Research Article

An Improved Self-Adaptive PSO Algorithm with Detection Function for Multimodal Function Optimization Problems

YingChao Zhang, 1,2 Xiong Xiong, 2 and Qi Dong Zhang 3

1 Institute of Information and System Science, Nanjing University of Information Science and Technology, Nanjing 210044, China
2 Key Laboratory of Meteorological Disaster of Ministry of Education, Nanjing University of Information Science and Technology, Nanjing 210044, China
3 College of Electrical Engineering, Southeast University, Nanjing 210018, China

Correspondence should be addressed to Xiong Xiong; nxgxiong@163.com

Received 8 June 2013; Revised 20 November 2013; Accepted 4 December 2013

Academic Editor: Jianming Shi

Copyright © 2013 YingChao Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presented an improved self-adaptive particle swarm optimization (IDPSO) algorithm with detection function to solve multimodal function optimization problems. To overcome the premature convergence of PSO in a short time, the evolution direction of each particle is redirected dynamically by tuning the three parameters of IDPSO in the evolution process. Numerical results on several benchmark functions indicate that the IDPSO strategy outperformed other variants of PSO.

1. Introduction

Particle swarm optimization (PSO), introduced by Kennedy and Eberhart [1, 2] in 1995, originates from the simulation of birds behavior and fish behavior. Essentially, PSO algorithm is a stochastic global optimization method to find the optimal or global optimum in the landscape of objective function, similar to other intelligent optimization algorithms, such as the mechanism of genetic algorithm (GA) and simulated annealing algorithm (SA). Compared with other evolutionary methods, PSO has an advantage of its implementation and the good tradeoff between exploration and exploitation ability [3–8]. The fundamental idea of PSO is inspired by research done on the behavior of bird flocking and is conducted by means of biological population model put forward by the biologist Heppner [9–11].

PSO is inspired from this organism’s behavior characteristic and used to solve the optimization problem. In PSO, each potential solution to the optimization problem can be conceived as a point in N-dimensional search space; we call it “particles” [12, 13]. And all particles have an adaptive value determined by an objective function, and each particle has speed that determines the direction and distance for flying; then the particles search in the solution space following the optimal particles. Through the study of birds flying, scholars found that the birds only trail a limited number of their neighbours, but it seemed that the birds are in control of a certain range; that is to say, the complex global action is caused by the interaction of simple rules.

PSO is an algorithm with simple structure, simple parameter setting, and fast convergence speed, which has been widely applied in function optimization, mathematical modeling, system control, and some other areas [14–18]. The research, which has been done since particle swarm optimization was proposed, mainly includes the following aspects:

(i) improvement of PSO,
(ii) theoretical analysis of PSO,
(iii) biological basis of PSO,
(iv) comparative analysis between PSO and other evolutionary algorithms,
(v) application of PSO.

How to improve the convergence speed and how to guarantee the convergence of PSO are the main problems of PSO improvement [19] and are gradually turning into a hot
2. Mathematical Problems in Engineering

Table 1: Basic characters of the benchmark functions.

| Function       | Range     | Opt. value | Opt. position |
|----------------|-----------|------------|---------------|
| Multimodal     | [−1, 2]   | 3.3099     | (1.64, 2)     |
| Schaffer       | [−10, 10] | 0          | (0, 0)        |
| Griewank       | [−600, 600]| 0         | (0, 0, ... , 0)|
| Shubert        | [−10, 10] | −186.7309  | (−1.43, 0.8)  |

In this field. In order to weigh the relationship between local search and global search, Shi et al. [20–22] proposed improved particle swarm optimization with inertia weight to control the exploitation and exploration of PSO better, and he also formed a topical improved particle swarm algorithm with inertia factor, which is called topical particle swarm optimization [23].

The remainder of this paper is organized as follows. Section 2 describes the basic PSO and presents the details of the proposed algorithm. Section 3 introduces the stability and convergence analysis of IDPSO and indicates stability and optimization [23].

