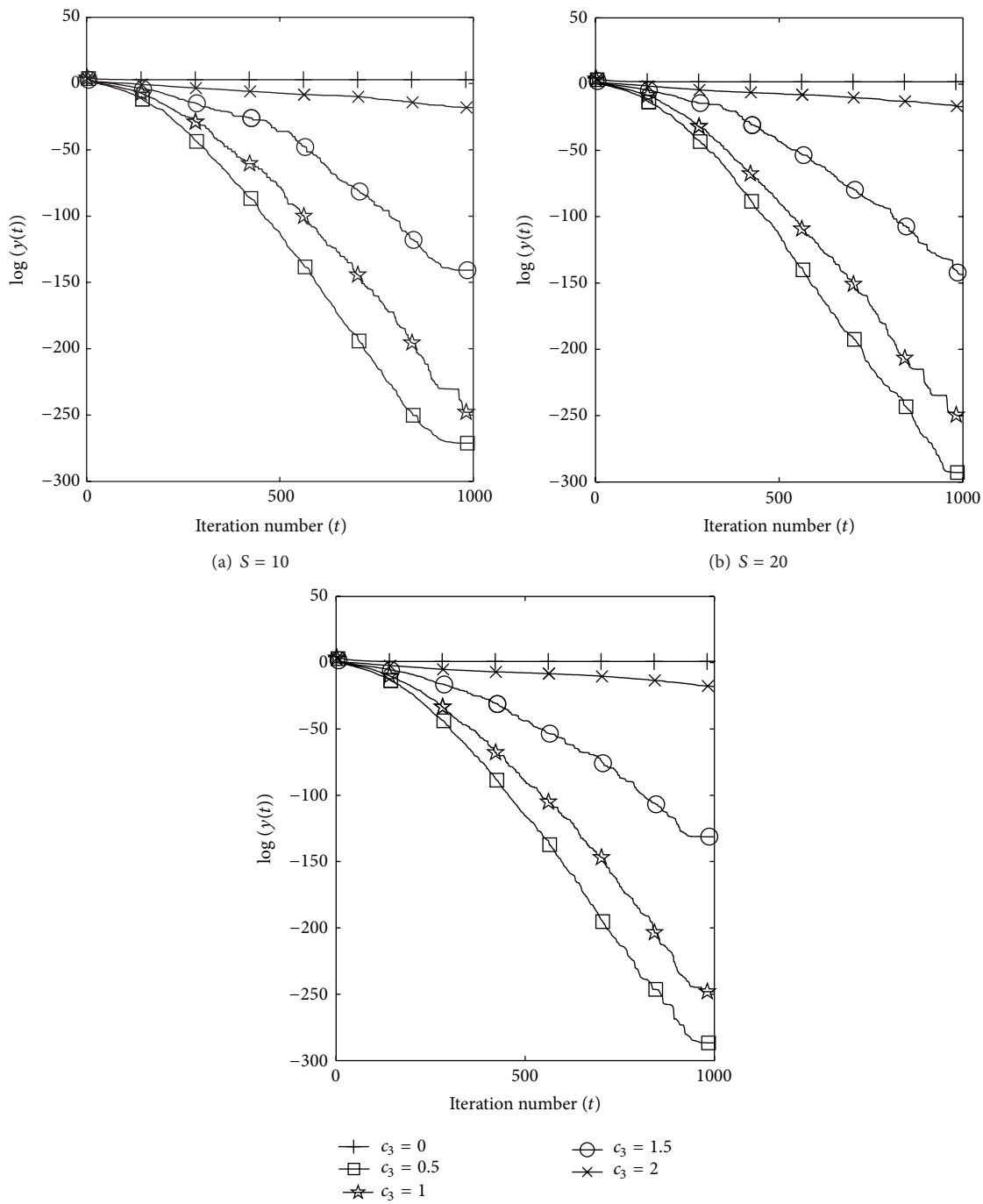

Figure 4: $(t - \log(y))$ curves of f_4 according to different S.

functions, we set $v_{\min} = x_{\min}$, $v_{\max} = x_{\max}$, $S = 30$, $CS = 1000$, and other parameters related to those compared with algorithms are listed in Table 3.