2. Method

2.1. Basic Particle Swarm Optimization. The basic principle of particle swarm optimization [1, 2] is that each individual in the N-dimensional search space is considered as a particle with no weight and volume, and that each particle flies at a certain speed; the quality of particle is measured by the fitness function, and particles can modify the speed dynamically according to the flying experience of themselves and other particles, in order to fly to the best position in the group. If \( X_i = (x_{i1}, x_{i2}, ..., x_{in}) \) is the current position of particle \( i \), \( V_i = (v_{i1}, v_{i2}, ..., v_{in}) \) is the current speed, \( P_i = (p_{i1}, p_{i2}, ..., p_{in}) \) is the best position that particle \( i \) has visited, and \( P_g = (p_{g1}, p_{g2}, ..., p_{gn}) \) is the best positions that all particles have visited, then the speed and position of each particle \( i \) for each generation are shown as follows:

\[
\begin{align*}
v_i(t + 1) &= \omega \cdot v_i(t) + c_1 \cdot \text{rand}() \cdot (p_i - x_i(t)) \\
&\quad + c_2 \cdot \text{Rand}() \cdot (p_g - x_i(t)),
\end{align*}
\]

\[
x_i(t + 1) = x_i(t) + v_i(t + 1),
\]

where \( \omega \) is inertia weight, \( c_1 \) and \( c_2 \) are positive acceleration factors, and \( \text{rand}() \) and \( \text{Rand}() \) are random functions changing in the range of \([0, 1]\). In (1), the first part is the previous speed of particles, the second part is cognition, which stands for particle’s own thinking, and the third part is social component, which represents the information sharing and cooperation of particles. The flight speed of particles is limited by the maximum and minimum speed of all particles. The PSO with larger inertia weight has a faster convergence speed and works well in global search, while the PSO with smaller inertia weight can reach a more accurate optimum value but only works well in local search [24].

2.2. Improved Particle Swarm Optimization. Researchers have shown that the three parameters \( \omega, c_1, \) and \( c_2 \) have a significant impact on performance of the algorithm, especially the impact of inertia weight. The impact is different on different conditions and is also different at different times under the same condition. Now dynamic inertia weight modification is used to train the appropriate value of \( \omega \), in order to coordinate between search accuracy and search speed. Usually, \( \omega \) is modified gradually in descending within the limit of \( 0.8 < \omega < 1.4 \) [20], so that search space can be changed steadily from the global to the local. Decreasing inertia weight particle swarm optimization is a topical algorithm, of which inertia weight \( \omega \) decreases linearly from 0.9 to 0.4 [20–22]. Some scholars propose the increasing inertia weight particle swarm optimization of which inertia weight \( \omega \) increases linearly from 0.4 to 0.9 [25]. The further development of these algorithms is fuzzy adaptive PSO, which optimizes the value of \( \omega \) dynamically using adaptive fuzzy inertia weight controller and can solve many problems satisfactorily.

However, the search process of PSO is a nonlinear complex process, where mechanical linear transformation of \( \omega \) cannot achieve the accurate balance between local search and global search. Some researchers show that performance, when the inertia weight decreases gradually as the iteration proceeds, is not always better than when the inertia weight is at an appropriate fixed value.

In this paper, a self-adapting particle swarm optimization is proposed, which introduces a detection function \( \varphi(t) = \frac{|p_g - x_i(t - 1)|/(|p_i - x_i(t - 1)|)}{|p_i - x_i(t - 1)|} \) is the Euclidean distance from the position \( X_i \) of particle \( i \) to the best position \( p_g \); all particles have passed at time \( t - 1 \), and \( p_i - x_i(t - 1) \) is the Euclidean distance from the position \( X_i \), of particle \( i \) to the best position \( p_i \) that the particle \( i \) has passed at time \( t - 1 \). The algorithm may adjust \( \omega, c_1, \) and \( c_2 \) dynamically via the value of function \( \varphi(t) \), so that it can optimize dynamically by taking both global search and local search into account. The modification of inertia weight in the new algorithm is based on the variable sigmoid function. According to the different values of detection function, the inertia weight is decreasing irregularly in real time to find the optimum solution in a relatively short time and to reduce the time cost. The values of these two acceleration factors \( c_1, c_2 \) are modified dynamically in real time according to detection function \( \varphi(t) \).