Run each of above five algorithms 50 times independently. Record five indicators, which are the minimum object value (Min), the maximum object value (Max), the mean object value (Mean), the deviation of object value (Dev), and the average convergence iteration number (ACIN), for every run. Ours experimental results are shown in Table 4 and Figures 5, 6, 7, and 8.

From Table 4, it can be seen that there are higher accuracy for the IPSO than that for the BPSO, EPSO, CPSO, and ACL-PSO. From the mean and deviation in Table 4, the IPSO is better than the BPSO, EPSO, CPSO, and ACL-PSO, with

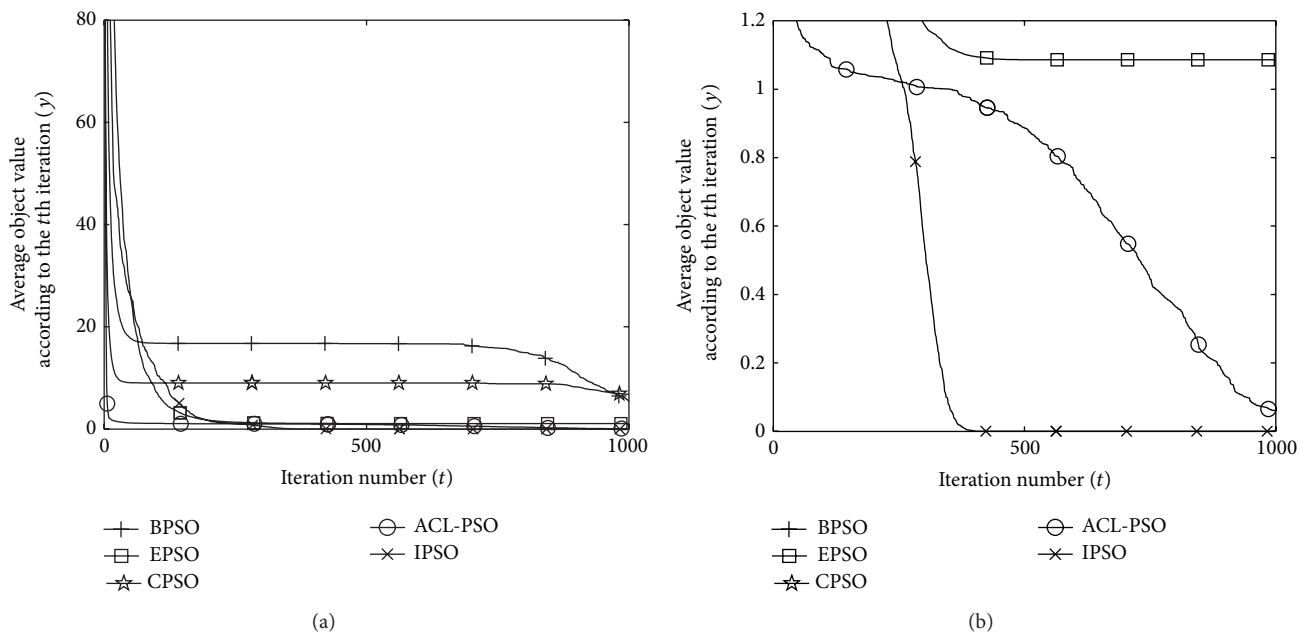

Figure 5: $(t - y(t))$ curves of f_1 according to results of five algorithms. (b) is a partial graph of (a) corresponding to $y(t) \in [0, 1.2]$.