Improved PSO is shown as follows (it is called IDPSO):

\[
v_i(t + 1) = \omega \cdot v_i(t) + c_1 \cdot \text{rand}() \cdot (p_i - x_i(t)) \\
&\quad + c_2 \cdot \text{Rand}() \cdot (p_g - x_i(t)),
\]

\[
\omega(t) = \frac{\omega_{\text{initial}} - \omega_{\text{final}}}{1 + e^{\theta(t) - ((1+\ln(\varphi(t)))/K_{\text{max}})/\mu}} + \omega_{\text{final}},
\]

\[
c_1(t) = c_1 \cdot \varphi^{-1}(t),
\]

\[
c_2(t) = c_2 \cdot \varphi(t),
\]

where \( \omega_{\text{initial}} \) and \( \omega_{\text{final}} \) represent the initial value and the final value of \( \omega \), respectively (in (4), we can obtain that \( 0 < \omega < 2 \)), \( K_{\text{max}} \) is the maximum number of evolutionary generation, \( t \)
is the current number of evolutionary generation, \( \varphi(t) \) is the
detection function, and \( \mu \) is an adjustment factor to ensure that \( \omega, \omega_{\text{initial}}, \) and \( \omega_{\text{final}} \) keep the reverse s change.

3. Stability and Convergence Analysis for IDPSO

Convergence analysis plays a great role in understanding the essence and convergence condition of PSO algorithm [26–30]. In this paper we reduce a multidimensional system to one-dimensional for discussion when we analyze the stability and convergence of IDPSO. This is decided by the independence of the multidimensional variables in IDPSO. In (3), (4), (5), and (6), \( \omega \) is inertia weight, \( c_1 \) and \( c_2 \) are positive acceleration factors, and \( \text{rand}(t) \) and \( \text{Rand}(t) \) are random functions fluctuating in the range of \([0, 1] \). Suppose that \( \varphi_0 = \omega(t), \varphi_1 = c_1(t) \cdot \text{rand}(t) \) and \( \varphi_2 = c_2(t) \cdot \text{rand}(t) \); then we have

\[
v(t + 1) = \varphi_0 \cdot v(t) + \varphi_1 \cdot \left( p_i - x_i(t) \right) + \varphi_2 \cdot \left( g_i - x_i(t) \right).
\]

(7)

By formula transformation, we have

\[
v(t + 1) = \varphi_0 \cdot v(t) + \varphi_1 \cdot \left( 1 - \varphi_1 - \varphi_2 \right) \cdot x_i(t) + \varphi_1 \cdot p_i + \varphi_2 \cdot p_g,
\]

(8)

Taking (8) and (2) together, one obtains

\[
x_i(t + 1) = \varphi_0 \cdot v(t) + \varphi_1 \cdot \left( 1 - \varphi_1 - \varphi_2 \right) \cdot x_i(t) + \varphi_1 \cdot p_i + \varphi_2 \cdot p_g,
\]

(9)

Equation (9) can also be rewritten as the matrix form as follows:

\[
\begin{bmatrix}
v_i(t + 1) \\
x_i(t + 1)
\end{bmatrix} = \begin{bmatrix}
\varphi_0 - (\varphi_1 + \varphi_2) \\
\varphi_1 - (\varphi_1 + \varphi_2)
\end{bmatrix} \cdot \begin{bmatrix}
v_i(t) \\
x_i(t)
\end{bmatrix}
+ \begin{bmatrix}
\varphi_1 & \varphi_2 \\
\varphi_1 & \varphi_2
\end{bmatrix} \cdot \begin{bmatrix}
p_i(t) \\
p_g(t)
\end{bmatrix}.
\]

(10)

Here, the parameters \( \varphi_0 \) and \( \varphi_1 + \varphi_2 \) are assumed to be constants. Then the IDPSO system is a linear time-invariant discrete system. When \( t \to \infty \) and \( v_i(t) \) and \( x_i(t) \) approach fixed values, the necessary and sufficient condition for the IDPSO system reaching stability is that \( \lambda_1, \lambda_2 \) amplitudes should be less than one, where \( \lambda_1, \lambda_2 \) are the eigenvalues of the vector matrix \( \begin{bmatrix}
\varphi_0 - (\varphi_1 + \varphi_2) \\
\varphi_1 - (\varphi_1 + \varphi_2)
\end{bmatrix} \) and we obtain

\[
\lambda_{1,2} = \frac{1 + \varphi_0 - (\varphi_1 + \varphi_2) \pm \sqrt{(\varphi_1 + \varphi_2 - \varphi_0 - 1)^2 - 4\varphi_0}}{2}.
\]

(11)

From (11), we will discuss the following three cases.

(1) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| > 2\sqrt{\varphi_0} \), \( \lambda_1, \lambda_2 \) are both real roots. For the IDPSO system to be stable, the amplitudes of the two real roots must be less than one; then we have

\[
-2\sqrt{\varphi_0} < 1 + \varphi_0 - (\varphi_1 + \varphi_2) \quad \text{and} \quad -2\sqrt{\varphi_0} < 1 + \varphi_0 - (\varphi_1 + \varphi_2)
\]

(12)

\[
- \sqrt{(\varphi_1 + \varphi_2 - \varphi_0 - 1)^2 - 4\varphi_0} < 2\sqrt{\varphi_0},
\]

(13)

Combining (12) and (13), we obtain

\[
-2\sqrt{\varphi_0} < \varphi_1 + \varphi_2 - \varphi_0 - 1 < 2\sqrt{\varphi_0}.
\]

(14)

It is obvious that the contradiction between (14) the assumption that \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| > 2\sqrt{\varphi_0} \), which leads to the system (10) being unstable.

(2) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| < 2\sqrt{\varphi_0}, \lambda_1, \lambda_2 \) are both complex roots; their amplitude is

\[
\text{if } \varphi_0 > 1, \text{ we have}
\]

\[
0 < \varphi_1 + \varphi_2 < 1 + \varphi_0 + \sqrt{2 + 2\varphi_0}.
\]

(17)

(i) when \( 0 < \varphi_0 < 1 \), we have

\[
\varphi_0 - 1 < \varphi_1 + \varphi_2 < 3 + \varphi_0.
\]

(18)

Then when \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| < 2\sqrt{\varphi_0} \), and if \( \varphi_0, (\varphi_1 + \varphi_2) \) satisfy the condition, as the shaded part shown in Figure 1, it will make the system (10) stable.

(3) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| = 2\sqrt{\varphi_0} \) and \( \lambda_1 = \lambda_2 = \pm \sqrt{\varphi_0} \), the system (10) is stable.

From the above, we analyze the stability of IDPSO for the case of one-dimensional particles, which is also suitable for its convergence problem. In this case, we ignore the subscript of \( v_i \) and denote the local best position and global best position as \( p_g \) and \( g_i \). By (2) and (7), we obtain

\[
v(t + 2) = v(t + 1) + \varphi_1 \cdot \left( p_g - x(t + 1) \right)
\]

(19)

\[
x(t + 2) = x(t + 1) + v(t + 2).
\]

(20)
Combining (19) and (20), we obtain
\[ x(t+2) = x(t+1) + \nu(t+2) \]
\[ = x(t+1) + \varphi_0 \cdot \nu(t+1) + \varphi_1 \cdot (p_b - x(t+1)) + \varphi_2 \cdot (g_b - x(t+1)) \]
\[ = (1 - \varphi_1 - \varphi_2) \cdot x(t+1) + \varphi_0 \cdot \nu(t+1) + \varphi_1 \cdot p_b + \varphi_2 \cdot g_b. \]  
(21)