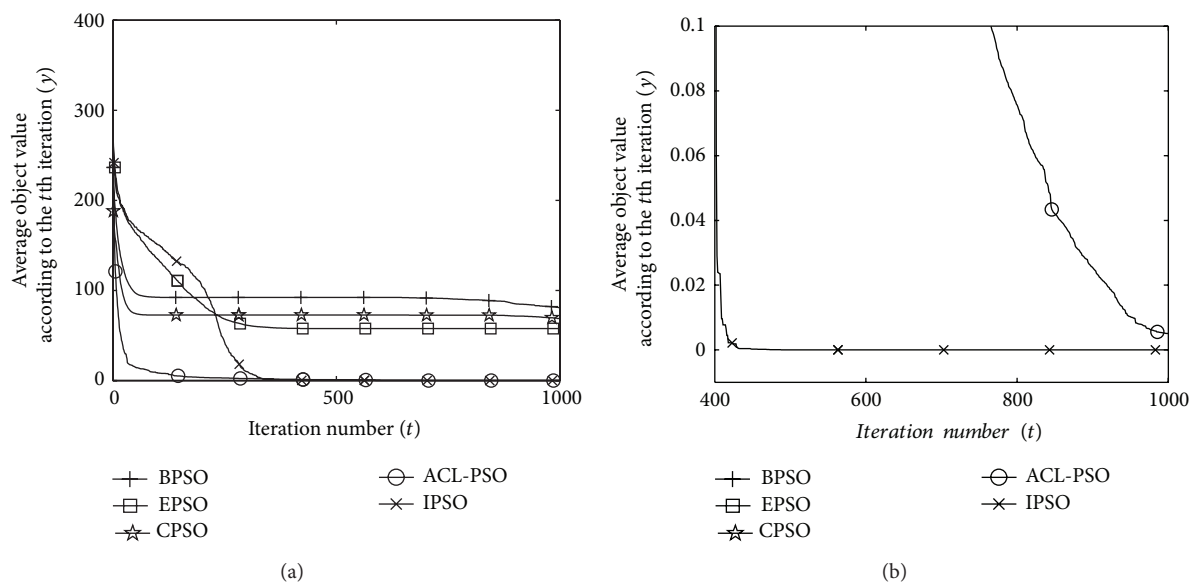

Figure 6: $(t - y(t))$ curves of f_2 according to results of five algorithms. (b) is a partial graph of (a) corresponding to $y(t) \in [0, 0.1]$.

a steady convergence. In addition, IPSO algorithm needs less number of iteration for coming to convergence than ACL-PSO algorithm does.

Let iteration number t be as x-coordinate and average best object value according to the tth iteration $y(t)$ as y-coordinate; we plot $(t - y(t))$ curves of these four benchmark functions in Figures 5–8.

From Figures 5–8, it can be seen that IPSO algorithm, compared to other four algorithms, searches more excellent object value at later iteration. EPSO, CPSO, ACL-PSO, and IPSO algorithm all can find more optimal solution than BPSO algorithm does after litter iteration number, and the performance of IPSO algorithm is the best among these five algorithms. In addition, IPSO algorithm needs less iteration number to come to convergence than ACL-PSO algorithm does. This phenomenon is considered to be due to the combination of inertia weight $w_2(t)$ working with population centroid and $w_1(t)$ working with best individual centroid.

(3) Compare Success Rate and Average Number of Iteration Corresponding to Target Precision Arriving. In order to validate the effectiveness of IPSO algorithm further, we set