According to \( x(t+1) = x(t) + \nu(t+1) \), we have
\[ x(t+2) = (1 - \varphi_1 - \varphi_2) \cdot x(t+1) + \varphi_0 \cdot \nu(t+1) + \varphi_1 \cdot p_b + \varphi_2 \cdot g_b. \]  
(22)

Obviously, (22) is a two-order constant coefficient homogeneous difference equation and its characteristic equation is
\[ \lambda^2 = (1 + \varphi_0 - \varphi_1 - \varphi_2) \cdot \lambda - \varphi_0. \]  
(23)
And the solution of (23) is
\[ \lambda_{1,2} = \frac{1 + \varphi_0 - (\varphi_1 + \varphi_2) \pm \sqrt{(\varphi_1 + \varphi_2 - \varphi_0 - 1)^2 - 4\varphi_0}}{2}. \]  
(24)

Then we have the following.

(1) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| > 2\sqrt{\varphi_0}, \lambda_1 \) and \( \lambda_2 \) are both real roots:
\[ \lambda_1 = \lambda_2 \]
\[ = \frac{1 + \varphi_0 - (\varphi_1 + \varphi_2) \pm \sqrt{(\varphi_1 + \varphi_2 - \varphi_0 - 1)^2 - 4\varphi_0}}{2}. \]  
(25)

Solving (22), we obtain
\[ x(t) = c^{-1} \cdot (\varphi_1 \cdot p_b + \varphi_2 \cdot g_b) \]
\[ + \frac{\lambda_2 \cdot b_1 - b_2}{\lambda_2 - \lambda_1} \cdot \lambda_1^{k+1} + \frac{b_2 - \lambda_1 \cdot b_1}{\lambda_2 - \lambda_1} \cdot \lambda_2^k, \]
\[ b_1 = x(0) - c \cdot (\varphi_1 \cdot p_b + \varphi_2 \cdot g_b), \]
\[ b_2 = (1 - c) \cdot x(t) + \varphi_0 \cdot \nu(0) + \varphi_1 \cdot p_b + \varphi_2 \cdot g_b, \]  
(26)

(2) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| < 2\sqrt{\varphi_0}, \lambda_1 \) and \( \lambda_2 \) are both complex roots:
\[ \lambda_1 = \lambda_2 = \frac{1 + \varphi_0 - \varphi_1 - \varphi_2}{2} \]
\[ + i \sqrt{(\varphi_1 + \varphi_2 - \varphi_0 - 1)^2 - 4\varphi_0}. \]  
(27)

Solving (22), we obtain
\[ x(t) = c^{-1} \cdot (\varphi_1 \cdot p_b + \varphi_2 \cdot g_b) \]
\[ + \frac{\lambda_2 \cdot b_1 - b_2}{\lambda_2 - \lambda_1} \cdot \lambda_1^{k+1} + \frac{b_2 - \lambda_1 \cdot b_1}{\lambda_2 - \lambda_1} \cdot \lambda_2^k, \]
\[ b_1 = x(0) - c \cdot (\varphi_1 \cdot p_b + \varphi_2 \cdot g_b), \]
\[ b_2 = (1 - c) \cdot x(t) + \varphi_0 \cdot \nu(0) + \varphi_1 \cdot p_b + \varphi_2 \cdot g_b, \]  
(28)

(3) When \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| = 2\sqrt{\varphi_0}, \lambda_1 \) and \( \lambda_2 \) are both equal roots:
\[ \lambda = \lambda_1 = \lambda_2 = \frac{1 + \varphi_0 - \varphi_1 - \varphi_2}{2}. \]  
(29)

Solving (22), we obtain
\[ x(t) = x(0) + \frac{(1 - c) \cdot x(0) + \varphi_0 \cdot \nu(0) + \varphi_1 \cdot p_b + \varphi_2 \cdot g_b}{\lambda}. \]  
(30)

In (30), \( c \) is a nonzero constant.