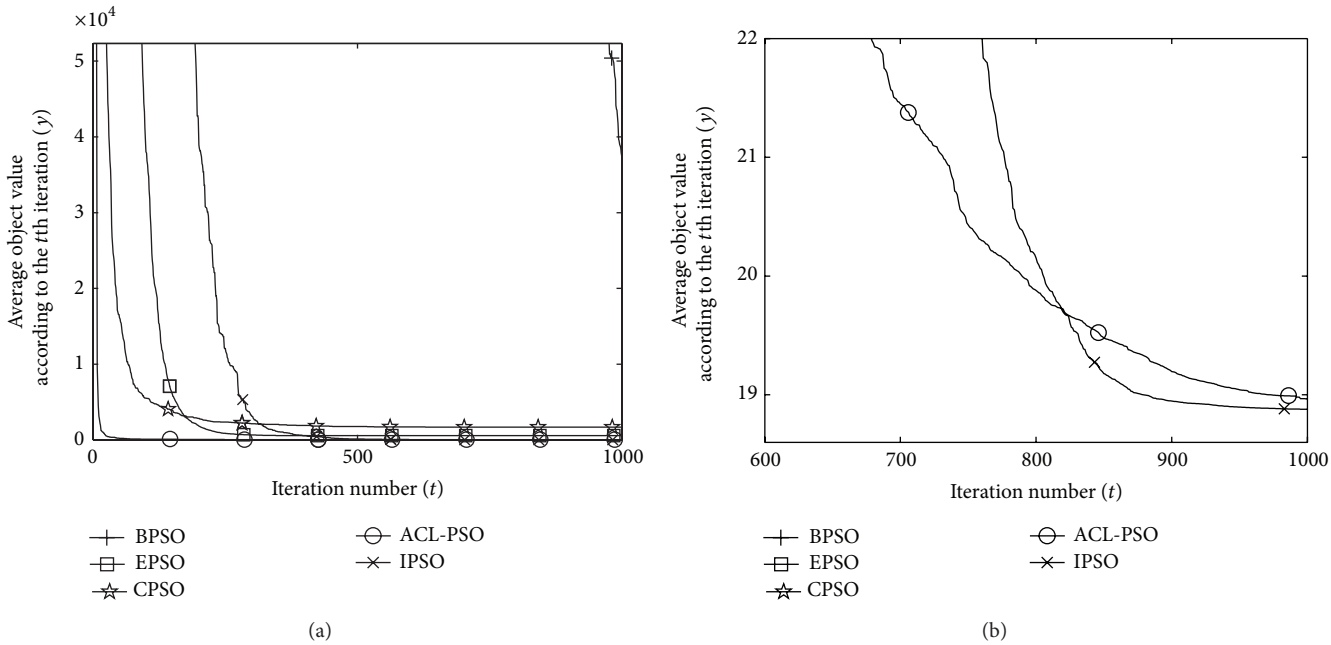

Figure 7: $(t - y(t))$ curves of f_3 according to results of five algorithms. (b) is a partial graph of (a) corresponding to $y(t) \in [18.6, 22]$.

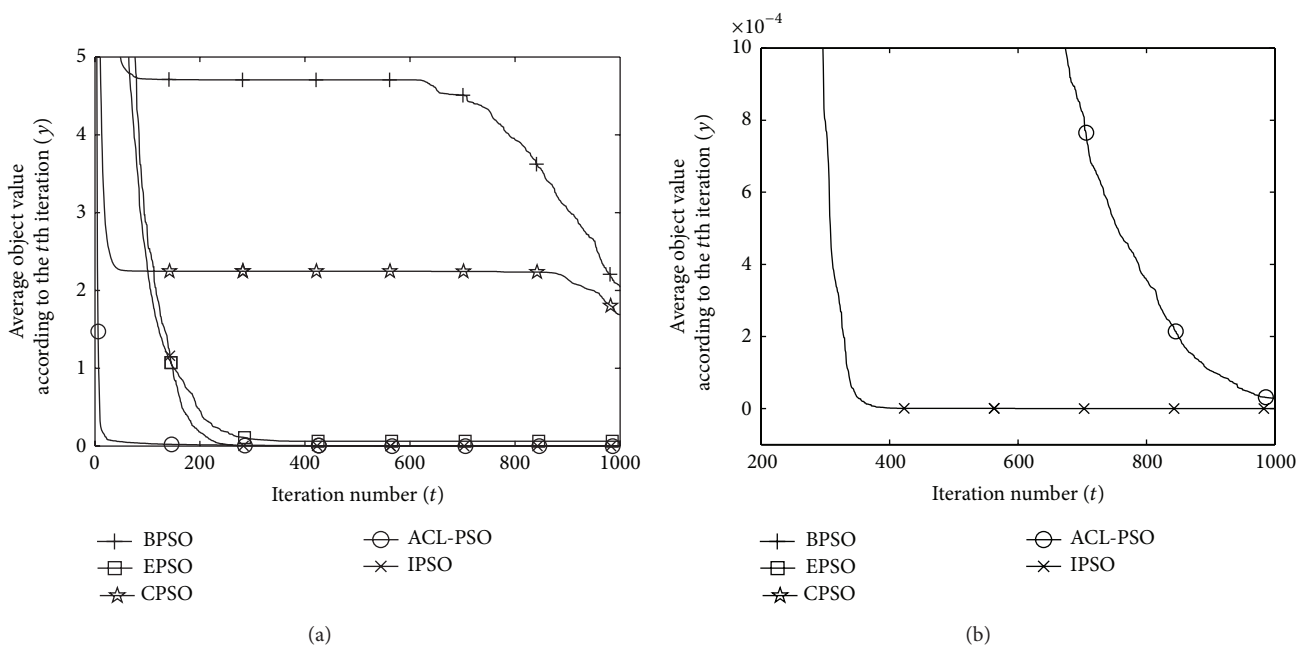

Figure 8: $(t - y(t))$ curves of f_4 according to results of five algorithms. (b) is a partial graph of (a) corresponding to $y(t) \in [0, 10^{-3}]$.

target precisions of above four benchmark functions, which are listed in Table 5 and run every algorithm 100 turns for each of test functions, respectively. Setting the maximum iteration number as 1000, while object value is less than or equal to its target precision, we plus success convergence number with one and record the current iteration number at one turn. Then, success rate (SR) is equal to the success convergence number divided by total number of turns. Average convergence iteration (ACI) is the mean iteration numbers at all. Four group target accuracies, which are listed in Table 5, are used to evaluate the stability of those algorithms. Our experimental results are shown in Table 5.

From Table 5, it can be seen that success rates of those five algorithms are affected by the target accuracies. Success rate of IPSO algorithm for each test function reaches 100% and is obviously better than those of BPSO, EPSO, CPSO, and ACL-PSO. Average convergence iteration of EPSO is smaller than CPSO algorithm, and ACI of IPSO is smaller

than EPSO algorithm. So as to the ability of finding optimal solutions, ACL-PSO algorithm is better than BPSO, EPSO, and CPSO algorithms, and IPSO algorithm is better than ACL-PSO algorithm.

5. Conclusions

The particle swarm optimization algorithm is a global stochastic tool, which has ability to search the global optima. PSO algorithm is easily trapped into local optima. In this paper, in order to overcome the shortcoming of PSO algorithm, we propose an improved particle swarm optimization algorithm (IPSO) based on two forms of exponential inertia weight and two kinds of centroids. By means of comparing optimization ability of IPSO algorithm with BPSO, EPSO, CPSO, and ACL-PSO algorithms, experimental results of these four benchmark functions show that the proposed IPSO algorithm is more efficient and outperforms other PSO algorithms in accuracy investigated in this paper. Inertia weight is one of the important parameters in PSO algorithm. How can (5) respect the constraint on $w_{\min}$? Moreover can *a and b* be chosen to be adaptive throughout a single evolution to guarantee a suitable trade-off between exploration and exploitation phase? These are good future research directions for us.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the Natural Science Foundation of Jiangsu Province of China under Grant No. BK20140857 and Grant No. BK20141420. It was also supported by the Excellent Young Talents Fund of Anhui Province of China (Grant No. 2012SQRL154).

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, December 1995.