For the above, three cases can be identified; if \( t \to \infty \) and \( x(t) \) has its limit, then IDPSO converges algorithm. Generally, if we want to prove that algorithms converge with probability 1, we just need \( \|A_1\| < 1 \) and \( \|A_2\| < 1 \). Then we obtain the following:

(1) when \( |\varphi_1 + \varphi_2 - \varphi_0 - 1| > 2\sqrt{\varphi_0}, \) if \( 0 < (\varphi_1 + \varphi_2) \) and \( 0 < 2\varphi_0 - (\varphi_1 + \varphi_2) + 2 \), then we wish to show that IDPSO is convergent;
In IDPSO, particles can “rationally” control the “exploitation” based on the value of the detection function. If \( \varphi(t) \geq 1 \), the algorithm prefers “exploration,” chooses the smaller step size with improved local search, so that it can achieve the smooth transition from “exploration” to “exploitation” adaptively based on sigmoidal function.

4.2. Analysis of the Value of \( c_1 \) and \( c_2 \) in IDPSO. The values of \( c_1 \) and \( c_2 \) are modified dynamically according to the detection function \( \varphi(t) \) in IDPSO. If \( \varphi(t) \geq 1 \), the weight of social component in (3) should be increased to improve the information sharing and cooperation of particles, so the value of \( c_1 \) is decreased and the value of \( c_2 \) is increased. If \( \varphi(t) < 1 \), weight of “cognition” in (3) should be increased to strengthen the particle’s own influence, so the value of \( c_1 \) is increased and the value of \( c_2 \) is decreased.

In IDPSO, particles can “rationally” control the “exploitation” and “exploitation” of PSO based on the dynamic modification of parameters \( \omega, c_1, \) and \( c_2 \), play a critical role in the performance of the algorithm, and effectively avoid falling into the local minimum quickly.

5. Experiments

5.1. Benchmark Functions. In this section, three variants of PSO are compared with the IDPSO. In the first variant of PSO(PSO+SIW), the inertia weight was set to 0.6 [6]. In the second variant of PSO(PSO+LIW), the inertia weight was set to 0.9 at the beginning of the run and was made to decrease linearly to 0.4 at the maximum number of iterations [2]. In the second variant of PSO(PSO+NLIW), the inertia weight was set to 0.9 at the beginning of the run and made to decrease nonlinearly to 0.4 at the maximum number of iterations [14]. For comparison, four benchmark functions were used to compare the performance of the IDPSO algorithm with the three variants of PSO above:

Multimodal function

\[
f(x, y) = x \cdot \sin(4\pi x) - y \cdot \sin(4\pi y + x + 1); \quad (31)
\]
Figure 4: Evolution of the optimal value and average value of multimodal function for the algorithms (a) PSO+SIW, (b) PSO+LIW, (c) PSO+NLIW, and (d) IDPSO.

Figure 5: Evolution of the optimal value and average value of Schaffer function for the algorithms (a) PSO+SIW, (b) PSO+LIW, (c) PSO+NLIW, and (d) IDPSO.

Figure 6: Evolution of the optimal value and average value of Griewank function for the algorithms (a) PSO+SIW, (b) PSO+LIW, (c) PSO+NLIW, and (d) IDPSO.
The optimal value of fitness  
The average value of fitness  

Schaffer function  
\[ f(x_1, x_2) = 0.5 + \left( \frac{\sin \sqrt{x_1^2 + x_2^2} - 0.5}{1 + 0.001(x_1^2 + x_2^2)} \right)^2 \]  

Griewank function  
\[ f(x_i) = \frac{1}{4000} \sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1; \]  

Shubert function  
\[ f(x, y) = \frac{1}{\prod_{i=1}^{5} (i + 1)} \right\} \times \left\{ \sum_{i=1}^{5} i \cos \left[ (i + 1) x + 1 \right]\right\} \times \left\{ \sum_{i=1}^{5} i \cos \left[ (i + 1) y + 1 \right]\right\}. \]  