[2] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[3] S. Ding, H. Jiang, J. Li, and G. Tang, "Optimization of well placement by combination of a modified particle swarm optimization algorithm and quality map method," *Computational Geosciences*, vol. 18, no. 5, pp. 747–762, 2014.

[4] A. Khare and S. Rangnekar, "A review of particle swarm optimization and its applications in Solar Photovoltaic system," *Applied Soft Computing Journal*, vol. 13, no. 5, pp. 2997–3006, 2013.

[5] S. Alam, G. Dobbie, Y. S. Koh, P. Riddle, and S. Ur Rehman, "Research on particle swarm optimization based clustering: a systematic review of literature and techniques," *Swarm and Evolutionary Computation*, vol. 17, pp. 1–13, 2014.

[6] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[7] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 35, no. 6, pp. 1272–1282, 2005.

[8] M. S. Kıran, M. Gündüz, and Ö. K. Baykan, "A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum," *Applied Mathematics and Computation*, vol. 219, no. 4, pp. 1515–1521, 2012.

[9] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, and L. M. Wang, "An improved GA and a novel PSO-GA-based hybrid algorithm," *Information Processing Letters*, vol. 93, no. 5, pp. 255–261, 2005.

[10] J. Robinson, S. Sinton, and Y. R. Samii, "Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna," in *Proceedings of the IEEE International Symposium in Antennas and Propagation Society*, vol. 1, pp. 314–317, San Antonio, Tex, USA, 2002.

[11] Q. Shen, W.-M. Shi, and W. Kong, "Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data," *Computational Biology and Chemistry*, vol. 32, no. 1, pp. 52–59, 2008.

[12] H.-L. Shieh, C.-C. Kuo, and C.-M. Chiang, "Modified particle swarm optimization algorithm with simulated annealing behavior and its numerical verification," *Applied Mathematics and Computation*, vol. 218, no. 8, pp. 4365–4383, 2011.

[13] B. Alatas, E. Akin, and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons and Fractals*, vol. 40, no. 4, pp. 1715–1734, 2009.

[14] C.-H. Yang, S.-W. Tsai, L.-Y. Chuang, and C.-H. Yang, "An improved particle swarm optimization with double-bottom chaotic maps for numerical optimization," *Applied Mathematics and Computation*, vol. 219, no. 1, pp. 260–279, 2012.

[15] C. Li, J. Zhou, P. Kou, and J. Xiao, "A novel chaotic particle swarm optimization based fuzzy clustering algorithm," *Neurocomputing*, vol. 83, pp. 98–109, 2012.

[16] S. L. Song, L. Kong, and J. J. Cheng, "A novel particle swarm optimization algorithm model with centroid and its application," *International Journal of Intelligent Systems and Applications*, vol. 1, no. 1, pp. 42–49, 2009.

[17] J. Gou, Z. Y. Wu, and J. Wang, "An improved particle swarm optimization algorithm based on self-adapted comprehensive learning," *Advanced Science Letters*, vol. 11, no. 1, pp. 668–675, 2012.

[18] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, vol. 3, pp. 1945–1950, Washington, DC, USA, July 1999.

[19] C. Guimin, H. Xinbo, J. Jianyuan, and M. Zhengfeng, "Natural exponential inertia weight strategy in particle swarm optimization," in *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA '06)*, vol. 1, pp. 3672–3675, June 2006.

[20] J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *Proceedings of the 3rd World Congress on Nature and Biologically Inspired Computing (NaBIC '11)*, pp. 633–640, Salamanca, Spain, October 2011.

[21] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3658–3670, 2011.

[22] P. Chauhan, K. Deep, and M. Pant, "Novel inertia weight strategies for particle swarm optimization," *Memetic Computing*, vol. 5, no. 3, pp. 229–251, 2013.

[23] T. O. Ting, Y. H. Shi, S. Cheng, and S. Lee, "Exponential inertia weight for particle swarm optimization," in *Proceedings of the 3rd International Conference on Advances in Swarm Intelligence*, vol. 1, pp. 83–90, June 2012.