The basic characters of the four benchmark functions are summarized in Table 1. In all cases, the population size was set to 20, and the maximum number of iterations was set to 1000. The acceleration constants \( c_1 = c_2 = 2.0 \). In all cases for which IDPSO was used, \( \omega_{\text{initial}} = 0.9, \omega_{\text{final}} = 0.4, \) and \( \mu = 100. \)

5.2. Results and Discussion. The numerical results using the four different methods for the four benchmark test functions are shown in Figures 4–7. In all cases, the numerical results show that PSO+SIW can reach a local optimum and suffer from premature convergence in a short time. According to Figures 4 and 7, IDPSO significantly outperforms other algorithms for the benchmark functions. In Figure 4, only IDPSO can reach the global optimum and effectively avoid premature convergence. In Figure 7, both PSO+NLIW and IDPSO can reach the global optimum and converge to the criteria for the Shubert function. But IDPSO performs better than PSO+NLIW and IDPSO reaches the global optimum and converges to the criteria faster than PSO+NLIW. From Figures 5(b)–5(d) and 6(b)–6(d), IDPSO performs better than other algorithms in reaching the global optimum. And the average values of fitness present a vibration state, especially in Figure 5(d). From the above, it can be easily observed that the proposed variant IDPSO outperforms other variants of PSO significantly by providing a more accurate optimal solution and converges to the criteria with a greater probability.

6. Conclusions
A self-adapting particle swarm optimization with detection functions, referred to IDPSO algorithm, is presented in this paper, which may adjust the values of \( \omega, c_1, \) and \( c_2 \) dynamically in different circumstances based on the analysis of the detection function, \( \phi(t) \). This approach ensures that the PSO explores the search space effectively. Experimental results show that IDPSO achieves better performance than other variants of PSO investigated in this paper.

Acknowledgments
This work was supported by Grants from a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions, PAPD, Department of Education of Jiangsu Province and “Public Sector (meteorological) Special Research Projects” (GYHY201106040 to Yingchao Zhang), China Meteorological Administration, and “2013 Colleges and Universities in Jiangsu Province plans to graduate research and innovation (CXZZ13_0520 to Xiong Xiong).” The paper reflects the opinions of the authors who bear full responsibility for errors and omissions.

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, December 1995.
[2] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in Proceedings of the International Symposium on Micro Machine and Human Science, pp. 39–43, October 1995.
[3] J. Liu, X. Ren, and H. Ma, “A new PSO algorithm with random C/D switchings,” Applied Mathematics and Computation, vol. 218, no. 19, pp. 9579–9593, 2012.
[4] L. Sangwook, P. Haesun, and J. Moongu, “Binary particle swarm optimization with bit change mutation,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E90-A, no. 10, pp. 2253–2256, 2007.
[5] 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.
[6] I. C. Trelea, “The particle swarm optimization algorithm: convergence analysis and parameter selection,” Information Processing Letters, vol. 85, no. 6, pp. 317–325, 2003.
[7] L.-Y. Chuang, S.-W. Tsai, and C.-H. Yang, “Chaotic catfish particle swarm optimization for solving global numerical optimization problems,” Applied Mathematics and Computation, vol. 217, no. 16, pp. 6900–6916, 2011.
[8] W.-B. Liu and X.-J. Wang, “An evolutionary game based particle swarm optimization algorithm,” Journal of Computational and Applied Mathematics, vol. 214, no. 1, pp. 30–35, 2008.
[9] H. Pomery and F. Heppner, “Laboratory determination of startle reaction time of the starling (Sturnus vulgaris),” Animal Behaviour, vol. 25, no. 3, pp. 720–725, 1977.
[10] C. Grosan, A. Abraham, and M. Chis, Swarm Intelligence in Data Mining, vol. 34 of Studies in Computational Intelligence, Springer, New York, NY, USA, 2006.
[11] Y. Liu and K. M. Passino, Swarm Intelligence: Literature Overview, 2000.
[12] Y. Nomura, H. Furuta, and M. Hirokane, “An integrated fuzzy control system for structural vibration,” Computer-Aided Civil and Infrastructure Engineering, vol. 22, no. 4, pp. 306–316, 2007.
[13] J. Pugh and A. Martinoli, “Multi-robot learning with particle swarm optimization,” in Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS ’06), pp. 441–448, Hakodate, Japan, May 2006.
[14] R. Akbari and K. Ziarati, “A rank based particle swarm optimization algorithm with dynamic adaptation,” Journal of Computational and Applied Mathematics, vol. 235, no. 8, pp. 2694–2714, 2011.
[15] L. M. Almeida and T. B. Ludermir, “A multi-objective memetic and hybrid methodology for optimizing the parameters and performance of artificial neural networks,” Neurocomputing, vol. 73, no. 7–9, pp. 1438–1450, 2010.
[16] P. K. Tripathi, S. Bandyopadhyay, and S. K. Pal, “Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients,” Information Sciences, vol. 177, no. 22, pp. 5033–5049, 2007.
[17] Y. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. G. Harley, “Particle swarm optimization: basic concepts, variants and applications in power systems,” IEEE Transactions on Evolutionary Computation, vol. 12, no. 2, pp. 171–195, 2008.
[18] R. Brits, A. P. Engelbrecht, and F. van den Bergh, “Locating multiple optima using particle swarm optimization,” Applied Mathematics and Computation, vol. 189, no. 2, pp. 1859–1883, 2007.
[19] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: empirical results,” Evolutionary Computation, vol. 8, no. 2, pp. 173–195, 2000.
[20] R. C. Eberhart and Y. Shi, “Comparing inertia weights and constriction factors in particle swarm optimization,” in Proceedings of the Congress on Evolutionary Computation (CEC ’00), pp. 84–88, July 2000.
[21] Y. Shi and R. Eberhart, “Modified particle swarm optimizer,” in Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC ’98), pp. 69–73, May 1998.
[22] R. C. Eberhart and Y. Shi, “Comparison between genetic algorithms and particle swarm optimization,” in Proceedings of the 7th International Conference Evolutionary Programming VII, pp. 611–616, 1998.
[23] 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.
[24] A. Atiyabi, S. Phon-Amnuaisuk, and C. K. Ho, “Applying area extension PSO in robotic swarm,” Journal of Intelligent and Robotic Systems, vol. 58, no. 3–4, pp. 253–285, 2010.
[25] Y. Zhang, X. Xiong, C. Chen, and X. Huang, “A particle swarm optimization based on dynamic parameter modification,” Applied Mechanics and Materials, vol. 40–41, pp. 201–205, 2011.
[26] I. C. Trelea, “The particle swarm optimization algorithm: convergence analysis and parameter selection,” Information Processing Letters, vol. 85, no. 6, pp. 317–325, 2003.
[27] X. Dong, Y. Zhao, Y. Xu, Z. Zhang, and P. Shi, “Design of PSO fuzzy neural network control for ball and plate system,” International Journal of Innovative Computing, Information and Control, vol. 7, no. 12, pp. 7091–7103, 2011.
[28] T.-Y. Sun, C.-C. Liu, S.-J. Tsai, S.-T. Hsieh, and K.-Y. Li, “Cluster Guide Particle Swarm Optimization (CGPSO) for underdetermined blind source separation with advanced conditions,” IEEE Transactions on Evolutionary Computation, vol. 15, no. 6, pp. 798–811, 2011.
[29] Ž. Kanović, M. R. Rapaić, and Z. D. Jeličić, “Generalized particle swarm optimization algorithm—theoretical and empirical analysis with application in fault detection,” Applied Mathematics and Computation, vol. 217, no. 24, pp. 10175–10186, 2011.
[30] J. Zeng, J. Jie, and Z. Cui, Particle Swarm Optimization Algorithm, Science Press, Beijing, China, 2004.
